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. Templates 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. 4).
- 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 text file whose format complies with the JSON standard. You can save these files with any extension, such as `.json`, `.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.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A sample template",
  "Resources" : {
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "ami-2f726546",
        "InstanceType" : "t1.micro",
        "KeyName" : "testkey",
        "BlockDeviceMappings" : [
          {
            "DeviceName" : "/dev/sdm",
            "Ebs" : {
              "VolumeType" : "io1",
              "Iops" : "200",
              "DeleteOnTermination" : "false",
              "VolumeSize" : "20"
            }
          }
        ]
      }
    }
  }
}
```

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:

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A sample template",
  "Resources" : {
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "ami-2f726546",
        "InstanceType" : "t1.micro",
        "KeyName" : "testkey",
        "BlockDeviceMappings" : [
          {
            "DeviceName" : "/dev/sdm",
            "Ebs" : {
              "VolumeType" : "io1",
              "Iops" : "200",
              "DeleteOnTermination" : "false",
              "VolumeSize" : "20"
            }
          }
        ]
      }
    },
    "MyEIP" : {
      "Type" : "AWS::EC2::EIP",
      "Properties" : {
        "InstanceId" : {"Ref": "MyEC2Instance"}
      }
    }
  }
}
```

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. 115).

Stacks

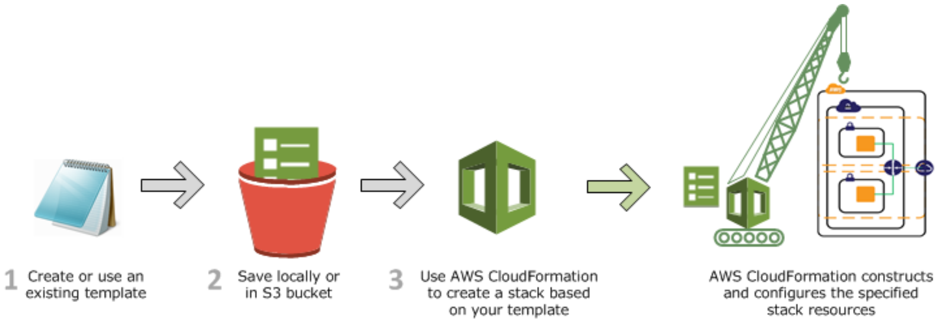
When you use AWS CloudFormation, you manage related resources as a single unit called a stack. In other words, you create, update, and delete a collection of resources by creating, updating, and deleting stacks. All the resources in a stack are defined by the stack's AWS CloudFormation template. Suppose you created a template that includes an Auto Scaling group, Elastic Load Balancing load balancer, and an Amazon 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. To update resources, you first modify the original stack template and then update your stack by submitting the modified template. 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. 70).

How Does AWS CloudFormation Work?

Whenever you create a stack, AWS CloudFormation makes underlying service calls to AWS to provision and configure your resources. Note that AWS CloudFormation can only perform actions that you have permission to do. For example, to create Amazon 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 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 Amazon 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 create stack workflow:


1 Create or use an existing template
2 Save locally or in S3 bucket
3 Use AWS CloudFormation to create a stack based on your template
AWS CloudFormation constructs and configures the specified stack resources

1. You can write an AWS CloudFormation template (a JSON-formatted document) in a text editor or pick an existing template. The template describes the resources you want and their settings. For example, suppose you want to create an Amazon EC2. Your template can declare an instance Amazon EC2 and describe its properties, as shown in the following example:

   ```
   {
     "AWSTemplateFormatVersion" : "2010-09-09",
     "Description" : "A simple Amazon EC2 instance",
     "Resources" : {
       "MyEC2Instance" : {
         "Type" : "AWS::EC2::Instance",
         "Properties" : {
           "ImageId" : "ami-2f726546",
           "InstanceType" : "t1.micro"
         }
       }
     }
   }
   ```

2. If you created a template, save the AWS CloudFormation template with any file extension like `.json` or `.txt`. You can save the file locally or in an Amazon S3 bucket.
3. You create an AWS CloudFormation stack and specify the location of your template file. The location can be a file on your local computer or an Amazon S3 URL. You can create stacks by using the AWS CloudFormation console (p. 72), API, or AWS CLI.

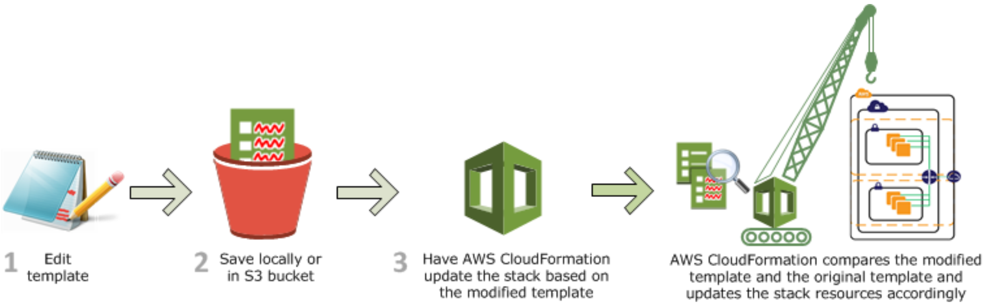
   > **Note**
   > If you specify a local template file, AWS CloudFormation uploads it to an Amazon S3 bucket in your AWS account. AWS CloudFormation creates a unique bucket for each region in which you upload a template file. The buckets are accessible to anyone with Amazon S3 permissions in your AWS account. 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.

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

After all the resources have been created, AWS CloudFormation signals that your stack has been successfully created. Then you can start to use all the resources in your stack. If the stack creation fails, AWS CloudFormation rolls back any changes by deleting any resources that were created.

Update Stack Workflow

When you update a stack, you modify the original stack template. AWS CloudFormation compares the modified template with the original stack template and updates only the resources that you modified. The following diagram summarizes the update stack workflow:


1 Edit template
2 Save locally or in S3 bucket
3 Have AWS CloudFormation update the stack based on the modified template
AWS CloudFormation compares the modified template and the original template and updates the stack resources accordingly

Important

Updates can cause interruptions. Depending on the resource and property that you are updating, the update might interrupt or even replace an existing resource. For more information, see AWS CloudFormation Stacks Updates (p. 88).

1. You modify an AWS CloudFormation stack template in a text editor. For example, suppose you want to change the instance type for an Amazon EC2 instance. In the original stack template, change the instance type property for that instance.
2. You save the AWS CloudFormation template locally or in an Amazon S3 bucket.
3. You select the AWS CloudFormation stack that you want to update and specify the location of the modified template file. The location can be a file on your local computer or an Amazon S3 URL. You can update stacks by using the AWS CloudFormation console (p. 88), API, or AWS CLI.

   Note

   If you specify a local template file, AWS CloudFormation automatically uploads your template to an Amazon S3 bucket in your AWS account.

AWS CloudFormation compares the modified template with the original stack template and updates only the resources that you modified.

After all the resources have been updated, AWS CloudFormation signals that your stack has been successfully updated. If the stack updates fails, AWS CloudFormation rolls back any changes to the last known working state.

Delete Stack Workflow

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. 78), 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. 539) 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. 115).

- For more information about creating, updating, or deleting stacks, see Working with Stacks (p. 70).

Getting Started with AWS CloudFormation

If you're new to AWS CloudFormation, the guides in this section will help get you started quickly, provide you with fundamental information about using CloudFormation from the AWS Console, and guide you through using the AWS command line interface (CLI) so that you can manage your CloudFormation stacks from your system's command prompt.

Topics

Signing Up for an AWS Account

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 http://aws.amazon.com, and then click **Sign Up**.
2. Follow the on-screen instructions.

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

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: Sign up for the Service

Signing up for AWS CloudFormation also automatically signs you up for other AWS products you need, such as Amazon Elastic Compute Cloud, Amazon Relational Database Service and Amazon Simple Notification Service. You're not charged for any services unless you use them.

> **Note**
> 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.

To sign up for AWS CloudFormation

1. Go to http://aws.amazon.com/cloudformation, and then click **Sign Up for AWS CloudFormation**.
2. Follow the on-screen instructions.

If you don't already have an AWS account, you'll be prompted to create one when you sign up for AWS CloudFormation.

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

Step 2: 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 and an Amazon RDS DB Instance. The template also creates an Amazon EC2 and Amazon RDS security group to control firewall settings for the Amazon EC2 instance and the database 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 download or view the WordPress sample template from https://s3.amazonaws.com/cloudformation-templates-us-east-1/WordPress_Single_Instance_With_RDS.template.

  You don't need to download it unless you want to inspect it. You will use the template URL later in this guide.

A template is a JavaScript Object Notation (JSON) text file that contains the configuration information about the AWS resources you want to create in the stack. In this particular sample template, it 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 Amazon RDS database instance, which in this example has the logical name DBInstance:

```
"Resources" : {
  ...
  "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": "WebServer
SecurityGroup"} },
      "GroupDescription"      : "Frontend Access"
    }
  },
  ...
},
```

If you have created database instances before, you can recognize properties, such as `Engine`, `DBInstanceClass`, and `AllocatedStorage`, that determine the configuration of the database 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 Parameters** 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 Amazon RDS database instance resource:

```
"Parameters" : {
  ...
  "DBName" : {
    "Default": "wordpress",
    "Description" : "The WordPress 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 alpha
numeric characters."
  },

  "DBUsername" : {
    "Default": "admin",
    "NoEcho": "true",
    "Description" : "The WordPress database admin account user name",
    "Type": "String",
    "MinLength": "1",
    "MaxLength": "16",
    "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
    "ConstraintDescription" : "must begin with a letter and contain only alpha
numeric characters."
  },

  "DBPassword" : {
    "Default": "admin",
    "NoEcho": "true",
    "Description" : "The WordPress database admin account password",
    "Type": "String",
    "MinLength": "8",
    "MaxLength": "41",
    "AllowedPattern" : "[a-zA-Z0-9]*",
    "ConstraintDescription" : "must contain only alphanumeric characters."
  },

  "DBAllocatedStorage" : {
    "Default": "5",
    "Description" : "The size of the database (Gb)",
    "Type": "Number",
    "MinValue": "5",
    "MaxValue": "1024",
    "ConstraintDescription" : "must be between 5 and 1024Gb."
  },
  ...
},
```

In the `DBInstance` resource declaration, you see the `DBName` property specified with the `DBName` parameter:

```
  "DBInstance" : {
    "Type": "AWS::RDS::DBInstance",
    "Properties": {
      "DBName" : { "Ref" : "DBName" },
      ...
    }
  },
```

The braces contain a call to the Ref (p. 565) function with `DBName` as its input. The Ref function returns the value of the object it refers to. In this case, the Ref function sets the `DBName` property to the value that was specified for `DBName` 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 `DBInstance` contains the following property declaration:

```
"DBInstance" : {
  "Type": "AWS::RDS::DBInstance",
  "Properties": {
    ...
    "DBSecurityGroups"  : [{ "Ref" : "DBSecurityGroup" }],
    ...
  }
},
```

The `DBSecurityGroups` property takes a list of Amazon RDS database security groups. The Ref function has an input of `DBSecurityGroup`, which is the logical name of a database security group in the template, and adds the name of `DBSecurityGroup` to the `DBSecurityGroups` 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. 14).

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

Step 3: 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 4: Create the stack

You will create your stack based on the *WordPress-1.0.0* file discussed earlier. The template contains several AWS resources including an Amazon RDS database instance and an Amazon 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 **Stack** section, enter a stack name in the **Name** field. For this example, use **`MyWPTestStack`**. The stack name cannot contain spaces.
4. 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.amazonaws.com/cloudformation-templates-us-east-1/Word-
   Press_Single_Instance_With_RDS.template
   ```

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

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 5: 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 MyWPTest-
Stack User initiated
```

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

```
2013-04-24 18:59 UTC-7 CREATE_COMPLETE AWS::RDS::DBSecurityGroup...
```

```
2013-04-24 18:54 UTC-7 CREATE_IN_PROGRESS AWS::RDS::DBSecurityGroup...
```

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. For example, if you specified an invalid database password, you might see something like the following event for the `AWS::RDS::DBInstance` resource:

```
2013-04-24 19:21 UTC-7 CREATE_FAILED AWS::RDS::DBInstance DBInstance The para-
meter MasterUserPassword is not a valid password because it is shorter than 8
characters.
```

Step 6: 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 8: 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. 14).

Learn Template Basics

Topics

In Get Started (p. 8), you learned how to use a template to create a stack. You took a brief walk through the resources declared in a template and saw 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?

Before we go any further, we should cover the basics of what a template is. 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) 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 JSON format, see http://www.json.org.

In the template, you use a JSON structure AWS CloudFormation can interpret to declare the AWS resources you want to create and configure. In the JSON format, an object is declared as a name-value pair or a pairing of a name with a set of child objects enclosed within braces. Multiple sibling objects are separated by commas. An AWS CloudFormation template begins with an open brace and ends with a close brace. Within those braces, you can declare top-level JSON objects, as described in the Template Anatomy (p. 115). 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 contained within braces. 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. 446). For a full list of resource types, see Template Reference (p. 243).

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.

```
{
    "Resources" : {
        "HelloBucket" : {
            "Type" : "AWS::S3::Bucket"
        }
    }
}
```

The syntactic elements are quoted strings. 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. 302) 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. 463).

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.

```
{
    "Resources" : {
        "HelloBucket" : {
            "Type" : "AWS::S3::Bucket",
            "Properties" : {
               "AccessControl" : "PublicRead"
            }
        }
    }
}
```

Some resources can have multiple properties, and some properties can have one or more subproperties. For example, the AWS::S3::Bucket (p. 446) 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.

```
{
    "Resources" : {
        "HelloBucket" : {
            "Type" : "AWS::S3::Bucket",
            "Properties" : {
               "AccessControl" : "PublicRead",
               "WebsiteConfiguration" : {
                    "IndexDocument" : "index.html",
                    "ErrorDocument" : "error.html"
               }
            }
        }
    }
}
```

Note how the sibling properties—AccessControl and WebsiteConfiguration, and IndexDocument and ErrorDocument—are separated with commas. One of the most common syntax errors in a template is a missing comma between sibling property declarations and between resources.

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. 565) 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. 299) 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. 565). The following template contains an AWS::EC2::Instance (p. 302) resource. The resource's SecurityGroups property calls the Ref function to refer to the AWS::EC2::SecurityGroup resource InstanceSecurityGroup.

```
{
  "Resources" : {
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "KeyName" : "mykey",
        "ImageId" : ""
      }
    },

    "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"
        } ]
      }
    }
  }
}
```

You probably noticed that the Ref function call is expressed like other JSON objects, as a name-value pair separated by a colon and surrounded by braces. The function name is the name, and the input parameter for the function is the value. You'll also notice that the function call is also surrounded by brackets. In JSON, lists are surrounded by brackets. 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.

```
{
  "Resources" : {
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" }, "MyExisting
SecurityGroup" ],
        "KeyName" : "mykey",
        "ImageId" : "ami-7a11e213"
      }
    },

    "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"
        } ]
      }
    }
  }
}
```

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. 302) 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 her account and in the region where the stack is being created.

```
{
  "Parameters" : {
    "KeyName" : {
      "Description" : "The EC2 Key Pair to allow SSH access to the instance",
      "Type" : "AWS::EC2::KeyPair::KeyName"
    }
  },
  "Resources" : {
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" }, "MyExisting
SecurityGroup" ],
        "KeyName" : { "Ref" : "KeyName"},
        "ImageId" : "ami-7a11e213"
      }
    },

    "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"
        } ]
      }
    }
  }
}
```

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. 558) 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.

```
    "Resources" : {
        "myBucket" : {
            "Type" : "AWS::S3::Bucket"
        },
        "myDistribution" : {
           "Type" : "AWS::CloudFront::Distribution",
           "Properties" : {
              "DistributionConfig" : {
                 "Origins" : [ {
                    "DomainName": {"Fn::GetAtt" : ["myBucket", "DomainName"]},

                    "Id" : "myS3Origin",
                    "S3OriginConfig" : { }
                  } ],
                  "Enabled" : "true",
                  "DefaultCacheBehavior" : {
                    "TargetOriginId" : "myS3Origin",
                    "ForwardedValues" : {
                        "QueryString" : "false"
                    },
                    "ViewerProtocolPolicy" : "allow-all"
                  }
               }
            }
        }
    }
```

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. 558). 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.

```
  "Parameters": {
    "KeyName": {
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access
into the WordPress web server",
      "Type": "AWS::EC2::KeyPair::KeyName"
    },
    "WordPressUser": {
      "Default": "admin",
      "NoEcho": "true",
      "Description" : "The WordPress database admin account user name",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*"
    },
    "WebServerPort": {
      "Default": "8888",
      "Description" : "TCP/IP port for the WordPress web server",
      "Type": "Number",
      "MinValue": "1",
      "MaxValue": "65535"
    }
  },
```

The KeyName parameter is of type `AWS::EC2::KeyPair::KeyName` (an AWS-specific parameter type) and has a description. You'll notice that KeyName has no Default attribute and the other parameters do. Because KeyName has no default value, it must be specified at stack creation time: AWS CloudFormation will not create the stack without a value for KeyName. When a user uses the template in the Create Stack wizard, the console will show a drop-down list of valid values for AWS-specific parameter types.

For parameters with default values, AWS CloudFormation will use the default values unless users specify another value. If you omit the default attribute, users will be 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.

For AWS-specific parameter types, AWS CloudFormation validates input values against existing values in a user's AWS account and in the region where he is creating the stack. For example, another AWS-specific type is `AWS::EC2::VPC::Id`, which requires users to specify VPC IDs that are already created in their accounts and in the region that they are creating their stacks.

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.

```
{
  "Parameters" : {
    "KeyName" : {
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to
 the instance",
      "Type" : "String"
    }
  },

  "Mappings" : {
    "RegionMap" : {
      "us-east-1" : {
          "AMI" : "ami-76f0061f"
      },
      "us-west-1" : {
          "AMI" : "ami-655a0a20"
      },
      "eu-west-1" : {
          "AMI" : "ami-7fd4e10b"
      },
      "ap-southeast-1" : {
          "AMI" : "ami-72621c20"
      },
      "ap-northeast-1" : {
          "AMI" : "ami-8e08a38f"
      }
    }
  },

  "Resources" : {
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region"
 }, "AMI" ]},
        "UserData" : { "Fn::Base64" : "80" }
      }
    }
  }
}
```

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. 557) 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. 302), AWS::AutoScaling::AutoScalingGroup (p. 245), and AWS::ElasticLoadBalancing::LoadBalancer (p. 376), have a property that specifies availability zones. You can use the Fn::GetAZs function (p. 562) 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.

```
  "Resources" : {
    "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"
        }
      }
    },
```

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.

```
"Outputs": {
  "InstallURL": {
    "Value": {
      "Fn::Join": [
        "",
        [
          "http://",
          {
            "Fn::GetAtt": [
              "ElasticLoadBalancer",
              "DNSName"
            ]
          },
          "/wp-admin/install.php"
        ]
      ]
    },
    "Description" : "Installation URL of the WordPress website"
  },
  "WebsiteURL": {
    "Value": {
      "Fn::Join": [
        "",
        [
          "http://",
          {
            "Fn::GetAtt": [
              "ElasticLoadBalancer",
              "DNSName" ]
          }
        ]
      ]
    }
  }
}
```

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 string and this description appears in the Specify Parameters page of the Create Stack wizard. For more information, see Format Version (p. 116) and Description (p. 116).

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. 540) enables you to specify that one resource must be created after another.
- DeletionPolicy attribute (p. 539) enables you to specify how AWS CloudFormation should handle the deletion of a resource.
- Metadata (p. 543) attribute enables you to specify structured data with a resource.

AWS::CloudFormation::Stack (p. 278) 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. 31)—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. 32)—update one of the files in the application and deploy the software using AWS CloudFormation.
3. Update the Instance Type (p. 34)—change the instance type of the underlying Amazon EC2 instance.
4. Update the AMI on an Amazon EC2 instance (p. 36)—change the Amazon Machine Image (AMI) for the Amazon EC2 instance in your stack.

5. Add a Key Pair to an Instance (p. 37)—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. 38)—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 update -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.WebServerHost.Metadata.AWS::CloudForma
tion::Init\n",
                "action=/opt/aws/bin/cfn-init -s ", { "Ref" : "AWS::StackId"
}, " -r WebServerHost ",
                " --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.WebServerHost.Metadata.AWS::CloudForma
tion::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" },

                "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 update -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" : [ "WebServer
Instance", "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. 34).

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:

```
"WebServerHost": {
      "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 WebServerHost was updated, which caused AWS CloudFormation to also reevaluate the Elastic IP address and the WaitCondition resource to ensure that there were no changes that affected the update. 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:

```
    "WebServerHost": {
      "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 also use UpdateStack, along with 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 using the update attribute on the Auto Scaling group. For more information, see UpdatePolicy (p. 543).

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 existing template**, and then click **Next**.

   The Specify Parameters 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. 32) 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. 543).

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 ac
cess",
      "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.

```
          "WebServerHost": {
          "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"
 : …
        ]
      }
    },
```

4. Update the stack, either from the AWS Management Console as explained in Update the Application (p. 32) 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. 37) 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" : "PT10M"
        }
      },
      "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" : "PT5M"
        }
      },
      "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.WebServerHost.Metadata.AWS::CloudForma
tion::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" },

                "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 update -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"
        ]]}}
      },
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT10M"
        }
      }
    },

    "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" : [ "ElasticLoad
Balancer", "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 ElastiCache 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 Custom Resource Walkthrough

What is a Custom Resource?

Custom resources are special AWS CloudFormation resources that provide a way for a template developer to include non-AWS resources in an AWS CloudFormation stack. The custom resource provider can be either a template developer or a separate third-party resource provider.

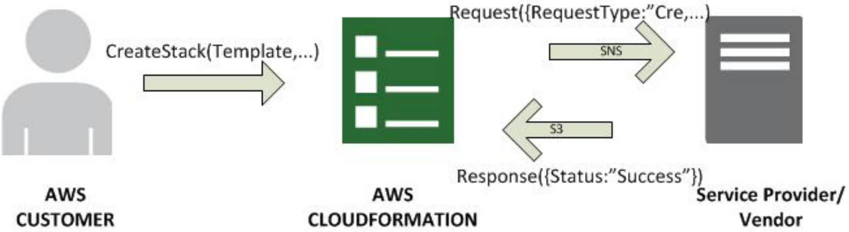
In an AWS CloudFormation template, custom resources are specified by the `AWS::CloudFormation::CustomResource` (p. 265) type or `Custom::`*`String`* (p. 266).

How Custom Resources Work

Any action taken for a custom resource involves three parties: the template developer, AWS CloudFormation, and the custom resource provider. The template developer and custom resource provider may be the same person or entity, but the process will be the same. The following steps describe the general process:

1. The template developer creates, updates, or deletes a stack that contains a custom resource. The template includes a service token and any input/output data parameters for the custom resource.
2. AWS CloudFormation communicates with the custom resource provider using an SNS topic, sending it the type of request (create, update, or delete) and any input data stored in the stack template. AWS CloudFormation provides the custom resource provider with an S3 URL for the response.
3. The custom resource provider processes the message and returns a response of `SUCCESS` or `FAILED`. The custom resource provider can also send the names and values of resource attributes that can be accessed by the template developer if the request succeeded (output data), or send a string that provides detail about the failure if the request failed.
4. AWS CloudFormation sets the stack status according to the response received and provides the values of any custom resource output data to the template developer with `Fn::GetAtt` (p. 558).

The following figure illustrates the relationships between the template developer, AWS CloudFormation, and the custom resource provider:


CreateStack(Template,...)
Request({RequestType:"Cre,...)
SNS
S3
Response({Status:"Success"})
AWS CUSTOMER
AWS CLOUDFORMATION
Service Provider/ Vendor

What's in this Walkthrough?

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.

It is divided into three parts:

- Part 1: Stack Creation (p. 48)
- Part 2: Stack Updates (p. 50)

- Part 3: Stack Deletion (p. 52)

Part 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 name is declared with a *service token*, optional *provider-specific properties*, and optional `Fn::GetAtt` (p. 558) 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-east-1:84969EXAMPLE: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 `"Request-Type" : "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-east-1:EXAMPLE/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. 588) 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-east-1:EXAMPLE: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. 558).

   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. 588) in the Custom Resource Reference (p. 585).

Part 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. 88).

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 Part 3: Stack Deletion (p. 52).

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-east-1:84969EXAMPLE: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-east-1:EXAMPLE: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-east-1:EXAMPLE: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. 593) in the Custom Resource Reference (p. 585).

Part 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-east-1:EXAMPLE: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`.

   The following is an example of a custom resource provider response to a `Delete` request:

```
{
   "Status" : "SUCCESS",
   "StackId" : "arn:aws:cloudformation:us-east-1:EXAMPLE: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. 590) in the Custom Resource Reference (p. 585).

See Also

- AWS CloudFormation Custom Resource Reference (p. 585)
- AWS::CloudFormation::CustomResource (p. 265)
- Fn::GetAtt (p. 558)
- Amazon Simple Notification Service Getting Started Guide
- Amazon Simple Storage Service Developer Guide

Using CloudFormer to Create AWS CloudFormation Templates from Existing AWS Resources

CloudFormer is a tool that creates an AWS CloudFormation template from existing AWS resources in your account. The basic procedure is:

1. Provision and configure the required resources using your existing processes and tools.
2. Create and launch a CloudFormer stack.

   CloudFormer is itself an AWS CloudFormation stack. You run CloudFormer by launching the stack from your AWS environment. It runs on a t1.micro 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.

Some general notes on how CloudFormer works:

- CloudFormer supports all AWS CloudFormation resources.
- CloudFormer automatically selects dependent resources.

  For example, if you include an Amazon EC2 instance that has a dependent EC2 security group, CloudFormer automatically selects the security group resource.

- You have full control over which resources are included in the template.

  You can override automatically selected resources and add others, as appropriate.
- You can specify the resource names to be used in the template.

  The default names are based on the existing resource names.
- You can add output parameters based on a resource's available attributes.

This guide describes how to use CloudFormer by walking you through a basic scenario—a simple website on an 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. There are several ways to perform this task.

- The AWS CloudFormation console.
- The URLs on the CloudFormer tool page.
- The URLs on the AWS CloudFormation Templates page

Because the AWS CloudFormation console is a good way to learn how to work with AWS resources, this walkthrough launches a CloudFormer stack by using the 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. On the wizard's **Create Stack** page:

   1. In the **Name** box, specify a name for this CloudFormer stack.
   2. In the **Template** section, select **Use a sample template** and select **CloudFormer - create a template from your existing resources** from the list.

   Click **Next Step** to move to the next page.
3. On the **Specify Parameters** screen:

   - Under **Access Control**, specify the IP address range that can be used to access the tool.

     The default IP address range is 0.0.0.0/0, which leaves the tool fully open. We recommend that you specify a more restrictive address range.
4. Click **Next Step**
5. Select **I acknowledge that this template may create IAM resources**, and then click **Next Step**.

   This example doesn't use tags.

6. On the **Review** screen, examine the information about the stack that will be created, then click **Create** to begin creating the CloudFormer stack.

   Note: CloudFormer is an AWS CloudFormation stack itself, so must go through the normal stack creation process, which takes a few minutes.

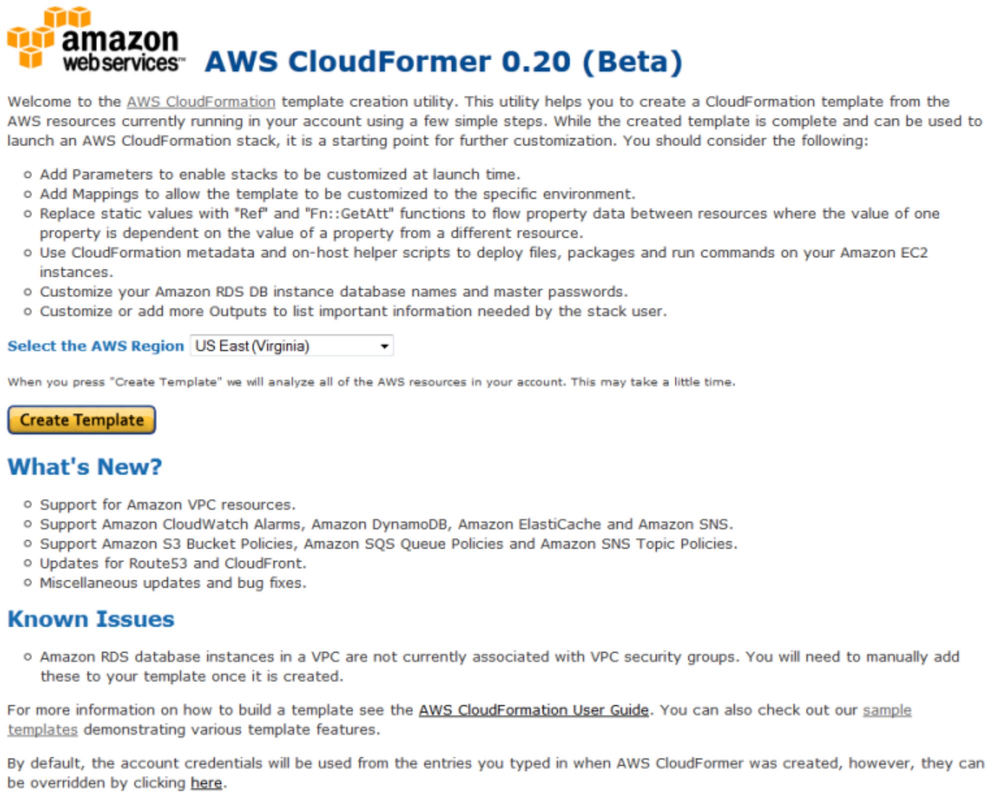
Step 2: Launch the CloudFormer Stack

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 **Outputs Value** column, click the URL to launch the CloudFormer tool.

After the stack launches, it displays the first page of the CloudFormer tool in your browser, which you use to create your template, as described in the next section.



AWS CloudFormer 0.20 (Beta)

Welcome to the AWS CloudFormation template creation utility. This utility helps you to create a CloudFormation template from the AWS resources currently running in your account using a few simple steps. While the created template is complete and can be used to launch an AWS CloudFormation stack, it is a starting point for further customization. You should consider the following:

- Add Parameters to enable stacks to be customized at launch time.
- Add Mappings to allow the template to be customized to the specific environment.
- Replace static values with "Ref" and "Fn::GetAtt" functions to flow property data between resources where the value of one property is dependent on the value of a property from a different resource.
- Use CloudFormation metadata and on-host helper scripts to deploy files, packages and run commands on your Amazon EC2 instances.
- Customize your Amazon RDS DB instance database names and master passwords.
- Customize or add more Outputs to list important information needed by the stack user.

Select the AWS Region: US East (Virginia)

When you press "Create Template" we will analyze all of the AWS resources in your account. This may take a little time.

What's New?

- Support for Amazon VPC resources.
- Support Amazon CloudWatch Alarms, Amazon DynamoDB, Amazon ElastiCache and Amazon SNS.
- Support Amazon S3 Bucket Policies, Amazon SQS Queue Policies and Amazon SNS Topic Policies.
- Updates for Route53 and CloudFront.
- Miscellaneous updates and bug fixes.

Known Issues

- Amazon RDS database instances in a VPC are not currently associated with VPC security groups. You will need to manually add these to your template once it is created.

For more information on how to build a template see the AWS CloudFormation User Guide. You can also check out our sample templates demonstrating various template features.

By default, the account credentials will be used from the entries you typed in when AWS CloudFormer was created, however, they can be overridden by clicking here.

> **Note**
> The CloudFormer stack launches a t1.micro Amazon EC2 instance, which must be manually terminated after you are 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.

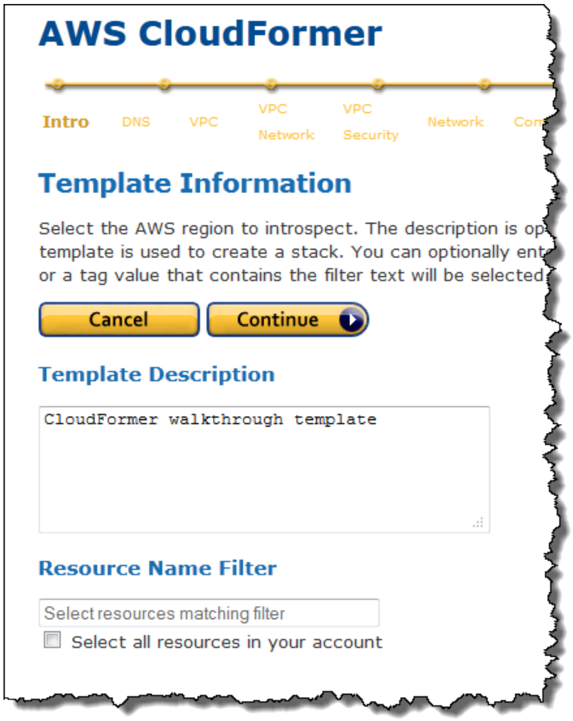
Step 3: Use CloudFormer to Create a Template

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**.



Template Information

Select the AWS region to introspect. The description is op
template is used to create a stack. You can optionally ent
or a tag value that contains the filter text will be selected

Template Description

CloudFormer walkthrough template

Resource Name Filter

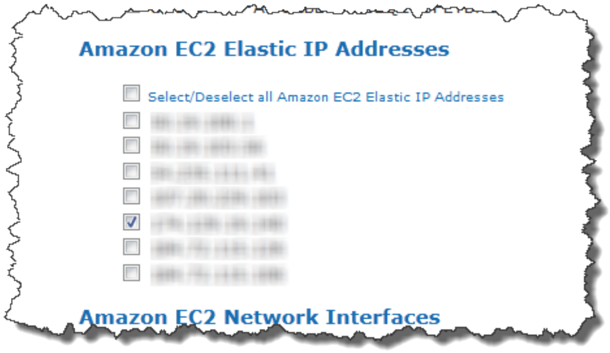
Select resources matching filter

□ Select all resources in your account

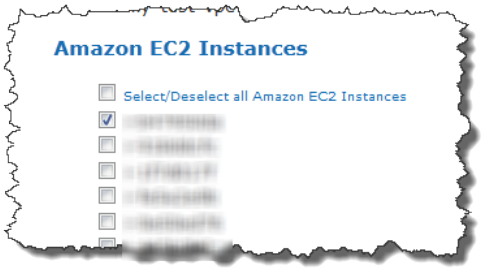
3. The following pages are for resources that are not used by this walkthrough, so just examine the page for future reference and click **Continue**. In order:

   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.


Amazon EC2 Elastic IP Addresses
□ Select/Deselect all Amazon EC2 Elastic IP Addresses
Amazon EC2 Network Interfaces

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

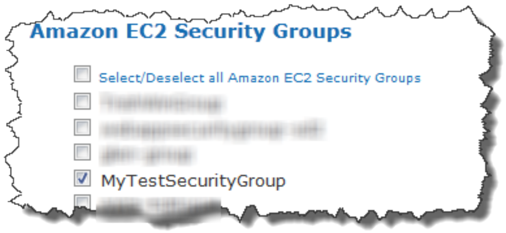
   You can manually include additional instances, as needed. If you don't want to include an automatically selected instance, just clear the check box.

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**.



Amazon EC2 Security Groups

- □ Select/Deselect all Amazon EC2 Security Groups
- □ [redacted]
- □ [redacted]
- □ [redacted]
- ☑ MyTestSecurityGroup
- □ [redacted]

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.

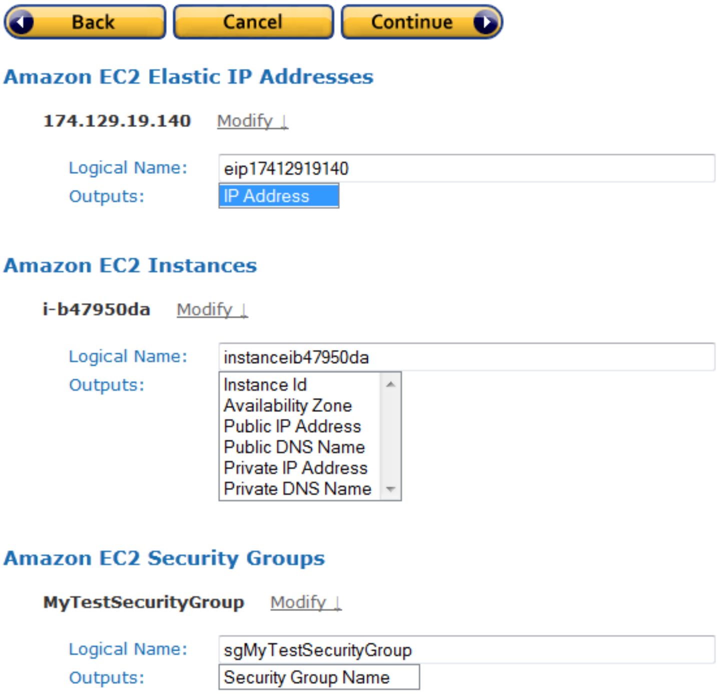
8. The **Operational Resources** page allows you to include Auto Scaling policies and CloudWatch alarms. This walkthrough uses neither, so just click **Continue**.
9. The **Summary** page serves several purposes:

   - 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.



Amazon EC2 Elastic IP Addresses

174.129.19.140 Modify ↓

Logical Name: eip17412919140
Outputs: IP Address

Amazon EC2 Instances

i-b47950da Modify ↓

Logical Name: instanceib47950da
Outputs: Instance Id
Availability Zone
Public IP Address
Public DNS Name
Private IP Address
Private DNS Name

Amazon EC2 Security Groups

MyTestSecurityGroup Modify ↓

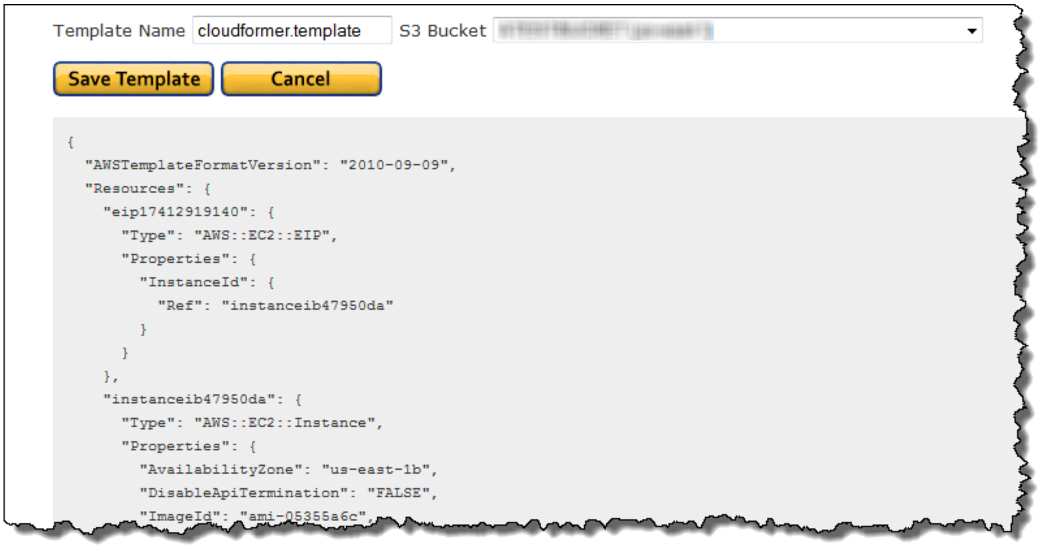
Logical Name: sgMyTestSecurityGroup
Outputs: Security Group Name

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.



Template Name cloudformer.template S3 Bucket [redacted]

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "eip17412919140": {
      "Type": "AWS::EC2::EIP",
      "Properties": {
        "InstanceId": {
          "Ref": "instanceib47950da"
        }
      }
    },
    "instanceib47950da": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "AvailabilityZone": "us-east-1b",
        "DisableApiTermination": "FALSE",
        "ImageId": "ami-05355a6c",
```

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.

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.

The standard AWS CloudFormation endpoints are:

Region Name	Endpoint
Asia Pacific (Singapore) Region	cloudformation.ap-southeast-1.amazonaws.com
Asia Pacific (Sydney) Region	cloudformation.ap-southeast-2.amazonaws.com
Asia Pacific (Tokyo) Region	cloudformation.ap-northeast-1.amazonaws.com
China (Beijing) Region	cloudformation.cn-north-1.amazonaws.com
EU (Frankfurt) Region	cloudformation.eu-central-1.amazonaws.com
EU (Ireland) Region	cloudformation.eu-west-1.amazonaws.com
South America (Sao Paulo) Region	cloudformation.sa-east-1.amazonaws.com
US East (Northern Virginia) Region	cloudformation.us-east-1.amazonaws.com
US West (Northern California) Region	cloudformation.us-west-1.amazonaws.com
US West (Oregon) Region	cloudformation.us-west-2.amazonaws.com

Note
All AWS CloudFormation endpoints use the HTTPS protocol for access.

For more information about regions and endpoints for AWS CloudFormation and other services, go to Regions and Endpoints in the *Amazon Web Services General Reference*.

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

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 setting remain the same. For more information about parameters, mappings, and conditions, see Template Anatomy (p. 115).

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 20 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*.

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. 116).

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 the your AWS account and in the region where the 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. 116).

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. 116).

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. 268) 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. 224).

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.

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. 24).

Use Stack Policies

Stack policies help protect critical stack resources from unintentional updates, which 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. 96).

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. 596).

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.

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 AWS CloudFormation actions users can perform, such as 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 use AWS CloudFormation to launch Amazon EC2 instances, terminate database instances, or update VPCs.

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

AWS CloudFormation Actions and Resources

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

A sample policy that grants view stack permissions

```
{
    "Version":"2012-10-17",
    "Statement":[{
        "Effect":"Allow",
        "Action":[
            "cloudformation:DescribeStacks",
            "cloudformation:DescribeStackEvents",
            "cloudformation:DescribeStackResources"
        ],
        "Resource":"*"
    }]
}
```

The policy grants permissions to all the listed actions in the `Action` element. In the `Resource` element, we specify an asterisk (*), a wild card that allows the actions to be performed on all AWS CloudFormation stacks.

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

A sample policy that grants create and view stack actions and all Amazon SQS actions

```
{
    "Version":"2012-10-17",
    "Statement":[{
        "Effect":"Allow",
        "Action":[
            "sqs:*",
            "cloudformation:CreateStack",
            "cloudformation:DescribeStacks",
            "cloudformation:DescribeStackEvents",
            "cloudformation:DescribeStackResources",
            "cloudformation:GetTemplate",
            "cloudformation:ValidateTemplate"
        ],
        "Resource":"*"
    }]
}
```

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

A sample policy that denies the delete and update stack actions for the MyProductionStack

```
{
    "Version":"2012-10-17",
    "Statement":[{
        "Effect":"Deny",
        "Action":[
            "cloudformation:DeleteStack",
            "cloudformation:UpdateStack"
        ],
        "Resource":"arn:aws:cloudformation:us-east-1:123456789012:stack/MyPro
ductionStack/*"
    }]
}
```

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`).

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

AWS CloudFormation Conditions

In an IAM policy, you can optionally specify conditions that control when a policy is in effect. AWS CloudFormation does not have service-specific conditions. However, you can use the AWS-wide conditions, such as `DateLessThan`, which specifies when a policy stops taking effect. For more information about AWS-wide conditions, see Condition in IAM Policy Elements Reference in *Using IAM*.

> **Note**
> Do not use the `aws:SourceIp` 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 Amazon EC2 instance or to create an Amazon S3 bucket, not the IP address from the `CreateStack` call or the `aws cloudformation create-stack` command.

IAM Resources in AWS CloudFormation Templates

Before you can create a stack, AWS CloudFormation validates your template. During the validation, AWS CloudFormation also checks your template for AWS resources that you should be aware of. Currently, AWS CloudFormation checks only for IAM resources in your templates. We recommend that you review the permissions associated with each IAM resource. IAM resources, such as an IAM user with full access, can access and modify any resource in your AWS account. To ensure that you've reviewed all IAM resources, you must acknowledge that the template is creating those resources before AWS CloudFormation creates the stack.

You can acknowledge the capabilities of AWS CloudFormation templates by using the AWS CloudFormation console, command line, or API:

- In the AWS CloudFormation console, select **I acknowledge that this template may create IAM resources** on the **Specify Parameters** page of the Create Stack or Update Stack wizards.

- For the AWS Command Line Interface, specify the *`CAPABILITY_IAM`* value for the `--`*`capabilities`* parameter when you use the `aws cloudformation create-stack` and `aws cloudformation update-stack` commands.
- For the API, specify the *`Capabilities.member.1=CAPABILITY_IAM`* parameter when you use the CreateStack and UpdateStack actions.

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 used when an Amazon EC2 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. 183).

> **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 Comparing Features of AWS STS APIs 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

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

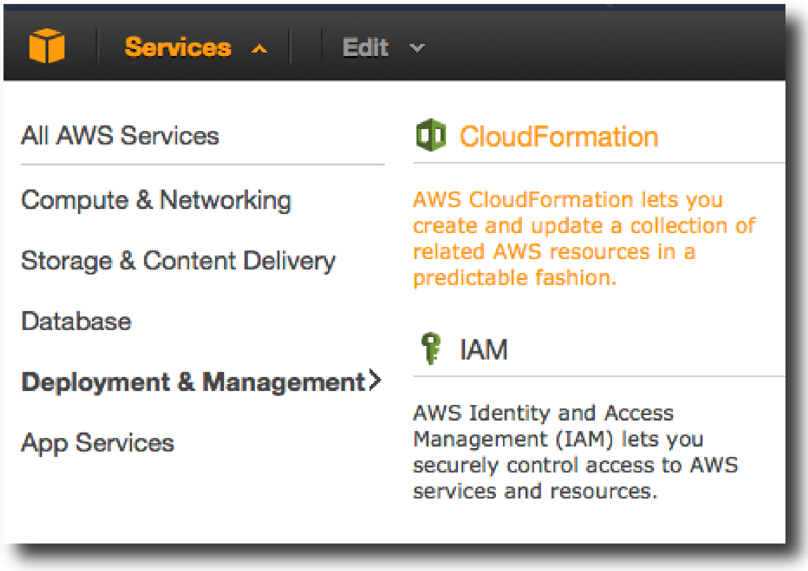
- Estimating the Cost of Your AWS CloudFormation Stack (p. 76)
- Viewing Stack Data and Resources (p. 77)
- Deleting a Stack (p. 78)
- Viewing Deleted Stacks (p. 79)

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:

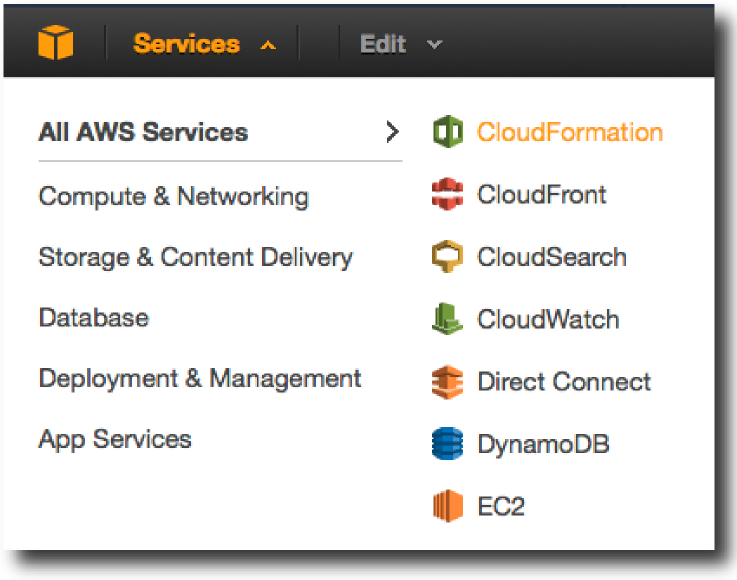
- Open the AWS CloudFormation console directly with the URL `https://console.aws.amazon.com/cloudformation/` . If you are not logged in to the AWS Management Console yet, you need to log in before using the AWS CloudFormation console.
- If you are logged into and using the AWS Management Console, you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



AWS CloudFormation lets you create and update a collection of related AWS resources in a predictable fashion.

AWS Identity and Access Management (IAM) lets you securely control access to AWS services and resources.

  - **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

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. 72)
2. Selecting a stack template (p. 73)
3. Specifying stack parameters (p. 74)
4. Setting Stack Options (p. 75)
5. Reviewing your stack (p. 75)

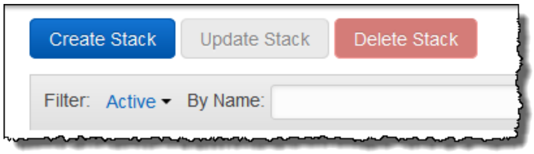
After creating a stack, you can monitor the stack's progress, view the stack's resources and outputs, update the stack, and delete it. Information about these actions are provided in their associated topics.

Starting the Create Stack Wizard

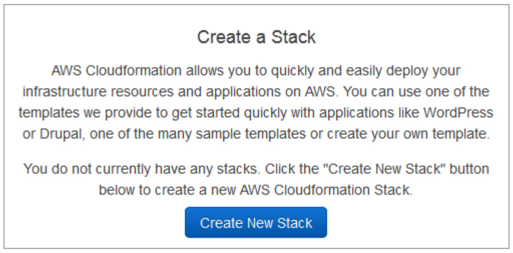
To create a stack on the AWS CloudFormation console

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.

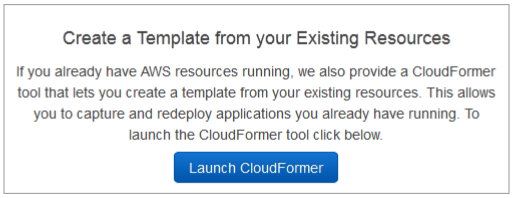


Create a Stack

AWS Cloudformation allows you to quickly and easily deploy your infrastructure resources and applications on AWS. You can use one of the templates we provide to get started quickly with applications like WordPress or Drupal, one of the many sample templates or create your own template.

You do not currently have any stacks. Click the "Create New Stack" button below to create a new AWS Cloudformation Stack.

- 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.



Create a Template from your Existing Resources

If you already have AWS resources running, we also provide a CloudFormer tool that lets you create a template from your existing resources. This allows you to capture and redeploy applications you already have running. To launch the CloudFormer tool click below.

  For more information about using CloudFormer to create AWS CloudFormation stacks, see Using CloudFormer to Create Templates (p. 53).

Next, you choose a stack template (p. 73).

Selecting a Stack Template on the AWS CloudFormation Console

After starting the Create Stack wizard (p. 72), you specify a stack name and select the template AWS CloudFormation uses to create your stack.

AWS CloudFormation templates are JSON files that specify the AWS resources that make up your stack. For more information about AWS CloudFormation templates, see Template Anatomy (p. 115).

To choose a stack name and select a stack template:

1. On the **Create A New Stack** page of the **Create Stack** wizard, type a stack name in the **Name** box.

   A stack name can contain only alphanumeric characters (case sensitive) and hyphens. It must start with an alphabetic character and cannot be longer than 255 characters.
2. Choose a stack using one of the following options:

Use a sample template

Select an AWS CloudFormation template from among those available on the menu. The list of available templates in the menu is generally the same as the list of templates on the AWS CloudFormation Sample Templates web page.

You can select **CloudFormer** from the list to create a stack from existing AWS resources, using the CloudFormer tool. For more information, see Using CloudFormer to Create Templates (p. 53).

Upload a template file

Select an AWS CloudFormation template on your local system. Specify the full path or click **Browse** to select the file that you want to upload.

An uploaded template can be, at most, 51200 bytes.

> **Note**
> If you upload 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.

Provide a template URL

Specify a URL to a template in an Amazon S3 bucket.

The URL must point to a template (max size: 460,800 bytes) in an Amazon S3 bucket that you have read permissions to, located in the same region as the stack. The URL itself can be, at most, 1024 characters long.

3. Click **Next Step** to accept your settings and proceed with specifying stack parameters (p. 74).

Specifying Stack Parameters on the AWS CloudFormation Console

After selecting a stack template (p. 73), you specify parameters (p. 116) defined in the template.

You can use parameters to customize your stack at creation time. Data you enter here can be referred to by logical ID in the stack template and be used to modify how AWS or custom resources are configured. For more information about how parameters are specified in an AWS CloudFormation template, see Parameters (p. 116).

To enter parameter values for your stack

1. On the **Specify Parameters** page of the **Create Stack** wizard, specify parameters defined in the stack template. Default values might already be present for some parameters.

   > **Note**
   > You might need to acknowledge some resources before AWS CloudFormation creates your stack, such as IAM resources. For more, information see IAM Resources in AWS CloudFormation Templates in Controlling Access with AWS Identity and Access Management (p. 65).

2. When you are satisfied with the parameter values, click **Next Step** to proceed with setting options for your stack (p. 75).

Setting AWS CloudFormation Stack Options

After specifying parameters (p. 116) 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 but must not contain any spaces. Tag keys up to 127 characters long. A **Value** consists of any alphanumeric characters or spaces. Tag values can be up to 255 characters long.

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. 96).

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. 75).

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. 77). 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 your 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 by using the following procedure.

To create an EC2 key pair

1. In the AWS Management Console, switch from the AWS CloudFormation console to the Amazon EC2 console by clicking the **Services** button in the top-left corner of the screen, and select **EC2**.

   The console display now shows the Amazon EC2 console dashboard.
2. In the Amazon EC2 console, in the **Navigation** pane, click **Key Pairs**.

   You see the **Key Pairs** page, displaying your Amazon EC2 key pairs. If you haven't created any yet, the list is empty, and instead shows the **Create Key Pair** button.
3. Click the **Create Key Pair** button.
4. Type a key pair name, and click **Create**. It doesn't matter what you name it, but make it something you can easily remember.

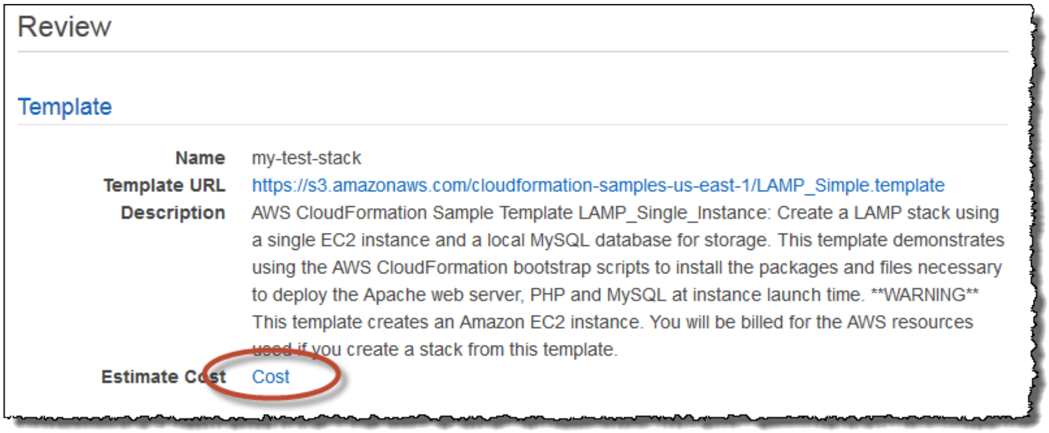
   The key pair is created, and the download of your private key begins. It will be called `name`.pem, where *name* represents the name you gave to your key pair.
5. Download the key pair, and set the permissions to 400 (on a Linux or Mac OS system).

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** or **Update Stack** dialog, click the **Cost** link.



Review

Template

Name	my-test-stack
Template URL	https://s3.amazonaws.com/cloudformation-samples-us-east-1/LAMP_Simple.template
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.
Estimate Cost	Cost

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.

Parameters
Displays the stack's parameters and their values.

Tags
Displays any tags that were associated with the stack.

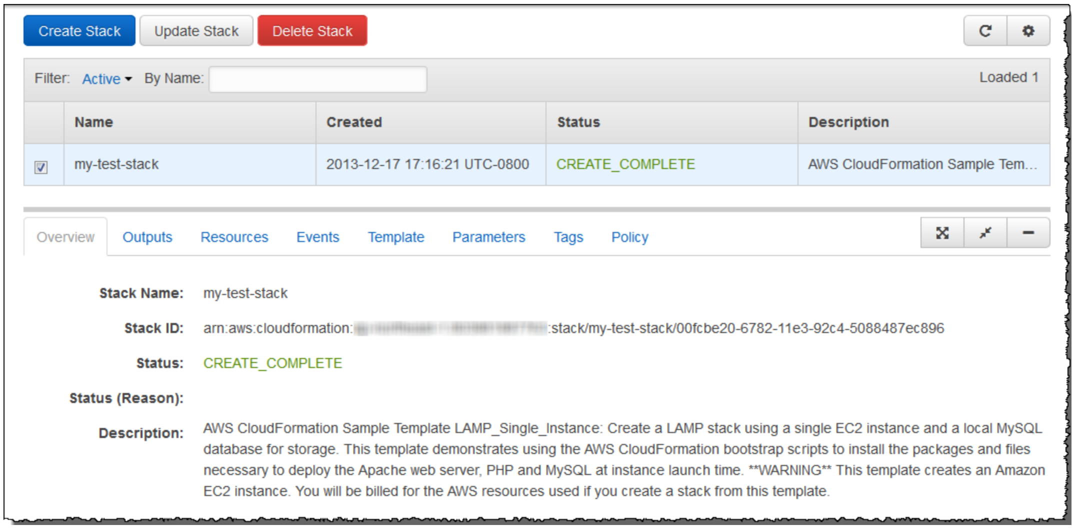
Stack Policy
Describes the stack resources that are protected against stack updates. To update these resources, they must be explicitly allowed during a stack update.

To view outputs for 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.



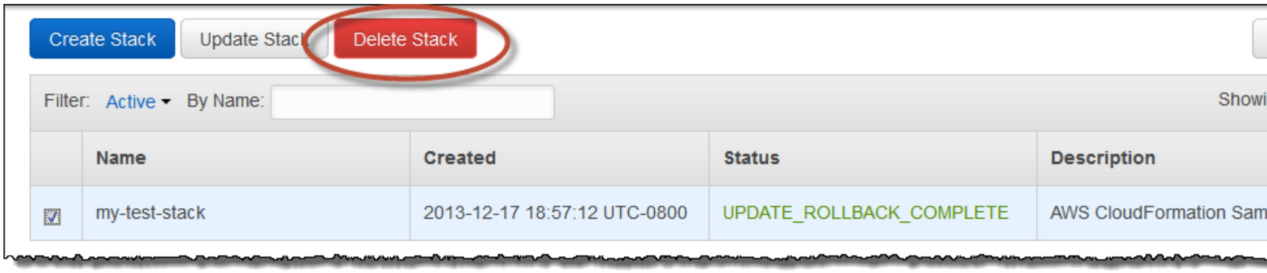
	Name	Created	Status	Description
☑	my-test-stack	2013-12-17 17:16:21 UTC-0800	CREATE_COMPLETE	AWS CloudFormation Sample Tem...

Stack Name:	my-test-stack
Stack ID:	arn:aws:cloudformation:[redacted]:stack/my-test-stack/00fcbe20-6782-11e3-92c4-5088487ec896
Status:	CREATE_COMPLETE
Status (Reason):	
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.

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. Click **Delete Stack**.



	Name	Created	Status	Description
☑	my-test-stack	2013-12-17 18:57:12 UTC-0800	UPDATE_ROLLBACK_COMPLETE	AWS CloudFormation Sam

3. Click **Yes, Delete** when prompted.

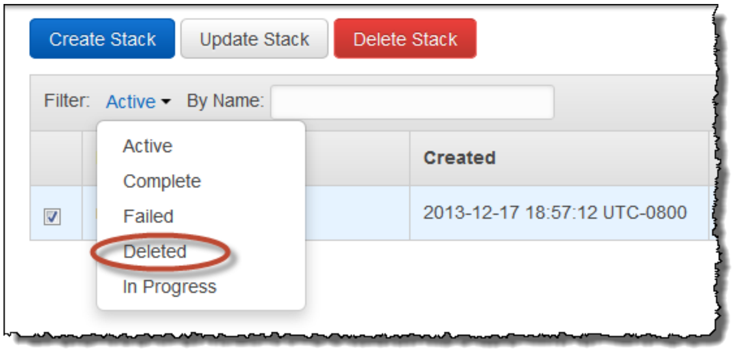
Note
After stack deletion has begun, you cannot abort it. The stack proceeds to the **DELETE_IN_PROGRESS** state.
After the stack deletion is complete, the stack will be in the **DELETE_COMPLETE** state. Stacks in the **DELETE_COMPLETE** state are not displayed in the AWS CloudFormation console by default.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

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. 78)
- Viewing Stack Data and Resources (p. 77)

Related Topics

- Using the AWS CLI (p. 79)

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

- Listing Resources (p. 85)
- Retrieving a Template (p. 86)
- Validating a Template (p. 87)
- Deleting a Stack (p. 88)

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.

> **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,Para
meterValue=test1 ParameterKey=Parm2,ParameterValue=test2
{
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mytest
stack/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`.

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. **WARN
ING** 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.
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.

Stack Status	Description
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`	Resource status; one of *IN_PROGRESS* | *CREATE_FAILED* | *CREATE_COMPLETE* | *DELETE_IN_PROGRESS* | *DELETE_FAILED* | *DELETE_COMPLETE*.
`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 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.

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.amazon
aws.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/loc
al/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)
```

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

You can update a stack that has been successfully created to update resources in the stack, such as an Amazon EC2 instance, or to update the stack's settings, such as the stack's Amazon SNS notification topic. For example, if your stack included an Amazon EC2 instance, you can update that instance by updating the stack. You don't need to create a new stack. You can use the AWS CloudFormation console, the `aws cloudformation update-stack` CLI command, or the `UpdateStack` API to update a stack.

Updates to stack resources

You modify stack resources by submitting an updated template or by submitting updated input parameters. 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. Resources that are updated could be interrupted or replaced, depending on the resources and properties that are being updated. AWS CloudFormation uses one of the following techniques to update resources:

Update with No Interruption

AWS CloudFormation updates the resource without disrupting operation of that resource and without changing the resource's physical name. For example, if you update any properties on an AWS::CloudWatch::Alarm (p. 287) resource, AWS CloudFormation updates the alarm's configuration and, during the update, the alarm's operation continues without disruption.

Updates with Some Interruption

AWS CloudFormation updates the resource with some interruption but the physical name is retained. For example, if you update certain properties on an AWS::EC2::Instance (p. 302) 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. 423) resource, AWS CloudFormation creates a new resource and replaces the current DBInstance resource with the new one.

To learn more about updating a particular resource, see the documentation that is associated with that resource. For example, the Amazon EC2 documentation provides details about what changes interrupt an instance. See also the AWS Resource Types Reference (p. 243), where the effects of updating a resource are listed for each property.

Depending on the technique AWS CloudFormation uses to modify each updated resource in your stack, you can make decisions about when it's best 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 resource, AWS CloudFormation creates a new DB instance with the updated port setting and a new physical name. To plan for this, you should do the following:

1. Take a snapshot of the current databases.
2. Prepare a strategy for how applications that use that DB instance will handle an interruption while the DB instance is being replaced.
3. Ensure that the applications that use that DB instance take into account the updated port setting and any other updates you have made.
4. 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. 278), 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.

Topics

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 then 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.

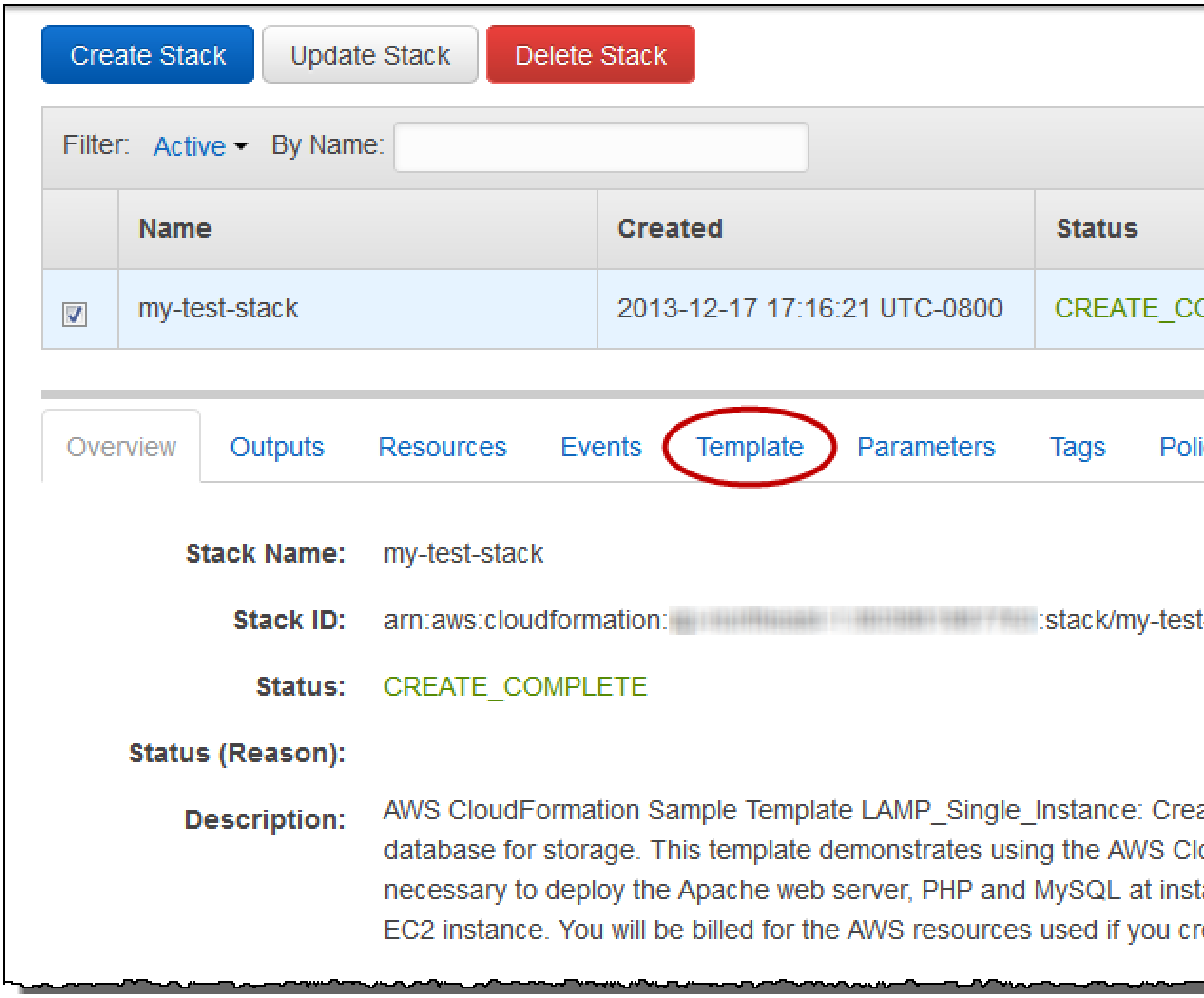
If you want to modify just the parameters or settings of a stack (like a stack's Amazon SNS topic), you can reuse the existing stack template. You don't need to get a copy of the stack template or make any modification to the stack template.

> **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

To get and modify a template for a stack from AWS CloudFormation by using the console

1. In the AWS CloudFormation console, select the stack that you want to update and then click the **Template** tab to view the stack template.



	Name	Created	Status
☑	my-test-stack	2013-12-17 17:16:21 UTC-0800	CREATE_CO

Stack Name: my-test-stack

Stack ID: arn:aws:cloudformation:[redacted]:stack/my-test

Status: CREATE_COMPLETE

Status (Reason):

Description: AWS CloudFormation Sample Template LAMP_Single_Instance: Crea database for storage. This template demonstrates using the AWS Cl necessary to deploy the Apache web server, PHP and MySQL at inst EC2 instance. You will be billed for the AWS resources used if you cr

2. From the **Template** tab, copy the template into a text file.
3. Modify the template file and then save it. 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. 243) 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. 88)
     - *Update requires*: Some interruptions (p. 88)
     - *Update requires*: Replacement (p. 88)
   - Add, modify, or delete attributes for resources (Metadata, DependsOn, and DeletionPolicy).

> **Important**
> You cannot update the DeletionPolicy attribute by itself. You can update the DeletionPolicy attribute only when you include changes that add, modify, or delete resources.

- 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.
- 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.

- Add, modify, or delete output value declarations.

> **Important**
> You cannot update outputs by themselves. You can update outputs only when you include changes that add, modify, or delete resources.

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. 423) 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 will fail. 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. 565) or the Fn::GetAtt function (p. 558) to specify an attribute from an updated resource as part of a property value in another resource in the template, AWS CloudFormation will also update 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.

4. If you want to specify the template as a URL when you update the stack, upload the update template to an Amazon S3 bucket. The bucket must be in the same region as the stack that you are updating.

To get and modify a template for a stack from AWS CloudFormation by using the command line

1. Use the command `aws cloudformation get-template` to get the template for the stack you want to update.
2. Copy the template, paste it into a text file, modify it, and save it. Make sure that you 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 a Stack

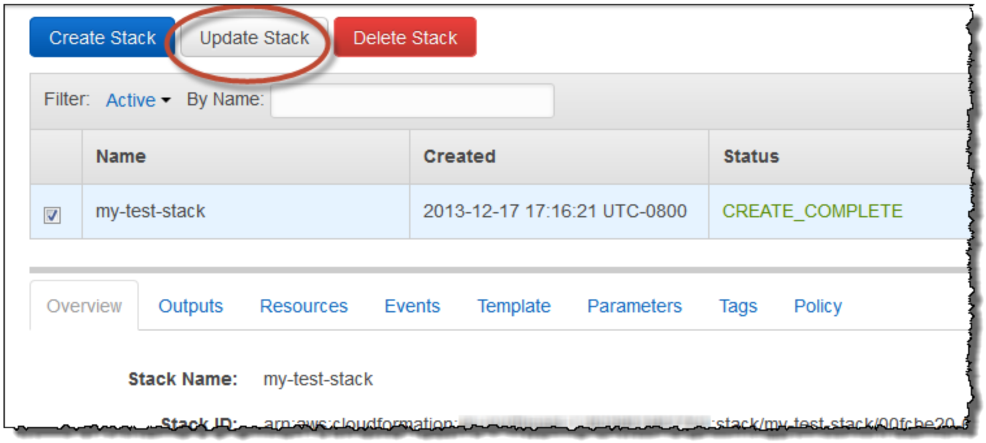
When you update a stack, you can modify resources in your stack, update stack settings, or both. For example, you can increase the capacity of an Amazon EC2 instance by changing the instance type, or you can update a stack's Amazon SNS notification topic.

When you update the stack, you can change the parameter values that are used for resources that support updates; however, you must keep the existing values in the current stack for parameters that affect resources that do not support updates.

Topics

To update an existing AWS CloudFormation stack by using the console

1. In the AWS CloudFormation console, from the list of stacks, select the running stack that you want to update.
2. Click **Update Stack**.



	Name	Created	Status
☑	my-test-stack	2013-12-17 17:16:21 UTC-0800	CREATE_COMPLETE

Stack Name: my-test-stack

Stack ID: arn:aws:cloudformation:[redacted]:stack/my-test-stack/00fcbe20

3. Depending on whether you modified the stack template, you can reuse the existing template or specify another one.

   - If you did not modify the stack template, select **Use existing template**.
   - If you modified the stack template, specify the location of the updated template:
     - For a template stored locally on your computer, select **Upload a template to Amazon S3**. Enter the location for the template file, or click **Browse** 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**.

4. 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 existing values by selecting **Use existing value**.
5. On the **Options** page, you can enter an overriding stack policy or update the Amazon SNS notification topic. The overriding stack policy enables you to update protected resources. For more information, see Prevent Updates to Stack Resources (p. 96).

   After you have completed modifying any options, click **Next**.
6. Review the information for the stack. If you have IAM resources in the template, 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. 65).
7. Click **Update**.

Your stack enters the **UPDATE_IN_PROGRESS** state. After it has finished updating, the stack state is set to **UPDATE_COMPLETE**.

If the stack update fails, AWS CloudFormation automatically roll back any changes, and the stack is set to **UPDATE_ROLLBACK_COMPLETE**.

> **Note**
> After your stack has begun updating, you can cancel the update while it's still in the **UPDATE_IN_PROGRESS** state. For more information, see Canceling a Stack Update (p. 95).

To update an existing AWS CloudFormation stack by using the command line

- Use the command `aws cloudformation update-stack` to update a stack by specifying the stack to update, updated template, parameter values, and capabilities.

  The following sample update stack command 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=Sub
  netIDs,ParameterValue=SampleSubnetID1\\,SampleSubnetID2
  ```

  The following sample update stack command 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=Subnet
  IDs,ParameterValue=SampleSubnetID1\\,UpdatedSampleSubnetID2
  ```

  The following sample update stack command 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 sample update stack command 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. 423) 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. 423) 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_PRO
GRESS 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

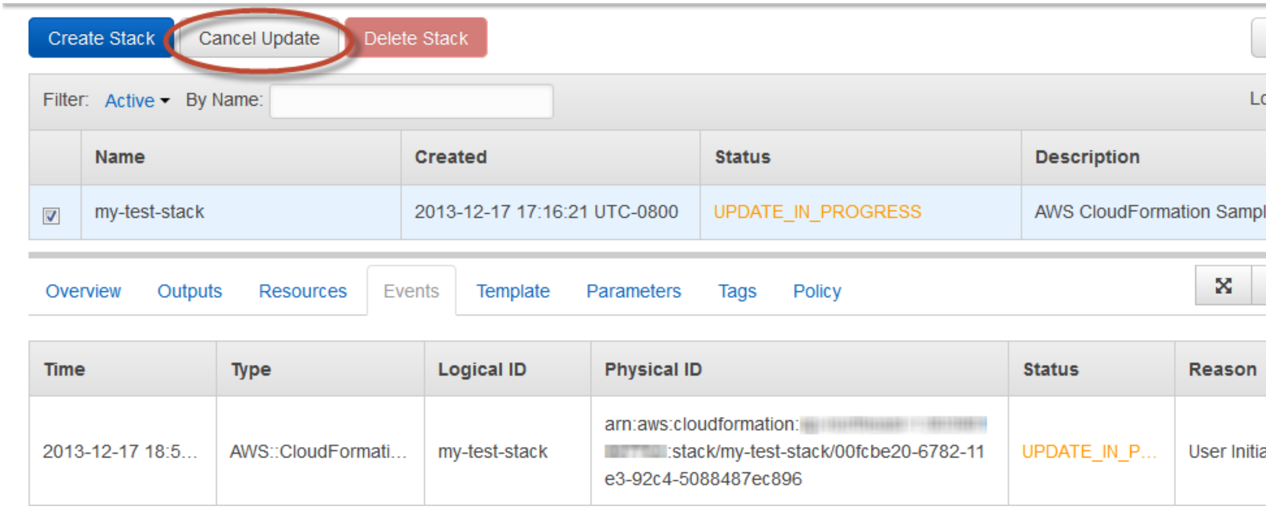
After a stack update has begun, you can cancel the stack update if the stack is still in the `UPDATE_IN_PROGRESS` state. After an update has finished, you cannot cancel it. You can, however, update a stack again with any previous settings.

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. Click **Cancel Update**.



	Name	Created	Status	Description
☑	my-test-stack	2013-12-17 17:16:21 UTC-0800	UPDATE_IN_PROGRESS	AWS CloudFormation Sampl

Time	Type	Logical ID	Physical ID	Status	Reason
2013-12-17 18:5…	AWS::CloudFormati…	my-test-stack	arn:aws:cloudformation:[redacted]:stack/my-test-stack/00fcbe20-6782-11e3-92c4-5088487ec896	UPDATE_IN_P…	User Initia

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

You can prevent stack resources (p. 243) from being unintentionally updated or deleted during a stack update by using stack policies. Stack policies apply only during stack updates and should be used only as a fail-safe mechanism to prevent accidental updates to certain stack resources. Do not use stack policies to control access to AWS resources or actions; instead, use AWS Identity and Access Management (IAM).

By default, all resources in a stack can be updated by anyone with update permissions. However, during an update, some resources might require an interruption or might be completely replaced, which could result in new physical IDs or completely new storage. To ensure that no one inadvertently updates these resources, you can set a stack policy. The stack policy prevents anyone from accidentally updating resources that are protected. If you want to update protected resources, you must explicitly specify those resources during a stack update.

> **Important**
> After you set a stack policy, all resources in the stack are protected by default. For any resources that you still want to allow updates on, you must specify an explicit `Allow` statement for those resources.

Stack policies are JSON documents that define which update actions can be performed on designated resources. You can define only one stack policy per stack; however, you can protect multiple resources within a single policy. The following sample stack policy prevents updates to the resource with the `ProductionDatabase` logical ID:

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "LogicalResourceId/ProductionDatabase"
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

In the sample, we want to prevent updates, so we specify `Deny` for the `Effect` element to prevent any actions defined in the `Action` element. To specify all update actions, we use a wild card (an asterisk). In the `Resource` element, we specify the resource that we want to protect, which has the logical ID of `ProductionDatabase`. The `Principal` element is required but supports only the wild card (`*`).

Note that when you set a stack policy, all resources are protected by default. Therefore, to protect only the `ProductionDatabase` resource, we add an `Allow` statement to cover all actions and resources so that all other resources can be updated. Even though the `Allow` specifies all resources, the explicit `Deny` overrides any allows.

Stack policies apply to all users who want to update the stack. You cannot apply stack policies that reference specific users. Users who must update protected resources need permission to use the `SetStackPolicy` action. For more information, see Updating Protected Resources (p. 99).

For more information about the stack policy syntax, see Stack Policy Reference (p. 100).

Setting a Stack Policy

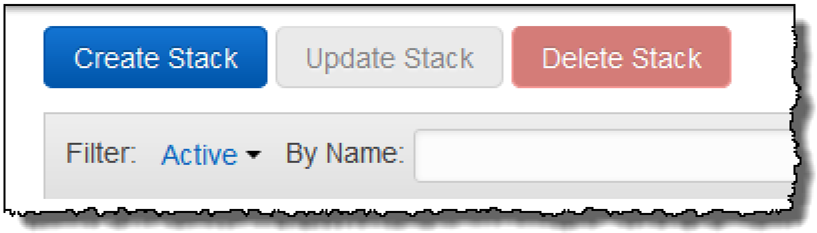
When you want to protect stack resources from unintentional updates, you define a stack policy in JSON format and then associate it with a stack when you create or update the stack. For more information about writing stack policies, see Stack Policy Reference (p. 100).

By default, when you create a stack, no stack policy is set on the stack, so you can update any resources. However, after you set a stack policy, all stack resources are protected by default. For any resources that you still want to allow updates on, you must specify an explicit `Allow` statement for those resources.

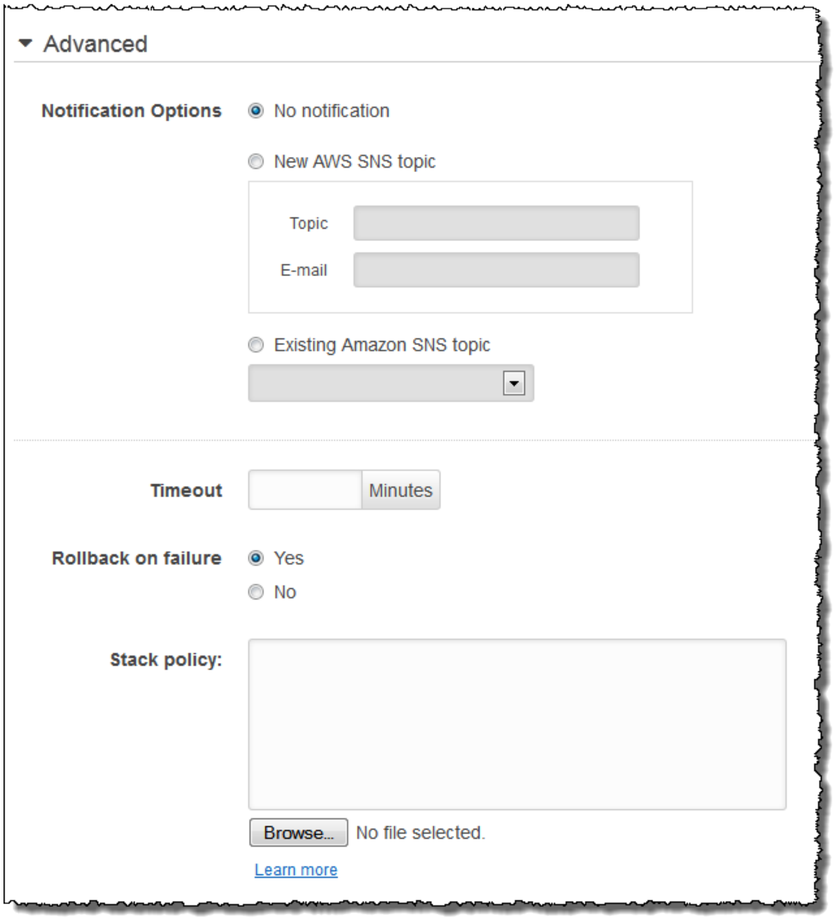
To set a stack policy when you create a stack:

AWS Management Console

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. On the **CloudFormation Stacks** page, click **Create Stack**.



3. On the **Options** screen of the **Create Stack** wizard, expand the **Advanced** section.



Advanced

Notification Options: ● No notification ○ New AWS SNS topic ○ Existing Amazon SNS topic

Topic:

E-mail:

Timeout: Minutes

Rollback on failure: ● Yes ○ No

Stack policy:

No file selected.

> **Note**
> When you create a stack and include a policy, you don't require permission to use the AWS CloudFormation `SetStackPolicy` action. However, if you want to update the policy or update protected resources, you must have permission to use the `SetStackPolicy` action.

4. Select a file that defines a stack policy or enter one.

CLI

- Use the `aws cloudformation create-stack` command with the `--stack-policy-body` or `--stack-policy-url` option.

To modify a stack policy (currently, you can't use the console to do this):

CLI

- Use the `aws cloudformation set-stack-policy` command with the `--stack-policy-body` or `--stack-policy-url` option.

To modify a stack policy during a stack update (currently, you can't use the console to do this):

CLI

- Use the aws cloudformation update-stack command with the `--stack-policy-body` or `--stack-policy-url` option.

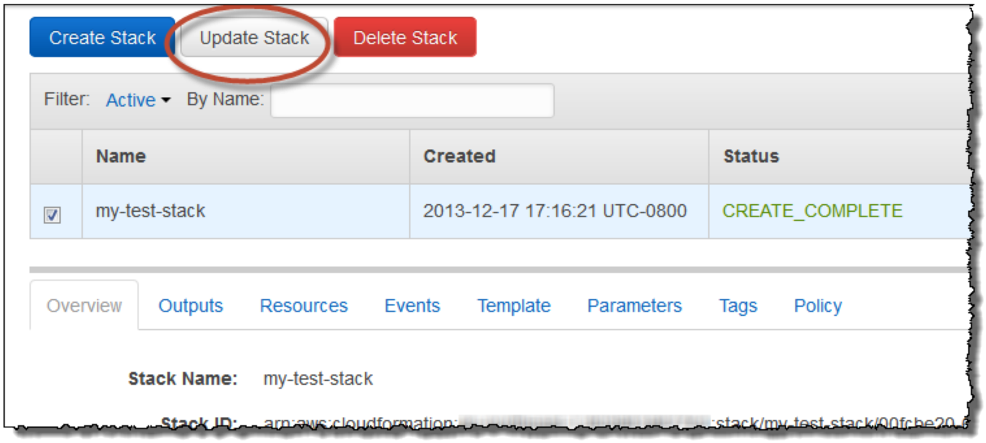
Updating Protected Resources

You can update protected resources by lifting their protections with a temporary policy. In the temporary policy you specify the resources that you want to update. You define the temporary policy when you update the stack. Before you begin, you must have permission to use the AWS CloudFormation `SetStackPolicy` action and define a temporary policy that allows you to update the resources that you want to change.

To update a protected resource:

AWS Management Console

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Select the stack that you want to update, and then click **Update Stack**.



	Name	Created	Status
☑	my-test-stack	2013-12-17 17:16:21 UTC-0800	CREATE_COMPLETE

Stack Name: my-test-stack

Stack ID: arn:aws:cloudformation:[redacted]:stack/my-test-stack/00fcbe20

3. On the **Policy** screen of the **Update Stack** wizard, select a file that defines an overriding stack policy or enter one.

   Note
   You must have permission to the AWS CloudFormation `SetStackPolicy` action in order to update protected resources.

   The override policy should specify an `Allow` for the protected resources that you want to update. The override policy is a temporary policy that is applied only during this update.

   For example, if you wanted to update all protected resources, you can specify a temporary override that allows all updates:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

AWS CLI

- Use the `aws cloudformation update-stack` command with the `--stack-policy-during-update-body` or `--stack-policy-during-update-url` option.

Removing a Stack Policy

After you set a stack policy, you cannot remove or delete the policy. If you want to remove all protections, you must update the policy to explicitly allow all actions on all resources. By default a stack policy denies all updates. The following sample policy allows all updates on all resources:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

To set an updated stack policy, see Setting a Stack Policy (p. 97).

Stack Policy Reference

Stack policies are JSON documents that define which update actions users can do and which resources they can take action on. These permissions are defined in the following elements: `Effect`, `Action`, `Resource`, and `Condition`. When you create a stack, no stack policy is set by default. In other words, all update actions on all resources are allowed. If you want to protect stack resources, you must set a stack policy. The following pseudo code shows the syntax for a stack policy:

```
{
  "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 that you specify. You can specify only `Deny` or `Allow` for this element, as shown in the following snippet:

```
"Effect" : "Deny"
```

Important

If a stack policy includes any overlapping statements, a `Deny` always overrides an `Allow`. If you want ensure that a resource is protected, use a `Deny` statement for that resource.

Action

Specifies the update actions that are denied or allowed. You can specify the following actions:

`Update:Modify`

Specifies update actions where resources might experience no interruptions or some interruptions while changes are being applied. All resources maintain their physical IDs.

`Update:Replace`

Specifies update actions where 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 resource might be different.

`Update:Delete`

Specifies update actions where resources are removed. Any 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 snippet shows how you can specify just the replace and delete actions:

```
"Action" : ["Update:Replace", "Update:Delete"]
```

You can also use a `Not` with actions. For example, if you wanted to allow all update actions, except for `Update:Delete`, you can use `NotAction`, as shown in the following sample:

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

For more information about stack updates, see AWS CloudFormation Stacks Updates (p. 88).

Principal

The `Principal` element is required but supports only the wild card (`*`).

Resource

Specifies the logical IDs of the resources that the policy applies to. If you want to specify types of resources (p. 243), use the `Condition` element.

You can specify a single resource by using its logical ID, as shown in the following snippet:

```
"Resource" : ["LogicalResourceId/myEC2instance"]
```

You can also use a wild card with logical IDs. For example, if you prefix the logical IDs of all related resources, you can specify them all with a wild card, as shown in the following snippet:

```
"Resource" : ["LogicalResourceId/MyPrefix*"]
```

You can also use a `Not` with resources. For example, if you wanted to allow updates to all resources, except for one, you can use a `NotResource`, as shown in the following sample:

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "NotResource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

When you set a stack policy, any update not explicitly allowed is denied by default. By allowing updates to all resources except for the `ProductionDatabase` resource, updates to the `ProductionDatabase` resource are denied.

Conditions

Specifies the resource type (p. 243) that the policy applies to. If you want to specify the logical IDs of specific resources, use the `Resource` element.

You can specify a resource type such as all Amazon EC2 instances and Amazon RDS DB instances, as shown in the following sample:

```
{
  "Statement" : [
  {
    "Effect" : "Deny",
    "Principal" : "*",
    "Action" : "Update:*",
    "Resource" : "*",
    "Condition" : {
      "StringEquals" : {
        "ResourceType" : ["AWS::EC2::Instance", "AWS::RDS::DBInstance"]
      }
    }
  },
  {
    "Effect" : "Allow",
    "Principal" : "*",
    "Action" : "Update:*",
    "Resource" : "*"
  }
  ]
}
```

When you set a stack policy, any update not explicitly allowed is denied by default. The `Allow` statement grants update permissions to all resources except for Amazon EC2 instances and Amazon RDS DB instances. The `Deny` statement always overrides any allows.

You can also 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 snippet:

```
"Condition" : {
  "StringLike" : {
    "ResourceType" : ["AWS::EC2::*"]
  }
}
```

You must use the `StringLike` condition when you use wild cards.

Sample Stack Policies

Prevent any updates to all stack resources

In order to prevent updates to all stack resources, the following policy specifies a `Deny` for all update actions on all resources:

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Prevent updates to a database only

The following policy denies all update actions for the database with the `MyDatabase` logical ID. To allow updates for all other stack resources, the policy also allows all update actions on all resources. The `Allow` statement doesn't affect the `MyDatabase` resource because the `Deny` statement always overrides any allows.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "LogicalResourceId/MyDatabase"
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Another way to achieve the same result is to use the default deny. When you set a stack policy, any update not explicitly allowed is denied by default. The following sample uses a `NotResource` to allow updates to all resources, except for the `ProductionDatabase` resource.

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "NotResource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

By allowing updates to all resources except for the `ProductionDatabase` resource, updates to the `ProductionDatabase` resource are denied by default. However, because an explicit deny overrides any allows, you can ensure that a resource is protected by using a `Deny` statement.

Prevent any updates to all Amazon RDS DB instances

The following policy denies all update actions for the Amazon RDS DB instance resource type. To allow updates for all other stack resources, the policy specifies an allow for all update actions on all resources. The `Allow` statement does not affect the Amazon RDS DB instance resources because the `Deny` statement always overrides any allows.

```
{
  "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 for the instance with the `MyInstance` logical ID. To allow updates for all other stack resources, the policy also allows all update actions on all resources. As always, however, the `Allow` statement doesn't affect the `MyInstance` resource because the `Deny` statement always overrides any allows.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:Replace",
      "Principal": "*",
      "Resource" : "LogicalResourceId/MyInstance"
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

Prevent updates to any nested stacks

The following policy denies all update actions for the AWS CloudFormation stack resource type (nested stacks). To updates for all other stack resources, the policy also allows all update actions on all resources. As always, however, the `Allow` statement does not affect the AWS CloudFormation stack resources because the `Deny` statement always overrides any allows.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "ResourceType" : ["AWS::CloudFormation::Stack"]
        }
      }
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

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. 268) 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:\\Share
Point\\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:\\Share
Point\\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" : "Instan
ceType" }, "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" : "SharePoint
FoundationWaitHandle" }}, "\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 retreive the auto-generated temporary Administrator password for the instance. Instructions about how to do this are provided in the Accessing AWS CloudFormation Windows Instances (p. 110) topic.

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-reload
er.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`

Accessing AWS CloudFormation Windows Instances

Once you've successfully created a Microsoft Windows stack on AWS CloudFormation, you can log in to your instance with Remote Desktop to configure it manually. There are a number of steps involved:

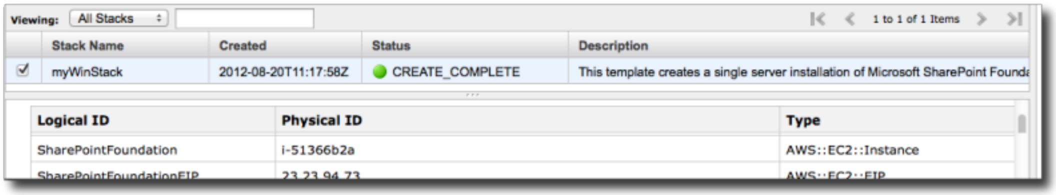
1. Find the physical id of your Windows instance.
2. Use the physical id to retrieve the login credentials from Amazon EC2.
3. Use the login credentials to access your instance with Remote Desktop.

> **Note**
> Before starting, you'll need to have an AWS CloudFormation Windows stack running, and you'll also need the private key of the key pair you used when creating the instance. For information about generating Amazon EC2 key pairs, see Creating an EC2 Key Pair (p. 76).

To retrieve the physical ID of your AWS CloudFormation Windows instance:

1. From the AWS CloudFormation console, click on your Windows-based stack. You will see your stack information appear in the lower pane of the window.
2. Click the **Resources** tab, and find the **Physical ID** of the AWS::EC2::Instance (p. 302). It will look something like this: `i-51366b2a`.

   If you have many instances running, you will probably want to remember the physical ID of your instance, or write it down. You'll need it to recover the Administrator password to log in to your instance.



Viewing: All Stacks

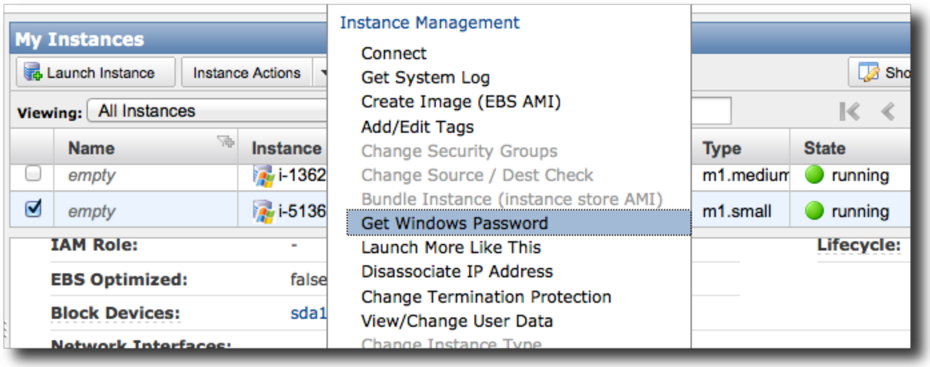
	Stack Name	Created	Status	Description
☑	myWinStack	2012-08-20T11:17:58Z	● CREATE_COMPLETE	This template creates a single server installation of Microsoft SharePoint Founda

Logical ID	Physical ID	Type
SharePointFoundation	i-51366b2a	AWS::EC2::Instance
SharePointFoundationEIP	23.23.94.73	AWS::EC2::EIP

Once you have the physical ID of your instance, you can use this to retrieve the Administrator password.

To retrieve the Administrator password for your Windows instance:

1. At the top left corner of the AWS CloudFormation console, click **Services** and then **EC2**. This will bring you to the **Amazon EC2 Console Dashboard**.
2. On the **Navigation Bar**, click **Instances**. This will bring up a list titled **My Instances**.
3. In the list, find your instance by its physical ID. Once you find it, right-click its entry on the list. This will display the **Instance Management** context menu.



My Instances

Viewing: All Instances

	Name	Instance	Type	State
□	*empty*	i-1362	m1.mediun	● running
☑	*empty*	i-5136	m1.small	● running

IAM Role: -

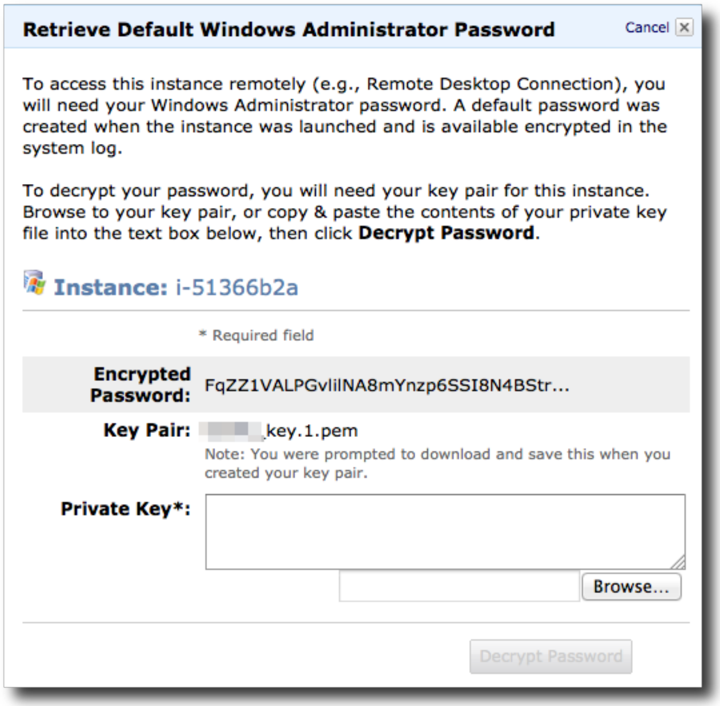
EBS Optimized: false

Block Devices: sda1

Network Interfaces:

Lifecycle:

4. On the context menu, click **Get Windows Password**. A dialog will appear, called **Retrieve Default Windows Administrator Password**. On this dialog, an encrypted password will be shown, as well as the Amazon EC2 key pair that you used when creating the AWS CloudFormation Windows stack.



Retrieve Default Windows Administrator Password

To access this instance remotely (e.g., Remote Desktop Connection), you will need your Windows Administrator password. A default password was created when the instance was launched and is available encrypted in the system log.

To decrypt your password, you will need your key pair for this instance. Browse to your key pair, or copy & paste the contents of your private key file into the text box below, then click **Decrypt Password**.

Instance: i-51366b2a

* Required field

Encrypted Password: FqZZ1VALPGvlilNA8mYnzp6SSI8N4BStr...

Key Pair: [redacted]key.1.pem

Note: You were prompted to download and save this when you created your key pair.

Private Key*:

5. Do *one* of the following (they are equivalent):

   - Locate the private key file you downloaded that corresponds to the key pair shown, copy its contents to the clipboard, and then paste it into the **Private Key** box on the dialog.
   - Click the **Browse** button to browse for the private key file on your system. When you select it, the contents of the file will appear in the **Private Key** box.

6. Click **Decrypt Password**. The connection information for your instance will be shown, consisting of:

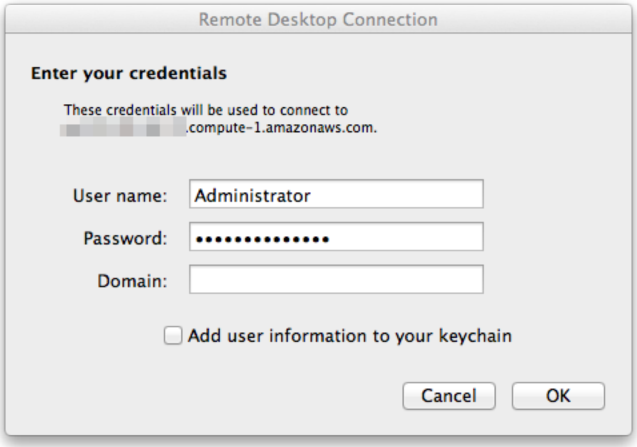
   - the IP address of your remote instance.
   - The user name to use when logging in.
   - The decrypted password.

Note
This password is meant to be temporary. Once you log in to your instance, you should change it to one of your own choice.

These user credentials can be used to log in to your Windows instance with Remote Desktop.

To log in to your AWS CloudFormation Windows stack:

1. Start your Remote Desktop client.
2. When prompted for the **Server**, enter the server name that you retrieved for your instance from EC2.



Remote Desktop Connection

Enter your credentials

These credentials will be used to connect to
[redacted].compute-1.amazonaws.com.

User name: Administrator

Password: ••••••••••••••

Domain:

□ Add user information to your keychain

3. Enter the **User name** ("Administrator") and the **Password** that you retrieved from EC2.
4. If you are prompted for a **Domain**, leave the field blank.
5. Click **OK** to finish connecting.

Once you're logged in to your server, you can configure it how you like. You can also use this credential information to log in to any secure outputs that your stack created, such as a Sharepoint site. It's your Windows instance, do what you want with it!

Working with AWS CloudFormation Templates

Topics

The key to getting the most out of AWS CloudFormation is a thorough understanding of templates. A template is a text file whose format complies with the JSON standard.

To get you started quickly on modifying and authoring templates, this section provides template anatomy details, example templates and template snippets. This section also discusses how to modify and validate templates.

- In Template Anatomy (p. 115), we provide the technical details for coding each of the template objects.
- In Template Snippets (p. 150), we provide a number of template sections that demonstrate how to write the JSON code for a particular section of a template. In this section you'll find starter snippets for Amazon EC2 instances, Amazon S3 domains, AWS CloudFormation mappings, and more. The snippets are selected to cover a range of resources and properties you are likely to include often in your templates. They are grouped by the resources they would be used to declare, with general-purpose AWS CloudFormation snippets in AWS CloudFormation Template Snippets (p. 210)).
- The section Example Templates (p. 129) contains a number of sample templates that will create stacks with little or no modification. The samples range in complexity, and highlight the use of AWS CloudFormation template features in the context of a complete application. Some of the templates require you to specify values in the command's `--parameters` option.

For details about the supported resources, type names, intrinsic functions, and pseudo parameters you can use in your templates, see the Template Reference (p. 243) section.

Template Anatomy

A template is a JSON-formatted text file that describes your AWS infrastructure. Templates include several major sections. The `Resources` section is the only section that is required. The first character in the template must be an open brace ({), and the last character must be a closed brace (}). The following template fragment shows the template structure and sections.

```
{
  "AWSTemplateFormatVersion" : "version date",

  "Description" : "JSON string",

  "Parameters" : {
    set of parameters
  },

  "Mappings" : {
    set of mappings
  },

  "Conditions" : {
    set of conditions
  },

  "Resources" : {
    set of resources
  },

  "Outputs" : {
    set of outputs
  }
}
```

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 previous example, as values in one section might refer to values from a previous section. The following list gives a brief overview of each section.

Format Version (optional) (p. 116)
: 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. 116)
: A text string that describes the template. This section must always follow the template format version section.

Parameters (optional) (p. 116)
: 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. 121)
: 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. 557) intrinsic function in the `Resources` and `Outputs` section.

Conditions (optional) (p. 124)
: 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.

Resources (required) (p. 126)

Specifies the stack resources and their properties, such as an Amazon EC2 instance or an Amazon S3 bucket. You can refer to resources in the `Resources` and `Outputs` sections of the template.

Outputs (optional) (p. 128)

Describes the values that are returned whenever you view your stack's properties. For example, you can declare an output for an Amazon S3 bucket name and then call the `aws cloudformation describe-stacks` AWS CLI command to view the name.

See Also

For more information about JSON, see http://www.json.org.

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:

```
"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:

```
"Description" : "Here are some details about the template."
```

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. For example, you can create a parameter for Amazon EC2 instance types, as shown in the following snippet:

```
"Parameters" : {
  "InstanceTypeParameter" : {
    "Type" : "String",
    "Default" : "t1.micro",
    "AllowedValues" : ["t1.micro", "m1.small", "m1.large"],
    "Description" : "Enter t1.micro, m1.small, or m1.large. Default is t1.micro."
  }
}
```

When you create a stack, you can specify the value for the `InstanceTypeParameter`. That way, you can choose what instance type you want when you create a stack. By default, the template uses `t1.micro`. Within the same template, you can use the `Ref` intrinsic function to specify the parameter value in other parts of the template, as shown in the following snippet:

```
"Ec2Instance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "InstanceType" : { "Ref" : "InstanceTypeParameter" },
    "ImageId" : "ami-2f726546"
  }
}
```

Syntax and Properties

The `Parameters` section consists of the key name `Parameters`, followed by a single colon. Braces enclose all parameter declarations. If you declare multiple parameters, they are delimited by commas. You have a maximum of 60 parameters in an AWS CloudFormation template.

For each parameter, you must declare a logical name in quotation marks followed by a colon. The logical name 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.

> **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 (`*****`).

The following table describes all the properties for a parameter and whether a property is required:

Property	Required	Description

Type	Yes	The data type for the parameter.
		`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"`.
		AWS-specific types
		For AWS-specific types, you can specify the following values:
		• `AWS::EC2::KeyPair::KeyName` (An Amazon EC2 key pair name)
		• `AWS::EC2::SecurityGroup::Id` (A security group ID)
		• `AWS::EC2::Subnet::Id` (A subnet ID)
		• `AWS::EC2::VPC::Id` (A VPC ID)
		• `List<AWS::EC2::VPC::Id>` (An array of VPC IDs)
		• `List<AWS::EC2::SecurityGroup::Id>` (An array of security group IDs)
		• `List<AWS::EC2::Subnet::Id>` (An array of subnet IDs)
		AWS CloudFormation validates input values for these types against existing values in a user's account. For example, with the `AWS::EC2::KeyPair::KeyName` type, a user must enter an existing Amazon EC2 key pair name that is in her account and in the region in which she is creating the stack.

Default	No	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.
NoEcho	No	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 (`*****`).

Property	Required	Description
AllowedValues	No	An array containing the list of values allowed for the parameter.
AllowedPattern	No	A regular expression that represents the patterns you want to allow for `String` types.
MaxLength	No	An integer value that determines the largest number of characters you want to allow for `String` types.
MinLength	No	An integer value that determines the smallest number of characters you want to allow for `String` types.
MaxValue	No	A numeric value that determines the largest numeric value you want to allow for `Number` types.
MinValue	No	A numeric value that determines the smallest numeric value you want to allow for `Number` types.
Description	No	A string of up to 4000 characters that describes the parameter.

ConstraintDescription	No	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`

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.

```
"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 Amazon EC2 instances.

If a user uses the AWS Management Console, AWS CloudFormation prepopulates AWS-specific parameter types with valid values. That way the userdoesn't have to remember and correctly enter a specific name or ID. He just selects one or more values from a drop-down list.

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.

```
"Parameters" : {
  "myKeyPair" : {
    "Description" : "Amazon EC2 Key Pair",
    "Type" : "AWS::EC2::KeyPair::KeyName"
  },
  "mySubnetIDs" : {
    "Description" : "Subnet IDs",
    "Type" : "List<AWS::EC2::Subnet::Id>"
  }
}
```

Currently, a user can't use the AWS CLI or AWS CloudFormation API to view a list of valid values for AWS-specific parameters. However, he 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:

```
"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"
  }
}
```

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:

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

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 base a mapping on a parameter, pseudo parameter, or intrinsic function.

Syntax

The `Mappings` section consists of the key name `Mappings`, followed by a single colon. Braces enclose all mapping declarations. If you declare multiple mappings, they are delimited by commas. The keys and values in mappings must be literal strings. For each mapping, you must declare a logical name in quotation marks followed by a colon and braces that enclose the sets of values to map. The following example shows a `Mappings` section containing a single mapping named `Mapping01`.

```
"Mappings" : {
  "Mapping01" : {
    "Key01" : {
      "Value" : "Value01"
    },
    "Key02" : {
      "Value" : "Value02"
    },
    "Key03" : {
      "Value" : "Value03"
    }
  }
}
```

Within a mapping, each map is a key followed by a colon and a set of name-value pairs that are enclosed by braces. The key identifies each map, and it must be unique within the mapping. Within the braces, you can declare multiple name-value pairs.

Examples

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.

```
"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"}
  }
}
```

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. The following example has region keys that are mapped to two sets of values: one named 32 and the other 64.

```
"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" }
}
```

You can use the `Fn::FindInMap (p. 557)` 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. 568)) and 32 as the name of the value to map to.

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

The following example shows a Mappings section with a mapping that contains three keys that map to arrays that contain multiple string values. The keys represent three regions, and the mapped values are the list of Availability Zones used in each region. The AWS::ElasticLoadBalancing::LoadBalancer (p. 376) resource uses the FindInMap function and the Region2AZ map to specify the AvailabilityZones property.

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

  "Mappings" : {
    "Region2AZ" : {
      "us-west-1" : { "AZ" : ["us-west-1a", "us-west-1b"] },
      "us-east-1" : { "AZ" : ["us-east-1a", "us-east-1b", "us-east-1c"] },
      "eu-west-1" : { "AZ" : ["eu-west-1a", "eu-west-1b"] }
    }
  },

  "Resources" : {
    "MyELB" : {
      "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
      "Properties" : {
        "AvailabilityZones" : { "Fn::FindInMap" : [ "Region2AZ", { "Ref" :
"AWS::Region" }, "AZ" ] },
        "Listeners" : [ {
          "LoadBalancerPort" : "8888" ,
          "InstancePort" : "8888" ,
          "Protocol" : "HTTP"
        } ],
        "HealthCheck" : {
          "Target" : { "Fn::Join" : [ "", ["HTTP:", "8888", "/"]]},
          "HealthyThreshold" : "5",
          "UnhealthyThreshold" : "2",
          "Interval" : "10",
          "Timeout" : "8"
        }
      }
    }
  }
}
```

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 parameters that you declare 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.

Syntax

The `Conditions` section consists of the key name `Conditions`, followed by a single colon. Braces enclose all condition declarations. If you declare multiple conditions, they are delimited by commas.

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:

```
"Conditions" : {
  "Logical ID" : {Intrinsic function}
}
```

You can use the following intrinsic functions to define conditions:

- `Fn::And`
- `Fn::Equals`
- `Fn::If`
- `Fn::Not`
- `Fn::Or`

For more information about the syntax of each intrinsic function, see Condition Functions (p. 547).

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.

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

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`.

Resources

The required `Resources` section declare the AWS resources that you want as part of your stack, such as an Amazon EC2 instance or an Amazon S3 bucket. You must declare each resource separately; however, you can specify multiple resources of the same type. If you declare multiple resources, separate them with commas.

Syntax

The `Resources` section consists of the key name `Resources`, followed by a single colon. Braces enclose all resource declarations. If you declare multiple resources, they are delimited by commas. The following pseudo template outlines the `Resources` section:

```
"Resources" : {
    "Logical ID" : {
        "Type" : "Resource type",
        "Properties" : {
            Set of properties
        }
    }
}
```

Logical ID

The logical ID must be alphanumeric (A-Za-z0-9) and unique within the template. You 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 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 Amazon EC2 instance ID or an Amazon S3 bucket name. You 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 Amazon 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 Amazon S3 bucket that stores logs as `MyPerformanceLogs`. For more information, see Name Type (p. 514).

Resource type

The resource type identifies the type of resource that you are declaring. For example, the `AWS::EC2::Instance` declares an Amazon EC2 instance. For a list of all the resource types, see AWS Resource Types Reference (p. 243).

Resource properties

Resource properties are additional options that you can specify for a resource. For example, for each Amazon EC2 instance, you must specify an AMI ID for that instance. You declare the AMI ID as a property of the instance, as shown in the following snippet:

```
"Resources" : {
  "MyEC2Instance" : {
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
      "ImageId" : "ami-2f726546"
    }
  }
}
```

If a resource does not require any properties to 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. When a property value is a literal string, the value is enclosed in double quotes. If a value is the result of a list of any kind, it is enclosed in brackets (`[ ]`). If a value is the result of an intrinsic function or reference, it is enclosed in braces (`{ }`). These rules apply when you combine literals, lists, references, and functions to obtain a value. The following sample shows you how to declare different property value types:

```
"Properties" : {
    "String" : "one-string-value",
    "LiteralList" : [ "first-value", "second-value" ],
    "Boolean" : "true"
    "ReferenceForOneValue" :  { "Ref" : "MyLogicalResourceName" } ,
    "FunctionResultWithFunctionParams" : {
        "Fn::Join" : [ "%", [ "Key=", { "Ref" : "MyParameter" } ] ] }
}
```

Examples

The following example shows a typical Resource declaration. It defines two resources. The `MyInstance` resource includes the `MyQueue` resource as part of its `UserData` property:

```
  "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" : {
        }
    }
}
```

Outputs

The optional `Outputs` section declares the values that you want to return in response to describe stack calls. For example, you can output the Amazon S3 bucket name for your stack so that you can easily find it.

Important
During a stack update, you cannot update outputs by themselves. You can update outputs only when you include changes that add, modify, or delete resources.

Syntax

The `Outputs` section consists of the key name `Outputs`, followed by a single colon. Braces enclose all output declarations. If you declare multiple outputs, they are delimited by commas. You can declare a maximum of 60 outputs in an AWS CloudFormation template. The following pseudo template outlines the `Outputs` section:

```
"Outputs" : {
  "Logical ID" : {
    "Value" : "Value to return",
    "Description" : "Information about the value"
  }
}
```

Value (required)
The value of the property that is returned by the aws cloudformation describe-stacks command.

Description (optional)
A `String` type up to 4K in length describing the output value.

Note
You can conditionally create outputs by adding a `Condition` property and then refer to a condition that is defined in the `Conditions` section of a template.

Examples

Output properties are declared like any other property. In the following example, the output named `LoadBalancer` returns the information for the resource with the logical name `BackupLoadBalancer` if the `CreateProdResources` condition is true.

```
"Outputs" : {
  "LoadBalancer" : {
    "Condition" : "CreateProdResources",
    "Value" : { "Ref" : "BackupLoadBalancer" }
  }
}
```

Example Templates

The example AWS CloudFormation templates are written to show the features of AWS CloudFormation, and to serve as a starting point for you to create custom stacks. We provide the two stack applications below. In the following sections we describe the template, its parts, and detail any special features it may have. A link to the latest source code for the template is also included.

Topics

More sample templates are available at http://aws.amazon.com/cloudformation/aws-cloudformation-templates/. In addition, we add new sample templates regularly to provide examples for newly supported features. Please check the AWS CloudFormation Discussion Forum for announcements. Also, other AWS CloudFormation users may have developed templates to provide custom solutions, and may post their AWS CloudFormation solutions to the forum as well.

Auto Scaling Group with LoadBalancer, Auto Scaling Policies, and CloudWatch Alarms

Topics

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.

> **Important**
> 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.

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 AutoScalingMultiAZWithNo
tifications: 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" : "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."
    },

    "OperatorEMail": {
      "Description": "EMail address to notify if there are any scaling opera
tions",
      "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" : {
    "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" : {
    "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=\"https://s3.amazonaws.com/cloudformation-ex
amples/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::CloudForma
tion::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" : [ "AWSInstance
Type2Arch", { "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" : [ "Elastic
LoadBalancer", "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. 245) 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. 562) 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. 251) 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. 322) 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. 558) 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. 376) 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. 257) 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. 287) 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.

Amazon EC2 Running an Amazon Linux 32-bit AMI

This template declares one parameter and four mappings. Resources include an Amazon EC2 instance and a security group. The mapping uses the AWS::Region pseudo parameter to select the appropriate AMI. The Outputs section prints the instance ID of the instance, the Availability Zone in which it is created, and its public IP address.

You can get the latest version of this sample template at https://s3.amazonaws.com/cloudformation-templates-us-east-1/EC2InstanceWithSecurityGroupSample.template.

Amazon Linux 32-bit AMI Sample Template

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

  "Description" : "AWS CloudFormation Sample Template EC2InstanceWithSecurity
GroupSample: Create an Amazon EC2 instance running the Amazon Linux AMI. The
AMI is chosen based on the region in which the stack is run. This example creates
 an EC2 security group for the instance to give you SSH access. **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."

    },

    "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" : {
    "EC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" :
"AWS::Region" },
                         { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref"
 : "InstanceType" }, "Arch" ] } ] }
      }
    },

    "InstanceSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable SSH access via port 22",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "22",
          "ToPort" : "22",
          "CidrIp" : { "Ref" : "SSHLocation"}
        } ]
      }
    }
  },

  "Outputs" : {
    "InstanceId" : {
      "Description" : "InstanceId of the newly created EC2 instance",
      "Value" : { "Ref" : "EC2Instance" }
    },
    "AZ" : {
      "Description" : "Availability Zone of the newly created EC2 instance",
      "Value" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone" ] }
    },
    "PublicDNS" : {
      "Description" : "Public DNSName of the newly created EC2 instance",
      "Value" : { "Fn::GetAtt" : [ "EC2Instance", "PublicDnsName" ] }
    },
    "PublicIP" : {
      "Description" : "Public IP address of the newly created EC2 instance",
      "Value" : { "Fn::GetAtt" : [ "EC2Instance", "PublicIp" ] }
    }
  }
}
```

Create a Load-Balanced Apache Website

This template declares two parameters and four mappings. Resources include an Elastic Load Balancing load balancer with listeners and health check, two Amazon EC2 instances, and a security group. The Outputs section prints the URL of the load balancer.

Load-Balanced Apache Website Sample Template

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

  "Description" : "Create a load balanced sample web site. The AMI is chosen
based on the region in which the stack is run. This example creates 2 EC2 in
stances behind a load balancer with a simple health check. The instances may
be created in one or more AZs. The web site is available on port 80, however,
the instances can be configured to listen on any port (8888 by default).
**WARNING** 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.",


  "Parameters" : {
    "InstanceType" : {
      "Description" : "Type of EC2 instance to launch",
      "Type" : "String",
      "Default" : "m1.small"
    },
    "WebServerPort" : {
      "Description" : "TCP/IP port of the web server",
      "Type" : "String",
      "Default" : "8888"
    },
    "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."

    }
  },

  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "64" },
      "m1.small"    : { "Arch" : "32" },
      "m1.large"    : { "Arch" : "64" },
      "m1.xlarge"   : { "Arch" : "64" },
      "m2.xlarge"   : { "Arch" : "64" },
      "m2.2xlarge"  : { "Arch" : "64" },
      "m2.4xlarge"  : { "Arch" : "64" },
      "c1.medium"   : { "Arch" : "32" },
      "c1.xlarge"   : { "Arch" : "64" },
      "cc1.4xlarge" : { "Arch" : "64" }
    },
    "AWSRegionArch2AMI" : {
      "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" : {
    "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"
        }
      }
    },
    "Ec2Instance1" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" :
"AWS::Region" },
                                          { "Fn::FindInMap" : [ "AWSInstance
Type2Arch", { "Ref" : "InstanceType" },
                                          "Arch" ] } ] },
        "UserData" : { "Fn::Base64" : { "Ref" : "WebServerPort" }}
      }
    },

    "Ec2Instance2" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" :
"AWS::Region" },
                                          { "Fn::FindInMap" : [ "AWSInstance
Type2Arch", { "Ref" : "InstanceType" },
                                          "Arch" ] } ] },
        "UserData" : { "Fn::Base64" : { "Ref" : "WebServerPort" }}
      }
    },

    "InstanceSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable SSH access and HTTP access on the inbound
 port",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "22",
          "ToPort" : "22",
          "CidrIp" : "0.0.0.0/0"
        },
        {
          "IpProtocol" : "tcp",
          "FromPort" : { "Ref" : "WebServerPort" },
          "ToPort" : { "Ref" : "WebServerPort" },
          "CidrIp" : "0.0.0.0/0"
        } ]
      }
    }
  },

  "Outputs" : {
    "URL" : {
      "Description" : "URL of the sample website",
      "Value" :  { "Fn::Join" : [ "", [ "http://", { "Fn::GetAtt" : [ "Elastic
LoadBalancer", "DNSName" ]}]]}
    }
  }
}
```

Auto-Scaled Worker that uses Spot Instances to Monitor Work in an SQS Queue

This template uses spot instances to create an auto-scaled worker that monitors work (messages) in an SQS queue. The application is auto-scaled based on the amount of work in the queue. When there is work, Auto Scaling scales up; when there is no work, Auto Scaling scales down. Each message contains a command or script to run, an input file location, and an output location for the results.

WorkerRole Template

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

  "Description" : "AWS CloudFormation Sample Template WorkerRole: Create a multi-
az, Auto Scaled worker that pulls command messages from a queue and execs the
command. Each message contains a command/script to run, an input file location and
 an output location for the results. The application is Auto-Scaled based on the
amount of work in the queue. **WARNING** This template creates one or more Amazon
 EC2 instances and an Amazon SQS queue. You will be billed for the AWS resources
used if you create a stack from this template.",

  "Parameters" : {
    "InstanceType" : {
      "Description" : "Worker EC2 instance type",
      "Type" : "String",
      "Default" : "m1.small",
      "AllowedValues" : [ "t1.micro","m1.small","m1.medi
um","m1.large","m1.xlarge","m2.xlarge","m2.2xlarge","m2.4xlarge","c1.medi
um","c1.xlarge","cc1.4xlarge","cc2.8xlarge","cg1.4xlarge"],
      "ConstraintDescription" : "must be a valid EC2 instance type."
    },

    "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."
    },

    "MinInstances" : {
      "Description" : "The minimum number of Workers",
      "Type" : "Number",
      "MinValue" : "0",
      "Default"  : "0",
      "ConstraintDescription" : "Enter a number >=0"
    },

    "MaxInstances" : {
      "Description" : "The maximum number of Workers",
      "Type" : "Number",
      "MinValue" : "1",
      "Default"  : "1",
      "ConstraintDescription" : "Enter a number >1"
    }
  },
  "Mappings" : {
    "AWSInstanceType2Arch" : {
      "t1.micro"    : { "Arch" : "64" },
      "m1.small"    : { "Arch" : "64" },
      "m1.medium"   : { "Arch" : "64" },
      "m1.large"    : { "Arch" : "64" },
      "m1.xlarge"   : { "Arch" : "64" },
      "m2.xlarge"   : { "Arch" : "64" },
      "m2.2xlarge"  : { "Arch" : "64" },
      "m2.4xlarge"  : { "Arch" : "64" },
      "c1.medium"   : { "Arch" : "64" },
      "c1.xlarge"   : { "Arch" : "64" },
      "cc1.4xlarge" : { "Arch" : "64HVM" },
      "cc2.8xlarge" : { "Arch" : "64HVM" },
      "cg1.4xlarge" : { "Arch" : "64HVM" }
    },

    "AWSRegionArch2AMI" : {
      "us-east-1"      : { "32" : "ami-31814f58", "64" : "ami-1b814f72", "64HVM"
: "ami-0da96764" },
      "us-west-2"      : { "32" : "ami-38fe7308", "64" : "ami-30fe7300", "64HVM"
: "NOT_YET_SUPPORTED" },
      "us-west-1"      : { "32" : "ami-11d68a54", "64" : "ami-1bd68a5e", "64HVM"
: "NOT_YET_SUPPORTED" },
      "eu-west-1"      : { "32" : "ami-973b06e3", "64" : "ami-953b06e1", "64HVM"
: "NOT_YET_SUPPORTED" },
      "ap-southeast-1" : { "32" : "ami-b4b0cae6", "64" : "ami-beb0caec", "64HVM"
: "NOT_YET_SUPPORTED" },
      "ap-northeast-1" : { "32" : "ami-0644f007", "64" : "ami-0a44f00b", "64HVM"
: "NOT_YET_SUPPORTED" },
      "sa-east-1"      : { "32" : "ami-3e3be423", "64" : "ami-3c3be421", "64HVM"
: "NOT_YET_SUPPORTED" }
    }
  },

  "Resources" : {

    "WorkerUser" : {
      "Type" : "AWS::IAM::User",
      "Properties" : {
        "Path": "/",
        "Policies": [{
          "PolicyName": "root",
          "PolicyDocument": {
            "Version": "2012-10-17",
            "Statement":[{
              "Effect": "Allow",
              "Action": [
                "cloudformation:DescribeStackResource",
                "sqs:ReceiveMessage",
                "sqs:DeleteMessage",
                "sns:Publish"
              ],
              "Resource": "*"
            }]
          }
        }]
      }
    },

    "WorkerKeys" : {
      "Type" : "AWS::IAM::AccessKey",
      "Properties" : {
        "UserName" : {"Ref": "WorkerUser"}
      }
    },

    "InputQueue" : {
      "Type" : "AWS::SQS::Queue"
    },

    "InputQueuePolicy" : {
      "Type" : "AWS::SQS::QueuePolicy",
      "DependsOn" : "LaunchConfig",
      "Properties" : {
        "Queues" : [ { "Ref" : "InputQueue" } ],
        "PolicyDocument":  {
          "Version": "2012-10-17",
          "Id": "ReadFromQueuePolicy",
          "Statement" : [ {
            "Sid": "ConsumeMessages",
            "Effect": "Allow",
            "Principal" : { "AWS": {"Fn::GetAtt" : ["WorkerUser", "Arn"]} },
            "Action": ["sqs:ReceiveMessage", "sqs:DeleteMessage"],
            "Resource": { "Fn::GetAtt" : [ "InputQueue", "Arn" ] }
          } ]
        }
      }
    },

    "InstanceSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable SSH access",
        "SecurityGroupIngress" : [ { "IpProtocol" : "tcp", "FromPort" : "22",
"ToPort" : "22", "CidrIp" : "0.0.0.0/0" } ]
      }
    },

    "LaunchConfig" : {
      "Type" : "AWS::AutoScaling::LaunchConfiguration",
      "Metadata" : {
        "Comment" : "Install a simple PHP application",
        "AWS::CloudFormation::Init" : {
          "configSets" : {
            "ALL" : ["XML", "Time", "LWP", "AmazonLibraries", "WorkerRole"]
          },
          "XML" : {
            "packages" : {
              "yum" : {
                "perl-XML-Simple"         : []
              }
            }
          },
          "Time" : {
            "packages" : {
              "yum" : {
                "perl-LWP-Protocol-https" : []
              }
            }
          },
          "LWP" : {
            "packages" : {
              "yum" : {
                "perl-Time-HiRes"         : []
              }
            }
          },
          "AmazonLibraries" : {
            "sources" : {
              "/home/ec2-user/sqs" : "http://s3.amazonaws.com/awscode/amazon-
queue/2009-02-01/perl/library/amazon-queue-2009-02-01-perl-library.zip"
            }
          },
          "WorkerRole" : {
            "files" : {
              "/etc/cron.d/worker.cron" : {
                "content" : "*/1 * * * * ec2-user /home/ec2-user/worker.pl &>
/home/ec2-user/worker.log\n",
                "mode"    : "000644",
                "owner"   : "root",
                "group"   : "root"
              },

              "/home/ec2-user/worker.pl" : {
                "content" : { "Fn::Join" : ["", [
                  "#!/usr/bin/perl -w\n",
                  "#\n",
                  "use strict;\n",
                  "use Carp qw( croak );\n",
                  "use lib qw(/home/ec2-user/sqs/amazon-queue-2009-02-01-perl-
library/src);  \n",
                  "use LWP::Simple qw( getstore );\n",
                  "\n",
                  "my $AWS_ACCESS_KEY_ID     = \"", { "Ref" : "WorkerKeys" },
"\";\n",
                 "my $AWS_SECRET_ACCESS_KEY = \"", { "Fn::GetAtt": ["WorkerKeys",
 "SecretAccessKey"]}, "\";\n",
                  "my $QUEUE_NAME            = \"", { "Ref" : "InputQueue" },
"\";\n",
                  "my $COMMAND_FILE          = \"/home/ec2-user/command\";\n",
                  "\n",
                  "eval {\n",
                  "\n",
                  "  use Amazon::SQS::Client; \n",
                  "  my $service = Amazon::SQS::Client->new($AWS_ACCESS_KEY_ID,
$AWS_SECRET_ACCESS_KEY);\n",
                  " \n",
               "  my $response = $service->receiveMessage({QueueUrl=>$QUEUE_NAME,
 MaxNumberOfMessages=>1});\n",
                  "  if ($response->isSetReceiveMessageResult) {\n",
                  "    my $result = $response->getReceiveMessageResult();\n",
                  "    if ($result->isSetMessage) {\n",
                  "      my $messageList = $response->getReceiveMessageResult()-
>getMessage();\n",
                  "      foreach(@$messageList) {\n",
                  "        my $message = $_;\n",
                  "        my $messageHandle = 0;\n",
                  "        if ($message->isSetReceiptHandle()) {\n",
                  "          $messageHandle = $message->getReceiptHandle();\n",
                  "        } else {\n",
                  "          croak \"Couldn't get message Id from message\";\n",
                  "        }\n",
                  "        if ($message->isSetBody()) {\n",
                  "          my %parameters = split(/[=;]/, $message->get
Body());\n",
                  "          if (defined($parameters{\"Input\"}) &&
defined($parameters{\"Output\"}) && defined($parameters{\"Command\"})) {\n",
                  "            getstore($parameters{\"Command\"}, $COM
MAND_FILE);\n",
                  "            chmod(0755, $COMMAND_FILE);\n",
                  "            my $command = $COMMAND_FILE . \" \" . $paramet
ers{\"Input\"} . \" \" . $parameters{\"Output\"};\n",
                  "            my $result = `$command`;\n",
                  "            print \"Result = \" . $result . \"\\n\";\n",
                  "          } else {\n",
                  "            croak \"Invalid message\";\n",
                  "          }\n",
                  "        } else {\n",
                 "          croak \"Couldn't get message body from message\";\n",

                  "        }\n",
                  "        my $response = $service->deleteMes
sage({QueueUrl=>$QUEUE_NAME, ReceiptHandle=>$messageHandle});\n",
                  "      }\n",
                  "    } else {\n",
                  "      printf \"Empty Poll\\n\";\n",
                  "    }\n",
                  "  } else {\n",
                  "    croak \"Call failed\";\n",
                  "  }\n",
                  "}; \n",
                  "\n",
                  "my $ex = $@;\n",
                  "if ($ex) {\n",
                  "  require Amazon::SQS::Exception;\n",
                  "  if (ref $ex eq \"Amazon::SQS::Exception\") {\n",
                  "    print(\"Caught Exception: \" . $ex->getMessage() .
\"\\n\");\n",
                  "  } else {\n",
                  "    croak $@;\n",
                  "  }\n",
                  "}\n"
                ]]},
                "mode"    : "000755",
                "owner"   : "ec2-user",
                "group"   : "ec2-user"
              }
            }
          }
        }
      },
      "Properties" : {
        "KeyName" : { "Ref" : "KeyName" },
        "SpotPrice" : "0.05",
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" :
"AWS::Region" },
                                          { "Fn::FindInMap" : [ "AWSInstance
Type2Arch", { "Ref" : "InstanceType" },
                                          "Arch" ] } ] },
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "InstanceType" : { "Ref" : "InstanceType" },
        "UserData"       : { "Fn::Base64" : { "Fn::Join" : ["", [
          "#!/bin/bash\n",
          "yum update -y aws-cfn-bootstrap\n",
          "# Install the Worker application\n",
          "/opt/aws/bin/cfn-init ",
          "         --stack ", { "Ref" : "AWS::StackName" },
          "         --resource LaunchConfig ",
          "         --configset ALL",
          "         --access-key ", { "Ref" : "WorkerKeys" },
          "         --secret-key ", {"Fn::GetAtt": ["WorkerKeys", "SecretAccess
Key"]},
          "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      }
    },

    "WorkerGroup" : {
      "Type" : "AWS::AutoScaling::AutoScalingGroup",
      "Properties" : {
        "AvailabilityZones" : { "Fn::GetAZs" : ""},
        "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },
        "MinSize" : { "Ref" : "MinInstances" },
        "MaxSize" : { "Ref" : "MaxInstances" }
      }
    },

    "WorkerScaleUpPolicy" : {
      "Type" : "AWS::AutoScaling::ScalingPolicy",
      "Properties" : {
        "AdjustmentType" : "ChangeInCapacity",
        "AutoScalingGroupName" : { "Ref" : "WorkerGroup" },
        "Cooldown" : "60",
        "ScalingAdjustment" : "1"
      }
    },

    "WorkerScaleDownPolicy" : {
      "Type" : "AWS::AutoScaling::ScalingPolicy",
      "Properties" : {
        "AdjustmentType" : "ChangeInCapacity",
        "AutoScalingGroupName" : { "Ref" : "WorkerGroup" },
        "Cooldown" : "60",
        "ScalingAdjustment" : "-1"
      }
    },

    "TooManyMessagesAlarm": {
     "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": "Scale-Up if queue depth grows beyond 10 messages",
        "Namespace": "AWS/SQS",
        "MetricName": "ApproximateNumberOfMessagesVisible",
        "Dimensions": [{ "Name": "QueueName", "Value" : { "Fn::GetAtt" : ["In
putQueue", "QueueName"] } }],
        "Statistic": "Sum",
        "Period": "60",
        "EvaluationPeriods": "3",
        "Threshold": "1",
        "ComparisonOperator": "GreaterThanThreshold",
        "AlarmActions": [ { "Ref": "WorkerScaleUpPolicy" } ]
      }
    },

    "NotEnoughMessagesAlarm": {
     "Type": "AWS::CloudWatch::Alarm",
     "Properties": {
        "AlarmDescription": "Scale-down if there are too many empty polls, indic
ating there is not enough work",
        "Namespace": "AWS/SQS",
        "MetricName": "NumberOfEmptyReceives",
        "Dimensions": [{ "Name": "QueueName", "Value" : { "Fn::GetAtt" : ["In
putQueue", "QueueName"] } }],
        "Statistic": "Sum",
        "Period": "60",
        "EvaluationPeriods": "10",
        "Threshold": "3",
        "ComparisonOperator": "GreaterThanThreshold",
        "AlarmActions": [ { "Ref": "WorkerScaleDownPolicy" } ]
      }
    }
  },

  "Outputs" : {
    "QueueURL" : {
      "Description" : "URL of input queue",
      "Value" : { "Ref" : "InputQueue" }
    }
  }
}
```

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

Auto Scaling 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.

```
"SimpleConfig" : {
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
   "Properties" : {
      "ImageId" : "ami-6411e20d",
      "SecurityGroups" : [ { "Ref" : "myEC2SecurityGroup" }, "myExistingEC2Se
curityGroup" ],
      "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. 245) 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. 562) 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. 376) resource LB.

```

"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. 257) 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. 287) resource CPUAlarmHigh specifies the scaling policy ScaleUpPolicy as the action to execute when the alarm is in an *`ALARM`* state (*`AlarmActions`*).

```

"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. 245) resource that sends Amazon SNS notifications when the specified events take place. The *NotificationConfiguration* 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.

```
"MyAsGroupWithNotification" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones" : { "Ref" : "azList" },
      "LaunchConfigurationName" : { "Ref" : "myLCOne" },
      "MinSize" : "0",
      "MaxSize" : "2",
      "DesiredCapacity" : "1",
      "NotificationConfiguration" : {
         "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. 543) with an auto-scaling group.

```
"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"
   }
}
```

Amazon CloudFront Template Snippets

Topics

Amazon CloudFront Distribution Resource with an Amazon S3 Origin

This example shows an Amazon CloudFront Distribution (p. 283) using an S3Origin (p. 479).

```
"myDistribution" : {
    "Type" : "AWS::CloudFront::Distribution",
    "Properties" : {
        "DistributionConfig" : {
            "Origins" : [ {
                "DomainName": "mybucket.s3.amazonaws.com",
                "Id" : "myS3Origin",
                "S3OriginConfig" : {
                    "OriginAccessIdentity" : "origin-access-identity/cloud
front/E127EXAMPLE51Z"
                }
            }],
            "Enabled" : "true",
            "Comment" : "Some comment",
            "DefaultRootObject" : "index.html",
            "Logging" : {
                "IncludedCookies" : "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

This example shows an Amazon CloudFront Distribution (p. 283) using a CustomOrigin (p. 478).

```
"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": {
                "IncludedCookies" : "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.

This template snippet shows how to declare a CloudFront Distribution (p. 283) with multi-origin support. In the DistributionConfig (p. 470), a list of origins is provided and a DefaultCacheBehavior (p. 475) is set.

```
{
    "AWSTemplateFormatVersion" : "2010-09-09",
    "Resources" : {
        "myDistribution" : {
            "Type" : "AWS::CloudFront::Distribution",
            "Properties" : {
                "DistributionConfig" : {
                    "Origins" : [ {
                        "Id" : "myS3Origin",
                        "DomainName" : "mybucket.s3.amazonaws.com",
                        "S3OriginConfig" : {
                            "OriginAccessIdentity" : "origin-access-iden
tity/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" : {
                       "IncludedCookies" : "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", "OP
TIONS", "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", "OP
TIONS", "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 Logs Sample

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 Developer 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.

```
{
    "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 op
erations",
          "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:*",
                                        "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": "Web
ServerLogGroup"}, "\n",
                                            "log_stream_name = {in

stance_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.WebServer
Host.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 update -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-cloud
watch/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"
                    }
                ],
                "Unit": "Count",
                "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"
                    }
                ],
                "Unit": "Kilobytes",
                "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" : [ "Web
ServerHost", "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"
            }
        }
    }
}
```

See Also

For more information about CloudWatch Logs resources, see AWS::Logs::LogGroup (p. 397) or AWS::Logs::MetricFilter (p. 398).

AWS CloudFormation Amazon EC2 Template Snippets

Topics

EC2 Block Device Mapping Examples

EC2 Instance with Block Device Mapping

```
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : 
"AWS::Region" },
                                          { "Fn::FindInMap" : [ "AWSInstance
Type2Arch", { "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

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

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. 299).

```
"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. 301).

```
"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. 301).

```
"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.

```
  "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" : "con
trolXface"}, "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"]]}
  }
   }
    }
  }
```

Amazon EC2 Instance Resource

This snippet shows a simple AWS::EC2::Instance resource.

```
"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::ETC::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.

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

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.

```
"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"
 }
}
```

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.

```
"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.

```
"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"}

         }
     ]
 }
}
```

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.

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

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, the *SecurityGroupIngress* properties must include *VpcId* and you must use *GroupId* and *SourceSecurityGroupId* instead of *GroupName* and *SourceSecurityGroupName*

```
        "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.

```
"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.

```
"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"
   }
 }
}
```

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. 558) 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.

```
"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"}
  }
}
```

AWS Elastic Beanstalk Snippets

With AWS 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 AWS Elastic Beanstalk resources in your AWS CloudFormation template.

AWS Elastic Beanstalk Sample PHP

The following sample template deploys a sample PHP web application that is stored in an Amazon S3 bucket. The AWS 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.

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

Elastic Load Balancing Snippets

Topics

Elastic Load Balancing Load Balancer Resource

This example shows an Elastic Load Balancing load balancer with a single listener, and no instances.

```
"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.

```
"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. 65).

Declaring an IAM User Resource

This snippet shows how to declare an AWS::IAM::User (p. 394) 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 SQS actions on the SQS queue resource myqueue, and denies access to all other SQS queue resources. The Fn::GetAtt (p. 558) function gets the Arn attribute of the AWS::SQS::Queue (p. 458) resource myqueue.

The policy document named `giveaccesstotopiconly` is added to the user to give the user permission to perform all SNS actions on the SNS topic resource mytopic and to deny access to all other SNS resources. The Ref function (p. 565) gets the ARN of the AWS::SNS::Topic (p. 455) resource mytopic.

```

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

Declaring an IAM Access Key Resource

This snippet shows an AWS::IAM::AccessKey (p. 383) resource. The myaccesskey resource creates an access key and assigns it to an IAM user that is declared as an AWS::IAM::User (p. 394) resource in the template.

```
"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. 558) 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 by putting 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.

```
"AccessKeyformyaccesskey" : {
   "Value" : { "Ref" : "myaccesskey" }
},
"SecretKeyformyaccesskey" : {
   "Value" : {
      "Fn::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. 302) declaration uses the UserData property to pass the access key and secret key for the myaccesskey resource.

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

Declaring an IAM Group Resource

This snippet shows an AWS::IAM::Group (p. 384) 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 SQS actions on the SQS queue resource myqueue and deny access to all other 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. 558) function gets the ARN of the AWS::SQS::Queue (p. 458) resource queue.

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

Adding Users to a Group

The AWS::IAM::UserToGroupAddition (p. 395) resource adds users to a group. In the following snippet, the addUserToGroup resource adds the following users to an existing group named myexistinggroup2: an existing user existinguser1 and a user myuser that is declared as an AWS::IAM::User (p. 394) resource in the template.

```
"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. 387) 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. 384) resource. This example shows how 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.

```
"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. 453) resource. The mybucketpolicy resource declares a policy document that allows the group mygroup 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. 558) function gets the ARN of the mygroup resource. The mybucketpolicy resource applies the policy to the AWS::S3::Bucket (p. 446) resource mybucket. The Ref function (p. 565) gets the bucket name of the mybucket resource.

```

"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" : [ "mygroup", "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. 457) resource. The mysnspolicy resource contains a PolicyDocument property that allows an AWS::IAM::User (p. 394) resource myuser to perform the publish action on an AWS::SNS::Topic (p. 455) resource mytopic. In the snippet, the Fn::GetAtt (p. 558) function gets the ARN for the myuser resource and the Ref (p. 565) function gets the ARN for the mytopic resource.

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

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. 462) resource. The PolicyDocument property allows an existing user myapp (specified by its ARN) to perform the send message action on an existing queue, which is specified by its URL, and an AWS::SQS::Queue (p. 458) resource myqueue. The Ref (p. 565) function gets the URL for the myqueue resource.

```
"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

Example IAM Role with External Policy and Instance Profiles wired to an EC2 Instance

In this example, the Instance Profile is referenced by the IamInstanceProfile property of the EC2 Instance. Both the Instance Policy and Role Policy reference the AWS::IAM::Role (p. 390).

```
{
   "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

Example IAM Roles With External Policy And Instance Profiles Wired to an AutoScaling Group

In this example, the Instance Profile is referenced by the IamInstanceProfile property of an AutoScaling Group Launch Configuration.

```
{
   "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 OpsWorks 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.

```
{
  "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 alpha
numeric 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 alpha
```

```
numeric 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 alpha
numeric 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-cook
books.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","NumberOf
Disks":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 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`.

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

See Also

AWS::Redshift::Cluster (p. 413)

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. 423). 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.

```
"MyDB" : {
 "Type" : "AWS::RDS::DBInstance",
 "Properties" : {
     "DBSecurityGroups" : [
       {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" : "MyDbSecurityByCID
RIPGroup"} ],
     "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. 423). 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.

```
"MyDB" : {
 "Type" : "AWS::RDS::DBInstance",
 "Properties" : {
     "DBSecurityGroups" : [
       {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" : "MyDbSecurityByCID
RIPGroup"} ],
     "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. 435) and Amazon RDS Security Group Rule (p. 522).

```
"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. 435) 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.

```
"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": "WebServer
SecurityGroup" } },
      "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"}
      ]
   }
}
```

The full template from which this example is extracted can be seen at Drupal_Single_Instance_With_RDS.template

Multiple VPC security groups

This example shows an AWS::RDS::DBSecurityGroup (p. 435) resource with ingress authorization for multiple Amazon EC2 VPC security groups in AWS::RDS::DBSecurityGroupIngress (p. 437).

```
{
   "Resources" : {
      "DBinstance" : {
         "Type" : "AWS::RDS::DBInstance",
         "Properties" : {
            "AllocatedStorage" : "5",
            "DBInstanceClass" : "db.m1.small",
            "DBName" : { "MyDBName" },
            "DBSecurityGroups" : [ { "Ref" : "DbSecurityByEC2SecurityGroup" }
],
            "DBSubnetGroupName" : { "Ref" : "MyDBSubnetGroup" },
            "Engine" : "MySQL",
            "MasterUserPassword": { "MyDBPassword" }
            "MasterUsername"    : { "MyDBUsername" },
        },
         "DeletionPolicy" : "Snapshot"
      },
      "DbSecurityByEC2SecurityGroup" : {
         "Type" : "AWS::RDS::DBSecurityGroup",
         "Properties" : {
            "GroupDescription" : "Ingress for Amazon EC2 security group",
            "EC2VpcId" : { "MyVPC" },
            "DBSecurityGroupIngress" : [ {
               "EC2SecurityGroupId" : "sg-b0ff1111",
               "EC2SecurityGroupOwnerId" : "111122223333"
            }, {
               "EC2SecurityGroupId" : "sg-ffd722222",
               "EC2SecurityGroupOwnerId" : "111122223333"
            } ]
         }
      }
   }
}
```

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.

```
"myDNSRecord" : {
  "Type" : "AWS::Route53::RecordSet",
  "Properties" :
  {
    "HostedZoneId" : "/hostedzone/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.

```
"myDNSRecord2" : {
            "Type" : "AWS::Route53::RecordSet",
            "Properties" : {
                "HostedZoneName" : "example.com.",
                "Comment" : "A records for my frontends.",
                "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. 444) 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.

```
        "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. 444) 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. 523) property specifies the hosted zone ID and DNS name for the myELB LoadBalancer by using the GetAtt (p. 558) intrinsic function to retrieve the CanonicalHostedZoneNameID and CanonicalHostedZoneName 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*.

```
      "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", "CanonicalHosted
ZoneNameID"] },
                  "DNSName" : { "Fn::GetAtt" : ["myELB","CanonicalHostedZone
Name"] }
              }
            }
          ]
        }
    }
```

Amazon S3 Template Snippets

Topics

Creating an Amazon S3 Bucket with Defaults

This example uses a AWS::S3::Bucket (p. 446) to create a bucket with default settings.

```
"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. 539) 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.

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

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.`*`domain-name`*`.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*.

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Mappings" : {
        "RegionMap" : {
            "us-east-1" : { "S3hostedzoneID" : "Z3AQBSTGFYJSTF", "websiteend
point" : "s3-website-us-east-1.amazonaws.com" },
            "us-west-1" : { "S3hostedzoneID" : "Z2F56UZL2M1ACD", "websiteend
point" : "s3-website-us-west-1.amazonaws.com" },
            "us-west-2" : { "S3hostedzoneID" : "Z3BJ6K6RIION7M", "websiteend
point" : "s3-website-us-west-2.amazonaws.com" },
            "eu-west-1" : { "S3hostedzoneID" : "Z1BKCTXD74EZPE", "websiteend
point" : "s3-website-eu-west-1.amazonaws.com" },
            "ap-southeast-1" : { "S3hostedzoneID" : "Z3O0J2DXBE1FTB", "websit
eendpoint" : "s3-website-ap-southeast-1.amazonaws.com" },
            "ap-southeast-2" : { "S3hostedzoneID" : "Z1WCIGYICN2BYD", "websit
eendpoint" : "s3-website-ap-southeast-2.amazonaws.com" },
            "ap-northeast-1" : { "S3hostedzoneID" : "Z2M4EHUR26P7ZW", "websit
eendpoint" : "s3-website-ap-northeast-1.amazonaws.com" },
            "sa-east-1" : { "S3hostedzoneID" : "Z31GFT0UA1I2HV", "websiteend
point" : "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"
        }
    }
}
```

Amazon SimpleDB Snippets

Amazon SimpleDB Domain Resource

This example shows an Amazon SimpleDB domain resource.

```
"MySDBDomain" : {
    "Type" : "AWS::SDB::Domain",
    "Properties" : {
       "Description" : "Other than this AWS CloudFormation Description property,
 SDB Domains have no properties."
    }
}
```

Amazon SNS Snippets

Amazon SNS Topic Resource

This example shows an Amazon SNS topic resource. It requires a valid email address.

```
"MySNSTopic" : {
    "Type" : "AWS::SNS::Topic",
    "Properties" : {
        "Subscription" : [ {
            "Endpoint" : "add valid email address",
            "Protocol" : "email"
        } ]
    }
}
```

Amazon SQS Queue Snippet

This example shows an Amazon SQS queue.

```
"MyQueue" : {
    "Type" : "AWS::SQS::Queue",
    "Properties" : {
        "VisibilityTimeout" : "value"
    }
}
```

Stack Resource Snippets

Topics

Nesting a Stack in a Template

This example template contains an 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.*`nested-stackoutputname`* 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. 278).

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

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. 278).

```
{
    "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 Template Snippets

Topics

- Using a Wait Condition with an Amazon EC2 Instance (p. 208)
- Using Curl to signal a Wait Condition (p. 209)

> **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 has been successfully created.

Using a Wait Condition with an Amazon EC2 Instance

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. 540) 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. 220), AWS::CloudFormation::WaitCondition (p. 280), and AWS::CloudFormation::WaitConditionHandle (p. 282).

```
{
    "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."
        }
    }
}
```

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.amazon
aws.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=AKI
AIOSFODNN7EXAMPLE&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 com
pleted configuration."}'  https://cloudformation-waitcondition-test.s3.amazon
aws.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=AKI
AIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wyOoRejVoo%3D
```

AWS CloudFormation Template Snippets

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*.

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

```
"UserData" : {
    "Fn::Base64" : {
        "Fn::Join" : [ "", [
            "ACCESS_KEY=", { "Ref" : "AccessKey" },
            "SECRET_KEY=", { "Ref" : "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.

```
"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.

```
"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.

```
"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.

```
"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.

```
"Parameters" : {
    "UserRoles" : {
        "Type" : "CommaDelimitedList",
        "Default" : "guest,newhire",
        "NoEcho" : "TRUE"
    }
}
```

Parameters Section with Parameter Value Based on Pseudo Parameter

This example shows a parameter assignment based on the value returned from the pseudo parameter, "AWS::StackName".

```
"Parameters" : {
    "StackName" : {
        "Type" : "String",
        "Default" : { "Ref" : "AWS::StackName"}
    }
},
```

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.

```
"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.

```
"Description" : "Replace this value"
```

Outputs Section with One Literal String Output

This example shows a output assignment based on a literal string.

```
"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.

```

"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.

```
"Outputs" : {
    "MyOutput" : {
        "Value" : { "Fn::Join" :
            [ "%", [ "A-string", {"Ref" : "AWS::StackName" } ] ]
        }
    }
}
```

Template Format Version

The following snippet depicts a valid Template Format Version section declaration.

```
"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.

```
"Tags" : [
      {
        "Key" : "keyname1",
        "Value" : "value1"
      },
      {
        "Key" : "keyname2",
        "Value" : "value2"
      }
    ]
},
```

Creating Templates

Topics

- Creating Wait Conditions in a Template (p. 220)
- Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 224)

Specifying Intrinsic Functions

AWS CloudFormation intrinsic functions are special actions you use in your template to assign values to properties not available until runtime. Each function is declared with a double-quoted name, a single colon, and its parameters. When an argument is a literal string, it is enclosed in double quotes (**""**). When arguments are in a list of any kind, they are enclosed in brackets (**[ ]**). If an argument is a value that is returned from an intrinsic function, it is enclosed in braces (**{ }**).

The following example shows the function "Fn::GetAtt" being used to assign a value to the *MyLBDNSName*, which it does by retrieving the value of the attribute *DNSName* from the Elastic Load Balancing load balancer named *MyLoadBalancer*.

```
"Properties" : {
   "MyMyLBDNSName" : {
      "Fn::GetAtt" : [ "MyLoadBalancer", "DNSName" ]
   }
}
```

For more information about intrinsic functions, see Intrinsic Function Reference (p. 546).

Adding Input Parameters to Your Template

You can configure your templates to require input parameters by adding them to the Parameters section. Each parameter you add must contain a value at runtime. You can specify a default value for each parameter to make the parameter optional. If you do not specify a default value, you must provide a value for that parameter when you create the stack.

A parameter can be declared as a *String*, *Number*, *CommaDelimitedList*, or AWS-specific type. The *String*, *Number*, and AWS-specific types can have constraints that AWS CloudFormation uses to validate the value of the parameter. For more information about parameter constraints, see Parameters (p. 116).

The following sample configures a single parameter, *Email*:

```
"Parameters" : {
    "Email" : {
        "Type" : "String"
    }
}
```

The parameter has no default, so you must provide a value to create the stack. After you create the CloudWatch Alarms stack with a value for *Email*, the `aws cloudformation describe-stacks` command returns the following:

```
STACK  myAlarms
arn:aws:aws cloudformation:us-east-1:165024647323:stack/f5b4cbb0-24d7-11e0-93a-
508be05d086/myAlarms
Email=Joe@Joe.com  2011-01-20T20:57:57Z  CREATE_COMPLETE
User Initiated  false  Instance=i-0723826b
```

You can configure the parameter to not display with the *NoEcho* parameter:

```
"Parameters" : {
    "Email" : {
        "Type" : "String",
        "NoEcho" : "TRUE"
    }
}
```

Here's the output from a stack created with the same template, but with the *NoEcho* set to *TRUE*:

```
STACK  myAlarms2
arn:aws:aws cloudformation:us-east-1:165024647323:stack/ff6ff540-24db-11e0-94f8-
5081b017c4b/myAlarms2
Email=******  2011-01-20T21:26:52Z  CREATE_COMPLETE  User Initiated
false  Instance=i-f734959b
```

The value for *Email* is masked with asterisks.

To supply the values for parameters, you include the `--parameters` option to the `aws cloudformation create-stack` command.

For example, the following command adds a value for the *UserName* and *Password* parameters:

```
PROMPT> aws cloudformation create-stack --stack-name MyStack --template-body
file:///home/local/test/sampletemplate.json
--parameters ParameterKey=UserName,ParameterValue=Joe ParameterKey=Password,Para
meterValue=JoesPw
```

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

Validate AWS-Specific Values

For some AWS values, such as Amazon EC2 key pair names and VPC IDs, you can use AWS-specific parameter types to validate input parameter values against existing values in users' AWS accounts. For example, you can use the `AWS::EC2::KeyPair::KeyName` parameter type to ensure that users specify a valid key pair name before AWS CloudFormation creates or updates any resources. AWS-specific parameter types are helpful in catching invalid values early. For more information, see Parameters (p. 116).

Use Parameters and Mappings to Specify Values in Your Template

You can use an input parameter to refer to a specific value in a map by using the `Fn::FindInMap` function. For example, suppose you have a list of regions that map to a specific AMI. You can select the AMI that your stack uses by specifying a region parameter when you create the stack.

1. Add one parameter to your Parameters section for every mapping you want to include. The parameter is how you pass in the desired mapping key.
2. Create the mappings that contain the key options and key values.
3. Use the `Fn::FindInMap` function as the value for the resource property or output you want to assign conditionally.

Note

When you use input parameters for keys and values in the `Fn::FindInMap` function, set default values for those parameters. Otherwise, if the parameters in the `Fn::FindInMap` function are not defined, stack creation fails.

Consider this example. Suppose you want the `aws cloudformation describe-stacks` command to print the AMI name of the AMI you want to run based on a particular region. You could do this with the following:

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

  "Description" : "TemplateName - ShortMapExample.template",

  "Parameters" : {
    "Region" : {
      "Default" : "us-east-1",
     "Description" : " 'us-east-1' | 'us-west-1' | 'eu-west-1' | 'ap-southeast-
1' "
    }
  },

  "Mappings" : {
    "RegionMap" : {
      "us-east-1" : {
          "AMI" : "ami-76f0061f"
      },
      "us-west-1" : {
          "AMI" : "ami-655a0a20"
      },
      "eu-west-1" : {
          "AMI" : "ami-7fd4e10b"
      },
      "ap-southeast-1" : {
          "AMI" : "ami-72621c20"
      }
    }
  },

  "Resources" : {

   ...other resources...

    },

  "Outputs" : {
    "OutVal" : {
      "Description" : "Return the name of the AMI matching the RegionMap key",

      "Value" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "Region" }, "AMI"
 ]}
    }
  }

}
```

The parameter *Region* accepts a string value, ideally one of the region identifiers in the template. The Mappings section declares the *RegionMap* mapping. Each mapping key assigns a value to the *AMI* attribute. The Outputs section declares the *OutVal* output, which gets its value based on the value returned from `Fn::FindInMap`.

The following shows the value assigned to *OutVal* based on the listed command:

Command Lines	Value Assigned to OutVal

`aws cloudformation create-stack --stack-name MyTestStack --template-body file:///home/local/test/ShortRegionExample.json --parameters ParameterKey=Region,ParameterValue=us-west-1`
`...`
`aws cloudformation describe-stacks --stack-name MyTestStack`
	ami-655a0a20

`aws cloudformation create-stack --stack-name MyTestStack --template-body file:///home/local/test/ShortRegionExample.json --parameters ParameterKey=Region,ParameterValue=eu-west-1`
`...`
`aws cloudformation describe-stacks --stack-name MyTestStack`
	ami-7fd4e10b

`aws cloudformation create-stack --stack-name MyTestStack --template-body file:///home/local/test/ShortRegionExample.json`
`...`
`aws cloudformation describe-stacks MyTestStack`
	ami-76f0061f

In the first two cases, the value specified as part of the *--parameters* option determines the value of *OutVal*. In the third example, a mapping key is not specified, so the default region, *us-east-1*, will be used.

Conditionally Creating Resources

When you create or update a stack, you can create resources conditioned on input parameters and mappings. You can set up multiple conditions with different outcomes for each. For example, you can specify an Amazon EC2 security group as an input parameter and use that security group in your stack. However, if a security group isn't provided, a security group that you specified in the template is created.

You can conditionally create resources by completing the following steps:

1. In the Parameters section of the template, define input parameters that you can use in your conditions. For more information, see Adding Input Parameters to Your Template (p. 215).
2. In the Conditions section of the template, define the conditions that you want to use by using the intrinsic functions for conditions. For more information, see Conditions (p. 124).

3. In the Resources and Outputs sections of the template, associate conditions with related resources or properties. For more information, see Conditions (p. 124).

For additional sample templates and information about the syntax of conditions, see Condition Functions (p. 547).

Tagging Your Member Resources

AWS CloudFormation automatically tags your resources with the stack name that you can filter on when viewing those resources in the AWS Management Console.

In addition to the stack name tags that AWS CloudFormation adds for you, you can add custom tags to the resources that support tagging.

> **Note**
>
> Tags you add to a member resource do not appear in the output from `aws cloudformation describe-stack-resources`. However, they do appear in the AWS Management Console on the tab for the tagged resource.

Suppose you wanted to customize a template to include the tag *`Stage`* for deployment stage, and *`QA`* for its value. You could write the definition for the *`MyInstance`* resource as follows:

```
"MyInstance" : {
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
        "SecurityGroups" : [ { "Ref" : "MySecurityGroup" } ],
        "AvailabilityZone" : "us-east-1a",
        "ImageId" : "ami-20b65349",
        "Volumes" : [
            { "VolumeId" : { "Ref" : "MyEBS" },
                       "Device" : "/dev/sdk" }
        ],
        "Tags" : [
            {
                "Key" : "Stage",
                "Value" : "QA"
            }
       ]
    }
}
```

After you created the stack, you could then filter on the *`Stage`* tag in the AWS Management Console.

Specifying Output Values

You can use the template Outputs section to specify custom values that are included in the values returned by `aws cloudformation describe-stacks` command. You specify each custom value according to template property rules (Resources (p. 126)), so you can base their value on literals, parameter references, pseudo parameters, mapping value, and intrinsic functions.

For a simple example, a sample template declares two outputs, *`IPAddress`* and *`InstanceId`*:

```
"Outputs" : {
    "IPAddress" : {
        "Value" : { "Ref" : "MyIp" }
    },

    "InstanceId" : {
        "Value" : { "Ref" : "MyInstance" }
    }
}
```

Both values are based on logical names declared within the template. *`IPAddress`* refers to the AWS::EC2::EIP type with the logical name *`MyIp`*, and *`InstanceId`* refers to the AWS::EC2::Instance type with the logical name *`MyInstance`*.

After the stack is created, and `aws cloudformation describe-stacks` reports its status as being `CREATE_COMPLETE`, it also reports the following:

```
PROMPT> aws cloudformation describe-stacks --stack-name StackName
...
    "Outputs": [
        {
            "OutputKey": "IPAddress",
            "OutputValue": "184.72.229.56"
        },
        {
            "OutputKey": "InstanceId",
            "OutputValue": "i-47ab0a2b"
        }
    ],
...
```

The custom output values `IPAddress` and `InstanceId` are present at the end of the report.

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 has been successfully created.
> For more information, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 224).

Using the AWS::CloudFormation::WaitCondition (p. 280) resource and CreationPolicy (p. 537) 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

> **Important**
> Generally it's best to use the CreationPolicy (p. 537) attribute instead of a wait condition handle. However, if for some reason you cannot use the creation policy attribute, you can still use a wait condition and wait condition handle.

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.
- 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. 540) 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. 223). For an example of a Curl command that sends a JSON statement to a presigned URL, see Wait Condition Template Snippets (p. 208).
- 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. 223).

   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.

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.amazon
aws.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&AWSAccess
KeyId=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-Wait
Condition--VEYW%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitCondition
Handle?Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signa
ture=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. 558) 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.

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.

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" : [ "WebServer
Instance", "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,'ht
tp://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,'ht
tp://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" : ["", [
             "yum update -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"
  ]]}}
      }
    },


    "WebServerSecurityGroup" : { ...  }
  },

  "Outputs" : { ... }
}
```

The `UserData` property runs two shell commands: install the AWS CloudFormation helper scripts and then run the cfn-init (p. 570) helper script. When you run cfn-init, it reads metadata from the AWS::CloudFormation::Init (p. 268) 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 al
phanumeric 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 al
phanumeric 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::CloudForma
tion::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 update -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 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. 577) 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 is installation 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. 537) 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. 573) 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 3 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 al
phanumeric 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 al
phanumeric 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,'ht
tp://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,'ht
tp://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::CloudForma
tion::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 update -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" : [ "WebServer
Instance", "PublicDnsName" ]}]] }
    }
  }
}
```

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. 116).

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.

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::CloudFront::Distribution (p. 283)
- AWS::CloudTrail::Trail (p. 284)
- AWS::CloudWatch::Alarm (p. 287)
- AWS::DynamoDB::Table (p. 291)
- AWS::EC2::CustomerGateway (p. 295)
- AWS::EC2::DHCPOptions (p. 297)
- AWS::EC2::EIP (p. 299)
- AWS::EC2::EIPAssociation (p. 301)
- AWS::EC2::Instance (p. 302)
- AWS::EC2::InternetGateway (p. 309)
- AWS::EC2::NetworkAcl (p. 310)
- AWS::EC2::NetworkAclEntry (p. 311)
- AWS::EC2::NetworkInterface (p. 313)
- AWS::EC2::NetworkInterfaceAttachment (p. 317)
- AWS::EC2::Route (p. 318)
- AWS::EC2::RouteTable (p. 321)
- AWS::EC2::SecurityGroup (p. 322)
- AWS::EC2::SecurityGroupEgress (p. 325)
- AWS::EC2::SecurityGroupIngress (p. 328)
- AWS::EC2::Subnet (p. 332)
- AWS::EC2::SubnetNetworkAclAssociation (p. 334)
- AWS::EC2::SubnetRouteTableAssociation (p. 335)
- AWS::EC2::Volume (p. 337)
- AWS::EC2::VolumeAttachment (p. 339)
- AWS::EC2::VPC (p. 341)
- AWS::EC2::VPCDHCPOptionsAssociation (p. 343)
- AWS::EC2::VPCGatewayAttachment (p. 344)
- AWS::EC2::VPCPeeringConnection (p. 346)
- AWS::EC2::VPNConnection (p. 354)
- AWS::EC2::VPNConnectionRoute (p. 356)
- AWS::EC2::VPNGateway (p. 357)
- AWS::EC2::VPNGatewayRoutePropagation (p. 358)
- AWS::ElastiCache::CacheCluster (p. 360)
- AWS::ElastiCache::ParameterGroup (p. 364)
- AWS::ElastiCache::SecurityGroup (p. 366)
- AWS::ElastiCache::SecurityGroupIngress (p. 366)
- AWS::ElastiCache::SubnetGroup (p. 367)
- AWS::ElasticBeanstalk::Application (p. 368)
- AWS::ElasticBeanstalk::ApplicationVersion (p. 369)
- AWS::ElasticBeanstalk::ConfigurationTemplate (p. 371)
- AWS::ElasticBeanstalk::Environment (p. 373)
- AWS::ElasticLoadBalancing::LoadBalancer (p. 376)
- AWS::IAM::AccessKey (p. 383)
- AWS::IAM::Group (p. 384)
- AWS::IAM::InstanceProfile (p. 385)
- AWS::IAM::Policy (p. 387)

AWS::AutoScaling::AutoScalingGroup

The AWS::AutoScaling::AutoScalingGroup type creates an Auto Scaling group.

You can add an UpdatePolicy (p. 543) 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. 251) or subnet group membership (p. 249).

Syntax

```
{
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Properties" : {
      "AvailabilityZones (p. 246)" : [ String, ... ],
      "Cooldown (p. 246)" : String,
      "DesiredCapacity (p. 246)" : String,
      "HealthCheckGracePeriod (p. 246)" : Integer,
      "HealthCheckType (p. 247)" : String,
      "InstanceId (p. 247)" : String,
      "LaunchConfigurationName (p. 247)" : String,
      "LoadBalancerNames (p. 247)" : [ String, ... ],
      "MaxSize (p. 248)" : String,
      "MetricsCollection (p. 248)" : [ MetricsCollection, ... ]
      "MinSize (p. 248)" : String,
      "NotificationConfiguration (p. 248)" : NotificationConfiguration,
      "PlacementGroup (p. 248)" : String,
      "Tags (p. 248)" : [ Auto Scaling Tag, ..., ],
      "TerminationPolicies (p. 248)" : [ String, ..., ],
      "VPCZoneIdentifier (p. 249)" : [ String, ... ]
   }
}
```

Properties

AvailabilityZones

Contains a list of availability zones for the group.

Required: Yes

Type: A list of strings

Update requires: No interruption (p. 88)

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. 88)

DesiredCapacity

Specifies the desired capacity for the Auto Scaling group.

If *SpotPrice* is not set in the AWS::AutoScaling::LaunchConfiguration (p. 251) 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. 88)

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. 88)

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. 88)

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. 88)

LaunchConfigurationName

Specifies the name of the associated AWS::AutoScaling::LaunchConfiguration (p. 251).

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. 540).

Required: Conditional; you must specify this property if you don't specify the `InstanceId` property.

Type: String

Update requires: No interruption (p. 88)

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. 543).

LoadBalancerNames

A list of load balancers associated with this Auto Scaling group.

Required: No

Type: A list of strings

Update requires: Replacement (p. 88)

Important

When you update LoadBalancerNames, the entire Auto Scaling group is replaced.

MaxSize
: The maximum size of the Auto Scaling group.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

MetricsCollection
: Enables the monitoring of group metrics of an Auto Scaling group.

  Required: No

  Type: A list of Auto Scaling MetricsCollection (p. 467)

  Update requires: No interruption (p. 88)

MinSize
: The minimum size of the Auto Scaling group.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

NotificationConfiguration
: An embedded property that configures an Auto Scaling group to send notifications when specified events take place.

  Required: No

  Type: NotificationConfiguration (p. 467)

  Update requires: No interruption (p. 88)

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. 88)

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 Developer Guide*.

  Required: No

  Type: List of Auto Scaling Tags (p. 468)

  Update requires: No interruption (p. 88)

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 Developer Guide*.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

VPCZoneIdentifier

A list of subnet identifiers of Amazon Virtual Private Cloud (Amazon VPCs).

The subnets that you specify for *`VPCZoneIdentifier`* must reside in the Availability Zones that you specify with the *`AvailabilityZones`* parameter.

For more information, go to Using EC2 Dedicated Instances Within Your VPC in the *Auto Scaling Developer Guide*.

Required: No

Type: A list of strings

Update requires: Some interruptions (p. 88)

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, it returns the resource name. For example:

```
{ "Ref": "MyASGroup" }
```

For an Auto Scaling group with the logical ID `"MyASGroup"`, `Ref` will return:

`mystack-myasgroup-NT5EUXTNTXXD`

For more information about using the `Ref` function, see Ref (p. 565).

Examples

To view more Auto Scaling examples, see Auto Scaling Snippets (p. 151).

Auto Scaling Group with an Elastic Load Balancing Load Balancer, Launch Configuration, and Metric Collection

```
"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. 543) 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. 543) attribute, the `WaitOnResourceSignals` flag is set to `true`. You can use the cfn-signal (p. 573) helper script on each instance to signal the Auto Scaling group.

```
"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. 543)
- UpdateAutoScalingGroup in the *Auto Scaling API Reference*
- AWS CloudFormation Stacks Updates (p. 88)

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. 543).

Syntax

```
{
   "Type" : "AWS::AutoScaling::LaunchConfiguration",
   "Properties" : {
      "AssociatePublicIpAddress (p. 252)" : Boolean,
      "BlockDeviceMappings (p. 252)" : [ BlockDeviceMapping, ... ],
      "EbsOptimized (p. 252)" : Boolean,
      "IamInstanceProfile (p. 252)" : String,
      "ImageId (p. 252)" : String,
      "InstanceId (p. 253)" : String,
      "InstanceMonitoring (p. 253)" : Boolean,
      "InstanceType (p. 253)" : String,
      "KernelId (p. 253)" : String,
      "KeyName (p. 253)" : String,
      "RamDiskId (p. 253)" : String,
      "SecurityGroups (p. 253)" : [ SecurityGroup, ... ],
      "SpotPrice (p. 254)" : String,
      "UserData (p. 254)" : 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. 540).

Required: No

Type: Boolean

Update requires: Replacement (p. 88)

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. 465).

Update requires: Replacement (p. 88)

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 Elastic Compute Cloud User Guide*.

Required: No If this property is not specified, "false" is used.

Type: Boolean

Update requires: Replacement (p. 88)

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. 88)

ImageId

Provides the unique ID of the Amazon Machine Image (AMI) that was assigned during registration.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

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. 88)

InstanceMonitoring

Indicates whether or not instance monitoring should be enabled for this autoscaling group. This is enabled by default. To turn it off, set *`InstanceMonitoring`* to "false".

Required: No Default value is "true".

Type: Boolean

Update requires: Replacement (p. 88)

InstanceType

Specifies the instance type of the EC2 instance.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

KernelId

Provides the ID of the kernel associated with the EC2 AMI.

Required: No

Type: String

Update requires: Replacement (p. 88)

KeyName

Provides the name of the EC2 key pair.

Required: No

Type: String

Update requires: Replacement (p. 88)

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. 88)

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. 88)

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. 88)

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. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name. For example:

```
{ "Ref": "MyAutoScalingGroup" }
```

For the resource with the logical ID "MyAutoScalingGroup", `Ref` will return the AutoScaling launch config name, such as: `mystack-mylaunchconfig-1DDYF1E3B3I`.

For more information about using the `Ref` function, see Ref (p. 565).

Template Examples

Example LaunchConfig with block device

This example shows a launch configuration that describes two Amazon Elastic Block Store mappings.

```
"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).

```
"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. 252) 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. 385) object referenced here as "RootInstanceProfile", see: auto_scaling_with_instance_profile.template.

```
"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:

```
"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. 151).

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::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 an CloudWatch alarm. An 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 an CloudWatch alarm, see Auto Scaling Policy Triggered by CloudWatch Alarm (p. 152).

This type supports updates. For more information about updating this resource, see PutScalingPolicy. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

Syntax

```
{
   "Type" : "AWS::AutoScaling::ScalingPolicy",
   "Properties" : {
      "AdjustmentType (p. 258)" : String,
      "AutoScalingGroupName (p. 258)" : String,
      "Cooldown (p. 258)" : String,
      "ScalingAdjustment (p. 258)" : String
   }
}
```

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. 88)

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. 88)

Cooldown

The amount of time, in seconds, after a scaling activity completes before any further trigger-related scaling activities can start.

Required: No

Type: String

Update requires: No interruption (p. 88)

ScalingAdjustment

The number of instances by which to scale. AdjustmentType determines the interpretation of this number (e.g., as an absolute number or as a percentage of the existing Auto Scaling group size). A positive increment adds to the current capacity and a negative value removes from the current capacity.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Return Value

When you specify an AWS::AutoScaling::ScalingPolicy type as an argument to the `Ref` function, AWS CloudFormation returns the policy name.

For more information about using the `Ref` function, see Ref (p. 565).

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 Developer 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. 543) 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.

Syntax

```
{
  "Type" : "AWS::AutoScaling::ScheduledAction",
  "Properties" : {
    "AutoScalingGroupName (p. 259)" : String,
    "DesiredCapacity (p. 259)" : Integer,
    "EndTime (p. 260)" : Time stamp,
    "MaxSize (p. 260)" : Integer,
    "MinSize (p. 260)" : Integer,
    "Recurrence (p. 260)" : String,
    "StartTime (p. 260)" : Time stamp,
  }
}
```

Properties

AutoScalingGroupName

The name or ARN of the Auto Scaling group.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

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. 88)

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. 88)

MaxSize

The maximum number of Amazon EC2 instances in the Auto Scaling group.

Required: No

Type: Integer

Update requires: No interruption (p. 88)

MinSize

The minimum number of Amazon EC2 instances in the Auto Scaling group.

Required: No

Type: Integer

Update requires: No interruption (p. 88)

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. 88)

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. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

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 5 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 1 Amazon EC2 instance.

```
"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::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. 268) 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. 272). For more information about sources, see Sources (p. 277).

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.

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. 262) property for more details.

```
{
   "Type" : "AWS::CloudFormation::Authentication" {
      "String" : {
         "accessKeyId (p. 262)" : String,
         "buckets (p. 262)" : [ String, ... ],
         "password (p. 262)" : String,
         "secretKey (p. 262)" : String,
         "type (p. 262)" : String,
         "uris (p. 263)" : [ String, ... ],
         "username (p. 263)" : String,
         "roleName (p. 263)" : 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: A 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"`, you must also specify the `username`, `password`, and `uris` properties.

If you specify `"S3"`, you must also specify the `accessKeyId`, `secretKey`, and `buckets` 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: A 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.

Required: Conditional Can be specified only if the type property is set to `"S3"`.

Type: String.

Examples

Example 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.

```
"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 ...
   }
}
```

Example Specifying Both Basic and S3 Authentication

The following example template snippet includes both *basic* and *S3* authentication types.

```
"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" ]
   }
}
```

Example IAM Roles

The following example shows how to use IAM roles.

```
"AWS::CloudFormation::Authentication": {
    "rolebased" : {
        "type": "s3",
        "buckets": [ "myBucket" ],
        "roleName": { "Ref": "myRole" }
    }
}
```

The example assumes the following:

- `myRole` is an AWS::IAM::Role (p. 390) resource.
- The Amazon EC2 instance that is running cfn-init is associated with `myRole` through an instance profile.
- The example specifies the authentication by using the buckets property, like normal Amazon S3 authentication. You can also specify the authentication by name.

Full Template Examples

For full template samples that feature the AWS::CloudFormation::Authentication resource, view the following templates on the AWS CloudFormation Sample Templates web page:

- S3Bucket_Auth_1.template
- S3Bucket_Auth_2.template
- S3Bucket_SourceAuth.template

AWS::CloudFormation::CustomResource

Custom resources are special AWS CloudFormation resources that provide a way for a template developer to include non-AWS resources in an AWS CloudFormation stack. The custom resource provider can be either a template developer or a separate third-party resource provider.

In a template, a custom resource is represented by either `AWS::CloudFormation::CustomResource` or `Custom::`*`String`* (a custom resource type name).

Syntax

```
{
   "Type" : "AWS::CloudFormation::CustomResource",
   "Version" : "1.0",
   "Properties" : {
      "ServiceToken (p. 266)" : String,
      ... provider-defined properties ...
   }
}
```

or

```
{
   "Type" : "Custom::String",
   "Version" : "1.0",
   "Properties" : {
      "ServiceToken (p. 266)" : String,
      ... provider-defined properties ...
   }
}
```

> **Note**
> Only one property is defined by AWS for a custom resource: `ServiceToken`. All other properties are defined by the service provider.

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

ServiceToken

The service token that was given to the template developer by the service provider to access the service.

Required: Yes

Type: String

Return Values

For a custom resource, return values are defined by the custom resource provider, and are retrieved by calling `Fn::GetAtt` (p. 558) 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.

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MyFrontEndTest" : {
         "Type": "AWS::CloudFormation::CustomResource",
         "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"] }
      }
   }
}
```

All properties other than `ServiceToken`, and all `Fn::GetAtt` resource attributes, are defined by the custom resource provider.

Creating a user-defined resource type for a custom resource

The following example demonstrates how to create a type name for a custom resource.

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

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. 50).

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. 77).
- For more information about resource behavior during updates, see AWS CloudFormation Stacks Updates (p. 88).

See Also

- AWS CloudFormation Custom Resource Walkthrough (p. 47)

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. 570).

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*.

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.

```
"Resources": {
  "MyInstance": {
    "Type": "AWS::EC2::Instance",
    "Metadata" : {
      "AWS::CloudFormation::Init" : {
        "config" : {
          "packages" : {
            :
          },
          "groups" : {
            :
          },
          "users" : {
            :
          },
          "sources" : {
            :
          },
          "files" : {
            :
          },
          "commands" : {
            :
          },
          "services" : {
            :
          }
        }
      }
    },
    "Properties": {
      :
    }
  }
}
```

> **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.

`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. 224).

For an example that shows how to use cfn-init to create a Windows stack, see Bootstrapping AWS CloudFormation Windows Stacks (p. 106).

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. For example, the following template snippet creates configsets named `ascending` and `descending` that each contain two config keys.

```
"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" : "~"
            }
        }
    }
}
```

The following example calls to cfn-init refer to the preceding example configsets. The example calls are abbreviated for clarity, see cfn-init (p. 570) 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`.

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`.

```
"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" : "~"
            }
        }
    }
}
```

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. 570) 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.
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 command that must return the value `true` in order for cfn-init to process the command contained in the command key.
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.

The following example snippet calls the echo command.

```
"commands" : {
    "test" : {
        "command" : "echo \"$MAGIC\" > test.txt",
        "env" : { "MAGIC" : "I come from the environment!" },
        "cwd" : "~",
        "test" : "test ! -e ~/test.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.
source	A URL to load the file from. This option cannot be specified with the content key.

Key	Description
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.
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. 261) 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.

The following example snippet creates a file named setup.mysql as part of a larger installation.

```
"files" : {
  "/tmp/setup.mysql" : {
    "content" : { "Fn::Join" : ["", [
      "CREATE DATABASE ", { "Ref" : "DBName" }, ";\n",
      "CREATE USER '", { "Ref" : "DBUsername" }, "'@'localhost' IDENTIFIED BY
'",
                       { "Ref" : "DBPassword" }, "';\n",
      "GRANT ALL ON ", { "Ref" : "DBName" }, ".* TO '", { "Ref" : "DBUsername"
 },
                       "'@'localhost';\n",
      "FLUSH PRIVILEGES;\n"
      ]]},
    "mode"  : "000644",
    "owner" : "root",
    "group" : "root"
  }
},
```

The full template is available at: https://s3.amazonaws.com/cloudformation-templates-us-east-1/Drupal_Single_Instance.template

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. 563). The following example snippet outputs "Content for test9" to `/tmp/test9.txt`.

```
"files" : {
    "/tmp/test9.txt" : {
        "content" : "Content for {{name}}",
        "context" : { "name" : "test9" }
    }
}
```

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`.

```
"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

Cfn-init 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

The following snippet specifies a version URL for rpm, requests the latest versions from yum, and version 0.10.2 of chef from rubygems:

```
"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" ]
}
```

The following snippet specifies a URL for an MSI package:

```
"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
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.

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.

```
"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"
    }
  }
}
```

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:

```
"services" : {
  "windows" : {
    "cfn-hup" : {
      "enabled" : "true",
      "ensureRunning" : "true",
      "files" : ["c:\\cfn\\cfn-hup.conf", "c:\\cfn\\hooks.d\\cfn-auto-reload
er.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.

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.

```
"sources" : {
  "/etc/puppet" : https://github.com/user1/cfn-demo/tarball/master
  }
```

Example

The following example downloads a zip file from an Amazon S3 bucket and unpacks it into /etc/myapp:

```
"sources" : {
  "/etc/myapp" : "https://s3.amazonaws.com/mybucket/myapp.tar.gz"
  }
```

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. 261).

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.

Users are created as non-interactive system users with a shell of /sbin/nologin. This is by design and cannot be modified.

```
"users" : {
    "myUser" : {
        "groups" : ["groupOne", "groupTwo"],
        "uid" : "50",
        "homeDir" : "/tmp"
    }
}
```

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. 558) 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. 88).

AWS::CloudFormation::Stack Snippets: Stack Resource Snippets (p. 206).

> **Note**
> Nested stacks require that you acknowledge IAM capabilities even if the nested stack doesn't contain any IAM resources. For more information about acknowledging IAM capabilities, see IAM Resources in AWS CloudFormation Templates in Controlling Access with AWS Identity and Access Management (p. 65).

Syntax

```
{
   "Type" : "AWS::CloudFormation::Stack",
   "Properties" : {
      "NotificationARNs (p. 279)" : [ String, ... ],
      "Parameters (p. 279)" : { CloudFormation Stack Parameters Property
Type (p. 469) },
      "TemplateURL (p. 279)" : String,
      "TimeoutInMinutes (p. 279)" : String
   }
}
```

Properties

NotificationARNs

A list of existing Amazon SNS topics where notifications about stack events are sent.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

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. 469)

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.

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: String

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-sgg
frhxhum7w/f449b250-b969-11e0-a185-5081d0136786
```

For more information about using the `Ref` function, see Ref (p. 565).

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. 558).

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 has been successfully created.

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. 537) 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. 220). If you use a CreationPolicy with a wait condition, do not specify any of the wait condition's properties.

Syntax

```
{
   "Type" : "AWS::CloudFormation::WaitCondition",
   "Properties" : {
      "Count (p. 280)" : String,
      "Handle (p. 281)" : String,
      "Timeout (p. 281)" : 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: String

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. 282) 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, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 223).

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. 558).

Examples

Example WaitCondition that waits for the desired number of instances in a web server group

```
"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. 220)
- DependsOn Attribute (p. 540)

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 has been successfully created.
> For more information, see Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 224).

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. 280) 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` re-

source. 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.

Syntax

```
{
   "Type" : "AWS::CloudFormation::WaitConditionHandle",
   "Properties" : {
   }
}
```

Note
Updates are not supported for this resource.

Related Resources

- For information about how to use wait conditions, see Creating Wait Conditions in a Template (p. 220).
- For a AWS::CloudFormation::WaitCondition snippet, see Wait Condition Template Snippets (p. 208)

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 POST Distribution in the *Amazon CloudFront API Reference*.

Syntax

```
{
   "Type" : "AWS::CloudFront::Distribution",
   "Properties" : {
      "DistributionConfig (p. 283)" : DistributionConfig
   }
}
```

Properties

DistributionConfig
The distribution's configuration information.

Required: Yes

Type: DistributionConfig (p. 470) type

Update requires: No interruption (p. 88)

Return Values

Ref

Returns: The CloudFront distribution ID. For example: `E27LVI50CSW06W`.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

AWS::CloudTrail::Trail

The `AWS::CloudTrail::Trail` resource creates a trail and specifies where logs are published. A CloudTrail trail can capture AWS API calls made by your AWS account and publishes the logs to an Amazon S3 bucket.

Syntax

```
{
  "Type" : "AWS::CloudTrail::Trail",
  "Properties" : {
    "IncludeGlobalServiceEvents (p. 284)" : Boolean,
    "IsLogging (p. 284)" : Boolean,
    "S3BucketName (p. 285)" : String,
    "S3KeyPrefix (p. 285)" : String,
    "SnsTopicName (p. 285)" : String
  }
}
```

Properties

IncludeGlobalServiceEvents
Indicates whether the trail is publishing events from global services, such as IAM, to the log files.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

IsLogging
Indicates whether the CloudTrail trail is currently logging AWS API calls.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 88)

S3BucketName

The name of the Amazon S3 bucket where CloudTrail publishes log files.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

S3KeyPrefix

An Amazon S3 object key prefix that precedes the name of all log files.

Required: No

Type: String

Update requires: No interruption (p. 88)

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

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 all 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.

```
{
  "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": {
                "AWS": [
                  "arn:aws:iam::903692715234:root",
                  "arn:aws:iam::859597730677:root",
                  "arn:aws:iam::814480443879:root",
                  "arn:aws:iam::216624486486:root",
                  "arn:aws:iam::086441151436:root",
                  "arn:aws:iam::388731089494:root",
                  "arn:aws:iam::284668455005:root",
                  "arn:aws:iam::113285607260:root"
                ]
              },
              "Action": "s3:GetBucketAcl",
              "Resource": { "Fn::Join" : ["", ["arn:aws:s3:::",
{"Ref":"S3Bucket"}]]}
            },
            {
              "Sid": "AWSCloudTrailWrite",
              "Effect": "Allow",
              "Principal": {
                "AWS": [
                  "arn:aws:iam::903692715234:root",
                  "arn:aws:iam::859597730677:root",
                  "arn:aws:iam::814480443879:root",
                  "arn:aws:iam::216624486486:root",
                  "arn:aws:iam::086441151436:root",
                  "arn:aws:iam::388731089494:root",
                  "arn:aws:iam::284668455005:root",
                  "arn:aws:iam::113285607260:root"
                ]
              },
              "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": {
                "AWS": [
                  "arn:aws:iam::903692715234:root",
                  "arn:aws:iam::859597730677:root",
                  "arn:aws:iam::814480443879:root",
                  "arn:aws:iam::216624486486:root",
                  "arn:aws:iam::086441151436:root",
                  "arn:aws:iam::388731089494:root",
                  "arn:aws:iam::284668455005:root",
                  "arn:aws:iam::113285607260:root"
                ]
              },
              "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 an 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. 88).

Syntax

```
{
   "Type" : "AWS::CloudWatch::Alarm",
   "Properties" : {
      "ActionsEnabled (p. 288)" : Boolean,
      "AlarmActions (p. 288)" : [ String, ... ],
      "AlarmDescription (p. 288)" : String,
      "AlarmName (p. 288)" : String,
      "ComparisonOperator (p. 289)" : String,
      "Dimensions (p. 289)" : [ Metric dimension, ... ],
      "EvaluationPeriods (p. 289)" : String,
      "InsufficientDataActions (p. 289)" : [ String, ... ],
      "MetricName (p. 289)" : String,
      "Namespace (p. 289)" : String,
      "OKActions (p. 290)" : [ String, ... ],
      "Period (p. 290)" : String,
      "Statistic (p. 290)" : String,
      "Threshold (p. 290)" : String,
      "Unit (p. 290)" : 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. 88)

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). Currently the only action supported is publishing to an Amazon SNS topic or an Amazon Auto Scaling policy.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

AlarmDescription

The description for the alarm.

Required: No

Type: String

Update requires: No interruption (p. 88)

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. 514).

> **Important**
> If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: String

Update requires: Replacement (p. 88)

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. 88)

Dimensions

The dimensions for the alarm's associated metric.

Required: No

Type: List of Metric Dimension (p. 483)

Update requires: No interruption (p. 88)

EvaluationPeriods

The number of periods over which data is compared to the specified threshold.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

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: A list of strings

Update requires: No interruption (p. 88)

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 Developer Guide*.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Namespace

The namespace for the alarm's associated metric.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

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: A list of strings

Update requires: No interruption (p. 88)

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: String

Update requires: No interruption (p. 88)

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. 88)

Threshold

The value against which the specified statistic is compared.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

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. 88)

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. 565).

AWS::DynamoDB::Table

Creates a DynamoDB table.

> **Note**
> AWS CloudFormation typically creates DynamoDB tables in parallel. However, if your template includes DynamoDB tables with indexes, you must declare dependencies so that the tables are created sequentially. For a sample snippet, see DynamoDB Table with a DependsOn Attribute (p. 294).

Syntax

```
{
   "Type" : "AWS::DynamoDB::Table",
   "Properties" : {
      "AttributeDefinitions (p. 291)" : [ AttributeDefinitions, ... ],
      "GlobalSecondaryIndexes (p. 291)" : [ GlobalSecondaryIndexes, ... ],
      "KeySchema (p. 291)" : [ KeySchema, ... ],
      "LocalSecondaryIndexes (p. 292)" : [ LocalSecondaryIndexes, ... ],
      "ProvisionedThroughput (p. 292)" : { ProvisionedThroughput },
      "TableName (p. 292)" : String
   }
}
```

Properties

AttributeDefinitions

A list of `AttributeName` and `AttributeType` objects that describe the key schema for the table and indexes.

Required: Yes

Type: DynamoDB Attribute Definitions (p. 486)

Update requires: Replacement (p. 88)

GlobalSecondaryIndexes

Global secondary indexes to be created on the table. You can create up to 5 global secondary indexes.

Required: No

Type: DynamoDB Global Secondary Indexes (p. 486)

Update requires: Replacement (p. 88)

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: DynamoDB Key Schema (p. 487)

Update requires: Replacement (p. 88)

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: DynamoDB Local Secondary Indexes (p. 488)

Update requires: Replacement (p. 88)

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. 489).

Required: Yes

Type: DynamoDB Provisioned Throughput (p. 489)

Update requires: No interruption (p. 88)

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. 514).

Important

If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: Name Type (p. 514)

Update requires: Replacement (p. 88)

Note

For detailed information about the limits in DynamoDB, see Limits in Amazon DynamoDB in the *Amazon DynamoDB Developer Guide*.

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

DynamoDB Table with Local and Secondary Indexes

The following sample creates an DynamoDB table with `Album`, `Artist`, and `Sales` as attributes. The primary key includes the `Album` attribute as the hash key and `Artist` attribute as the range key. The table also includes a global and a 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 of an album, the local secondary index uses the same hash key as the table but uses the `Sales` attribute as the range key.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myDynamoDBTable" : {
      "Type" : "AWS::DynamoDB::Table",
      "Properties" : {
        "AttributeDefinitions" : [
          {
            "AttributeName" : "Album",
            "AttributeType" : "S"
          },
          {
            "AttributeName" : "Artist",
            "AttributeType" : "S"
          },
          {
            "AttributeName" : "Sales",
            "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"],
            "ProjectionType" : "INCLUDE"
          },
          "ProvisionedThroughput" : {
            "ReadCapacityUnits" : "5",
            "WriteCapacityUnits" : "5"
          }
        }],
        "LocalSecondaryIndexes" :[{
          "IndexName" : "myLSI",
          "KeySchema" : [
            {
              "AttributeName" : "Album",
              "KeyType" : "HASH"
            },
            {
              "AttributeName" : "Sales",
              "KeyType" : "RANGE"
            }
          ],
          "Projection" : {
            "NonKeyAttributes" : ["Artist"],
            "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. 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.

```
"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.

Syntax

```
{
   "Type" : "AWS::EC2::CustomerGateway",
   "Properties" : {
      "BgpAsn (p. 295)" : Number,
      "IpAddress (p. 296)" : String,
      "Tags (p. 296)" :  [ Resource Tag, ... ],
      "Type (p. 296)" : 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. 88)

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. 88)

Tags

The tags that you want to attach to the resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521).

Update requires: No interruption (p. 88).

Type

The type of VPN connection that this customer gateway supports.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Example: `ipsec.1`

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Example

```
{
   "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*.

Syntax

```
{
   "Type" : "AWS::EC2::DHCPOptions",
   "Properties" : {
      "DomainName (p. 297)" : String,
      "DomainNameServers (p. 297)" : [ String, ... ],
      "NetbiosNameServers (p. 297)" : [ String, ... ],
      "NetbiosNodeType (p. 298)" : Number,
      "NtpServers (p. 298)" : [ String, ... ],
      "Tags (p. 298)" : [ Resource Tag, ... ]
   }
}
```

Properties

DomainName

A domain name of your choice.

Required: Conditional; see note (p. 298).

Type: String

Update requires: Replacement (p. 88)

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. 298).

Type: A list of strings

Update requires: Replacement (p. 88)

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. 298).

Type: A list of strings

Update requires: Replacement (p. 88)

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: A list of numbers

Update requires: Replacement (p. 88)

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. 298).

Type: A list of strings

Update requires: Replacement (p. 88)

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. 521)

Update requires: No interruption (p. 88).

Conditional Properties

At least one of the following properties must be specified:

- DomainNameServers (p. 297)
- NetbiosNameServers (p. 297)
- NtpServers (p. 298)

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, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
{
   "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.

Syntax

```
{
   "Type" : "AWS::EC2::EIP",
   "Properties" : {
      "InstanceId (p. 300)" : String,
      "Domain (p. 300)" : 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. 88)

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. 540).

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. 88)

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Examples

To view AWS::EC2::EIP snippets, see Assigning an Amazon EC2 Elastic IP Using AWS::EC2::EIP Snippet (p. 167).

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. 299).

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

Syntax

```
{
   "Type": "AWS::EC2::EIPAssociation",
   "Properties": {
      "AllocationId (p. 301)": String,
      "EIP (p. 301)": String,
      "InstanceId (p. 301)": String,
      "NetworkInterfaceId (p. 302)": String,
      "PrivateIpAddress (p. 302)": String
   }
}
```

Properties

AllocationId

Allocation ID for the VPC Elastic IP address you want to associate with an Amazon EC2 instance in your VPC.

Required: Conditional. Required for a VPC.

Type: String

Update requires: Replacement (p. 88) if you also change the `InstanceId` or `NetworkInterfaceId` property. If not, update requires No interruption (p. 88).

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. 299).

Required: Conditional. Required for Elastic IP addresses for use in EC2-Classic.

Type: String

Update requires: Replacement (p. 88) if you also change the `InstanceId` or `NetworkInterfaceId` property. If not, update requires No interruption (p. 88).

InstanceId

Instance ID of the Amazon EC2 instance that you want to associate with the Elastic IP address specified by the EIP property.

Required: No

Type: String

Update requires: Replacement (p. 88) if you also change the `AllocationId` or `EIP` property. If not, update requires No interruption (p. 88).

NetworkInterfaceId

The ID of the network interface to associate with the Elastic IP address (VPC only).

Required: No

Type: String

Update requires: Replacement (p. 88) if you also change the `AllocationId` or `EIP` property. If not, update requires No interruption (p. 88).

PrivateIpAddress

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

For AWS::EC2::EIPAssociation snippets, see Assigning an Amazon EC2 Elastic IP Using AWS::EC2::EIP Snippet (p. 167).

AWS::EC2::Instance

The AWS::EC2::Instance type creates an Amazon 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. 88).

Syntax

```
{
   "Type" : "AWS::EC2::Instance",
   "Properties" : {
      "AvailabilityZone (p. 303)" : String,
      "BlockDeviceMappings (p. 303)" : [ EC2 Block Device Mapping, ... ],
      "DisableApiTermination (p. 303)" : Boolean,
      "EbsOptimized (p. 303)" : Boolean,
      "IamInstanceProfile (p. 304)" : String,
      "ImageId (p. 304)" : String,
      "InstanceInitiatedShutdownBehavior (p. 304)" : String,
      "InstanceType (p. 304)" : String,
      "KernelId (p. 304)" : String,
      "KeyName (p. 305)" : String,
      "Monitoring (p. 305)" : Boolean,
      "NetworkInterfaces (p. 305)" : [ EC2 Network Interface, ... ],
      "PlacementGroupName (p. 305)" : String,
      "PrivateIpAddress (p. 305)" : String,
      "RamdiskId (p. 306)" : String,
      "SecurityGroupIds (p. 306)" : [ String, ... ],
      "SecurityGroups (p. 306)" : [ String, ... ],
      "SourceDestCheck (p. 306)" : Boolean,
      "SubnetId (p. 306)" : String,
      "Tags (p. 306)" : [ Resource Tag, ... ],
      "Tenancy (p. 307)" : String,
      "UserData (p. 307)" : String,
      "Volumes (p. 307)" : [ EC2 MountPoint (p. 494), ... ]
   }
}
```

Properties

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. 88)

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. 490).

Update requires: Replacement (p. 88). If you change only the `DeleteOnTermination` property for one or more block devices, update requires No interruption (p. 88).

DisableApiTermination

Specifies whether the instance can be terminated through the API.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

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. 88) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 88) for instance store-backed instances

IamInstanceProfile

The physical ID of an instance profile or a reference to an AWS::IAM::InstanceProfile (p. 385) 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. 88)

ImageId

Provides the unique ID of the Amazon Machine Image (AMI) that was assigned during registration.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

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. 88)

InstanceType

The instance type, such as `t2.micro`. The default type is `"m1.small"`. For a list of instance types, see Instance Families and Types.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 88) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 88) for instance store-backed instances

KernelId

The kernel ID.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 88) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 88) for instance store-backed instances

KeyName

Provides the name of the Amazon EC2 key pair.

Required: No

Type: String

Update requires: Replacement (p. 88)

Monitoring

Specifies whether monitoring is enabled for the instance.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

NetworkInterfaces

A list of NetworkInterface (p. 495) embedded objects that describe the network interfaces to associate with this instance.

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. 540).

Required: No

Type: A list of NetworkInterface (p. 495) objects

Update requires: Replacement (p. 88)

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. 88)

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. 88)

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. 88) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 88) 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: A list of strings

Update requires:

- *Update requires*: No interruption (p. 88) for instances that are in a VPC.
- *Update requires*: Replacement (p. 88) 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: A list of strings

Update requires: Replacement (p. 88).

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. 88)

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. 88)

Tags

An arbitrary set of tags (key–value pairs) for this instance.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

Tenancy

The tenancy of the instance that you want to launch. This value can be either `"default"` or `"dedicated"`. An instance that has a *`tenancy`* value of `"dedicated"` runs on single-tenant hardware and can be launched only into a VPC. For more information, see Using EC2 Dedicated Instances Within Your VPC in the *Amazon VPC User Guide*.

Required: No

Type: String

Update requires: Replacement (p. 88)

UserData

Base64-encoded MIME user data that is made available to the instances.

Required: No

Type: String

Update requires:

- *Update requires*: Some interruptions (p. 88) for Amazon EBS-backed instances
- *Update requires*: Replacement (p. 88) 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. 494).

Update requires: No interruption (p. 88)

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 562) 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. 558).

Examples

EC2 Instance with an EBS Block Device Mapping

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

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 Snippets (p. 166).

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.

Syntax

```
{
   "Type" : "AWS::EC2::InternetGateway",
   "Properties" : {
      "Tags (p. 309)" : [ Resource Tag, ... ]
   }
}
```

Properties

Tags
: An arbitrary set of tags (key–value pairs) for this resource.

  Required: No

  Type: AWS CloudFormation Resource Tags (p. 521)

  Update requires: No interruption (p. 88).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myInternetGateway" : {
         "Type" : "AWS::EC2::InternetGateway",
         "Properties" : {
            "Tags" : [ {"Key" : "foo", "Value" : "bar"}]
         }
      }
   }
}
```

See Also

- CreateInternetGateway in the *Amazon EC2 API Reference*.
- Using Tags in the *Amazon Elastic Compute Cloud User Guide*.

AWS::EC2::NetworkAcl

Creates a new network ACL in a VPC.

Syntax

```
{
   "Type" : "AWS::EC2::NetworkAcl",
   "Properties" : {
      "Tags (p. 310)" : [ Resource Tag, ... ],
      "VpcId (p. 310)" : String
   }
}
```

Properties

Tags
An arbitrary set of tags (key–value pairs) for this ACL.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

VpcId
The ID of the VPC where the network ACL will be created.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
{
   "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., 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.

Syntax

```
{
   "Type" : "AWS::EC2::NetworkAclEntry",
   "Properties" : {
      "CidrBlock (p. 311)" : String,
      "Egress (p. 312)" : Boolean,
      "Icmp (p. 312)" : EC2 ICMP,
      "NetworkAclId (p. 312)" : String,
      "PortRange (p. 312)" : EC2 PortRange,
      "Protocol (p. 312)" : Integer,
      "RuleAction (p. 312)" : String,
      "RuleNumber (p. 312)" : Integer
   }
}
```

Properties

CidrBlock

The CIDR range to allow or deny, in CIDR notation (e.g., 172.16.0.0/24).

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Egress
: Whether this rule applies to egress traffic from the subnet ("true") or ingress traffic to the subnet ("false").

  Required: Yes

  Type: Boolean

  Update requires: Replacement (p. 88).

Icmp
: The Internet Control Message Protocol (ICMP) code and type.

  Required: Conditional required if specifying 1 (ICMP) for the protocol parameter.

  Type: EC2 ICMP Property Type (p. 493)

  Update requires: No interruption (p. 88)

NetworkAclId
: ID of the ACL where the entry will be created.

  Required: Yes

  Type: String

  Update requires: Replacement (p. 88).

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 PortRange Property Type (p. 499)

  Update requires: No interruption (p. 88)

Protocol
: IP protocol the rule applies to. You can use -1 to mean all protocols. This must be -1 or a protocol number (go to Protocol Numbers at iana.org).

  Required: Yes

  Type: Number

  Update requires: No interruption (p. 88)

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. 88)

RuleNumber
: Rule number to assign to the entry (e.g., 100). This must be a positive integer from 1 to 32766.

  Required: Yes

  Type: Number

  Update requires: Replacement (p. 88).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
{
   "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. 302).

Syntax

```
{
   "Type" : "AWS::EC2::NetworkInterface",
   "Properties" : {
      "Description (p. 314)" : String,
      "GroupSet (p. 314)" : [ String, ... ],
      "PrivateIpAddress (p. 314)" : String,
      "PrivateIpAddresses (p. 314)" : [ PrivateIpAddressSpecification, ... ],
      "SecondaryPrivateIpAddressCount (p. 314)" : Integer,
      "SourceDestCheck (p. 315)" : Boolean,
      "SubnetId (p. 315)" : String,
      "Tags (p. 315)" : [ Resource Tag, ... ],
   }
}
```

Properties

Description

The description of this network interface.

Required: No

Type: String

Update requires: No interruption (p. 88).

GroupSet

A list of security group IDs associated with this network interface.

Required: No

Type: List of strings.

Update requires: No interruption (p. 88)

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. 88).

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. 499).

Update requires: Replacement (p. 88) if you change the primary private IP address. If not, update requires No interruption (p. 88).

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. 88).

SourceDestCheck
Flag indicating whether traffic to or from the instance is validated.

Required: No

Type: Boolean

Update requires: No interruption (p. 88).

SubnetId
The ID of the subnet to associate with the network interface.

Required: Yes

Type: String

Update requires: Replacement (p. 88).

Tags
An arbitrary set of tags (key–value pairs) for this network interface.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Template Examples

Tip
For more NetworkInterface template examples, see Elastic Network Interface (ENI) Template Snippets (p. 168).

Simple Standalone ENI

This is a simple standalone Elastic Network Interface (ENI), using all of the available properties.

```
{
   "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.

```
"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" }}
   }
}
```

See Also

- 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.

Syntax

```
{
   "Type" : "AWS::EC2::NetworkInterfaceAttachment",
   "Properties" : {
      "DeleteOnTermination (p. 317)": Boolean,
      "DeviceIndex (p. 317)": String,
      "InstanceId (p. 317)": String,
      "NetworkInterfaceId (p. 317)": 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. 88)

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. 88)

InstanceId

The ID of the instance to which you will attach the ENI.

Required: Yes.

Type: String.

Update requires: No interruption (p. 88)

NetworkInterfaceId

The ID of the ENI that you want to attach.

Required: Yes.

Type: String.

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

Example Attaching `MyNetworkInterface` to `MyInstance`

```
"NetworkInterfaceAttachment" : {
    "Type" : "AWS::EC2::NetworkInterfaceAttachment",
        "Properties" : {
            "InstanceId" : {"Ref" : "MyInstance"},
            "NetworkInterfaceId" : {"Ref" : "MyNetworkInterface"},
            "DeviceIndex" : "1"
        }
}
```

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.

Syntax

```

{
   "Type" : "AWS::EC2::Route",
   "Properties" : {
      "DestinationCidrBlock (p. 318)" : String,
      "GatewayId (p. 319)" : String,
      "InstanceId (p. 319)" : String,
      "NetworkInterfaceId (p. 319)" : String,
      "RouteTableId (p. 319)" : String,
      "VpcPeeringConnectionId (p. 319)" : String
   }
}
```

Properties

DestinationCidrBlock

The 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: Yes

Type: String

Update requires: Replacement (p. 88)

GatewayId

The ID of a gateway 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. 540).

Required: Conditional. You must provide *only one* of the following: a `GatewayID`, `InstanceID`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 88)

InstanceId

The ID of a NAT instance in your VPC. For example, `"i-1a2b3c4d"`.

Required: Conditional. You must provide *only one* of the following: a `GatewayID`, `InstanceID`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 88)

NetworkInterfaceId

Allows the routing of network interface IDs.

Required: Conditional. You must provide *only one* of the following: a `GatewayID`, `InstanceID`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 88)

RouteTableId

The ID of the route table (p. 321) where the route will be added.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

VpcPeeringConnectionId

The ID of a VPC peering connection.

Required: Conditional. You must provide *only one* of the following: a `GatewayID`, `InstanceID`, `NetworkInterfaceId`, or `VpcPeeringConnectionId`.

Type: String

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

Example Route with Gateway ID

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

Example Route with Instance ID

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myRoute" : {
         "Type" : "AWS::EC2::Route",
         "Properties" : {
            "RouteTableId" : { "Ref" : "myRouteTable" },
            "DestinationCidrBlock" : "0.0.0.0/0",
            "InstanceId" : { "Ref" : "myInstance" }
         }
      }
   }
}
```

Example Route with Network Interface ID.

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myRoute" : {
         "Type" : "AWS::EC2::Route",
         "Properties" : {
            "RouteTableId" : { "Ref" : "myRouteTable" },
            "DestinationCidrBlock" : "0.0.0.0/0",
            "NetworkInterfaceId" : { "Ref" : "eni-1a2b3c4d" }
         }
      }
   }
}
```

Example Route with VPC peering connection ID.

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myRoute" : {
         "Type" : "AWS::EC2::Route",
         "Properties" : {
            "RouteTableId" : { "Ref" : "myRouteTable" },
            "DestinationCidrBlock" : "0.0.0.0/0",
            "VpcPeeringConnectionId" : { "Ref" : "myVPCPeeringConnectionID" }
         }
      }
   }
}
```

See Also

- AWS::EC2::RouteTable (p. 321)
- 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.

Syntax

```
{
   "Type" : "AWS::EC2::RouteTable",
   "Properties" : {
      "VpcId (p. 321)" : String,
      "Tags (p. 321)" : [ 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. 88)

Tags
: An arbitrary set of tags (key–value pairs) for this route table.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

Example

The following example snippet uses the VPC ID from a VPC named *myVPC* that was declared elsewhere in the same template.

```
{
   "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. 318)
- 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. 323) property.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

> **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. 325) and AWS::EC2::SecurityGroupIngress (p. 328) 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.

Syntax

```
{
   "Type" : "AWS::EC2::SecurityGroup",
   "Properties" :
   {
      "GroupDescription (p. 323)" : String,
      "SecurityGroupEgress (p. 323)" : [ Security Group Rule, ... ],
      "SecurityGroupIngress (p. 323)" : [ Security Group Rule, ... ],
      "Tags (p. 323)" :  [ Resource Tag, ... ],
      "VpcId (p. 323)" : String
   }
}
```

Properties

GroupDescription

Description of the security group.

Type: String

Required: Yes

Update requires: Replacement (p. 88)

SecurityGroupEgress

A list of Amazon EC2 security group egress rules.

Type: EC2 Security Group Rule (p. 500)

Required: No

Update requires: No interruption (p. 88)

SecurityGroupIngress

A list of Amazon EC2 security group ingress rules.

Type: EC2 Security Group Rule (p. 500)

Required: No

Update requires: No interruption (p. 88)

Tags

The tags that you want to attach to the resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521).

Update requires: No interruption (p. 88).

VpcId

The physical ID of the VPC. Can be obtained by using a reference to an AWS::EC2::VPC (p. 341), such as: `{ "Ref" : "myVPC" }`.

For more information about using the `Ref` function, see Ref (p. 565).

Type: String

Required: Yes, for VPC security groups

Update requires: Replacement (p. 88)

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 (for EC2-classic) or the security group ID (for EC2-VPC).

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Example

AWS::EC2::SecurityGroup exists as a top-level element inside an AWS CloudFormation template. Here's an example:

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

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. 322). For more information, see Amazon EC2 Security Groups.

Syntax

```
{
   "CidrIp (p. 325)" : String,
   "DestinationSecurityGroupId (p. 325)" : String,
   "FromPort (p. 325)" : Integer,
   "GroupId (p. 326)" : String,
   "IpProtocol (p. 326)" : String,
   "ToPort (p. 326)" : 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

CIDR range.

Type: String

Required: Conditional. Cannot be used when specifying a destination security group.

Update requires: Replacement (p. 88)

DestinationSecurityGroupId

Specifies the group ID of the destination Amazon VPC security group.

Type: String

Required: Conditional. Cannot be used when specifying a CIDR IP address.

Update requires: Replacement (p. 88)

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

Update requires: Replacement (p. 88)

GroupId

ID of the Amazon VPC security group to modify. This value can be a reference to an AWS::EC2::SecurityGroup (p. 322) resource that has a valid VpcId property or the ID of an existing Amazon VPC security group.

Type: String

Required: Yes

Update requires: Replacement (p. 88)

IpProtocol

IP protocol name or number. For valid values, see the IpProtocol parameter in AuthorizeSecurityGroupIngress

Type: String

Required: Yes

Update requires: Replacement (p. 88)

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

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

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.

```
{
  "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. 322). For more information, see Amazon EC2 Security Groups.

Syntax

```
{
   "CidrIp (p. 328)" : String,
   "FromPort (p. 328)" : Integer,
   "GroupId (p. 328)" : String,
   "GroupName (p. 329)" : String,
   "IpProtocol (p. 329)" : String,
   "SourceSecurityGroupName (p. 329)" : String,
   "SourceSecurityGroupId (p. 329)" : String,
   "SourceSecurityGroupOwnerId (p. 329)" : String,
   "ToPort (p. 329)" : 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

Specifies a CIDR range.

For an overview of CIDR ranges, go to the Wikipedia Tutorial.

Type: String

Required: Conditional. If you specify SourceSecurityGroupName, do not specify CidrIp.

Update requires: Replacement (p. 88)

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. 88)

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. 88)

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. 322) 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. 88)

IpProtocol

IP protocol name or number. For valid values, see the IpProtocol parameter in AuthorizeSecurityGroupIngress

Type: String

Required: Yes

Update requires: Replacement (p. 88)

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. If you specify CidrIp, do not specify SourceSecurityGroupId.

Update requires: Replacement (p. 88)

SourceSecurityGroupName

Specifies the name of the Amazon EC2 security group (non-VPC security group) to allow access or uses the `Ref` intrinsic function to refer to the logical name of a security group defined in the same template. For instances in a VPC, specify the `SourceSecurityGroupId` property.

Type: String

Required: Conditional. If you specify CidrIp, do not specify SourceSecurityGroupName.

Update requires: Replacement (p. 88)

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. 88)

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. 88)

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. 565) to specify the value for SourceSecurityGroupName.

```
{
    "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": {
                "GroupName": { "Ref": "SGBase" },
                "IpProtocol": "tcp",
                "FromPort": "80",
                "ToPort": "80",
                "SourceSecurityGroupName": { "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.

```
{
  "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.

```
"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::Subnet

Creates a subnet in an existing VPC.

Syntax

```
{
   "Type" : "AWS::EC2::Subnet",
   "Properties" : {
      "AvailabilityZone (p. 332)" : String,
      "CidrBlock (p. 333)" : String,
      "Tags (p. 333)" : [ Resource Tag, ... ],
      "VpcId (p. 333)" : { "Ref" : 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. 88)

  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. 88)

Note
If you update this property, you must also update the `AvailabilityZone` property.

Tags

An arbitrary set of tags (key–value pairs) for this subnet.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

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. 88)

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 used.

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

AvailabilityZone

Returns the availability zone (for example, `"us-east-1a"`) of this subnet.

Example:

```
{ "Fn::GetAtt" : [ "mySubnet", "AvailabilityZone" ] }
```

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Example

The following example snippet uses the VPC ID from a VPC named *myVPC* that was declared elsewhere in the same template.

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

See Also

- CreateSubnet in the *Amazon EC2 API Reference*
- Using Tags in the *Amazon Elastic Compute Cloud User Guide*

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`*.

Syntax

```
"Type" : "AWS::EC2::SubnetNetworkAclAssociation",
"Properties" : {
   "SubnetId (p. 334)" : { String }
   "NetworkAclId (p. 334)" : { 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. 88)

NetworkAclId

The ID of the new ACL to associate with the subnet.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

AssociationId
    Returns the value of this object's SubnetId (p. 334) property.

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Template Examples

Example

```
{
   "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.

Syntax

```
{
   "Type" : "AWS::EC2::SubnetRouteTableAssociation",
   "Properties" : {
      "RouteTableId (p. 336)" : String,
      "SubnetId (p. 336)" : 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. 88). 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. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Example

```

{
   "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 volume.

You can set a deletion policy for your volume to control how AWS CloudFormation handles the volume when the stack is deleted. For Amazon Elastic Block Store volumes, 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. 539).

> **Note**
> If you set a deletion policy that creates a snapshot, all tags on the volume are included in the snapshot.

Syntax

```
{
   "Type":"AWS::EC2::Volume",
   "Properties" : {
      "AvailabilityZone (p. 337)" : String,
      "Encrypted (p. 337)" : Boolean,
      "Iops (p. 338)" : Number,
      "Size (p. 338)" : String,
      "SnapshotId (p. 338)" : String,
      "Tags (p. 338)" : [ Resource Tag, ... ],
      "VolumeType (p. 338) : String
   }
}
```

Properties

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. 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

Update requires: Updates are not supported.

Iops

The number of I/O operations per second (IOPS) that the volume supports. This can be any integer value from 1–4000.

Required: Conditional. *Required* when the volume type is `io1`; not used with other volume types.

Type: Number

Update requires: Updates are not supported.

Size

The size of the volume, in gibibytes (GiBs). This can be any value from 10–1024.

Note

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.

Required: Conditional. *Required* if you are not creating a volume from a snapshot. If you specify Size, *do not* specify SnapshotId.

Type: String

Update requires: Updates are not supported.

SnapshotId

The snapshot from which to create the new volume.

Required: Conditional *Required* if you are creating a volume from a snapshot. If you do not specify a value for SnapshotId, you must specify a value for Size.

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. 521)

Update requires: Updates are not supported.

VolumeType

The volume type. You can specify `standard`, `io1`, or `gp2`. If you set the type to `io1`, you must also set the Iops property. For more information about these values and the default value, see CreateVolume 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. 565).

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"
 ] }
   }
}
```

See Also

- CreateVolume in the *Amazon Elastic Compute Cloud API Reference*
- DeletionPolicy Attribute (p. 539)

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.

Syntax

```
{
   "Type":"AWS::EC2::VolumeAttachment",
   "Properties" : {
      "Device (p. 340)" : String,
      "InstanceId (p. 340)" : String,
      "VolumeId (p. 340)" : 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. 302) 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. 337) 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.

Syntax

```
{
   "Type" : "AWS::EC2::VPC",
   "Properties" : {
      "CidrBlock (p. 341)" : String,
      "EnableDnsSupport (p. 342)" : Boolean,
      "EnableDnsHostnames (p. 342)" : Boolean,
      "InstanceTenancy (p. 342)" : String,
      "Tags (p. 342)" : [ 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. 88)

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. 88)

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. 88)

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 will automatically be dedicated, regardless of the tenancy option you specify when you launch the instance.

Required: No

Type: String

Valid values: `"default"` or `"dedicated"`

Update requires: Replacement (p. 88)

Tags

An arbitrary set of tags (key–value pairs) for this VPC.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

You can obtain the following default resource IDs, which AWS creates whenever you create a VPC.

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`.

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Example

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

See Also

- CreateVpc in the *Amazon EC2 API Reference*.

AWS::EC2::VPCDHCPOptionsAssociation

Associates a set of DHCP options (that you've previously created) with the specified VPC.

Syntax

```
{
   "Type" : "AWS::EC2::VPCDHCPOptionsAssociation",
   "Properties" : {
      "DhcpOptionsId (p. 343)" : String,
      "VpcId (p. 344)" : 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. 88)

VpcId

The ID of the VPC to associate with this DHCP options set.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

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. 341) or AWS::EC2::DHCPOptions (p. 297).

```
"myVPCDHCPOptionsAssociation" : {
  "Type" : "AWS::EC2::VPCDHCPOptionsAssociation",
  "Properties" : {
    "VpcId" : {"Ref" : "myVPC"},
    "DhcpOptionsId" : {"Ref" : "myDHCPOptions"}
  }
}
```

See Also

- AssociateDhcpOptions in the *Amazon EC2 API Reference*.

AWS::EC2::VPCGatewayAttachment

Attaches a gateway to a VPC.

Syntax

```

{
   "Type" : "AWS::EC2::VPCGatewayAttachment",
   "Properties" : {
      "InternetGatewayId (p. 345)" : String,
      "VpcId (p. 345)" : String,
      "VpnGatewayId (p. 345)" : 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. 88)

VpcId
: The ID of the VPC to associate with this gateway.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

Example Attaching both an Internet gateway and a VPN gateway to a VPC

To attach both an Internet gateway and a VPN gateway to a VPC, you must specify two separate AWS::EC2::VPCGatewayAttachment resources:

```
"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 by means of a private IP addresses. For more information about VPC peering and its limitation, see VPC Peering Overview in the *Amazon VPC Peering Guide*.

> **Note**
> With AWS CloudFormation, you can create a peering connection only between VPCs in the same AWS account. You cannot create a peering connection with another AWS account.

Syntax

```
{
   "Type" : "AWS::EC2::VPCPeeringConnection",
   "Properties" : {
      "PeerVpcId (p. 346)" : String,
      "Tags (p. 347)" : [ Resource Tag, ... ],
      "VpcId (p. 347)" : String
   }
}
```

Properties

PeerVpcId

The ID of the VPC with which you are creating the peering connection.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Tags
: An arbitrary set of tags (key–value pairs) for this resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

VpcId
: The ID of the VPC that is requesting a peering connection.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

Example A sample VPC peering connection

The following sample 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 sample by `PeeringRoute1` and `PeeringRoute2`. If you launch the template, you can SSH into the `myInstance` instance and then ping the `myPrivateInstance` instance even though both instances are in separate VPCs.

```
{
    "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*.

Syntax

```
{
   "Type" : "AWS::EC2::VPNConnection",
   "Properties" : {
      "Type (p. 354)" : String,
      "CustomerGatewayId (p. 354)" : GatewayID,
      "StaticRoutesOnly (p. 354)" : Boolean,
      "Tags (p. 355)" :  [ Resource Tag, ... ],
      "VpnGatewayId (p. 355)" : GatewayID
   }
}
```

Properties

Type
: The type of VPN connection this virtual private gateway supports.

  Example: "ipsec.1"

  Required: Yes

  Type: String

  Update requires: Replacement (p. 88)

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. 88)

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. 88)

Tags

The tags that you want to attach to the resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521).

Update requires: No interruption (p. 88).

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. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Examples

Example VPNConnection

```
{
   "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.

Syntax

```
{
   "Type" : "AWS::EC2::VPNConnectionRoute",
   "Properties" : {
      "DestinationCidrBlock (p. 356)" : String
      "VpnConnectionId (p. 356)" : 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. 88)

VpnConnectionId
: The ID of the VPN connection.

  Required: Yes.

  Type: String

  Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

Example Specifying a static route

```
"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.

Syntax

```
{
   "Type" : "AWS::EC2::VPNGateway",
   "Properties" : {
      "Type (p. 357)" : String,
      "Tags (p. 357)" : [ 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. 88)

Tags

An arbitrary set of tags (key–value pairs) for this resource.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Example

```
{
   "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. 540) attribute in the `AWS::EC2::VPNGatewayRoutePropagation` resource to explicitly declare a dependency on the VPN gateway attachment.

Syntax

```
{
   "Type" : "AWS::EC2::VPNGatewayRoutePropagation",
   "Properties" : {
      "RouteTableIds (p. 359)" : [ String, ... ],
      "VpnGatewayId (p. 359)" : 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. 88)

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. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Example

```
"myVPNGatewayRouteProp" : {
  "Type" : "AWS::EC2::VPNGatewayRoutePropagation",
  "Properties" : {
    "RouteTableIds" : [{"Ref" : "PrivateRouteTable"}],
    "VpnGatewayId" : {"Ref" : "VPNGateway"}
  }
}
```

See Also

- EnableVgwRoutePropagation in the *Amazon EC2 API Reference*.

AWS::ElastiCache::CacheCluster

The AWS::ElastiCache::CacheCluster type creates an Amazon ElastiCache cache cluster.

Syntax

```
{
   "Type" : "AWS::ElastiCache::CacheCluster",
   "Properties" :
   {
      "AutoMinorVersionUpgrade (p. 360)" : Boolean,
      "CacheNodeType (p. 360)" : String,
      "CacheParameterGroupName (p. 360)" : String,
      "CacheSecurityGroupNames (p. 361)" : [ String, ... ],
      "CacheSubnetGroupName (p. 361)" : String,
      "ClusterName (p. 361)" : String,
      "Engine (p. 361)" : String,
      "EngineVersion (p. 361)" : String,
      "NotificationTopicArn (p. 361)" : String,
      "NumCacheNodes (p. 362)" : String,
      "Port (p. 362)" : Integer,
      "PreferredAvailabilityZone (p. 362)" : String,
      "PreferredMaintenanceWindow (p. 362)" : String,
      "SnapshotArns (p. 362)" : [String, ... ],
      "VpcSecurityGroupIds (p. 362)" : [String, ...]
   }
}
```

Properties

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. 88)

CacheNodeType

The compute and memory capacity of nodes in a cache cluster.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

CacheParameterGroupName

The name of the cache parameter group that is associated with this cache cluster.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

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: If your cache cluster isn't in a VPC, you must specify this property.

Type: List of Strings

Update requires: No interruption (p. 88)

CacheSubnetGroupName

The cache subnet group that you associate with a cache cluster.

Required: No

Type: String

Update requires: Replacement (p. 88)

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. 514).

> **Important**
>
> If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: String

Update requires: Replacement (p. 88)

Engine

The name of the cache engine to be used for this cache cluster, such as `memcached` or `redis`.

> **Note**
>
> AWS CloudFormation does not currently support replication groups and read replicas for Redis.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

EngineVersion

The version of the cache engine to be used for this cluster.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

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. 88)

NumCacheNodes

The number of cache nodes that the cache cluster should have.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Port

The port number on which each of the cache nodes will accept connections.

Required: No

Type: Integer

Update requires: Replacement (p. 88)

PreferredAvailabilityZone

The EC2 Availability Zone in which the cache cluster is created.

Required: No

Type: String

Update requires: Replacement (p. 88)

PreferredMaintenanceWindow

The weekly time range (in UTC) during which system maintenance can occur.

Required: No

Type: String

Update requires: No interruption (p. 88)

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: String list

Update requires: Replacement (p. 88)

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: If your cache cluster is in a VPC, you must specify this property.

Type: List of strings

Update requires: No interruption (p. 88)

Notes

Obtaining the Cache Cluster Node Addresses

The ElastiCache cache cluster does not have a single endpoint, but you can obtain the endpoints for individual cache nodes by defining a `get-cache-nodes` script and installing it in the AWS::CloudFormation::Init (p. 268) section of the template.

You can view a full sample templates for implementation details:

- For Memcached, see https://s3.amazonaws.com/cloudformation-templates-us-east-1/ElastiCache.template
- For Redis, see https://s3.amazonaws.com/cloudformation-templates-us-east-1/ElastiCache_Redis.template

The Amazon ElastiCache template uses the AWS CloudFormation bootstrap script cfn-hup (p. 577) to detect changes to the Amazon ElastiCache cache cluster configuration, such as the number of instances in the cache cluster. It then runs a script to update the on-host configuration for the application.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

> **Note**
> Currently, you can use `Fn::GetAtt` only with Memcached cache clusters.

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.

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Template Snippet

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.

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. Because the security group doesn't specify a VPC ID, the `Ref` function will return the security group name.

```
"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.t1.micro",
    "NumCacheNodes": "1",
    "VpcSecurityGroupIds": [{"Fn::GetAtt": [ "ElasticacheSecurityGroup",
"GroupId"]}]
  }
}
```

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.

Syntax

```

{
   "Type": "AWS::ElastiCache::ParameterGroup",
   "Properties": {
      "CacheParameterGroupFamily" : String,
      "Description" : String,
      "Properties" : { "prop1" : "value1", ... }
   }
}
```

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: Yes

Type: JSON object

Update requires: Updates are not supported.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
"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. 88)

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. 322) resource. For more information, see the `VpcSecurityGroupIds` property in the AWS::ElastiCache::CacheCluster (p. 360) resource.

Syntax

```
{
  "Type" : "AWS::ElastiCache::SecurityGroup",
  "Properties" :
  {
    "Description (p. 366)" : 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. 565).

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*.

Syntax

```
{
  "Type" : "AWS::ElastiCache::SecurityGroupIngress",
  "Properties" :
  {
    "CacheSecurityGroupName (p. 367)" : String,
    "EC2SecurityGroupName (p. 367)" : String,
    "EC2SecurityGroupOwnerId (p. 367)" : 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*.

When you specify an AWS::ElastiCache::SubnetGroup type as an argument to the `Ref` function, AWS CloudFormation returns the name of the cache subnet group.

Syntax

```
"SubnetGroup" : {
    "Type" : "AWS::ElastiCache::SubnetGroup",
    "Properties" : {
        "Description (p. 368)" : String,
        "SubnetIds (p. 368)" : [ String, ... ]
    }
}
```

Properties

Description
: The description for the cache subnet group.

  Type: String

  Required: Yes

  Update requires: No interruption (p. 88)

SubnetIds
: The Amazon EC2 subnet IDs for the cache subnet group.

  Type: String list

  Required: Yes

  Update requires: No interruption (p. 88)

Example

```
"SubnetGroup" : {
    "Type" : "AWS::ElastiCache::SubnetGroup",
    "Properties" : {
        "Description" : "Cache Subnet Group",
        "SubnetIds" : [ { "Ref" : "Subnet1" }, { "Ref" : "Subnet2" } ]
    }
}
```

AWS::ElasticBeanstalk::Application

Creates an AWS Elastic Beanstalk application.

Syntax

```

{
   "Type" : "AWS::ElasticBeanstalk::Application",
   "Properties" : {
      "ApplicationName (p. 368)" : String,
      "Description (p. 369)" : String
   }
}
```

Properties

ApplicationName
: A name for the AWS 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. 514).

  Important
  If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: String

Update requires: Replacement (p. 88)

Description

An optional description of this application.

Required: No

Type: String

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
{
   "Type" : "AWS::ElasticBeanstalk::Application",
   "Properties" : {
      "ApplicationName" : "SampleAWSElasticBeanstalkApplication",
      "Description" : "AWS Elastic Beanstalk PHP Sample Application"
   }
}
```

See Also

- For a complete AWS Elastic Beanstalk sample template, see AWS Elastic Beanstalk Snippets (p. 175).

AWS::ElasticBeanstalk::ApplicationVersion

Creates an application version, an iteration of deployable code, for an AWS Elastic Beanstalk application.

Syntax

```
{
  "Type" : "AWS::ElasticBeanstalk::ApplicationVersion",
  "Properties" : {
    "ApplicationName (p. 370)" : String,
    "Description (p. 370)" : String,
    "SourceBundle (p. 370)" : { SourceBundle }
  }
}
```

Members

ApplicationName

Name of the AWS Elastic Beanstalk application that is associated with this application version.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Description

A description of this application version.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

SourceBundle

The location of the source bundle for this version.

Required: No

Type: Source Bundle (p. 505)

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Example

```
"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 AWS Elastic Beanstalk sample template, see AWS Elastic Beanstalk Snippets (p. 175).

AWS::ElasticBeanstalk::ConfigurationTemplate

Creates a configuration template for an AWS 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.

Syntax

```
{
  "Type" : "AWS::ElasticBeanstalk::ConfigurationTemplate",
  "Properties" : {
    "ApplicationName (p. 371)" : String,
    "Description (p. 371)" : String,
    "EnvironmentId (p. 371)" : String,
    "OptionSettings (p. 371)" : [ OptionSetting, ... ],
    "SolutionStackName (p. 372)" : String,
    "SourceConfiguration (p. 372)" : Source configuration
  }
}
```

Members

ApplicationName

Name of the AWS Elastic Beanstalk application that is associated with this configuration template.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Description

An optional description for this configuration.

Type: String

Required: No

Update requires: Some interruptions (p. 88)

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. 88)

OptionSettings

A list of OptionSettings (p. 504) for this Elastic Beanstalk configuration. 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. 504).

Required: No

Update requires: Some interruptions (p. 88)

SolutionStackName

The name of an AWS 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. 88)

SourceConfiguration

A configuration template that is associated with another AWS 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: AWS Elastic Beanstalk SourceConfiguration Property Type (p. 506)

Required: Conditional

Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

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.

```
"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. 368)
- Option Values in the *AWS Elastic Beanstalk Developer Guide*
- For a complete AWS Elastic Beanstalk sample template, see AWS Elastic Beanstalk Snippets (p. 175).

AWS::ElasticBeanstalk::Environment

Creates or updates an AWS Elastic Beanstalk environment.

Syntax

```
{
   "Type" : "AWS::ElasticBeanstalk::Environment",
   "Properties" : {
      "ApplicationName (p. 373)" : String,
      "CNAMEPrefix (p. 373)" : String,
      "Description (p. 374)" :  String,
      "EnvironmentName (p. 374)" :  String,
      "OptionSettings (p. 374)" : [ OptionSettings, ... ],
      "SolutionStackName (p. 374)" : String,
      "TemplateName (p. 374)" : String,
      "Tier (p. 374)" : Environment Tier,
      "VersionLabel (p. 375)" : String
   }
}
```

Properties

ApplicationName
: The name of the application that is associated with this environment.

  Required: Yes

  Type: String

  Update requires: Replacement (p. 88)

CNAMEPrefix
: A prefix for your AWS Elastic Beanstalk environment URL.

  Required: No

  Type: String

  Update requires: Replacement (p. 88)

Description
: A description that helps you identify this environment.

  Required: No

  Type: String

  Update requires: No interruption (p. 88)

EnvironmentName
: A name for the AWS 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. 514).

  > **Important**
  > If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

  Required: No

  Type: String

  Update requires: Replacement (p. 88)

OptionSettings
: Key-value pairs defining configuration options for this environment. These options override the values that are defined in the solution stack or the configuration template. If you remove any options during a stack update, the removed options revert to default values.

  Required: No

  Type: A list of OptionSettings (p. 504).

  Update requires: Some interruptions (p. 88)

SolutionStackName
: The name of an AWS 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 AWS Elastic Beanstalk configuration template name.

  Required: No

  Type: String

  Update requires: Replacement (p. 88)

TemplateName
: The name of the AWS 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. 88)

Tier
: Specifies the tier to use in creating this environment. The environment tier that you choose determines whether AWS 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: AWS Elastic Beanstalk Environment Tier Property Type (p. 503)

Update requires: See AWS Elastic Beanstalk Environment Tier Property Type (p. 503)

VersionLabel

The version to associate with the environment.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Examples

Simple Environment

```
{
   "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

```
{
   "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 AWS Elastic Beanstalk sample template, see AWS Elastic Beanstalk Snippets (p. 175).

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. 540).

Syntax

```

{
   "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
   "Properties": {
      "AccessLoggingPolicy (p. 377)" : AccessLoggingPolicy,
      "AppCookieStickinessPolicy (p. 377)" : [ AppCookieStickinessPolicy, ...
],
      "AvailabilityZones (p. 377)" : [ String, ... ],
      "ConnectionDrainingPolicy (p. 377)" : ConnectionDrainingPolicy,
      "CrossZone (p. 377)" : Boolean,
      "HealthCheck (p. 377)" : HealthCheck,
      "Instances (p. 378)" : [ String, ... ],
      "LBCookieStickinessPolicy (p. 378)" : [ LBCookieStickinessPolicy, ... ],
      "LoadBalancerName (p. 378)" : String,
      "Listeners (p. 378)" : [ Listener, ... ],
      "Policies (p. 378)" : [ ElasticLoadBalancing Policy, ... ],
      "Scheme (p. 378)" : String,
      "SecurityGroups (p. 379)" : [ Security Group, ... ],
      "Subnets (p. 379)" : [ String, ... ],
      "Tags (p. 379)" : [ 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. 506)

Update requires: No interruption (p. 88)

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. 507) objects.

Update requires: No interruption (p. 88)

AvailabilityZones

The Availability Zones in which to create the load balancer. You can specify *either* AvailabilityZones or Subnets, but not both.

Required: No

Type: A list of strings

Update requires: Replacement (p. 88) 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. 88).

ConnectionDrainingPolicy

Whether deregistered or unhealthy instances can complete all in-flight requests.

Required: No

Type: Elastic Load Balancing ConnectionDrainingPolicy (p. 508)

Update requires: No interruption (p. 88)

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. 88)

HealthCheck

Application health check for the instances.

Required: No

Type: ElasticLoadBalancing HealthCheck Type (p. 509).

Update requires: Replacement (p. 88) 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. 88).

Instances

A list of EC2 instance IDs for the load balancer.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

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. 510) objects.

Update requires: No interruption (p. 88)

LoadBalancerName

A name for the load balancer. 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. 514).

Important

If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: String

Update requires: Replacement (p. 88)

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. 510) objects.

Update requires: No interruption (p. 88)

Policies

A list of elastic load balancing policies to apply to this elastic load balancer.

Required: No

Type: A list of ElasticLoadBalancing policy (p. 512) objects.

Update requires: No interruption (p. 88)

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.

Required: No

Type: String

Update requires: Replacement (p. 88)

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. 88)

Subnets

A list of subnet IDs in your virtual private cloud (VPC) to attach to your load balancer. You can specify *either* AvailabilityZones or Subnets, 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: A list of strings

Update requires: Replacement (p. 88) 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. 88).

Tags

An arbitrary set of tags (key-value pairs) for this load balancer.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name. For example, `mystack-myelb-1WQN7BJGDB5YQ`.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Examples

A load balancer with a health check and access logs

```
"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.

```
"S3LoggingBucket": {
  "Type": "AWS::S3::Bucket"
},
"S3LoggingBucketPolicy": {
  "Type": "AWS::S3::BucketPolicy",
  "Properties": {
    "Bucket": {
      "Ref": "S3LoggingBucket"
    },
    "PolicyDocument": {
      "Version": "2008-10-17",
      "Statement": [ {
        "Sid": "ELBAccessLogs20130930",
        "Effect": "Allow",
        "Resource": {
          "Fn::Join": [
            "",
            [
              "arn:aws:s3:::",
              { "Ref": "S3LoggingBucket" },
              "/",
              "Logs",
              "/AWSLogs/",
              { "Ref": "AWS::AccountId" },
              "/*"
            ]
          ]
        },
        "Principal": "*",
        "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.

```
"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"
  }
 }
}
```

More examples

Examples of AWS CloudFormation templates can be viewed and downloaded from the AWS CloudFormation Sample Templates. These include:

- ELBSample.template: A load balancer with a health check.
- ELBStickinessSample.template: A load balancer example configured with cookie-based stickiness.
- ELBWithLockedDownEC2Instances.template: A load balancer with instances that receive traffic only from the load balancer.
- ELBWithLockedDownAutoScaledInstances.template: A load balancer with an auto scaling group that receives traffic only from the load balancer.
- ELBZoneApex.template: Maps a load balancer to a DNS zone apex.

See Also

- CreateLoadBalancer in the *Elastic Load Balancing API Reference*

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. 88).

Syntax

```
{
   "Type": "AWS::IAM::AccessKey",
   "Properties": {
      "Serial (p. 383)": Integer,
      "Status (p. 383)": String,
      "UserName (p. 383)": 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. 88)

Status

> The status of the access key.
>
> *Required*: Yes
>
> *Type*: String
>
> *Valid values:* `"Active" or "Inactive"`
>
> *Update requires*: No interruption (p. 88)

UserName

> The name of the user that the new key will belong to.
>
> *Required*: Yes
>
> *Type*: String
>
> *Update requires*: Replacement (p. 88)

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Template Examples

To view AWS::IAM::AccessKey snippets, see Declaring an IAM Access Key Resource (p. 179).

AWS::IAM::Group

The AWS::IAM::Group type creates an Identity and Access Management (IAM) group.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

Syntax

```
{
   "Type": "AWS::IAM::Group",
   "Properties": {
      "Path (p. 384)": String,
      "Policies (p. 384)": [ Policies, ... ]
   }
}
```

Properties

Path

The path to the group. For more information about paths, see Identifiers for IAM Entities in *Using IAM*.

Required: No

Type: String

Update requires: No interruption (p. 88)

Policies

The policies to associate with this group. For information about policies, see Overview of Policies in *Using IAM*.

Required: No

Type: List of IAM Policies (p. 514)

Update requires: No interruption (p. 88)

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Template Examples

To view AWS::IAM::Group snippets, see Declaring an IAM Group Resource (p. 180)

AWS::IAM::InstanceProfile

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*.

Syntax

```
{
   "Type": "AWS::IAM::InstanceProfile",
   "Properties": {
      "Path (p. 385)": String,
      "Roles (p. 386)": [ IAM Roles ]
   }
}
```

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*.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Roles

The roles associated with this IAM instance profile.

Required: Yes

Type: List of references to AWS::IAM::Roles. Currently, a maximum of one role can be assigned to an instance profile.

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name. For example:

```
{ "Ref": "MyProfile" }
```

For the IAM::InstanceProfile with the logical ID "MyProfile", `Ref` will return the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

Arn

Returns the Amazon Resource Name (ARN) for the instance profile. For example:

```
{"Fn::GetAtt" : ["MyProfile", "Arn"] }
```

This will return a value such as “`arn:aws:iam::1234567890:instance-profile/MyProfile-ASDNSDLKJ`”.

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Template Examples

Example 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.

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

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 Policies in *Using IAM*.

Syntax

```
{
   "Type": "AWS::IAM::Policy",
   "Properties": {
      "Groups (p. 388)" : [ String, ... ],
      "PolicyDocument (p. 388)" : JSON,
      "PolicyName (p. 388)" : String,
      "Roles (p. 388)" : [ String, ...
      "Users (p. 388)" : [ String, ... ],
   }
}
```

Properties

Groups

The names of groups to which you want to add the policy.

Required: Conditional

Type: A list of strings

Update requires: No interruption (p. 88)

PolicyDocument

A policy document that contains permissions to add to the specified users or groups.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 88)

PolicyName

The name of the policy.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Roles

The names of AWS::IAM::Role (p. 390)s to attach to this policy.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

Users

The names of users for whom you want to add the policy.

Required: Conditional

Type: A list of strings

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

IAM Policy with policy group

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

This snippet is from IAM_Users_Groups_and_Policies.template

IAM Policy with specified role

```
{
   "Type": "AWS::IAM::Policy",
   "Properties": {
      "PolicyName": "root",
      "PolicyDocument": {
         "Version" : "2012-10-17",
         "Statement": [
            { "Effect": "Allow", "Action": "*", "Resource": "*" }
         ]
      },
      "Roles": [ { "Ref": "RootRole" } ]
   }
}
```

This snippet is from auto_scaling_with_instance_profile.template.

To view more AWS::IAM::Policy snippets, see Declaring an IAM Policy (p. 181).

AWS::IAM::Role

Creates an AWS Identity and Access Management (IAM) role. An IAM role can be used to enable applications running on an Amazon 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*.

Syntax

```
{
   "Type": "AWS::IAM::Role",
   "Properties": {
      "AssumeRolePolicyDocument (p. 390)": { JSON },
      "Path (p. 390)": String,
      "Policies (p. 390)": [ Policies, ... ]
   }
}
```

Properties

AssumeRolePolicyDocument

The IAM assume role policy that is associated with this role.

Required: Yes

Type: A JSON policy document.

Update requires: No interruption (p. 88)

Note

You can associate only one assume role policy with a role. For an example of an assume role policy, see Template Examples (p. 392).

Path

The path associated with this role. For information about IAM paths, see Friendly Names and Paths in *Using IAM*.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Policies

The policies to associate with this role. Policies can also be specified externally. For sample templates that demonstrates both embedded and external policies, see Template Examples (p. 392).

Required: No

Type: List of IAM Policies (p. 514)

Update requires: No interruption (p. 88)

Notes on policies for IAM roles

For general information about IAM policies and policy documents, see How to Write a Policy in *Using IAM*.

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Template Examples

Example 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.

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

Example 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.

```
{
   "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. 178)

- AWS::IAM::InstanceProfile (p. 385)

AWS::IAM::User

The AWS::IAM::User type creates a user.

This type supports updates. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

Syntax

```
{
   "Type": "AWS::IAM::User",
   "Properties": {
      "Path (p. 394)": String,
      "Groups (p. 394)": [ String, ... ],
      "LoginProfile (p. 394)": { "Password" : String },
      "Policies (p. 395)": [ Policies, ... ]
   }
}
```

Properties

Path

The path for the user name. For more information about paths, see Identifiers for IAM Entities in Using AWS Identity and Access Management.

Required: No

Type: String

Update requires: No interruption (p. 88)

Groups

A name of a group to which you want to add the user.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

LoginProfile

Creates a login profile for the user so the user can access AWS services such as the AWS Management Console.

The LoginProfile type is an embedded property in the AWS::IAM::User type. The LoginProfile property contains a single field: *Password*, which takes a string as its value. For example:

```
"LoginProfile": { "Password": "myP@ssW0rd" }
```

Required: No

Type: LoginProfile type

Update requires: No interruption (p. 88)

Policies

The policies to associate with this user. For information about policies, see Overview of Policies in [Using IAM].

Required: No

Type: List of IAM Policies (p. 514)

Update requires: No interruption (p. 88)

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Template Examples

To view AWS::IAM::User snippets, see: Declaring an IAM User Resource (p. 178)

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. 88).

Syntax

```
{
   "Type": "AWS::IAM::UserToGroupAddition",
   "Properties": {
      "GroupName (p. 396)": String,
      "Users (p. 396)": [ User1, ... ]
   }
}
```

Properties

GroupName

The name of group to add users to.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Users

Required: Yes

Type: List of users

Update requires: No interruption (p. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Examples

To view AWS::IAM::UserToGroupAddition snippets, see Adding Users to a Group (p. 181).

AWS::Kinesis::Stream

Creates an Amazon Kinesis stream that captures and transports data records that are emitted from data sources. For specific information about creating streams, see CreateStream in the *Amazon Kinesis API Reference*.

Syntax

```
{
   "Type" : "AWS::Kinesis::Stream",
   "Properties" : {
      "ShardCount (p. 396)" : Integer
   }
}
```

Properties

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. 88)

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. 565).

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.

Syntax

```
{
  "Type" : "AWS::Logs::LogGroup",
  "Properties" : {
    "RetentionInDays (p. 397)" : Integer
  }
}
```

Properties

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

Examples

The following example creates a CloudWatch Logs log group that retains events for 7 days.

```
"myLogGroup": {
    "Type": "AWS::Logs::LogGroup",
    "Properties": {
        "RetentionInDays": 7
    }
}
```

For an additional sample template, see Amazon CloudWatch Logs Sample (p. 158).

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.

Syntax

```
{
  "Type": "AWS::Logs::MetricFilter",
  "Properties": {
    "FilterPattern (p. 398)": [String, ...],
    "LogGroupName (p. 398)": String,
    "MetricTransformations (p. 398)": [ 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: A list of strings

  Update requires: No interruption (p. 88)

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. 88)

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. 485)

> **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. 88)

Examples

The following example sends a value of `1` to the `404Count` metric whenever the status code field includes a `404` value.

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

For an additional sample template, see Amazon CloudWatch Logs Sample (p. 158).

AWS::OpsWorks::App

Defines an AWS OpsWorks app for an AWS OpsWorks stack. The app represents code that you want to run on an application server.

Syntax

```
{
  "Type": "AWS::OpsWorks::App",
  "Properties": {
    "AppSource (p. 400)" : Source,
    "Attributes (p. 400)" : { String:String, ... },
    "Description (p. 400)" : String,
    "Domains (p. 400)" :  [ String, ... ],
    "EnableSsl (p. 400)" : Boolean,
    "Name (p. 400)" : String,
    "Shortname (p. 400)" : String,
    "SslConfiguration (p. 401)" : { SslConfiguration },
    "StackId (p. 401)" : String,
    "Type (p. 401)" : String
  }
}
```

Properties

AppSource

Contains the information required to retrieve an app from a repository.

Required: No

Type: AWS OpsWorks Source Type (p. 517)

Update requires: No interruption (p. 88)

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. 88)

Description

A description of the app.

Required: No

Type: String

Update requires: No interruption (p. 88)

Domains

The app virtual host settings, with multiple domains separated by commas. For example, `'www.example.com, example.com'`.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

EnableSsl

Whether to enable SSL for this app.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

Name

The AWS OpsWorks app name.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Shortname

The app short name, which is used internally by AWS OpsWorks and by Chef recipes.

Required: No

Type: String

Update requires: Replacement (p. 88)

SslConfiguration

The SSL configuration

Required: No

Type: AWS OpsWorks SslConfiguration Type (p. 518)

Update requires: No interruption (p. 88)

StackId

The AWS OpsWorks stack ID that this app will be associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Type

The app type. Each supported type is associated with a particular layer. For more information, see CreateApp in the *AWS OpsWorks API Reference*.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name. For example:

```
{ "Ref": "myApp" }
```

For the AWS OpsWorks stack `myApp`, `Ref` returns the AWS OpsWorks app ID.

For more information about using the `Ref` function, see Ref (p. 565).

Template Snippet

The following snippet creates an AWS OpsWorks app that uses a PHP application in a Git repository:

```
"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"
    }
  }
}
```

See Also

- AWS::OpsWorks::Stack (p. 409)
- AWS::OpsWorks::Layer (p. 406)
- AWS::OpsWorks::Instance (p. 403)

AWS::OpsWorks::ElasticLoadBalancerAttachment

Attaches an Elastic Load Balancing load balancer to an AWS OpsWorks layer that you specify.

Syntax

```
{
  "Type": "AWS::OpsWorks::ElasticLoadBalancerAttachment",
  "Properties": {
    "ElasticLoadBalancerName (p. 402)" : String,
    "LayerId (p. 402)" : String
  }
}
```

Properties

ElasticLoadBalancerName
: Elastic Load Balancing load balancer name.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

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. 88)

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:

```
"ELBAttachment" : {
  "Type" : "AWS::OpsWorks::ElasticLoadBalancerAttachment",
    "Properties" : {
      "ElasticLoadBalancerName" : { "Ref" : "ELB" },
      "LayerId" : { "Ref" : "Layer" }
    }
}
```

See Also

- AWS::OpsWorks::Layer (p. 406)

AWS::OpsWorks::Instance

Creates an instance for an AWS OpsWorks stack. These instances represent the Amazon EC2 instances that, for example, handle the work of serving applications and balancing traffic.

Syntax

```
{
  "Type": "AWS::OpsWorks::Instance",
  "Properties": {
    "AmiId (p. 403)" : String,
    "Architecture (p. 403)" : String,
    "AvailabilityZone (p. 403)" : String,
    "InstallUpdatesOnBoot (p. 404)" : Boolean,
    "InstanceType (p. 404)" : String,
    "LayerIds (p. 404)" :  [ String, ... ],
    "Os (p. 404)" : String,
    "RootDeviceType (p. 404)" : String,
    "SshKeyName (p. 404)" : String,
    "StackId (p. 404)" : String,
    "SubnetId (p. 405)" : String
  }
}
```

Properties

AmiId

The ID of the custom AMI to be used to create the instance. The AMI should be based on one of the standard AWS OpsWorks APIs.

Required: No

Type: String

Update requires: Updates are not supported.

Architecture

The instance architecture.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

AvailabilityZone

The instance Availability Zone.

Required: No

Type: String

Update requires: Replacement (p. 88)

InstallUpdatesOnBoot
Whether to install operating system and package updates when the instance boots.

Required: No

Type: Boolean

Update requires: Some interruptions (p. 88)

InstanceType
The instance type, which must be supported by AWS OpsWorks. For more information, see CreateInstance in the *AWS OpsWorks API Reference*.

Required: Yes

Type: String

Update requires: Some interruptions (p. 88)

LayerIds
The IDs of the AWS OpsWorks layers that will be associated with this instance.

Required: Yes

Type: A list of strings

Update requires: Some interruptions (p. 88)

Os
The instance operating system. For more information, see CreateInstance in the *AWS OpsWorks API Reference*.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

RootDeviceType
The instance root device type.

Required: No

Type: String

Update requires: Replacement (p. 88)

SshKeyName
The instance SSH key name.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

StackId
The ID of the AWS OpsWorks stack that this instance will be associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Snippet

The following snippet creates two AWS OpsWorks instances that are associated with the `myStack` AWS OpsWorks stack and the `myLayer` AWS OpsWorks layer:

```
"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"
  }
}
```

See Also

- AWS::OpsWorks::Stack (p. 409)
- AWS::OpsWorks::Layer (p. 406)

- AWS::OpsWorks::App (p. 399)

AWS::OpsWorks::Layer

Creates an AWS OpsWorks layer. A layer defines, for example, which packages and applications are installed and how they are configured.

Syntax

```
{
  "Type": "AWS::OpsWorks::Layer",
  "Properties": {
    "Attributes (p. 406)" : { String:String, ... },
    "AutoAssignElasticIps (p. 406)" : Boolean,
    "AutoAssignPublicIps (p. 406)" : Boolean,
    "CustomInstanceProfileArn (p. 407)" : String,
    "CustomRecipes (p. 407)" : Recipes,
    "CustomSecurityGroupIds (p. 407)" :  [ String, ... ],
    "EnableAutoHealing (p. 407)" : Boolean,
    "InstallUpdatesOnBoot (p. 407)" : Boolean,
    "Name (p. 407)" : String,
    "Packages (p. 407)" : [ String, ... ],
    "Shortname (p. 408)" : String,
    "StackId (p. 408)" : String,
    "Type (p. 408)" : String,
    "VolumeConfigurations (p. 408)" : [ 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. 88)

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. 88)

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. 88)

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. 88)

CustomRecipes

Custom event recipes for this layer.

Required: No

Type: AWS OpsWorks Recipes Type (p. 516)

Update requires: No interruption (p. 88)

CustomSecurityGroupIds

Custom security group IDs for this layer.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

EnableAutoHealing

Whether to automatically heal Amazon EC2 instances that have become disconnected or timed out.

Required: Yes

Type: Boolean

Update requires: No interruption (p. 88)

InstallUpdatesOnBoot

Whether to install operating system and package updates when the instance boots.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

Name

The AWS OpsWorks layer name.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Packages

The packages for this layer.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

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 '.'.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

StackId

The ID of the AWS OpsWorks stack that this layer will be associated with.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

Type

The layer type. A stack cannot have more than one layer of the same type. For more information, see CreateLayer in the *AWS OpsWorks API Reference*.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

VolumeConfigurations

Describes the Amazon EBS volumes for this layer.

Required: No

Type: AWS OpsWorks VolumeConfiguration Type (p. 519)

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Snippet

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.

```
"myLayer": {
  "Type": "AWS::OpsWorks::Layer",
  "DependsOn": "myApp",
  "Properties": {
    "StackId": {"Ref": "myStack"},
 "Name": "PHP App Server",
    "Type": "php-app",
    "Shortname" : "php-app",
    "EnableAutoHealing" : "true",
    "AutoAssignElasticIps" : "false",
    "AutoAssignPublicIps" : "true",
    "Name": "MyPHPApp"
  }
}
```

See Also

- AWS::OpsWorks::Stack (p. 409)
- AWS::OpsWorks::App (p. 399)
- AWS::OpsWorks::Instance (p. 403)

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.

Syntax

```
{
  "Type" : "AWS::OpsWorks::Stack",
  "Properties" : {
    "Attributes (p. 410)" : { String:String, ... },
    "ChefConfiguration (p. 410)" : { ChefConfiguration },
    "ConfigurationManager (p. 410)" : { StackConfigurationManager },
    "CustomCookbooksSource (p. 410)" : { Source },
    "CustomJson (p. 410)" : JSON,
    "DefaultAvailabilityZone (p. 410)" : String,
    "DefaultInstanceProfileArn (p. 411)" : String,
    "DefaultOs (p. 411)" : String,
    "DefaultRootDeviceType (p. 411)" : String,
    "DefaultSshKeyName (p. 411)" : String,
    "DefaultSubnetId (p. 411)" : String,
    "HostnameTheme (p. 411)" : String,
    "Name (p. 412)" : String,
    "ServiceRoleArn (p. 412)" : String,
    "UseCustomCookbooks (p. 412)" : Boolean,
    "UseOpsworksSecurityGroups (p. 412)" : Boolean,
    "VpcId (p. 412)" : String
  }
}
```

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. 88)

ChefConfiguration

Describes the Chef configuration. For more information, see the CreateStack ChefConfiguration parameter in the *AWS OpsWorks 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. 516)

Update requires: No interruption (p. 88)

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 API Reference*.

Required: No

Type: AWS OpsWorks StackConfigurationManager Type (p. 519)

Update requires: No interruption (p. 88)

CustomCookbooksSource

Contains the information required to retrieve a cookbook from a repository.

Required: No

Type: AWS OpsWorks Source Type (p. 517)

Update requires: No interruption (p. 88)

CustomJson

A string that contains user-defined custom JSON. The custom JSON is used to override the corresponding default stack configuration JSON values. For more information, see CreateStack in the *AWS OpsWorks 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. 88)

DefaultAvailabilityZone

The stack's default Availability Zone, which must be in the specified region.

Required: No

Type: String

Update requires: No interruption (p. 88)

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. 88)

DefaultOs

The stack's default operating system. For more information, see CreateStack in the *AWS OpsWorks API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 88)

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 API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 88)

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. 88)

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. 88)

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 API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 88)

Name

The name of the AWS OpsWorks stack.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

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: No interruption (p. 88)

UseCustomCookbooks

Whether the stack uses custom cookbooks.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

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. 88)

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

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:

```
"myStack" : {
  "Type" : "AWS::OpsWorks::Stack",
  "Properties" : {
    "Name" : {"Ref":"OpsWorksStackName"},
    "ServiceRoleArn" : { "Fn::Join": ["", ["arn:aws:iam::", {"Ref":"AWS::Accoun
tId"}, ":role/aws-opsworks-service-role"]] },
    "DefaultInstanceProfileArn" : { "Fn::Join": ["", ["arn:aws:iam::",
{"Ref":"AWS::AccountId"}, ":instance-profile/aws-opsworks-ec2-role"]] },
    "DefaultSshKeyName" : {"Ref":"KeyName"}
  }
}
```

For a complete sample AWS OpsWorks template, see AWS OpsWorks Snippets (p. 190).

See Also

- AWS::OpsWorks::Layer (p. 406)
- AWS::OpsWorks::App (p. 399)
- AWS::OpsWorks::Instance (p. 403)

AWS::Redshift::Cluster

Creates an Amazon Redshift cluster. A cluster is a fully managed data warehouse that consists of set of compute nodes. For more information about default values and valid values, see CreateCluster in the *Amazon Redshift API Reference*.

Syntax

```
{
  "Type": "AWS::Redshift::Cluster",
  "Properties": {
    "AllowVersionUpgrade (p. 414)" : Boolean,
    "AutomatedSnapshotRetentionPeriod (p. 414)" : Integer,
    "AvailabilityZone (p. 414)" : String,
    "ClusterParameterGroupName (p. 414)" : String,
    "ClusterSecurityGroups (p. 414)" : [ String, ... ],
    "ClusterSubnetGroupName (p. 415)" : String,
    "ClusterType (p. 415)" : String,
    "ClusterVersion (p. 415)" : String,
    "DBName (p. 415)" : String,
    "ElasticIp (p. 415)" : String,
    "Encrypted (p. 415)" : Boolean,
    "HsmClientCertificateIdentifier (p. 415)" : String,
    "HsmConfigurationIdentifier (p. 416)" : String,
    "MasterUsername (p. 416)" : String,
    "MasterUserPassword (p. 416)" : String,
    "NodeType (p. 416)" : String,
    "NumberOfNodes (p. 416)" : Integer,
    "OwnerAccount (p. 416)" : String,
    "Port (p. 416)" : Integer,
    "PreferredMaintenanceWindow (p. 417)" : String,
    "PubliclyAccessible (p. 417)" : Boolean,
    "SnapshotClusterIdentifier (p. 417)" : String,
    "SnapshotIdentifier (p. 417)" : String,
    "VpcSecurityGroupIds (p. 417)" : [ String, ... ]
  }
}
```

Properties

AllowVersionUpgrade

When a new version of the Amazon Redshift is released, indicates 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. 88)

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. 88)

AvailabilityZone

The Amazon EC2 Availability Zone in which you want to provision your Amazon Redshift cluster. For example, if you have several Amazon 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. 88)

ClusterParameterGroupName

The name of the parameter group that you want to associate with this cluster.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

ClusterSecurityGroups

A list of security groups that you want to associate with this cluster.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

ClusterSubnetGroupName

The name of a cluster subnet group that you want to associate with this cluster.

Required: No

Type: String

Update requires: Replacement (p. 88)

ClusterType

The type of cluster. You can specify `single-node` or `multi-node`.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

ClusterVersion

The Amazon Redshift engine version that you want to deploy on the cluster.

Required: No

Type: String

Update requires: No interruption (p. 88)

DBName

The name of the first database that is created when the cluster is created.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

ElasticIp

The Elastic IP (EIP) address for the cluster.

Required: No

Type: String

Update requires: Replacement (p. 88)

Encrypted

Indicates whether the data in the cluster is encrypted at rest.

Required: No

Type: Boolean

Update requires: Replacement (p. 88)

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. 88)

HsmConfigurationIdentifier
: Specifies the name of the 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. 88)

MasterUsername
: The user name that is associated with the master user account for this cluster.

  Required: Yes

  Type: String

  Update requires: Replacement (p. 88)

MasterUserPassword
: The password associated with the master user account for this cluster.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

NodeType
: The node type that is provisioned for this cluster.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

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.

  Required: Conditional

  Type: Integer

  Update requires: No interruption (p. 88)

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. 88)

Port
: The port number on which the cluster accepts incoming connections.

  Required: No

  Type: Integer

  Update requires: Replacement (p. 88)

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. 88)

PubliclyAccessible

Indicates whether the cluster can be accessed from a public network.

Required: No

Type: Boolean

Update requires: Replacement (p. 88)

SnapshotClusterIdentifier

The name of the cluster the source snapshot was created from.

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. 88)

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. 88)

VpcSecurityGroupIds

A list of VPC security groups that are associated with this cluster.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

Endpoint.Address
The connection endpoint for the Amazon Redshift cluster. For example: `examplecluster.cg034hp-kmmjt.us-east-1.redshift.amazonaws.com` .

Endpoint.Port
The port number on which the Amazon Redshift cluster accepts connections. For example: `5439`.

Template Snippet

The following snippet describes a single-node Amazon Redshift cluster. The master user password is referenced from an input parameter that is in the same template.

```
"myCluster" : {
  "Type": "AWS::Redshift::Cluster",
  "Properties": {
    "MasterUsername" : "master",
    "MasterUserPassword" : { "Ref" : "MasterUserPassword" },
    "NodeType" : "dw.hs1.xlarge",
    "ClusterType" : "single-node"
  }
}
```

For a complete sample template, see Amazon Redshift Snippets (p. 193).

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.

Syntax

```
{
  "Type": "AWS::Redshift::ClusterParameterGroup",
  "Properties": {
    "Description (p. 418)" : String,
    "ParameterGroupFamily (p. 419)" : String,
    "Parameters (p. 419)" : [ Parameter, ... ]
  }
}
```

Properties

Description
A description of the parameter group.

Required: Yes

Type: String

Update requires: Replacement (p. 88)

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. 88)

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. 520)

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Snippets

The following snippet describes a parameter group with one parameter that is specified:

```
"myClusterParameterGroup" : {
  "Type" : "AWS::Redshift::ClusterParameterGroup",
  "Properties" : {
    "Description" : "My parameter group",
    "ParameterGroupFamily" : "redshift-1.0",
    "Parameters" : [ {
      "ParameterName" : "enable_user_activity_logging",
      "ParameterValue" : "true"
    }]
  }
}
```

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 ("). Use only single quotation marks (') in the JSON object.

```
"RedshiftClusterParameterGroup" : {
  "Type" : "AWS::Redshift::ClusterParameterGroup",
  "Properties" : {
    "Description" : "Cluster parameter group",
    "ParameterGroupFamily" : "redshift-1.0",
    "Parameters" : [{
      "ParameterName" : "wlm_json_configuration",
      "ParameterValue" : "[{'user_group':['ex
ample_user_group1'],'query_group':['example_query_group1'],'query_concur
rency':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.

Syntax

```
{
  "Type": "AWS::Redshift::ClusterSecurityGroup",
  "Properties": {
    "Description (p. 420)" : String
  }
}
```

Properties

Description
: A description of the security group.

  Required: Yes

  Type: String

  Update requires: Replacement (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Snippet

The following snippet creates an Amazon Redshift cluster security group that you can associate cluster security group ingress rules with:

```
"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. 421)

AWS::Redshift::ClusterSecurityGroupIngress

Specifies inbound (ingress) rules for an Amazon Redshift security group.

Syntax

```
{
  "Type": "AWS::Redshift::ClusterSecurityGroupIngress",
  "Properties": {
    "ClusterSecurityGroupName (p. 421)" : String,
    "CIDRIP (p. 421)" : String,
    "EC2SecurityGroupName (p. 421)" : String,
    "EC2SecurityGroupOwnerId (p. 422)" : 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. 88)

CIDRIP
: The IP address range that has inbound access to the Amazon Redshift security group.

  Required: No

  Type: String

  Update requires: Replacement (p. 88)

EC2SecurityGroupName
: The Amazon EC2 security group that will be added the Amazon Redshift security group.

Required: No

Type: String

Update requires: Replacement (p. 88)

EC2SecurityGroupOwnerId

The 12-digit AWS account number of the owner of the Amazon EC2 security group that is specified by the `EC2SecurityGroupName` parameter.

Required: No

Type: String

Update requires: Replacement (p. 88)

Template Snippet

The following snippet describes a ingress rules for an Amazon Redshift cluster security group:

```
"myClusterSecurityGroupIngressIP" : {
  "Type": "AWS::Redshift::ClusterSecurityGroupIngress",
  "Properties": {
    "ClusterSecurityGroupName" : {"Ref":"myClusterSecurityGroup"},
    "CIDRIP" : "10.0.0.0/16"
  }
}
```

See Also

- AWS::Redshift::ClusterSecurityGroup (p. 420)

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.

Syntax

```
{
  "Type": "AWS::Redshift::ClusterSubnetGroup",
  "Properties": {
    "Description (p. 422)" : String,
    "SubnetIds (p. 423)" : [ String, ... ]
  }
}
```

Properties

Description

A description of the subnet group.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

SubnetIds

A list of VPC subnet IDs. You can modify a maximum of 20 subnets.

Required: Yes

Type: A list of strings

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

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::RDS::DBInstance

The AWS::RDS::DBInstance type creates an Amazon RDS database instance. For detailed information about configuring RDS DB instances, see CreateDBInstance.

Important

If a DB instance is deleted or replaced during an update, all automated snapshots are deleted. However, manual DB snapshot are retained. 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. 96).

Syntax

```

{
   "Type" : "AWS::RDS::DBInstance",
   "Properties" :
   {
      "AllocatedStorage (p. 424)" : String,
      "AllowMajorVersionUpgrade (p. 424)" : Boolean,
      "AutoMinorVersionUpgrade (p. 425)" : Boolean,
      "AvailabilityZone (p. 425)" : String,
      "BackupRetentionPeriod (p. 425)" : String,
      "DBInstanceClass (p. 425)" : String,
      "DBInstanceIdentifier (p. 425)" : String,
      "DBName (p. 425)" : String,
      "DBParameterGroupName (p. 426)" : String,
      "DBSecurityGroups (p. 426)" : [ String, ... ],
      "DBSnapshotIdentifier (p. 426)" : String,
      "DBSubnetGroupName (p. 426)" : String,
      "Engine (p. 426)" : String,
      "EngineVersion (p. 427)" : String,
      "Iops (p. 427)" : Number,
      "LicenseModel (p. 427)" : String,
      "MasterUsername (p. 427)" : String,
      "MasterUserPassword (p. 427)" : String,
      "MultiAZ (p. 427)" : Boolean,
      "Port (p. 428)" : String,
      "PreferredBackupWindow (p. 428)" : String,
      "PreferredMaintenanceWindow (p. 428)" : String,
      "PubliclyAccessible (p. 428)" : Boolean,
      "SourceDBInstanceIdentifier (p. 428)" : String,
      "Tags (p. 429)" : [ Resource Tag, ..., ],
      "VPCSecurityGroups (p. 429)" : [ String, ... ]
   }
}
```

Properties

AllocatedStorage

The allocated storage size specified in gigabytes (GB).

If any value is used in the `Iops` parameter, `AllocatedStorage` must be at least 100 GB, which corresponds to the minimum `Iops` value of 1000. If `Iops` is increased (in 1000 IOPS increments), then `AllocatedStorage` must also be increased (in 100 GB increments) correspondingly.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

AllowMajorVersionUpgrade

Indicates whether major version upgrades are allowed. Changing this parameter does not result in an outage, and the change is applied asynchronously as soon as possible.

Constraints: This parameter must be set to `true` when you specify an `EngineVersion` that differs from the DB instance's current major version.

Required: No

Type: Boolean

Update requires: No interruption (p. 88)

AutoMinorVersionUpgrade

Indicates that minor engine upgrades will be applied automatically to the DB instance during the maintenance window. The default value is `true`.

Required: No

Type: Boolean

Update requires: No interruption (p. 88) or some interruptions (p. 88). 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. 88)

BackupRetentionPeriod

The number of days for which automatic DB snapshots are retained.

Important

If this DB instance is deleted or replaced during an update, all automated snapshots are deleted. However, manual DB snapshot are retained.

Required: No

Type: String

Update requires: No interruption (p. 88) or some interruptions (p. 88). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

DBInstanceClass

The name of the compute and memory capacity class of the DB instance.

Required: Yes

Type: String

Update requires: Some interruptions (p. 88)

DBInstanceIdentifier

A name for the DB instance. 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. 514).

Important

If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: String

Update requires: Replacement (p. 88)

DBName

The name of the initial database of this instance that was provided at create time, if one was specified. This same name is returned for the life of the DB instance.

Required: No

Type: String

Update requires: Replacement (p. 88)

DBParameterGroupName

The name of an existing DB parameter group or a reference to an AWS::RDS::DBParameterGroup (p. 432) resource created in the template.

Required: No

Type: String

Update requires: No interruption (p. 88) or some interruptions (p. 88). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*. Also, if any of the data members of the referenced parameter group are changed during an update, the database instance may need to be restarted, causing some interruption.

DBSecurityGroups

A list of the DB security groups to assign to the Amazon RDS instance. The list can include both the name of existing DB security groups or references to AWS::RDS::DBSecurityGroup (p. 435) resources created in the template.

If you set DBSecurityGroups, you must not set VPCSecurityGroups (p. 429), and vice-versa.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88)

DBSnapshotIdentifier

The identifier for the DB snapshot to restore from.

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, the database is created as a new database. If the property contains a value (other than empty string), AWS CloudFormation creates a database from the specified snapshot. If a snapshot with the specified name does not exist, the database creation fails and the stack rolls back.

Required: No

Type: String

Update requires: Replacement (p. 88)

DBSubnetGroupName

A DB subnet group to associate with the DB instance.

If there is no DB subnet group, then it is a non-VPC DB instance.

For more information about using Amazon RDS in a VPC, go to 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. 88)

Engine

The name of 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. 88)

EngineVersion

The version number of the database engine to use.

Required: No

Type: String

Update requires: Some interruptions (p. 88)

Iops

The number of I/O operations per second (IOPS) that the database should provision. This can be any integer value from 1000 to 10,000, in 1000 IOPS increments.

If any value is used in the *`Iops`* parameter, *`AllocatedStorage`* must be at least 100 GB, which corresponds to the minimum *`Iops`* value of 1000. If *`Iops`* is increased (in 1000 IOPS increments), then *`AllocatedStorage`* must also be increased (in 100 GB increments) correspondingly.

For more information about this parameter, see Provisioned IOPS Storage in the *Amazon Relational Database Service User Guide*.

Required: No

Type: Number

Update requires: No interruption (p. 88)

LicenseModel

The license model information for the DB instance.

Required: No

Type: String

Update requires: Replacement (p. 88).

MasterUsername

The master user name for the DB instance. This property is optional when you specify the `DBSnapshotIdentifier` property to create DB instances.

Required: Conditional

Type: String

Update requires: Replacement (p. 88).

MasterUserPassword

The master password for the DB instance. This property is optional when you specify the `DBSnapshotIdentifier` property to create DB instances.

Required: Conditional

Type: String

Update requires: No interruption (p. 88).

MultiAZ

Specifies if the DB 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. 88).

Port

The port for the instance.

Required: No

Type: String

Update requires: Replacement (p. 88).

PreferredBackupWindow

The daily time range during which automated backups are created if automated backups are enabled, as determined by the BackupRetentionPeriod.

Required: No

Type: String

Update requires: No interruption (p. 88).

PreferredMaintenanceWindow

The weekly time range (in UTC) during which system maintenance can occur.

Required: No

Type: String

Update requires: No interruption (p. 88) or some interruptions (p. 88). For more information, see ModifyDBInstance in the *Amazon Relational Database Service API Reference*.

PubliclyAccessible

Indicates whether the database instance is an Internet-facing instance. If you specify `true`, an instance is created with a publicly resolvable DNS name, which resolves to a public IP address. If you specify `false`, an internal instance is created 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*.

Required: No

Type: Boolean

Update requires: Replacement (p. 88).

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 certain 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 current template and then update your stack, the read replica is deleted and a new DB instance (not a read replica) is created.

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.

Important

Note the following:

- Read replicas do not support deletion policies. Any deletion policy that's associated with a read replica is ignored.
- You must create read replicas that are in the same region as the source DB instance. Currently, cross-region replicas are not supported.

Required: No

Type: String

Update requires: Replacement (p. 88).

Tags

An arbitrary set of tags (key–value pairs) for this RDS database instance.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88).

VPCSecurityGroups

A list of the VPC security groups to assign to the Amazon RDS instance. The list can include both the physical IDs of existing VPC security groups or references to AWS::EC2::SecurityGroup (p. 322) resources created in the template.

If you set VPCSecurityGroups, you must not set DBSecurityGroups (p. 426), and vice-versa.

Important

You can migrate a database instance in your stack from an RDS DB security group to a VPC security group, but you should keep the following points in mind:

- You cannot revert to using an RDS security group once you have established a VPC security group membership.
- When you migrate your DB instance to VPC security groups, if your stack update rolls back because of another failure in the database instance update, or because of an update failure in another AWS CloudFormation resource, the rollback will fail because it cannot revert to an RDS security group.

To avoid this situation, only migrate your DB instance to using VPC security groups when that is the *only* change in your stack template.

Required: No

Type: A list of strings

Update requires: No interruption (p. 88).

Updating and Deleting AWS:RDS::DBInstances

When updates are made to properties labeled "*Update requires*: Replacement (p. 88)", AWS CloudFormation first creates a replacement DB instance resource, then changes references from other dependent resources to point to the replacement resource, and finally deletes the old resource.

Caution

If you do not take a snapshot of the database before updating the stack, you will lose the data when your DB instance is replaced. To preserve your data, take the following precautions:

1. Deactivate any applications that are using the DB instance so that there is no activity against 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 update 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 on this resource, see ModifyDBInstance. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

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. 539).

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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. 558).

Examples

Example DBInstance with a set MySQL version, Tags and DeletionPolicy

This example shows how to set the MySQL version that has a `DeletionPolicy Attribute (p. 539)` 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.

```
"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"
}
```

Example DBInstance with provisioned IOPS

This example sets a provisioned IOPS value in the Iops (p. 427) property. Note that the AllocatedStorage (p. 424) property is set according to the 10:1 ratio between IOPS and GiBs of storage.

```
"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" }
   }
}
```

Example Read replica DBInstance

This example creates a read replica named `MyDBreadreplica` for the `MyDB` DB instance.

```
"MyDB" : {
   "Type" : "AWS::RDS::DBInstance",
   "Properties" : {
      "DBName" : { "Ref" : "DBName" },
      "AllocatedStorage" : { "Ref" : "DBAllocatedStorage" },
      "DBInstanceClass" : { "Ref" : "DBClass" },
      "Engine" : "MySQL",
      "EngineVersion" : "5.6",
      "MasterUsername" : { "Ref" : "DBUser" } ,
      "MasterUserPassword" : { "Ref" : "DBPassword" },
      "Port" : "5804",
      "Tags" : [{"Key" : "Role", "Value" : "Primary"}]
   }
},

"MyDBreadreplica" : {
   "Type": "AWS::RDS::DBInstance",
   "Properties": {
      "SourceDBInstanceIdentifier": { "Ref" : "MyDB" },
      "Port" : "5802",
      "Tags" : [{"Key" : "Role", "Value" : "ReadRep"}]
      }
   }
}
```

To view more AWS::RDS::DBInstance template snippets, see Amazon RDS Template Snippets (p. 197).

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. 423).

> **Note**
> Applying a ParameterGroup to a DBInstance may require the instance to reboot, resulting in a database outage for the duration of the reboot.

Syntax

```
{
   "Type": "AWS::RDS::DBParameterGroup",
   "Properties" : {
      "Description (p. 433)" : String,
      "Family (p. 433)" : String,
      "Parameters (p. 433)" : DBParameters,
      "Tags (p. 433)" : [ Resource Tag, ... ]
   }
}
```

Properties

Description

A friendly description of the RDS parameter group. For example, `"My Parameter Group"`.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Family

The database family of this RDS parameter group. For example, `"MySQL5.1"`.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

Parameters

The parameters to set for this RDS parameter group.

Required: No

Type: DBParameters, a JSON object consisting of key/value pairs of Strings. For example:

```
"Parameters" : {
   "Key1" : "Value1",
   "Key2" : "Value2",
   "Key3" : "Value3"
}
```

Update requires: No interruption (p. 88)

Tags

The tags that you want to attach to the RDS parameter group.

Required: No

Type: A list of resource tags (p. 521).

Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

AWS::RDS::DBSubnetGroup

The AWS::RDS::DBSubnetGroup type creates an RDS database subnet group. Subnet groups must contain at least one subnet in two availability zones in the region.

Syntax

```
{
   "Type" : "AWS::RDS::DBSubnetGroup",
   "Properties" : {
      "DBSubnetGroupDescription (p. 434)" : String,
      "SubnetIds (p. 434)" : [ String, ... ],
      "Tags (p. 434)" : [ Resource Tag, ... ]
   }
}
```

Properties

DBSubnetGroupDescription
: The description for the DB Subnet Group.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

SubnetIds
: The EC2 Subnet IDs for the DB Subnet Group.

  Required: Yes

  Type: A list of strings

  Update requires: No interruption (p. 88)

Tags
: The tags that you want to attach to the RDS database subnet group.

  Required: No

  Type: A list of resource tags (p. 521).

  Update requires: No interruption (p. 88)

Example

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "myDBSubnetGroup" : {
         "Type" : "AWS::RDS::DBSubnetGroup",
         "Properties" : {
            "DBSubnetGroupDescription" : "description",
            "SubnetIds" : [ "subnet-7b5b4112", "subnet-7b5b4115" ],
            "Tags" : [ {"key" : "value", "key2" : "value2"} ]
         }
      }
   }
}
```

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. 88)

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.

When you specify an AWS::RDS::DBSecurityGroup as an argument to the `Ref` function, AWS CloudFormation returns the value of the *`DBSecurityGroupName`*.

Syntax

```
{
   "Type" : "AWS::RDS::DBSecurityGroup",
   "Properties" :
   {
      "EC2VpcId (p. 435)" : { "Ref" : "myVPC" },
     "DBSecurityGroupIngress (p. 435)" : [ RDS Security Group Rule (p. 522) object
 1, ... ],
      "GroupDescription (p. 436)" : String,
      "Tags (p. 436)" : [ 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. 88)

DBSecurityGroupIngress

Network ingress authorization for an Amazon EC2 security group or an IP address range.

Type: List of RDS Security Group Rules (p. 522).

Required: Yes

Update requires: No interruption (p. 88)

GroupDescription

Description of the security group.

Type: String

Required: Yes

Update requires: Replacement (p. 88)

Tags

The tags that you want to attach to the Amazon RDS DB security group.

Required: No

Type: A list of resource tags (p. 521).

Update requires: No interruption (p. 88)

Template Examples

Tip

For more RDS template examples, see Amazon RDS Template Snippets (p. 197).

Single VPC security group

This template snippet creates/updates a single VPC security group, referred to by EC2SecurityGroupName.

```
"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.

```
{
   "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. 88).

For details about the settings for DB security group ingress, see AuthorizeDBSecurityGroupIngress.

Syntax

```
{
   "CIDRIP (p. 437)": String,
   "DBSecurityGroupName (p. 437)": String,
   "EC2SecurityGroupId (p. 438)": String,
   "EC2SecurityGroupName (p. 438)": String,
   "EC2SecurityGroupOwnerId (p. 438)": 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. 88)

DBSecurityGroupName

The name (ARN) of the AWS::RDS::DBSecurityGroup (p. 435) to which this ingress will be added.

Type: String

Required: Yes

Update requires: No interruption (p. 88)

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. 88)

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. 88)

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. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

For more information about using the `Ref` function, see Ref (p. 565).

See Also

- AuthorizeDBSecurityGroupIngress in the *Amazon Relational Database Service API Reference*

AWS::Route53::HealthCheck

You can 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*.

Syntax

```
{
  "Type" : "AWS::Route53::HealthCheck",
  "Properties" : {
    "HealthCheckConfig (p. 439)" : {HealthCheckConfig}
  }
}
```

Properties

HealthCheckConfig

An Amazon Route 53 health check.

Required: Yes

Type: Amazon Route 53 HealthCheck Configuration (p. 524)

Update requires: No interruption (p. 88)

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 in the *Amazon Route 53 API Reference*.

Syntax

```
{
  "Type" : "AWS::Route53::HostedZone",
  "Properties" : {
    "HostedZoneConfig (p. 439)" : {HostedZoneConfig},
    "Name (p. 439)" : String
  }
}
```

Properties

HostedZoneConfig

A complex type that contains an optional comment about your hosted zone.

Required: No

Type: Amazon Route 53 Hosted Zone Configuration Property (p. 526)

Update requires: Updates are not supported.

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: Updates are not supported.

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. 444) type. Note that some AWS::Route53::RecordSet properties are valid only when used within AWS::Route53::RecordSetGroup.

Note that before you use AWS CloudFormation to add a record set to hosted zone, that hosted zone must already be created in Amazon Route 53. AWS CloudFormation does not create new hosted zones.

For information about changing resource record sets, see Editing Resource Record Sets. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

For more information about constraints and values for each property, see POST CreateHostedZone for hosted zones and POST ChangeResourceRecordSet for resource record sets.

Syntax

```
{
  "Type" : "AWS::Route53::RecordSet",
  "Properties" : {
    "AliasTarget (p. 440)" : AliasTarget (p. 523),
    "Comment (p. 440)" : String,
    "Failover (p. 441)" : String,
    "GeoLocation (p. 441)" : { GeoLocation },
    "HealthCheckId (p. 441)" : String,
    "HostedZoneId (p. 441)" : String,
    "HostedZoneName (p. 441)" : String,
    "Name (p. 442)" : String,
    "Region (p. 442)" : String,
    "ResourceRecords (p. 442)" : [ String ],
    "SetIdentifier (p. 442)" : String,
    "TTL (p. 443)" : String,
    "Type (p. 443)" : String,
    "Weight (p. 443)" : 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. 523)

Update requires: No interruption (p. 88)

Comment

Any comments you want to include about the hosted zone.

Required: No

Type: String

Update requires: No interruption (p. 88)

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 element in the *Amazon Route 53 API Reference*.

Required: No

Type: String

Update requires: No interruption (p. 88)

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. 523)

Update requires: No interruption (p. 88)

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. 88)

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. 88)

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. 88)

Name

The name of the domain. This must be a fully specified domain, ending with a period as the last label indication. If you omit the final period, Amazon Route 53 assumes the domain is relative to the root.

Required: Yes

Type: String

Update requires: No interruption (p. 88)

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. Required if `TTL` or `SetIdentifier` is set. Also, if you set `ResourceRecords`, you must set `TTL` or `SetIdentifier`.

Note

If you are creating an alias resource record set, you should omit `ResourceRecords`.

Type: A list of strings

Update requires: No interruption (p. 88)

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 resource record set. `ResourceRecords` must also be set.

For more information about weighted resource record sets, see Setting Up Weighted Resource Record Sets in the *Amazon Route 53 Developer Guide*.

Type: String

Update requires: No interruption (p. 88)

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 *`TTL`* is specified, then *`ResourceRecords`* is also required.

Required: No

Type: String

Update requires: No interruption (p. 88)

Type

The type of records to add.

Required: Yes

Type: String

Valid Values: A | AAAA | CNAME | MX | NS | PTR | SOA | SPF | SRV | TXT

Update requires: No interruption (p. 88)

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. 88)

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. 565).

Example

Example Mapping an Amazon Route 53 A record to the public IP of an Amazon EC2 instance

```
"Resources" : {
   "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
         "ImageId" : { "Fn::FindInMap" : [
            "RegionMap", { "Ref" : "AWS::Region" }, "AMI"
         ] }
      }
   },
   "myDNSRecord" : {
      "Type" : "AWS::Route53::RecordSet",
      "Properties" : {
         "HostedZoneName" : {
            "Fn::Join" : [ "", [
               { "Ref" : "HostedZone" }, "."
            ] ]
         },
         "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" ] }
         ]
      }
   }
},
```

For additional AWS::Route53::RecordSet snippets, see Amazon Route 53 Template Snippets (p. 200) .

AWS::Route53::RecordSetGroup

Note

Before you use AWS CloudFormation to add a record set to a hosted zone, that hosted zone must already be created in Amazon Route 53. AWS CloudFormation does not create new hosted zones.

This type supports updates. For details and constraints on updates for resource record sets, see Changing Your Resource Record Sets. For more information about updating stacks, see AWS CloudFormation Stacks Updates (p. 88).

For more information about constraints and values for each property, see POST CreateHostedZone for hosted zones and POST ChangeResourceRecordSet for resource record sets.

Syntax

```
{
   "Type" : "AWS::Route53::RecordSetGroup",
   "Properties" : {
      "HostedZoneId (p. 445)" : String,
      "HostedZoneName (p. 445)" : String,
      "RecordSets (p. 445)" : [ RecordSet1, ... ],
      "Comment (p. 445)" : String,
   }
}
```

Properties

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. 88)

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. 88)

RecordSets

List of resource record sets to add.

Required: Yes

Type: list of AWS::Route53::RecordSet (p. 440)

Update requires: No interruption (p. 88)

Comment

Any comments you want to include about the hosted zone.

Required: No

Type: String

Update requires: No interruption (p. 88)

Return Value

When the logical ID of this resource is provided to the `Ref` intrinsic function, it 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. 565).

Template Examples

For AWS::Route53::RecordSetGroup snippets, see Amazon Route 53 Template Snippets (p. 200).

AWS::S3::Bucket

The AWS::S3::Bucket type creates an Amazon S3 bucket.

You can set a deletion policy for your bucket to control how AWS CloudFormation handles the bucket when the stack is deleted. For Amazon S3 buckets, you can choose to *retain* the bucket or to *delete* the bucket. For more information, see DeletionPolicy Attribute (p. 539).

> **Important**
> Only Amazon S3 buckets that are empty can be deleted. Deletion will fail for buckets that have contents.

Syntax

```
{
   "Type" : "AWS::S3::Bucket",
   "Properties" : {
      "AccessControl (p. 446)" : String,
      "BucketName (p. 446)" : String,
      "CorsConfiguration (p. 447)" : CORS Configuration,
      "LifecycleConfiguration (p. 447)" : Lifecycle Configuration,
      "LoggingConfiguration (p. 447)" : Logging Configuration,
      "NotificationConfiguration (p. 447)" : Notification Configuration,
      "Tags (p. 447)" : [ Resource Tag, ... ],
      "VersioningConfiguration (p. 447)" : Versioning Configuration,
      "WebsiteConfiguration (p. 448)" : 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.

Required: No

Type: String

Valid values: `Private` | `PublicRead` | `PublicReadWrite` | `AuthenticatedRead` | `LogDeliveryWrite` | `BucketOwnerRead` | `BucketOwnerFullControl`

Update requires: No interruption (p. 88)

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. 514). The bucket name must contain only lowercase letters, numbers, periods (.), and dashes (-).

Important

If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: String

Update requires: Replacement (p. 88)

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. 526)

Update requires: No interruption (p. 88)

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. 528)

Update requires: No interruption (p. 88)

LoggingConfiguration

Settings that defines where logs are stored.

Required: No

Type: Amazon S3 Logging Configuration (p. 530)

Update requires: No interruption (p. 88)

NotificationConfiguration

Configuration that defines which Amazon SNS topic to send messages to and what events to report.

Required: No

Type: Amazon S3 Notification Configuration (p. 531)

Update requires: No interruption (p. 88)

Tags

An arbitrary set of tags (key-value pairs) for this Amazon S3 bucket.

Required: No

Type: AWS CloudFormation Resource Tags (p. 521)

Update requires: No interruption (p. 88)

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. 531)

Update requires: No interruption (p. 88)

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. 532)

  Update requires: No interruption (p. 88)

Return Values

Ref

When the logical ID of this resource is provided to the `Ref` intrinsic function, it returns the resource name.

Example: mystack-mybucket-kdwwxmddtr2g

For more information about using the `Ref` function, see Ref (p. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

DomainName
: Returns the DNS name of the specified bucket.

  Example: mystack-mybucket-kdwwxmddtr2g.s3.amazonaws.com

WebsiteURL
: Amazon S3 website endpoint for the specified bucket.

  Example: http://mystack-mybucket-kdwwxmddtr2g.s3-website-us-east-1.amazonaws.com/

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Examples

Example Static website configuration with a routing rule

In this example, AWS::S3::Bucket's `Fn::GetAtt` values are used to provide outputs. The routing rule redirects requests to an Amazon EC2 instance in the event of an HTTP 404 error and inserts a object key prefix `report-404/` in the redirect. For example, if you request a page `ExamplePage.html` and it results in a HTTP 404 error, the request is routed to a page `report-404/ExamplePage.html` on the specified instance. For all other HTTP error codes, `error.html` is returned.

```
"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"
   }
}
```

Example Enable cross-origin resource sharing

The following sample template shows an Amazon S3 bucket with two cross-origin resource sharing rules.

```
{
  "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 en
abled."
        }
    }
}
```

Example Manage the lifecycle for Amazon S3 objects

The following sample template shows an Amazon 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.

```
{
    "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."
        }
    }
}
```

Example Log access requests for a specific bucket

The following sample template creates two Amazon S3 buckets. The `LoggingBucket` bucket store the logs from the `S3Bucket` bucket. The logging bucket requires log delivery write permissions in order receive logs from the `S3Bucket` bucket.

```
{
    "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."
        }
    }
}
```

Example Receive bucket notifications to an Amazon SNS topic

The following sample template shows an Amazon S3 bucket with a notification configuration that sends an event to the specified topic when Amazon S3 has lost all replicas of an object.

```
{
    "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 notific
ation configuration."
        }
    }
}
```

For more examples, see Amazon S3 Template Snippets (p. 203).

See Also

- DeletionPolicy Attribute (p. 539)
- 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. 182)

Syntax

```

{
   "Type" : "AWS::S3::BucketPolicy",
   "Properties" : {
      "Bucket (p. 454)" : String,
      "PolicyDocument (p. 454)" : 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.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 88)

Examples

Example 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`:

```
"SampleBucketPolicy" : {
  "Type" : "AWS::S3::BucketPolicy",
  "Properties" : {
    "Bucket" : {"Ref" : "myExampleBucket"},
    "PolicyDocument": {
      "Statement":[{
     "Action":["s3:GetObject"],
     "Effect":"Allow",
    "Resource": { "Fn::Join" : ["", ["arn:aws:s3:::", { "Ref" : "myExampleBuck
et" } , "/*" ]]},
     "Principal":"*",
        "Condition":{
          "StringLike":{
            "aws:Referer":[
              "http://www.example.com/*",
              "http://example.com/*"
            ]
          }
        }
      }]
    }
  }
}
```

AWS::SDB::Domain

The AWS::SDB::Domain type does not have any properties.

Updates are not supported for this resource.

When you specify an AWS::SDB::Domain type as an argument to the `Ref` function, AWS CloudFormation returns the value of the *`DomainName`*.

AWS::SNS::Topic

The AWS::SNS::Topic type creates an Amazon SNS topic.

Syntax

```
{
  "Type" : "AWS::SNS::Topic",
  "Properties" : {
    "DisplayName (p. 456)" : String,
    "Subscription (p. 456)" : [ SNS Subscription, ... ],
    "TopicName (p. 456)" : String
  }
}
```

Properties

Important
After you create an Amazon SNS topic, you cannot update its properties by using AWS CloudFormation. You can modify an Amazon SNS topic by using the AWS Management Console.

DisplayName
A developer-defined string that can be used to identify this SNS topic.

Required: No

Type: String

Update requires: Updates are not supported.

Subscription
The SNS subscriptions (endpoints) for this topic.

Required: No

Type: List of SNS Subscriptions (p. 536)

Update requires: Updates are not supported.

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. 514).

Required: No

Type: Name Type (p. 514)

Update requires: Updates are not supported.

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding return values.

TopicName
Returns the name for an Amazon SNS topic.

For more information about using `Fn:GetAtt`, see Fn::GetAtt (p. 558).

Examples

An example of an SNS topic subscribed to by two SQS queues:

```
"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.

Syntax

```
{
  "Type" : "AWS::SNS::TopicPolicy",
  "Properties" :
    {
      "PolicyDocument (p. 457)" : JSON,
      "Topics (p. 457)" : [ 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. 88)

Topics

The Amazon Resource Names (ARN) of the topics to which you want to add the policy. You can use the Ref function (p. 565) to specify an AWS::SNS::Topic (p. 455) resource.

Required: Yes

Type: A list of Amazon SNS topics ARNs

Update requires: No interruption (p. 88)

For sample `AWS::SNS::TopicPolicy` snippts, see Declaring an Amazon SNS Topic Policy (p. 182).

AWS::SQS::Queue

The AWS::SQS::Queue type creates an Amazon SQS queue.

Syntax

```
{
   "Type": "AWS::SQS::Queue",
   "Properties": {
      "DelaySeconds (p. 458)": Integer,
      "MaximumMessageSize (p. 458)": Integer,
      "MessageRetentionPeriod (p. 458)": Integer,
      "QueueName (p. 458)": String,
      "ReceiveMessageWaitTimeSeconds (p. 459)": Integer,
      "RedrivePolicy (p. 459)": RedrivePolicy,
      "VisibilityTimeout (p. 459)": Integer
   }
}
```

Properties

DelaySeconds

The time in seconds that the delivery of all messages in the queue will be 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. 88)

MaximumMessageSize

The limit of how many bytes 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. 88)

MessageRetentionPeriod

The number of seconds 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. 88)

QueueName

A name for the queue. 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. 514).

> **Important**
>
> If you specify a name, you cannot do updates that require this resource to be replaced. You can still do updates to this resource if the update requires no or some interruption.

Required: No

Type: Name Type (p. 514)

Update requires: Replacement (p. 88)

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. 88)

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. 536)

Update requires: No interruption (p. 88)

VisibilityTimeout

The length of time during which the queue will be unavailable once 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 no value is specified, the default value of 30 seconds will be used.

For more information about SQS Queue visibility timeouts, see Visibility Timeout in the *Amazon Simple Queue Service Developer Guide*.

Required: No

Type: Integer

Update requires: No interruption (p. 88)

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. 565).

Fn::GetAtt

`Fn::GetAtt` returns a value for a specified attribute of this type. This section lists the available attributes and corresponding 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

```
{
  "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 re
sources 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.

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

See Also

- 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 SQS queues.

AWS::SQS::QueuePolicy Snippet: Declaring an Amazon SQS Policy (p. 183)

Syntax

```
{
   "Type" : "AWS::SQS::QueuePolicy",
   "Properties" : {
      "PolicyDocument (p. 462)" : JSON,
      "Queues (p. 463)" : [ String, ... ]
   }
}
```

Properties

PolicyDocument

A policy document containing permissions to add to the specified SQS queues.

Required: Yes

Type: JSON object

Update requires: No interruption (p. 88)

Queues

The URLs of the queues to which you want to add the policy. You can use the Ref function (p. 565) to specify an AWS::SQS::Queue (p. 458) resource.

Required: Yes

Type: A list of strings

Update requires: No interruption (p. 88)

Resource Property Types Reference

This section details the resource-specific properties for the resources supported by AWS CloudFormation.

Topics

- Amazon SNS Subscription Property Type (p. 536)
- Amazon SQS RedrivePolicy (p. 536)

AWS CloudFormation AutoScaling Block Device Mapping Property Type

The AutoScaling Block Device Mapping type is an embedded property of the AWS::AutoScaling::LaunchConfiguration (p. 251) type.

Syntax

```
{
   "DeviceName (p. 465)" : String,
   "Ebs (p. 465)" : AutoScaling EBS Block Device,
   "NoDevice (p. 465)" : Boolean,
   "VirtualName (p. 465)" : String
}
```

Properties

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. 466).

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 `ephemeral`*X* 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. 465) type.

Syntax

```
{
   "DeleteOnTermination (p. 466)" : Boolean,
   "Iops (p. 466)" : Integer,
   "SnapshotId (p. 466)" : String,
   "VolumeSize (p. 466)" : Integer,
   "VolumeType (p. 466)" : String
}
```

Properties

DeleteOnTermination

Indicates whether to delete the volume when the instance is terminated. By default, Auto Scaling uses `true`.

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. 251).

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. 88)

VolumeType

The volume type. By default, Auto Scaling uses the `standard` volume type.

Required: No

Type: String

Examples

For AutoScaling EBS Block Device snippets, see Auto Scaling Launch Configuration Resource (p. 151).

Auto Scaling MetricsCollection

The `MetricsCollection` is a property of the AWS::AutoScaling::AutoScalingGroup (p. 245) 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

```
{
  "Granularity (p. 467)" : String,
  "Metrics (p. 467)" : [ 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: A list of strings

Auto Scaling NotificationConfiguration Property Type

The `NotificationConfiguration` property is an embedded property of the `AWS::AutoScaling::AutoScalingGroup (p. 245)` resource that specifies the events for which the Auto Scaling group sends notifications.

Syntax

```
{
   "NotificationTypes (p. 468)" : [ String, ... ],
   "TopicARN (p. 468)" : 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: A list of strings

TopicARN

The Amazon Resource Name (ARN) of the Amazon Simple Notification Service (SNS) topic.

Required: Yes

Type: String

Examples

For NotificationConfiguration snippets, see Auto Scaling Group with Notifications (p. 153).

Auto Scaling Tags Property Type

The Auto Scaling Tags property is an embedded property of the AWS::AutoScaling::AutoScalingGroup (p. 245) type. For more information about tags, go to Tagging Auto Scaling Groups and Amazon EC2 Instances in the *Auto Scaling Developer 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

```
{
   "Key (p. 468)" : String,
   "Value (p. 468)" : String,
   "PropagateAtLaunch (p. 469)" : 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"
      } ]
   }
}
```

CloudFormation Stack Parameters Property Type

The Parameters type is an embedded property of the AWS::CloudFormation::Stack (p. 278) 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:

```
"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"
   }
}
```

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:

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

CloudFront DistributionConfig

DistributionConfig is a property of the AWS::CloudFront::Distribution (p. 283) property that describes which Amazon CloudFront origin servers to get your files from when users request the files through your website or application.

Syntax

```
{
   "Aliases (p. 471)" : [ String, ... ],
   "CacheBehaviors (p. 471)" : [ CacheBehavior, ... ],
   "Comment (p. 471)" : String,
   "CustomErrorResponses (p. 471)" : [ CustomErrorResponse, ... ],
   "DefaultCacheBehavior (p. 471)" : DefaultCacheBehavior,
   "DefaultRootObject (p. 471)" : String,
   "Enabled (p. 471)" : Boolean,
   "Logging (p. 472)" : Logging,
   "Origins (p. 472)" : [ Origin, ... ],
   "PriceClass (p. 472)" : String,
   "Restrictions (p. 472)" : Restriction,
   "ViewerCertificate (p. 472)" : ViewerCertificate
}
```

Properties

Note

For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic in the *Amazon CloudFront API Reference*.

Aliases

CNAMEs (alternate domain names), if any, for the distribution.

Required: No

Type: A list of strings

CacheBehaviors

A list of CacheBehavior types for the distribution.

Required: No

Type: List of CacheBehavior (p. 472)

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. 474)

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. 475)

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

Logging
: Controls whether access logs are written for the distribution. To turn on access logs, specify this property.

  Required: No

  Type: Logging (p. 476) 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. 477).

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*.

  Required: No

  Type: String

Restrictions
: Specifies restrictions on who or how viewers can access your content.

  Required: No

  Type: CloudFront DistributionConfiguration Restrictions (p. 479)

ViewerCertificate
: The certificate to use when viewers use HTTPS to request objects.

  Required: No

  Type: CloudFront DistributionConfiguration ViewerCertificate (p. 481)

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

See Also

- DistributionConfig Complex Type in the *Amazon CloudFront API Reference*

CloudFront DistributionConfig CacheBehavior

`CacheBehavior` is a property of the DistributionConfig (p. 470) property that describes the Amazon CloudFront cache behavior when the requested URL matches a pattern.

Syntax

```
{
   "AllowedMethods (p. 473)" : [ String ],
   "ForwardedValues (p. 473)" : ForwardedValues,
   "MinTTL (p. 473)" : String,
   "PathPattern (p. 473)" : String,
   "SmoothStreaming (p. 473)" : Boolean,
   "TargetOriginId (p. 473)" : String,
   "TrustedSigners (p. 474)" : [ String, ... ],
   "ViewerProtocolPolicy (p. 474)" : String
}
```

Properties

Note

For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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"]` or `["DELETE", "GET", "HEAD", "OPTIONS", "PATCH", "POST", "PUT"]`.

Required: No

Type: A list of strings

ForwardedValues

Specifies how CloudFront handles query strings or cookies.

Required: Yes

Type: ForwardedValues (p. 482) type

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

PathPattern

The pattern for which this cache behavior applies to. For example, you can specify `images/*.jpg`.

When CloudFront receives an end-user request, the requested path is compared with path patterns in the order in which cache behaviors are listed in the stack specification for the distribution.

Required: Yes

Type: String

SmoothStreaming

Indicates whether to distribute media files in the Microsoft Smooth Streaming format by using the origin that is associated with this cache behavior. 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: A 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.

Required: Yes

Type: String

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfig CustomErrorResponse

`CustomErrorResponse` is a property of the CloudFront DistributionConfig (p. 470) resource that defines custom error messages for certain HTTP status codes.

Syntax

```
{
  "ErrorCachingMinTTL (p. 474)" : Integer,
  "ErrorCode (p. 474)" : Integer,
  "ResponseCode (p. 475)" : Integer,
  "ResponsePagePath (p. 475)" : String
}
```

Properties

Note

For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DefaultCacheBehavior

`DefaultCacheBehavior` is a property of the DistributionConfig (p. 470) property that describes the default cache behavior for an Amazon CloudFront distribution.

Syntax

```
{
  "AllowedMethods (p. 475)" : [ String, ... ],
  "ForwardedValues (p. 475)" : ForwardedValues,
  "MinTTL (p. 476)" : String,
  "SmoothStreaming (p. 476)" : Boolean,
  "TargetOriginId (p. 476)" : String,
  "TrustedSigners (p. 476)" : [ String, ... ],
  "ViewerProtocolPolicy (p. 476)" : String
}
```

Properties

Note

For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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"]` (the default) or `["DELETE", "GET", "HEAD", "OPTIONS", "PATCH", "POST", "PUT"]`.

Required: No

Type: A list of strings

ForwardedValues

Specifies how CloudFront handles query strings or cookies.

Required: Yes

Type: ForwardedValues (p. 482) type

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 distribute media files in the Microsoft Smooth Streaming format by using the origin that is associated with this cache behavior.

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: A 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.

Required: Yes

Type: String

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront Logging

`Logging` is a property of the DistributionConfig (p. 470) property that enables Amazon CloudFront to deliver access logs for each distribution to an Amazon Simple Storage Service (S3) bucket.

Syntax

```
{
  "Bucket (p. 477)" : String,
  "IncludeCookies (p. 477)" : Boolean,
  "Prefix (p. 477)" : String
}
```

Properties

Note
For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfig Origin

`Origin` is a property of the DistributionConfig (p. 470) property that describes an Amazon CloudFront distribution origin.

Syntax

```
{
  "CustomOriginConfig (p. 477)" : Custom Origin,
  "DomainName (p. 478)" : String,
  "Id (p. 478)" : String,
  "S3OriginConfig (p. 478)" : S3 Origin
}
```

Properties

Note
For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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. 478) type

DomainName

The DNS name of the Amazon 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

S3OriginConfig

Origin information to specify an Amazon 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. 479) type

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfig Origin CustomOrigin

CustomOrigin is a property of the Amazon CloudFront Origin (p. 477) property that describes an HTTP server.

Syntax

```
{
   "HTTPPort (p. 478)" : String,
   "HTTPSPort (p. 479)" : String,
   "OriginProtocolPolicy (p. 479)" : String
}
```

Properties

Note

For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic in the *Amazon CloudFront API Reference*.

HTTPPort

The HTTP port the custom origin listens on.

Required: No

Type: String

HTTPSPort

The HTTPS port the custom origin listens on.

Required: No

Type: String

OriginProtocolPolicy

The origin protocol policy to apply to your origin.

Required: Yes

Type: String

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfig Origin S3Origin

S3Origin is a property of the Origin (p. 477) property that describes the Amazon Simple Storage Service (S3) origin to associate with an Amazon CloudFront origin.

Syntax

```
{
   "OriginAccessIdentity (p. 479)" : String
}
```

Properties

Note

For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfiguration Restrictions

`Restrictions` is a property of the CloudFront DistributionConfig (p. 470) property that lets you limit which viewers can access your content.

Syntax

```
{
  "GeoRestriction (p. 480)" : GeoRestriction
}
```

Properties

Note
For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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. 480)

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfig Restrictions GeoRestriction

`GeoRestriction` is a property of the CloudFront DistributionConfiguration Restrictions (p. 479) property that describes the countries in which Amazon CloudFront allows viewers to access your content.

Syntax

```
{
  "Locations (p. 480)" : [ String, ... ],
  "RestrictionType (p. 480)" : String
}
```

Properties

Note
For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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: A 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

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront DistributionConfiguration ViewerCertificate

ViewerCertificate is a property of the CloudFront DistributionConfig (p. 470) property that specifies which certificate to use when viewers use HTTPS to request objects.

Syntax

```
{
  "CloudFrontDefaultCertificate (p. 481)" : Boolean,
  "IamCertificateId (p. 481)" : String,
  "SslSupportMethod (p. 481)" : String
}
```

Properties

Note
For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic in the *Amazon CloudFront API Reference*.

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 either this property or `IamCertificateId`.

Type: Boolean

IamCertificateId
The IAM certificate ID to use if you're using an alternate domain name.

Required: Conditional. You must specify either this property or `CloudFrontDefaultCertificate`.

Type: String

SslSupportMethod
Specifies how CloudFront serves HTTPS requests.

Required: Conditional. Required if you specified the `IamCertificateId` property.

Type: String

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront ForwardedValues

`ForwardedValues` is a property of the DefaultCacheBehavior (p. 475) and CacheBehavior (p. 472) properties that indicates whether Amazon CloudFront forwards query strings or cookies.

Syntax

```
{
  "Cookies (p. 482)" : Cookies,
  "Headers (p. 482)" : [ String, ... ],
  "QueryString (p. 482)" : Boolean
}
```

Properties

> **Note**
> For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic in the *Amazon CloudFront API Reference*.

Cookies
: Forwards specified cookies to the origin of the cache behavior.

  Required: No

  Type: CloudFront ForwardedValues Cookies (p. 482)

Headers
: Specifies the headers that you want Amazon CloudFront to forward to the origin for this cache behavior. 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.

  Required: No

  Type: A 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`.

  Required: Yes

  Type: Boolean

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudFront ForwardedValues Cookies

`Cookies` is a property of the CloudFront ForwardedValues (p. 482) that describes which cookies are forwarded to the Amazon CloudFront origin.

Syntax

```
{
  "Forward (p. 483)" : String,
  "WhitelistedNames (p. 483)" : [ String, ... ]
}
```

Properties

> **Note**
>
> For more information about the constraints and valid values of each property, see the elements table in the DistributionConfig Complex Type topic 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: A list of strings

Template Examples

To view AWS::CloudFront::Distribution snippets, see Amazon CloudFront Template Snippets (p. 154).

CloudWatch Metric Dimension Property Type

The Metric Dimension is an embedded property of the AWS::CloudWatch::Alarm (p. 287) 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

```

{
   "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. 565) and Fn::GetAtt (p. 558) 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*

CloudWatch Logs MetricFilter MetricTransformation Property

`MetricTransformation` is a property of the AWS::Logs::MetricFilter (p. 398) resource that describes how to transform log streams into a CloudWatch metric.

Syntax

```
{
  "MetricName (p. 485)": String,
  "MetricNamespace (p. 485)": String,
  "MetricValue (p. 485)": 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. 398) or Amazon CloudWatch Logs Sample (p. 158).

DynamoDB Attribute Definitions

A list of attribute definitions for the AWS::DynamoDB::Table (p. 291) resource. Each element is composed of an `AttributeName` and `AttributeType`.

Syntax

```
{
  "AttributeName (p. 486)" : String,
  "AttributeType (p. 486)" : 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

Examples

For an example, see AWS::DynamoDB::Table (p. 291).

DynamoDB Global Secondary Indexes

Describes global secondary indexes for the AWS::DynamoDB::Table (p. 291) resource.

Syntax

```
{
  "IndexName (p. 486)" : String,
  "KeySchema (p. 487)" : [ KeySchema, ... ],
  "Projection (p. 487)" : { Projection },
  "ProvisionedThroughput (p. 487)" : { 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. 487)

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. 489)

ProvisionedThroughput

The provisioned throughput settings for the index.

Required: Yes

Type: DynamoDB Provisioned Throughput (p. 489)

Examples

For an example of a declared global secondary index, see AWS::DynamoDB::Table (p. 291).

DynamoDB Key Schema

Describes a primary key for the AWS::DynamoDB::Table (p. 291) 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

```
{
  "AttributeName (p. 487)" : String,
  "KeyType (p. 488)" : "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. 291).

DynamoDB Local Secondary Indexes

Describes local secondary indexes for the AWS::DynamoDB::Table (p. 291) 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

```
{
  "IndexName (p. 488)" : String,
  "KeySchema (p. 488)" : [ KeySchema, ...],
  "Projection (p. 488)" : { 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. 487)

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. 489)

Examples

For an example of a declared local secondary index, see AWS::DynamoDB::Table (p. 291).

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

```
{
  "NonKeyAttributes (p. 489)" : [ String, ... ],
  "ProjectionType (p. 489)" : 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

Examples

For an example, see AWS::DynamoDB::Table (p. 291).

DynamoDB Provisioned Throughput

Describes a set of provisioned throughput values for an AWS::DynamoDB::Table (p. 291) 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

```
{
   "ReadCapacityUnits (p. 490)" : Number,
   "WriteCapacityUnits (p. 490)" : 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*.

Examples

For an example of declared provisioned throughput values, see AWS::DynamoDB::Table (p. 291).

Amazon EC2 Block Device Mapping Property

The Amazon EC2 block device mapping property is an embedded property of the AWS::EC2::Instance (p. 302) resource. For block device mappings for an Auto Scaling launch configuration, see AutoScaling Block Device Mapping (p. 465).

Syntax

```

{
   "DeviceName (p. 490)" : String,
   "Ebs (p. 491)" : EC2 EBS Block Device,
   "NoDevice (p. 491)" : {},
   "VirtualName (p. 491)" : 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. 492).

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 `ephemeral`*X* 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. 490) property.

Syntax

```
{
   "DeleteOnTermination (p. 492)" : Boolean,
   "Encrypted (p. 492)" : Boolean,
   "Iops (p. 492)" : Number,
   "SnapshotId (p. 493)" : String,
   "VolumeSize (p. 493)" : String,
   "VolumeType (p. 493)" : 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. 493) 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). This can be a number from 1 – 1024. If the volume type is `io1`, the minimum value is 10.

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. 88)

VolumeType

The volume type. You can specify `standard`, `io1`, or `gp2`. If you set the type to `io1`, you must also set the Iops property. For more information about these values and the default value, see CreateVolume 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*

EC2 ICMP Property Type

The EC2 ICMP property is an embedded property of the AWS::EC2::NetworkAclEntry (p. 311) type.

The following properties are available with the EC2 ICMP type.

Property	Type	Required	Notes
Code	Integer	Conditional	The Internet Control Message Protocol (ICMP) code. You can use -1 to specify all ICMP codes for the given ICMP type. Condition: Required if specifying 1 (ICMP) for the CreateNetworkAclEntry protocol parameter.
Type	Integer	Conditional	The Internet Control Message Protocol (ICMP) type. You can use -1 to specify all ICMP types. Condition: Required if specifying 1 (ICMP) for the CreateNetworkAclEntry protocol parameter.

EC2 MountPoint Property Type

The EC2 MountPoint property is an embedded property of the AWS::EC2::Instance (p. 302) type.

Syntax

```
{
   "Device (p. 494)" : String,
   "VolumeId (p. 494)" : 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. 302)
- AWS::EC2::Volume (p. 337)

EC2 NetworkInterface Embedded Property Type

The EC2 Network Interface type is an embedded property of the AWS::EC2::Instance (p. 302) type. It specifies a network interface that is to be attached.

Syntax

```
{
   "AssociatePublicIpAddress (p. 496)" : Boolean,
   "DeleteOnTermination (p. 496)" : Boolean,
   "Description (p. 496)" : String,
   "DeviceIndex (p. 496)" : String,
   "GroupSet (p. 496)" : [ String, ... ],
   "NetworkInterfaceId (p. 496)" : String,
   "PrivateIpAddress (p. 496)" : String,
   "PrivateIpAddresses (p. 496)" : [ PrivateIpAddressSpecification, ... ],
   "SecondaryPrivateIpAddressCount (p. 497)" : Integer,
   "SubnetId (p. 497)" : 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`. 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

The network interface ID.

Required: No

Type: String

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. 499)

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 Network Interface Association

Describes a network interface association for an Elastic Network Interface (ENI). AWS::EC2::NetworkInterface (p. 313) takes an object of this type in its Association property.

Syntax

```
{
   "AttachmentID" : String,
   "InstanceID" : String,
   "PublicIp" : String,
   "IpOwnerId" : String
}
```

Properties

AttachmentID

The ID of the network interface attachment.

Required: Yes

Type: String

InstanceID

The ID of the instance attached to the network interface.

Required: Yes

Type: String

PublicIp

The address of the Elastic IP address bound to the network interface.

Required: Yes

Type: String

IpOwnerId

The ID of the Elastic IP address owner.

Required: Yes

Type: String

EC2 Network Interface Attachment

Describes a network interface attachment for an Elastic Network Interface (ENI). AWS::EC2::NetworkInterface (p. 313) takes an object of this type in its Attachment property.

Syntax

```
{
   "AttachmentID" : String,
   "InstanceID" : String
}
```

Properties

AttachmentID

The ID of the network interface attachment.

Required: Yes

Type: String

InstanceID

The ID of the instance attached to the network interface.

Required: Yes

Type: String

EC2 Network Interface Group Item

Refers to an individual Amazon EC2 security group by ID or name in a group set. AWS::EC2::NetworkInterface (p. 313) takes a list of objects of this type in its GroupSet property.

Syntax

```
{
   "GroupId" : String,
   "GroupName" : String
}
```

Properties

Key

ID of the security group.

Required: Yes

Type: String

Value

Name of the security group.

Required: Yes

Type: String

EC2 Network Interface Private IP Specification

The `PrivateIpAddressSpecification` type is an embedded property of the AWS::EC2::NetworkInterface (p. 313) type.

Syntax

```
{
   "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 PortRange Property Type

The EC2 PortRange property is an embedded property of the AWS::EC2::NetworkAclEntry (p. 311) type.

The following properties are available with the EC2 PortRange type.

Property	Type	Required	Notes
From	Integer	Conditional	The first port in the range. Condition: Required if specifying 6 (TCP) or 17 (UDP) for the CreateNetworkAclEntry protocol parameter.
To	Integer	Conditional	The last port in the range. Condition: Required if specifying 6 (TCP) or 17 (UDP) for the CreateNetworkAclEntry protocol parameter.

EC2 Security Group Rule Property Type

The EC2 Security Group Rule is an embedded property of the AWS::EC2::SecurityGroup (p. 322) type.

Syntax SecurityGroupIngress

```
{
   "CidrIp (p. 500)" : String,
   "FromPort (p. 500)" : Integer,
   "IpProtocol (p. 500)" : String,
   "SourceSecurityGroupId (p. 501)" : String,
   "SourceSecurityGroupName (p. 501)" : String,
   "SourceSecurityGroupOwnerId (p. 501)" : String,
   "ToPort (p. 501)" : Integer
}
```

Syntax SecurityGroupEgress

```
{
   "CidrIp (p. 500)" : String,
   "FromPort (p. 500)" : Integer,
   "IpProtocol (p. 500)" : String,
   "DestinationSecurityGroupId (p. 500)" : String,
   "ToPort (p. 501)" : Integer
}
```

Properties

CidrIp

Specifies a CIDR range.

Type: String

Required: Conditional If you specify SourceSecurityGroupName or SourceSecurityGroupId, do not specify CidrIp.

DestinationSecurityGroupId (SecurityGroupEgress only)

Specifies the GroupId of the destination Amazon VPC security group.

Type: String

Required: Conditional Cannot be used when specifying a CIDR IP address.

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).

Type: Integer

Required: Yes

IpProtocol

An IP protocol name or number. For valid values, go to the IpProtocol parameter in AuthorizeSecurityGroupIngress

Type: String

Required: Yes

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.

Type: String

Required: Conditional. If you specify `CidrIp`, do not specify `SourceSecurityGroupId`.

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.

Type: String

Required: Conditional. If you specify CidrIp, do not specify SourceSecurityGroupName.

SourceSecurityGroupOwnerId (SecurityGroupIngress only)

Specifies the AWS Account ID of the owner of the Amazon EC2 Security Group that is 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.

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).

Type: Integer

Required: Yes

Examples

Security Group with CidrIp

```
"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

```
"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.

```
"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*

AWS Elastic Beanstalk Environment Tier Property Type

Describes the environment tier for an AWS::ElasticBeanstalk::Environment (p. 373) resource. For more information, see Environment Tiers in the *AWS Elastic Beanstalk Developer Guide*.

Syntax

```
{
   "Name (p. 503)" : String,
   "Type (p. 503)" : String,
   "Version (p. 503)" : String
}
```

Members

Name
: The name of the environment tier. You can specify `WebServer` or `Worker`.

  Required: No

  Type: String

  Update requires: Replacement (p. 88)

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. 88)

Version
: The version of this environment tier.

  Required: No

Type: String

Update requires: No interruption (p. 88)

Example

```
"Tier" : {
  "Type" : "SQS/HTTP",
  "Name" : "Worker",
  "Version" : "1.0"
}
```

AWS Elastic Beanstalk OptionSettings Property Type

`OptionSettings` is an embedded property of the AWS::ElasticBeanstalk::Environment (p. 373) and AWS::ElasticBeanstalk::ConfigurationTemplate (p. 371) resources. You use the `OptionSettings` property to specify an array of options for the AWS Elastic Beanstalk environment.

> **Note**
> You can get the set of valid settings for an AWS Elastic Beanstalk configuration by using the `elastic-beanstalk-describe-configuration-settings` command. For more information, go to elastic-beanstalk-describe-configuration-settings in the *AWS Elastic Beanstalk Developer Guide*.

Syntax

```

{
   "Namespace (p. 504)" : String,
   "OptionName (p. 504)" : String,
   "Value (p. 504)" : String
}
```

Members

Namespace
: A unique namespace identifying the option's associated AWS resource.

  Required: Yes

  Type: String

OptionName
: The name of the configuration option. For a list of options that can be used here, see Option Values in the *AWS Elastic Beanstalk Developer Guide*.

  Required: Yes

  Type: String

Value
: The value of the setting.

  Required: Yes

Type: String

Example

This example of using `OptionSettings` is found in the AWS CloudFormation sample template: ElasticBeanstalkSample.template, which also provides an example of its use within an `AWS::ElasticBeanstalk::Application`.

```
"OptionSettings" : [ {
   "Namespace" : "aws:autoscaling:launchconfiguration",
   "OptionName" : "EC2KeyName",
   "Value" : { "Ref" : "KeyName" }
} ]
```

See Also

- ConfigurationOptionSetting in the *AWS Elastic Beanstalk Developer Guide*
- Option Values in the *AWS Elastic Beanstalk Developer Guide*

AWS Elastic Beanstalk SourceBundle Property Type

The SourceBundle property is an embedded property of the AWS::ElasticBeanstalk::ApplicationVersion (p. 369) resource.

Syntax

```
{
   "S3Bucket (p. 505)" : String,
   "S3Key (p. 505)" : 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

Example

```
{
   "S3Bucket" : { "Fn::Join" :
      ["-", [ "elasticbeanstalk-samples", { "Ref" : "AWS::Region" } ] ] },
   "S3Key" : "samplefolder/php-sample.zip"
}
```

AWS Elastic Beanstalk SourceConfiguration Property Type

Use settings from another AWS Elastic Beanstalk configuration template for the AWS::ElasticBeanstalk::ConfigurationTemplate (p. 371) resource type.

Syntax

```
{
   "ApplicationName (p. 506)" : String,
   "TemplateName (p. 506)" : String
}
```

Members

ApplicationName
: The name of the AWS 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

Elastic Load Balancing AccessLoggingPolicy

The `AccessLoggingPolicy` property describes where and how access logs are stored for the AWS::ElasticLoadBalancing::LoadBalancer (p. 376) resource.

Syntax

```
{
  "EmitInterval (p. 507)" : Integer,
  "Enabled (p. 507)" : Boolean,
  "S3BucketName (p. 507)" : String,
  "S3BucketPrefix (p. 507)" : 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: Conditional. If you enable logging, you must specify a bucket name.

  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. 376) type.

Syntax

```
{
   "CookieName (p. 507)" : String,
   "PolicyName (p. 508)" : 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.

Required: Yes

Type: String

See Also

- AWS::ElasticLoadBalancing::LoadBalancer (p. 376)
- ElasticLoadBalancing Policy Type (p. 512)
- ElasticLoadBalancing LBCookieStickinessPolicy Type (p. 510)
- CreateAppCookieStickinessPolicyin the *Elastic Load Balancing API Reference*

Elastic Load Balancing ConnectionDrainingPolicy

The `ConnectionDrainingPolicy` property describes how deregistered or unhealthy instances handle in-flight requests for the AWS::ElasticLoadBalancing::LoadBalancer (p. 376) 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 Developer Guide*.

Syntax

```
{
   "Enabled (p. 508)" : Boolean,
   "Timeout (p. 508)" : 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

ElasticLoadBalancing HealthCheck Type

The ElasticLoadBalancing HealthCheck is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 376) type.

Syntax

```
{
   "HealthyThreshold (p. 509)" : String,
   "Interval (p. 509)" : String,
   "Target (p. 509)" : String,
   "Timeout (p. 509)" : String,
   "UnhealthyThreshold (p. 510)" : 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. 376) type.

Syntax

```
{
   "CookieExpirationPeriod (p. 510)" : String,
   "PolicyName (p. 510)" : 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.

See Also

- AWS::ElasticLoadBalancing::LoadBalancer (p. 376)
- ElasticLoadBalancing Policy Type (p. 512)
- ElasticLoadBalancing AppCookieStickinessPolicy Type (p. 507)
- CreateLBCookieStickinessPolicyin the *Elastic Load Balancing API Reference*

ElasticLoadBalancing Listener Property Type

The Listener property is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 376) type.

Syntax

```
{
   "InstancePort (p. 511)" : String,
   "InstanceProtocol (p. 511)" : String,
   "LoadBalancerPort (p. 511)" : String,
   "PolicyNames (p. 511)" :  [ String, ... ],
   "Protocol (p. 511)" : String,
   "SSLCertificateId (p. 512)" : String
}
```

Properties

InstancePort

Specifies the TCP port on which the instance server is listening. This property cannot be modified for 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. This property cannot be modified for the life of the load balancer.

Required: No

Type: String

Note

- If the front-end protocol is HTTP or HTTPS, *InstanceProtocol* has to be at the same protocol layer, i.e., HTTP or HTTPS. Likewise, if the front-end protocol is TCP or SSL, *InstanceProtocol* has to be TCP or SSL.
- If there is another listener with the same InstancePort whose *InstanceProtocol* is secure, i.e., HTTPS or SSL, the listener's *InstanceProtocol* has to be secure, i.e., HTTPS or SSL. If there is another listener with the same InstancePort whose *InstanceProtocol* is HTTP or TCP, the listener's *InstanceProtocol* must be either HTTP or TCP.

LoadBalancerPort

Specifies the external load balancer port number. This property cannot be modified for the life of the load balancer.

Required: Yes

Type: String

PolicyNames

A list of ElasticLoadBalancing policy (p. 512) names to associate with the listener.

Required: No

Type: A list of strings

Protocol

Specifies the load balancer transport protocol to use for routing — HTTP, HTTPS, TCP or SSL. This property cannot be modified for 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 documentation.

Required: No

Type: String

ElasticLoadBalancing Policy Type

The ElasticLoadBalancing policy type is an embedded property of the AWS::ElasticLoadBalancing::LoadBalancer (p. 376) resource. You associate policies with a listener (p. 510) by referencing a policy's name in the listener's *PolicyNames* property.

Syntax

```
{
   "Attributes (p. 512)" : [ { "Name" : String, "Value" : String }, ... ],
   "InstancePorts (p. 512)" : [ String, ... ],
   "LoadBalancerPorts (p. 512)" : [ String, ... ],
   "PolicyName (p. 512)" : String,
   "PolicyType (p. 513)" : String
}
```

Properties

Attributes

A list of arbitrary attributes for this policy.

Required: No

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 ELB listener.

```
"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/51Aohx5Vrpm
lfGHZCzciMBa",
               "fkHve+MQYYJcxmNUKMdsWnz9WtVfKxxWUU7Cfor4lorYmENGCG8FWqCoLD
MFs7pN",
               "yGEtpsrlKhzZWtgY1d7eGrUrBil03bI90E2KW0j4qAwGYAC8xix
OkNClicojeEz4",
               "f4rr3sUf+ZBSsuMEuwIDAQAB" ]
         ] }
      } ]
   }, {
      "PolicyName" : "MyBackendServerAuthenticationPolicy",
      "PolicyType" : "BackendServerAuthenticationPolicyType",
      "Attributes" : [
         { "Name" : "PublicKeyPolicyName", "Value" : "MyPublicKeyPolicy" } ],
      "InstancePorts" : [ "8443" ]
   }
]
```

This example shows a snippet of the Policies section of an ELB using proxy protocol.

```
"Policies" : [{
   "PolicyName" : "EnableProxyProtocol",
   "PolicyType" : "ProxyProtocolPolicyType",
   "Attributes" : [{
      "Name"  : "ProxyProtocol",
      "Value" : "true"
   }],
   "InstancePorts" : [{"Ref" : "WebServerPort"}]
}]
```

See Also

- AWS::ElasticLoadBalancing::LoadBalancer (p. 376)
- ElasticLoadBalancing AppCookieStickinessPolicy Type (p. 507)
- ElasticLoadBalancing LBCookieStickinessPolicy Type (p. 510)

IAM Policies

Policies is a property of the AWS::IAM::Role (p. 390), AWS::IAM::Group (p. 384), and AWS::IAM::User (p. 394) resources. The Policies property describes what actions are allowed on what resources. For more information about IAM policies, see Overview of Policies in *Using IAM*.

Syntax

```
{
  "PolicyDocument (p. 514)" : JSON,
  "PolicyName (p. 514)" : 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. 88)

PolicyName
: The name of the policy.

  Required: Yes

  Type: String

  Update requires: No interruption (p. 88)

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.

> **Important**
> You can't perform an update that causes a custom-named resource to be replaced.

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:

```
"myDynamoDBTable" : {
   "Type" : "AWS::DynamoDB::Table",
   "Properties" : {
      "KeySchema" : {
         "HashKeyElement": {
            "AttributeName" : "AttributeName1",
            "AttributeType" : "S"
         },
         "RangeKeyElement" : {
            "AttributeName" : "AttributeName2",
            "AttributeType" : "N"
         }
      },
      "ProvisionedThroughput" : {
         "ReadCapacityUnits" : "5",
         "WriteCapacityUnits" : "10"
      },
      "TableName" : "Sample Table"
   }
}
```

Do not manage stack resources outside of AWS CloudFormation. For example, if you rename an Amazon S3 bucket 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.

The following resources that support custom names:

- AWS::CloudWatch::Alarm (p. 287)
- AWS::DynamoDB::Table (p. 291)
- AWS::ElasticBeanstalk::Application (p. 368)
- AWS::ElasticBeanstalk::Environment (p. 373)
- AWS::ElasticLoadBalancing::LoadBalancer (p. 376)
- AWS::ElastiCache::CacheCluster (p. 360)
- AWS::RDS::DBInstance (p. 423)
- AWS::S3::Bucket (p. 446)
- AWS::SNS::Topic (p. 455)
- AWS::SQS::Queue (p. 458)

AWS OpsWorks ChefConfiguration Type

Describes the Chef configuration for the AWS::OpsWorks::Stack (p. 409) resource type. For more information, see ChefConfiguration in the *AWS OpsWorks API Reference*.

Syntax

```
{
  "BerkshelfVersion (p. 516)" : String,
  "ManageBerkshelf (p. 516)" : Boolean
}
```

Properties

BerkshelfVersion
: The Berkshelf version.

  Required: No

  Type: String

ManageBerkshelf
: Whether to enable Berkshelf.

  Required: No

  Type: Boolean

AWS OpsWorks Recipes Type

Describes custom event recipes for the AWS::OpsWorks::Layer (p. 406) 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

```
{
  "Configure (p. 516)" : [ String, ... ],
  "Deploy (p. 517)" : [ String, ... ],
  "Setup (p. 517)" : [ String, ... ],
  "Shutdown (p. 517)" : [ String, ... ],
  "Undeploy (p. 517)" : [ 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: A 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: A 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: A 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: A 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: A 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. 409) or AWS::OpsWorks::App (p. 399) resource types. For more information, see Source in the *AWS OpsWorks API Reference*.

Syntax

```
{
  "Password (p. 517)" : String,
  "Revision (p. 518)" : String,
  "SshKey (p. 518)" : String,
  "Type (p. 518)" : String,
  "Url (p. 518)" : String,
  "Username (p. 518)" : 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.

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. 399) resource type.

Syntax

```
{
  "Certificate (p. 518)" : String,
  "Chain (p. 519)" : String,
  "PrivateKey (p. 519)" : 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 StackConfigurationManager Type

Describes the stack configuration manager for the AWS::OpsWorks::Stack (p. 409) resource type. For more information, see StackConfigurationManager in the *AWS OpsWorks API Reference*.

Syntax

```
{
  "Name (p. 519)" : String,
  "Version (p. 519)" : String
}
```

Properties

Name
The name of the configuration manager.

Required: No

Type: String

Version
The Chef version.

Required: No

Type: String

AWS OpsWorks VolumeConfiguration Type

Describes the Amazon EBS volumes for the AWS::OpsWorks::Layer (p. 406) resource type.

Syntax

```
{
  "MountPoint (p. 520)" : String,
  "NumberOfDisks (p. 520)" : Number,
  "RaidLevel (p. 520)" : Number,
  "Size (p. 520)" : Number
}
```

Properties

MountPoint
: The volume mount point, such as `/dev/sdh`.

  Required: Yes

  Type: String

NumberOfDisks
: The number of disks in the volume.

  Required: Yes

  Type: Number

RaidLevel
: The volume RAID level.

  Required: No

  Type: Number

Size
: The volume size.

  Required: Yes

  Type: Number

Amazon Redshift Parameter Type

Describes parameters for the AWS::Redshift::ClusterParameterGroup (p. 418) resource type.

Syntax

```
{
  "ParameterName (p. 520)" : String,
  "ParameterValue (p. 520)" : 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. 243).

> **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

```
{
   "Key (p. 521)" : String,
   "Value (p. 521)" : String
}
```

Properties

Key

The key name of the tag. You can specify a value that is 1 to 128 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 128 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. 75)

- Viewing Stack Data and Resources (p. 77)

Amazon RDS Security Group Rule

The Amazon RDS security group rule is an embedded property of the AWS::RDS::DBSecurityGroup (p. 435) type.

Syntax

```
{
   "CIDRIP (p. 522)": String,
   "EC2SecurityGroupId (p. 522)": String,
   "EC2SecurityGroupName (p. 522)": String,
   "EC2SecurityGroupOwnerId (p. 522)": 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. 88)

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. 88)

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. 88)

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. 88)

Route 53 AliasTarget Property

`AliasTarget` is a property of the AWS::Route53::RecordSet (p. 440) resource.

For more information about alias resource record sets, see Creating Alias Resource Record Sets in the *Amazon Route 53 Developer Guide*.

Syntax

```
{
  "DNSName (p. 523)" : String,
  "EvaluateTargetHealth (p. 523)" : Boolean,
  "HostedZoneId (p. 523)" : String
}
```

Properties

DNSName
: The DNS name of the load balancer 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 of the load balancer that is the target of the alias.

  Type: String

  Required: Yes

Amazon Route 53 Record Set GeoLocation Property

The `GeoLocation` property is part of the AWS::Route53::RecordSet (p. 440) resource that describes how Amazon Route 53 responds to DNS queries based on the geographic location of the query.

Syntax

```
{
  "ContinentCode (p. 524)" : String,
  "CountryCode (p. 524)" : String,
  "SubdivisionCode (p. 524)" : 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 HealthCheck Configuration

The `HealthCheckConfig` property is part of the AWS::Route53::HealthCheck (p. 438) resource that describes a health check that Amazon Route 53 uses before responding to a DNS query.

Syntax

```
{
  "FailureThreshold (p. 525)" : Integer,
  "FullyQualifiedDomainName (p. 525)" : String,
  "IPAddress (p. 525)" : String,
  "Port (p. 525)" : Integer,
  "RequestInterval (p. 525)" : Integer,
  "ResourcePath (p. 525)" : String,
  "SearchString (p. 525)" : String,
  "Type (p. 526)" : String
}
```

Properties

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

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

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.

Required: No

Type: Integer

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 `HTTP_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, which 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

Amazon Route 53 Hosted Zone Configuration Property

The `HostedZoneConfig` property is part of the AWS::Route53::HostedZone (p. 439) resource that can contain a comment about the hosted zone.

Syntax

```
{
  "Comment (p. 526)" : String
}
```

Properties

Comment

Any comments that you want to include about the hosted zone.

Type: String

Required: No

Amazon S3 Cors Configuration

Describes the cross-origin access configuration for objects in an AWS::S3::Bucket (p. 446) resource.

Syntax

```
{
  "CorsRules (p. 526)" : [ CorsRules, ... ]
}
```

Properties

CorsRules

A set of origins and methods that you allow.

Required: Yes

Type: Amazon S3 Cors Configuration Rule (p. 527)

Amazon S3 Cors Configuration Rule

Describes cross-origin access rules for the Amazon S3 Cors Configuration (p. 526) property.

Syntax

```
{
  "AllowedHeaders (p. 527)" : [ String, ... ],
  "AllowedMethods (p. 527)" : [ String, ... ],
  "AllowedOrigins (p. 527)" : [ String, ... ],
  "ExposedHeaders (p. 527)" : [ String, ... ],
  "Id (p. 527)" : String,
  "MaxAge (p. 527)" : 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: A 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: A list of strings

AllowedOrigins

An origin that you allow to send cross-domain requests.

Required: Yes

Type: A 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: A 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. 446) resource.

Syntax

```
{
  "Rules (p. 528)" : [ Lifecycle Rule, ... ]
}
```

Properties

Rules
: A lifecycle rule for individual objects in a bucket.

  Required: Yes

  Type: Amazon S3 Lifecycle Rule (p. 528)

Amazon S3 Lifecycle Rule

Describes lifecycle rules for the Amazon S3 Lifecycle Configuration (p. 528) property.

Syntax

```
{
  "ExpirationDate (p. 528)" : String,
  "ExpirationInDays (p. 529)" : Integer,
  "Id (p. 529)" : String,
  "Prefix (p. 529)" : String,
  "Status (p. 529)" : String,
  "Transition (p. 529)" : 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.

  You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, or `Transition`. 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

  Type: String

ExpirationInDays

Indicates the number of days after creation when objects are deleted from Amazon S3 and Amazon Glacier.

You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, or `Transition`. 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

Type: Integer

Id

A unique identifier for this rule. The value cannot be more than 255 characters.

Required: No

Type: String

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

Describes when an object transitions to a specified storage class.

You must specify at least one of the following properties: `ExpirationDate`, `ExpirationInDays`, or `Transition`. 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 transition time must also be earlier than the expiration time.

Required: Conditional

Type: Amazon S3 Lifecycle Rule Transition (p. 529)

Amazon S3 Lifecycle Rule Transition

Describes when an object transitions to a specified storage class for the Amazon S3 Lifecycle Rule (p. 528) property.

Syntax

```
{
  "StorageClass (p. 530)" : String,
  "TransitionDate (p. 530)" : String,
  "TransitionInDays (p. 530)" : Integer
}
```

Properties

StorageClass

The storage class to which you want the object to transition. Currently, you can specify only `Glacier`.

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. 446) 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

```
{
  "DestinationBucketName (p. 530)" : String,
  "LogFilePrefix (p. 530)" : 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 Notification Configuration

Describes the notification configuration for an AWS::S3::Bucket (p. 446) resource.

Syntax

```
{
  "TopicConfigurations (p. 531)" : [ Topic Configuration, ... ]
}
```

Properties

TopicConfigurations
The topic to which notifications are sent and the events for which notification are generated.

Required: Yes

Type: Amazon S3 Notification Topic Configurations (p. 531)

Amazon S3 Notification Topic Configurations

Describes the topic and events for the Amazon S3 Notification Configuration (p. 531) property.

Syntax

```
{
  "Event (p. 531)" : String,
  "Topic (p. 531)" : String
}
```

Properties

Event
The Amazon S3 bucket event about which to send notifications. Currently, `s3:ReducedRedundancyLostObject` is the only event supported for notifications.

Required: Yes

Type: String

Topic
The Amazon SNS topic to which Amazon S3 reports the specified events.

Required: Yes

Type: String

Amazon S3 Versioning Configuration

Describes the versioning state of an AWS::S3::Bucket (p. 446) resource. For more information, see PUT Bucket versioning in the *Amazon Simple Storage Service API Reference*.

Syntax

```
{
  "Status (p. 532)" : String
}
```

Properties

Status
: The versioning state of an Amazon S3 bucket. If you enable versioning, you must suspend versioning to disable it.

  Required: Yes

  Type: String

Amazon S3 Website Configuration Property

WebsiteConfiguration is an embedded property of the AWS::S3::Bucket (p. 446) resource.

Syntax

```

"WebsiteConfiguration" : {
   "ErrorDocument (p. 532)" : String,
   "IndexDocument (p. 532)" : String,
   "RedirectAllRequestsTo (p. 532)" : Redirect all requests rule,
   "RoutingRules (p. 533)" : [ 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: No

  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. 533)

RoutingRules

Rules that define when a redirect is applied and the redirect behavior.

Required: No

Type: Amazon S3 Website Configuration Routing Rules Property (p. 534)

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. 532) property that describes the redirect behavior of all requests to a website endpoint of an Amazon S3 bucket.

Syntax

```
"RedirectAllRequestsTo" : {
   "HostName (p. 533)" : String,
   "Protocol (p. 533)" : 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. 532) property. This property describes the redirect behavior and when a redirect is applied.

Syntax

```
"RoutingRules" : {
   "RedirectRule (p. 534)" : Redirect rule,
   "RoutingRuleCondition (p. 534)" : 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. 534)

RoutingRuleCondition
Rules that define when a redirect is applied.

Required: No

Type: Amazon S3 Website Configuration Routing Rules Routing Rule Condition Property (p. 535)

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. 534) that describes how requests are redirected. In the event of an error, you can specify a different error code to return.

Syntax

```
"RedirectRule" : {
   "HostName (p. 535)" : String,
   "HttpRedirectCode (p. 535)" : String,
   "Protocol (p. 535)" : String,
   "ReplaceKeyPrefixWith (p. 535)" : String,
   "ReplaceKeyWith (p. 535)" : 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. 534) that describes a condition that must be met for a redirect to apply.

Syntax

```
"RoutingRuleCondition" : {
   "HttpErrorCodeReturnedEquals (p. 536)" : String,
   "KeyPrefixEquals (p. 536)" : 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 SNS Subscription Property Type

`Subscription` is an embedded property of the `AWS::SNS::Topic (p. 455)` resource that describes the subscription endpoints for a topic.

Syntax

```
{
   "Endpoint (p. 536)" : String,
   "Protocol (p. 536)" : 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. 458) resource.

Syntax

```
{
  "deadLetterTargetArn (p. 537)" : String,
  "maxReceiveCount (p. 537)" : 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: No

Type: String

maxReceiveCount

The number of times a message is delivered to the source queue before being sent to the dead letter queue.

Required: No

Type: Integer

Resource Attribute Reference

This section details the attributes that you can add to a resource to control additional behaviors and relationships.

Topics

CreationPolicy Attribute

You 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. 573) helper script or `SignalResource` API. AWS CloudFormation publishes valid signals to the stack events so that you track the number of signals sent.

Currently, the only AWS CloudFormation resources that support creation policies are AWS::AutoScaling::AutoScalingGroup (p. 245), AWS::EC2::Instance (p. 302), and AWS::CloudFormation::WaitCondition (p. 280).

The CreationPolicy attribute is helpful when you want to wait on resource configuration actions before stack creation proceeds. For example, if you install and configure software applications on an Amazon EC2 instance, you might want those applications up and 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. 224).

Syntax

```
"CreationPolicy" : {
  "ResourceSignal (p. 538)" : {
    "Count (p. 538)" : Integer,
    "Timeout (p. 538)" : String
  }
}
```

ResourceSignal Properties

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.

Default: `PT5M` (5 minutes)

Type: String

Required: No

Examples

The following example shows how to add a creation policy to an Auto Scaling group. The creation policy requires 3 success signals and times out after 5 minutes.

```
"AutoScalingGroup": {
  "Type": "AWS::AutoScaling::AutoScalingGroup",
  "Properties": {
    "AvailabilityZones": { "Fn::GetAZs": "" },
    "LaunchConfigurationName": { "Ref": "LaunchConfig" },
    "DesiredCapacity": "3",
    "MinSize": "1",
    "MaxSize": "4"
  },
  "CreationPolicy": {
    "ResourceSignal": {
      "Count": "3",
      "Timeout": "PT5M"
    }
  },
  "UpdatePolicy" : {
    "AutoScalingScheduledAction" : {
      "IgnoreUnmodifiedGroupSizeProperties" : "true"
    },
    "AutoScalingRollingUpdate" : {
      "MinInstancesInService" : "1",
      "MaxBatchSize" : "2",
      "PauseTime" : "PT1M",
      "WaitOnResourceSignals" : "true"
    }
  }
},
"LaunchConfig": {
  "Type": "AWS::AutoScaling::LaunchConfiguration",
  "Properties": {
    "ImageId": "ami-018c9568",
    "EbsOptimized": "true",
    "InstanceType": "c3.xlarge",
    "UserData": {
      "Fn::Base64": {
        "Fn::Join" [ "", [
          "/opt/aws/bin/cfn-signal -e 0 --stack ", { "Ref": "AWS::StackName"
}, " --resource AutoScalingGroup\n"
        ] ]
      }
    }
  }
}
```

The following example shows how to add a creation policy to a wait condition.

```
"WaitCondition" : {
    "Type" : "AWS::CloudFormation::WaitCondition",
    "CreationPolicy" : {
        "ResourceSignal" : {
            "Timeout" : "PT5M",
            "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 an Amazon S3 bucket or an Amazon 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.

```
{
  "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::RDS::DBInstance`, 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.

DeletionPolicy Updates

During a stack update, you cannot add or update a DeletionPolicy by itself. You can add or update a DeletionPolicy only when you include changes that add, modify, or delete resources. If you need to add or modify a DeletionPolicy and don't want to make any changes to a resource, you can use a dummy resource, such as `AWS::CloudFormation::WaitConditionHandle`.

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. 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. 220).
- 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. 542).
- 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.

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. 302) resource with a `DependsOn` attribute that specifies myDB, an AWS::RDS::DBInstance (p. 423). When AWS CloudFormation creates this stack, it first creates myDB, then creates Ec2Instance.

```

{
    "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

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 group
- Amazon EC2 instances
- Elastic Load Balancing load balancers
- Elastic IP address

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:

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

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 format to the resource declaration. In addition, you can use intrinsic functions (such as GetAtt (p. 558) and Ref (p. 565)), parameters, and pseudo parameters within the Metadata attribute to add those interpreted values.

> **Note**
>
> AWS CloudFormation does not validate the JSON in 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.

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MyS3Bucket" : {
         "Type" : "AWS::S3::Bucket",
         "Metadata" : { "Object1" : "Location1",  "Object2" : "Location2" }
      }
   }
}
```

UpdatePolicy Attribute

You can use the UpdatePolicy attribute to specify how AWS CloudFormation handles updates to the AWS::AutoScaling::AutoScalingGroup (p. 245) resource.

The update policy is invoked under the following conditions:

- The `AutoScalingRollingUpdate` policy is applied when you make a change to the Auto Scaling placement group, launch configuration, or subnet group membership of the Auto Scaling group.
- The `AutoScalingScheduledAction` policy is applied when you update a stack that includes an Auto Scaling group with an associated scheduled action.

Syntax

```
"UpdatePolicy" : {
  "AutoScalingRollingUpdate (p. 544)" : {
    "MaxBatchSize (p. 544)" : String,
    "MinInstancesInService (p. 544)" : String,
    "PauseTime (p. 544)" : String,
    "SuspendProcesses (p. 544)" : [ List of processes ],
    "WaitOnResourceSignals (p. 545)" : Boolean
  },
  "AutoScalingScheduledAction (p. 545)" : {
    "IgnoreUnmodifiedGroupSizeProperties (p. 545)" : Boolean
  }
}
```

AutoScalingRollingUpdate Properties

You can use the AutoScalingRollingUpdate policy to specify how AWS CloudFormation handles rolling updates for a particular resource.

> **Important**
> If you have enabled rolling updates and scheduled actions, 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 Developer Guide*.

MaxBatchSize

The maximum number of instances that are terminated at a given time.

Default: `1`

Type: String

Required: No

MinInstancesInService

The minimum number of instances that must be in service within the autoscaling group while obsolete instances are being terminated.

Default: `0`

Type: String

Required: No

PauseTime

The amount of time to pause after AWS CloudFormation makes a change to the Auto Scaling group before making additional changes to a resource. For example, the amount of time to pause before adding or removing autoscaling instances when scaling up or terminating instances in an Auto Scaling group.

If you specify the `WaitOnResourceSignals` property, the amount of time to wait until the Auto Scaling group receives the required number of valid signals. If the pause time is exceeded before theAuto Scaling group receives the required number of signals, the update times out and fails.

The value must be in ISO8601 duration format, in the form: "PT#H#M#S", where each # is the number of hours, minutes, and/or seconds, respectively. The maximum amount of time that can be specified for the pause time is one hour ("PT1H").

Default: `PT0S` (zero seconds)

Type: String

Required: No

SuspendProcesses

The Auto Scaling processes to suspend during a stack update. Suspending processes is useful when you don't want Auto Scaling to potentially interfere with a stack update. For example, you can suspend process so that no alarms are triggered during an update. For valid values, see SuspendProcesses in the *Auto Scaling API Reference*.

Default: Not specified

Type: List of Auto Scaling processes

Required: No

WaitOnResourceSignals

Indicates whether the Auto Scaling group waits on signals during an update. AWS CloudFormation suspends the update of an Auto Scaling group after any new Amazon EC2 instances are launched into the group. AWS CloudFormation must receive a signal from each new instance within the specified pause time before AWS CloudFormation continues the update. You can use the cfn-signal (p. 573) helper script or `SignalResource` API to signal the Auto Scaling group. This property is useful when you want to ensure instances have completed installing and configuring applications before the Auto Scaling group update proceeds.

Default: `false`

Type: Boolean

Required: No

AutoScalingScheduledAction Properties

When the `AWS::AutoScaling::AutoScalingGroup` resource has an associated scheduled action, the AutoScalingScheduledAction policy describes how AWS CloudFormation handles updates for the `MinSize`, `MaxSize`, and `DesiredCapacity` properties..

With scheduled actions, the group size properties (minimum size, maximum size, and desired capacity) of an Auto Scaling group can change at any time. Whenever you update a stack with an Auto Scaling group and scheduled action, AWS CloudFormation always sets 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. 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 the AutoScalingScheduledAction update policy to prevent AWS CloudFormation from changing the min size, max size, or desired capacity unless you modified the individual values in your template.

IgnoreUnmodifiedGroupSizeProperties

During a stack update, indicates whether AWS CloudFormation ignores any group size property differences between your current Auto Scaling group and the Auto Scaling group that is described in the `AWS::AutoScaling::AutoScalingGroup` resource of your template. However, if you modified any group size property values in your template, AWS CloudFormation will always use the modified values and update your Auto Scaling group.

Default: `false`

Type: Boolean

Required: No.

Examples

Add an UpdatePolicy to an Auto Scaling Group

The following example shows how to adds an update policy. During an update, the Auto Scaling group will update instances in batches of two and keep a minimum of one instance in service. With the `WaitOnResourceSignals` flag, 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 proceeds to update the next batch of instances.

```
"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"
   }
}
```

Intrinsic Function Reference

AWS CloudFormation provides several built-in functions that help you manage your stacks.

Note

You can use intrinsic functions only in specific parts of a template. Currently, you can use intrinsic functions in resource properties, 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

{ "Fn::Base64" : *`valueToEncode`* }

Parameters

valueToEncode
: The string value you want to convert to Base64.

Return Value:

The original string, in Base64 representation.

Example

```
{ "Fn::Base64" : "AWS CloudFormation" }
```

See Also

- Intrinsic Function Reference (p. 546)

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. 124).

> **Note**
> You can only reference other conditions and values from the Parameters and Mappings sections of a template. For example, you cannot reference the logical ID of a resource in a condition, but you can reference a value from an input parameter.

Topics

Reference a condition

When you refer to a condition in another condition or associate the condition with a resource, you use the `Condition:` key. For the `Fn::If` function, you only need to specify the condition name.

The following snippet is from the Conditions section of a template. The `MyAndCondition` condition refers to the `SomeOtherCondition` condition:

```
"MyAndCondition": {
   "Fn::And": [
      {"Fn::Equals": ["sg-mysggroup", {"Ref": "ASecurityGroup"}]},
      {"Condition": "SomeOtherCondition"}
   ]
}
```

The following snippet is from the Resources section of a template that shows how to associate a resource with a condition. The `NewVolume` resource is associated with the `CreateProdResources` condition.

```
"NewVolume" : {
  "Type" : "AWS::EC2::Volume",
  "Condition" : "CreateProdResources",
  "Properties" : {
     "Size" : "100",
     "AvailabilityZone" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone"
 ]}
}
```

The following snippet is shows how to use `Fn::If` in order to conditionally specify a resource property value for the `NewVolume` resource. 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.

```
"NewVolume" : {
  "Type" : "AWS::EC2::Volume",
  "Properties" : {
    "Size" : {
      "Fn::If" : [
        "CreateLargeSize",
        {"Ref" : "100"},
        {"Ref" : "10"}
      ]},
    "AvailabilityZone" : { "Fn::GetAtt" : [ "Ec2Instance", "AvailabilityZone"
]}
  },
  "DeletionPolicy" : "Snapshot"
}
```

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

```
"Fn::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:

```
"MyAndCondition": {
   "Fn::And": [
      {"Fn::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

```
"Fn::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`:

```
"UseProdCondition" : {
   "Fn::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

```
"Fn::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

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`.

```
"SecurityGroups" : [{
  "Fn::If" : [
    "CreateNewSecurityGroup",
    {"Ref" : "NewSecurityGroup"},
    {"Ref" : "ExistingSecurityGroup"}
  ]
}]
```

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.

```
"Outputs" : {
  "SecurityGroupId" : {
    "Description" : "Group ID of the security group used.",
    "Value" : {
      "Fn::If" : [
        "CreateNewSecurityGroup",
        {"Ref" : "NewSecurityGroup"},
        {"Ref" : "ExistingSecurityGroup"}
      ]
    }
  }
}
```

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.

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

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.

```
"UpdatePolicy": {
  "AutoScalingRollingUpdate": {
    "Fn::If": [
      "RollingUpdates",
      {
        "MaxBatchSize": "2",
        "MinInstancesInService": "2",
        "PauseTime": "PT0M30S"
      },
      {
        "Ref" : "AWS::NoValue"
      }
    ]
  }
}
```

To view additional samples, see Sample Templates (p. 553).

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

```
"Fn::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`:

```
"MyNotCondition" : {
   "Fn::Not" : [{
      "Fn::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

```
"Fn::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:

```
"MyOrCondition" : {
   "Fn::Or" : [
      {"Fn::Equals" : ["sg-mysggroup", {"Ref" : "ASecurityGroup"}]},
      {"Condition" : "SomeOtherCondition"}
   ]
}
```

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:

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

  "Mappings" : {
    "RegionMap" : {
      "us-east-1"      : { "AMI" : "ami-aecd60c7"},
      "us-west-1"      : { "AMI" : "ami-734c6936"},
      "us-west-2"      : { "AMI" : "ami-48da5578"},
      "eu-west-1"      : { "AMI" : "ami-6d555119"},
      "sa-east-1"      : { "AMI" : "ami-fe36e8e3"},
      "ap-southeast-1" : { "AMI" : "ami-3c0b4a6e"},
      "ap-southeast-2" : { "AMI" : "ami-bd990e87"},
      "ap-northeast-1" : { "AMI" : "ami-2819aa29"}
    }
  },

  "Parameters" : {
    "EnvType" : {
      "Description" : "Environment type.",
      "Default" : "test",
      "Type" : "String",
      "AllowedValues" : ["prod", "dev", "test"],
      "ConstraintDescription" : "must specify prod, dev, or test."
    }
  },

  "Conditions" : {
    "CreateProdResources" : {"Fn::Equals" : [{"Ref" : "EnvType"}, "prod"]},
    "CreateDevResources" : {"Fn::Equals" : [{"Ref" : "EnvType"}, "dev"]}
  },

  "Resources" : {
    "EC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region"
 }, "AMI" ]},
        "InstanceType" : { "Fn::If" : [
          "CreateProdResources",
          "c1.xlarge",
          {"Fn::If" : [
            "CreateDevResources",
            "m1.large",
            "m1.small"
          ]}
        ]}
      }
    },

    "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"
 ]}
      }
    }
  }
}
```

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.

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

  "Parameters": {
    "DBUser": {
      "NoEcho": "true",
      "Description" : "The database admin account username",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "must begin with a letter and contain only al
phanumeric characters."
    },
    "DBPassword": {
      "NoEcho": "true",
      "Description" : "The database admin account password",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern" : "[a-zA-Z0-9]*",
      "ConstraintDescription" : "must contain only alphanumeric characters."
    },
    "DBSnapshotName": {
      "Description": "The name of a DB snapshot (optional)",
      "Default": "",
      "Type": "String"
    }
  },

  "Conditions": {
    "UseDBSnapshot": {"Fn::Not": [{"Fn::Equals" : [{"Ref" : "DBSnapshotName"},
 ""]}]}
  },

  "Resources" : {
    "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"}
          ]
        }
      }
    },

    "MyRDSParamGroup" : {
      "Type": "AWS::RDS::DBParameterGroup",
      "Properties" : {
        "Family" : "MySQL5.5",
        "Description" : "CloudFormation Sample Database Parameter Group",
          "Parameters" : {
            "autocommit" : "1" ,
            "general_log" : "1",
            "old_passwords" : "0"
        }
      }
    }
  }
}
```

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.

```
{
  "Parameters" : {
    "ExistingSecurityGroup" : {
      "Description" : "An existing security group ID (optional).",
      "Default" : "NONE",
      "Type" : "String",
      "AllowedValues" : ["default", "NONE"]
    }
  },

  "Conditions" : {
    "CreateNewSecurityGroup" : {"Fn::Equals" : [{"Ref" : "ExistingSecurity
Group"}, "NONE"] }
  },

  "Resources" : {
    "MyInstance" : {
      "Type" : "AWS::EC2::Instance",
        "Properties" : {
          "ImageId" : "ami-1b814f72",
          "SecurityGroups" : [{
            "Fn::If" : [
              "CreateNewSecurityGroup",
              {"Ref" : "NewSecurityGroup"},
              {"Ref" : "ExistingSecurityGroup"}
            ]
          }]
        }
    },

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

  "Outputs" : {
    "SecurityGroupId" : {
      "Description" : "Group ID of the security group used.",
      "Value" : {
        "Fn::If" : [
          "CreateNewSecurityGroup",
          {"Ref" : "NewSecurityGroup"},
          {"Ref" : "ExistingSecurityGroup"}
        ]
      }
    }
  }
}
```

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

"Fn::FindInMap" : ["*MapName*", "*TopLevelKey*", "*SecondLevelKey*"]

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.

```
{
  ...
  "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"
        }
     }
 }
}
```

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".

Fn::GetAtt

The intrinsic function Fn::GetAtt returns the value of an attribute from a resource in the template.

Declaration

"Fn::GetAtt" : ["*`logicalNameOfResource`*", "*`attributeName`*"]

Parameters

logicalNameOfResource
: The logical name of the resource that contains the attribute 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 LoadBalancer with the logical name *`MyLB`*.

```
"Fn::GetAtt" : [ "MyLB" , "DNSName" ]
```

Attributes

You can retrieve the following attributes using `Fn::GetAtt`.

Resource TypeName	Attribute	Description
AWS::CloudFormation::WaitCondition (p. 280)	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. 223). Example for wait condition with 2 signals: `{"Signal1":"Step 1 complete.","Signal2":"Step 2 complete."}`
AWS::CloudFormation::Stack (p. 278)	Outputs.*NestedStackOutputName*	Output value from the nested stack that you specified, where *`NestedStackOutputName`* is the name of the output value.
AWS::CloudFront::Distribution (p. 283)	DomainName	Example: `http://d2fadu0nynjpfn.cloudfront.net/`
AWS::EC2::EIP (p. 299)	AllocationId	ID that AWS assigns to represent the allocation of the address for use with Amazon VPC. Returned only for VPC elastic IP addresses. Example: `eipalloc-5723d13e`

Resource TypeName	Attribute	Description
AWS::EC2::Instance (p. 302)	AvailabilityZone	The Availability Zone where the instance that you specified is launched. Example: us-east-1b
AWS::EC2::Instance (p. 302)	PrivateDnsName	The private DNS name of the instance that you specified. Example: `ip-10-24-34-0.ec2.internal`
AWS::EC2::Instance (p. 302)	PublicDnsName	The public DNS name of the specified instance that you specified. Example: `ec2-107-20-50-45.compute-1.amazonaws.com`
AWS::EC2::Instance (p. 302)	PrivateIp	The private IP address of the instance that you specified. Example: `10.24.34.0`
AWS::EC2::Instance (p. 302)	PublicIp	The public IP address of the instance that you specified. Example: `192.0.2.0`
AWS::EC2::NetworkInterface (p. 313)	PrimaryPrivateIpAddress	The primary private IP address of the network interface that you specified. Example: `10.0.0.192`
AWS::EC2::NetworkInterface (p. 313)	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. 322)	GroupId	The group ID of the specified security group. Example: `sg-94b3a1f6`
AWS::EC2::Subnet (p. 332)	AvailabilityZone	The Availability Zone of the subnet. Example: `us-east-1a`
AWS::EC2::SubnetNetworkAclAssociation (p. 334)	AssociationId	NetworkAcl associationId that is attached to a subnet.
AWS::EC2::VPC (p. 341)	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. 341)	DefaultSecurityGroup	The default security group ID that is associated with the VPC, which AWS creates when you create a VPC. Example: `sg-b178e0d3`
AWS::ElastiCache::CacheCluster (p. 360)	ConfigurationEndpointAddress	The DNS address of the configuration endpoint for the cache cluster.

Resource TypeName	Attribute	Description
AWS::ElastiCache::CacheCluster (p. 360)	ConfigurationEndpoint.Port	The port number of the configuration endpoint for the cache cluster.
AWS::ElasticBeanstalk::Environment (p. 373)	EndpointURL	The URL to the LoadBalancer for this environment. Example: `awseb-myst-myen-132MQC4KRLAMD-1371280482.us-east-1.elb.amazonaws.com`
AWS::ElasticLoadBalancing::LoadBalancer (p. 376)	CanonicalHostedZoneName	The name of the Amazon Route 53 hosted zone that is associated with the LoadBalancer. Example: `mystack-myelb-15HMABG9ZCN57-1013119603.us-east-1.elb.amazonaws.com`
AWS::ElasticLoadBalancing::LoadBalancer (p. 376)	CanonicalHostedZoneNameID	The ID of the Amazon Route 53 hosted zone name that is associated with the LoadBalancer. Example: `Z3DZXE0Q79N41H`
AWS::ElasticLoadBalancing::LoadBalancer (p. 376)	DNSName	The DNS name for the LoadBalancer. Example: `mystack-myelb-15HMABG9ZCN57-1013119603.us-east-1.elb.amazonaws.com`
AWS::ElasticLoadBalancing::LoadBalancer (p. 376)	SourceSecurityGroup.GroupName	The security group that you can use as part of your inbound rules for your LoadBalancer's back-end Amazon EC2 application instances. Example: `amazon-elb`
AWS::ElasticLoadBalancing::LoadBalancer (p. 376)	SourceSecurityGroup.OwnerAlias	Owner of the source security group. Example: `amazon-elb-sg`
AWS::IAM::AccessKey (p. 383)	SecretAccessKey	Secret access key for the specified Access Key. Example: `wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY`
AWS::IAM::Group (p. 384)	Arn	Example: `arn:aws:iam::123456789012:group/mystack-mygroup-1DZETITOWEKVO`
AWS::IAM::InstanceProfile (p. 385)	Arn	Returns the Amazon Resource Name (ARN) for the instance profile. Example: `arn:aws:iam::1234567890:instance-profile/MyProfile-ASDNSDLKJ`
AWS::IAM::Role (p. 390)	Arn	Example: `arn:aws:iam::1234567890:role/MyRole-AJJHDSKSDF`
AWS::IAM::User (p. 394)	Arn	Example: `arn:aws:iam::123456789012:group/mystack-myuser-1CCXAFG2H2U4D`
AWS::Redshift::Cluster (p. 413)	Endpoint.Address	Connection endpoint for the cluster. Example: `examplecluster.cg034hpkmmjt.us-east-1.redshift.amazonaws.com`

Resource TypeName	Attribute	Description
AWS::Redshift::Cluster (p. 413)	Endpoint.Port	Connection endpoint for the cluster. Example: `5439`
AWS::RDS::DBInstance (p. 423)	Endpoint.Address	Connection endpoint for the database. Example: `mystack-mydb-1apw1j4phylrk.cg034hpkmmjt.us-east-1.rds.amazonaws.com`
AWS::RDS::DBInstance (p. 423)	Endpoint.Port	The port number on which the database accepts connections. Example: `3306`
AWS::S3::Bucket (p. 446)	DomainName	The DNS name of the specified bucket. Example: `mystack-mybucket-kdwwxmddtr2g.s3.amazonaws.com`
AWS::S3::Bucket (p. 446)	WebsiteURL	Amazon S3 website endpoint for the specified bucket. Example: `http://mystack-mybucket-kdwwxmddtr2g.s3-website-us-east-1.amazonaws.com/`
AWS::SNS::Topic (p. 455)	TopicName	The name of an Amazon SNS topic. Example: `my-sns-topic`
AWS::SQS::Queue (p. 458)	Arn	ARN for the specified queue. Example: `arn:aws:sqs:us-east-1:123456789012:mystack-myqueue-15PG5C2FC1CW8`
AWS::SQS::Queue (p. 458)	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. For the EC2-VPC platform, the `Fn::GetAZs` function returns only the Availablity Zones that have default subnets. For the EC2-Classic platform, the `Fn::GetAZs` function returns all Availability Zones for a 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.

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

"Fn::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

```
{ "Fn::GetAZs" : "" }
```

```
{ "Fn::GetAZs" : { "Ref" : "AWS::Region" } }
```

```
{ "Fn::GetAZs" : "us-east-1" }
```

For both of the previous 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" ]
```

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

"Fn::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

```
"Fn::Join" : [ ":", [ "a", "b", "c" ] ]
```

This example returns: "a:b:c".

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

{ "Fn::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

```
{ "Fn::Select" : [ "1", [ "apples", "grapes", "oranges", "mangoes" ] ] }
```

This example returns: "grapes".

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:

```
"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"
  }
}
```

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:

```
"Subnet0": {
  "Type": "AWS::EC2::Subnet",
    "Properties": {
      "VpcId": { "Ref": "VPC" },
      "CidrBlock": { "Fn::Select" : [ "0", {"Ref": "DbSubnetIpBlocks"} ] }
    }
},
```

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.

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. 245) 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. 299) resource, for instance, returns the IP address, and an AWS::EC2::Instance (p. 302) 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

"Ref" : "*`logicalName`*"

Parameters

logicalName
: The logical name of the resource or parameter you want to dereference.

Return Value

The value of the *`MyInputParameter`* 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:

```
"MyEIP" : {
   "Type" : "AWS::EC2::EIP",
   "Properties" : {
      "InstanceId" : { "Ref" : "MyEC2Instance" }
   }
}
```

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::AutoScaling::AutoScalingGroup (p. 245)	Name	mystack-myasgroup-NT5EUXTNTXXD
AWS::AutoScaling::LaunchConfiguration (p. 251)	Name	mystack-mylaunchconfig-1DDYF1E3B3I
AWS::AutoScaling::ScalingPolicy (p. 257)	Name	mystack-myaspolicy-1DDYF1E3B3I
AWS::AutoScaling::ScheduledAction (p. 259)	Name	mystack-myscheduledaction-NT5EUXTNTXXD
AWS::CloudFormation::Stack (p. 278)	Stack ID	[illegible]
AWS::CloudFormation::WaitCondition (p. 280)	Name	[illegible]
AWS::CloudFormation::WaitConditionHandle (p. 282)	Wait Condition Signal URL	[illegible]
AWS::CloudFront::Distribution (p. 283)	Distribution ID	E27LVI50CSW06W
AWS::CloudTrail::Trail (p. 284)	Trail name	awscloudtrail-example
AWS::CloudWatch::Alarm (p. 287)	Name	mystack-myalarm-3AOHFRGOXR5T
AWS::EC2::VPCPeeringConnection (p. 346)	VPC peering connection ID	pcx-75de3e1d
AWS::EC2::Volume (p. 337)	Volume ID	vol-3cdd3f56
AWS::EC2::VolumeAttachment (p. 339)	Name	mystack-myvola-ERXHJITXMRLT
AWS::EC2::EIP (p. 299)	Elastic IP Address	192.0.2.0
AWS::EC2::EIPAssociation (p. 301)	Name	mystack-myeipa-1NU3IL8LJ313N
AWS::EC2::Instance (p. 302)	Instance ID	i-636be302
AWS::EC2::SecurityGroup (p. 322)	Name or security group ID (for VPC security groups)	mystack-mysecuritygroup-QQB406M8FISX or sg-94b3a1f6
AWS::EC2::SecurityGroupIngress (p. 328)	Name	mysecuritygroupingress
AWS::EC2::Subnet (p. 332)	Name	subnet-e19f0178
AWS::ElasticBeanstalk::Application (p. 368)	Name	mystack-myapplication-FM6BIXY7U8PK
AWS::ElasticBeanstalk::ApplicationVersion (p. 369)	Name	mystack-myapplicationversion-iy8ptveuxjly
AWS::ElasticBeanstalk::ConfigurationTemplate (p. 371)	Name	mystack-myconfigurationtemplate-108RPH64J195

Resource Type	Reference Value	Example Return Value
AWS::ElasticBeanstalk::Environment (p. 373)	Name	mystack-myenv-LKGNQSFHO1DB
AWS::ElastiCache::SubnetGroup (p. 367)	Name	myCachesubnetgroup
AWS::ElasticLoadBalancing::LoadBalancer (p. 376)	Name	mystack-myelb-1WQN7BJGDB5YQ
AWS::IAM::AccessKey (p. 383)	AccessKeyId	AKIAIOSFODNN7EXAMPLE
AWS::IAM::Group (p. 384)	GroupName	mystack-mygroup-1DZETITOWEKVO
AWS::IAM::User (p. 394)	UserName	mystack-myuser-1CCXAFG2H2U4D
AWS::Kinesis::Stream (p. 396)	Name	mystack-mystream-1NAOH4L1RIQ7I
AWS::Logs::LogGroup (p. 397)	Name	mystack-myLogGroup-1341JS4M96031
AWS::OpsWorks::App (p. 399)	AWS OpsWorks Application ID	4fee5b96-0d10-4af1-bcc5-25f92e3c6acf
AWS::OpsWorks::Instance (p. 403)	AWS OpsWorks Instance ID	aa2e9ae2-2b4b-491c-aeb6-8bf3ce9400fe
AWS::OpsWorks::Layer (p. 406)	AWS OpsWorks Layer ID	730b238b-f7c4-461d-b7c0-3feb7ef1152a
AWS::OpsWorks::Stack (p. 409)	AWS OpsWorks Stack ID	5c9f04e8-370e-4bd3-ae09-a4bbcc2998bb
AWS::Redshift::Cluster (p. 413)	Name	mystack-myredshiftcluster-ranmiv3f0mad
AWS::Redshift::ClusterParameterGroup (p. 418)	Name	mysta-mypar-1AJYM1FL3WQBW
AWS::Redshift::ClusterSecurityGroup (p. 420)	Name	mystack-myredshiftclustersecuritygroup-bjy2afmhy3ee
AWS::Redshift::ClusterSubnetGroup (p. 422)	Name	mystack-myredshiftclustersubnetgroup-aq6rsdq8rp71
AWS::RDS::DBInstance (p. 423)	Name	mystack-mydb-ea5ugmfvuaxg
AWS::RDS::DBSecurityGroup (p. 435)	Name	mystack-mydbsecuritygroup-1k5u5dxjb0nxs
AWS::S3::Bucket (p. 446)	Name	mystack-mys3bucket-1hbsmonr9mytq
AWS::SDB::Domain (p. 455)	Name	mystack-mysdbdomain-IVNAOZTDFVXL
AWS::SNS::Topic (p. 455)	Topic ARN	arn:aws:sns:us-east-1:123456789012:mystack-mytopic-NZJ5JSMVGFIE
AWS::SQS::Queue (p. 458)	Queue URL	https://sqs.us-east-1.amazonaws.com/803981987763/aa4-MyQueue-Z5NOSZO2PZE9
Pseudo Parameter (p. 568)	AWS::AccountId	123456789012
Pseudo Parameter (p. 568)	AWS::NotificationARNs	[arn:aws:sns:us-east-1:123456789012:MyTopic]
Pseudo Parameter (p. 568)	AWS::NoValue	Does not return a value.
Pseudo Parameter (p. 568)	AWS::Region	us-east-1
Pseudo Parameter (p. 568)	AWS::StackId	arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1c2fa620-982a-11e3-aff7-50e2416294e0
Pseudo Parameter (p. 568)	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.

For example, the following fragment assigns the value of the *`AWS::Region`* pseudo parameter:

```
"Outputs" {
   "MyStacksRegion" : { "Value" : { "Ref" : "AWS::Region" } }
}
```

The currently available pseudo parameters are listed here.

AWS::AccountId
: Returns the AWS account ID of the account in which the stack is being created.

AWS::NotificationARNs
: Returns the list of notification Amazon Resource Names (ARNs) for the current stack.

For example:

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "MyNestedStack" : {
         "Type" : "AWS::CloudFormation::Stack",
         "Properties" : {
            "TemplateURL" : "https://my-website.com/stack-spec.json",
            "NotificationARNs" : {"Ref" : "AWS::NotificationARNs"}
         }
      }
   }
}
```

To get a single ARN from the list, use Fn::Select (p. 564):

```
"myASGrpOne" : {
   "Type" : "AWS::AutoScaling::AutoScalingGroup",
   "Version" : "2009-05-15",
   "Properties" : {
      "AvailabilityZones" : [ "us-east-1a" ],
      "LaunchConfigurationName" : { "Ref" : "MyLaunchConfiguration" },
      "MinSize" : "0",
      "MaxSize" : "0",
      "NotificationConfiguration" : {
         "TopicARN" : { "Fn::Select" : [ "0", { "Ref" : "AWS::Notification
ARNs" } ] },
         "NotificationTypes" : [ "autoscaling:EC2_INSTANCE_LAUNCH", "auto
scaling: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, as shown in the following snippet:

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

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.

AWS::Region

Returns a string representing the AWS Region in which the encompassing resource is being created.

AWS::StackId

Returns the ID of the stack as specified with the `aws cloudformation create-stack` command.

AWS::StackName

Returns the name of the stack as specified with the `aws cloudformation create-stack` command.

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. 570): Used to retrieve and interpret the resource metadata, installing packages, creating files and starting services.
- cfn-signal (p. 573): 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. 576): 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. 577): 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.
  - https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.zip
  - https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-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.

A complete list of available helper scripts and information regarding their use can be found on the AWS CloudFormation tools page: Bootstrapping Applications using AWS CloudFormation.

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. 268).

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 \
         -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*: None *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

Name	Description	Required
`--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
`-u, --url`	The AWS CloudFormation endpoint to use. *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

Examples

The following snippet is associated with a resource named WebServer.

```
"/opt/aws/bin/cfn-init -s ", { "Ref" : "AWS::StackName" },
"    -r WebServer ",
"    --region ", { "Ref" : "AWS::Region" }, "\n",
```

Several AWS CloudFormation sample templates use cfn-init, including the following templates.

- LAMP: Single EC2 Instance with local MySQL database
- WordPress: Single EC2 Instance with local MySQL database

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. 537) or an Auto Scaling group with a `WaitOnResourceSignals` (p. 543) 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. 224).

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 \
        --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 %ERRORCODE%` (for Windows)	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
`-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
`--resource` (resource signaling only)	The logical ID (p. 126) of the resource that contains the creations policy you want to signal. *Type*: String	Yes

Name	Description	Required
`--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

Examples

Example 1

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. 106).

```
"MyInstance": {
   "Type": "AWS::EC2::Instance",
   "Metadata": {
      :
   },
   "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 0 --stack ", { "Ref" : "AWS::Stack
Name" },
               "         --resource MyInstance \n"
            ] ]
         }
      }
   }
},
```

Examples in Sample Templates

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 the 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

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 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
`--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*: `10`	No

Name	Description	Required
`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|post.update|post.remove
path=Resources.<logicalResourceId> (.Metadata|PhysicalResourceId)(.optional
Metadatapath)
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
`triggers`	A comma-delimited list of conditions to detect. *Valid values*: `post.add | post.update | post.remove` *Example*: `post.add, post.update`	Yes

Name	Description	Required
`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).

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

AWS CloudFormation Limits

Your AWS account has AWS CloudFormation limits that you might need to know when authoring templates and creating stacks. The following tables summarizes these AWS CloudFormation limits.

AWS CloudFormation limits

Limit	Description	Value	Tuning Strategy
cfn-signal wait condition data (p. 573)	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. 265)	Maximum amount of data that a custom resource provider can pass.	4,096 bytes	
Mappings (p. 115)	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. 278).
Mapping attributes (p. 115)	Maximum number of mapping attributes for each mapping that you can declare in your AWS CloudFormation template.	30 attributes	To specify more mapping attributes, separate the attributes into multiple mappings.
Mapping name and mapping attribute name (p. 115)	Maximum size of each mapping name.	255 characters	

Limit	Description	Value	Tuning Strategy
Outputs (p. 115)	Maximum number of outputs that you can declare in your AWS CloudFormation template.	60 outputs	
Output name (p. 115)	Maximum size of an output name.	255 characters	
Parameters (p. 115)	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. 115)	Maximum size of a parameter name.	255 characters	
Parameter value (p. 115)	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 values into a single value.
Resources (p. 115)	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. 278).
Resource name (p. 115)	Maximum size of a resource name.	255 characters	
Stacks (p. 70)	Maximum number of AWS CloudFormation stacks that you can create.	20 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. 115)	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. 278). Or upload the template to an Amazon S3 bucket.

Limit	Description	Value	Tuning Strategy
Template body size in an Amazon S3 object (p. 115)	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. 278).
Template description (p. 115)	Maximum size of a template description.	1,024 bytes	

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. 265), 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 that is obtained from the custom resource provider to access the service.

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`.

Required: Yes

Type: String

ResponseURL

The response URL identifies a pre-signed Amazon 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 containing the custom resource.

Combining the *`StackId`* with the *`RequestId`* forms a value that can be used 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 can be used 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

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 1Kb in size.

Required: Yes

Type: String

StackId

The Amazon Resource Name (ARN) that identifies the stack containing 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. The values provided here can be accessed 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. 585) 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 responds to the S3 bucket with either a `SUCCESS` or `FAILED` result. 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 Amazon 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 containing 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 containing 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 1Kb in size.

Data

Optional, custom resource provider-defined name/value pairs to send with the response. The values provided here can be accessed 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 containing 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.

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 Amazon 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 containing 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 containing 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 1Kb in size.

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. 591).

RequestId
: The *RequestId* value copied from the delete request (p. 591).

StackId
: The *StackId* value copied from the delete request (p. 591).

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 Amazon S3 bucket that receives responses from the custom resource provider to AWS CloudFormation.

StackId
The Amazon Resource Name (ARN) that identifies the stack containing 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 containing 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 1Kb in size.

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"
}
```

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 containing 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 1Kb in size.

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"
}
```

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"
}
```

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.

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. 602) 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. 77).
- 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.
- For all other issues, if you have AWS Premium Support, you can create a Technical Support case. See Contacting Support (p. 604).

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. 540).

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. 65).

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. 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. 243).

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. 582).

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.

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::SecuirtyGroupIngress` resource has a `SourceSecurityGroupName` and `SourceSecurityGroupId` properties. For VPC security groups, you must use the `SourceSecurityGroupId` and specify the security group ID.

Update Rollback Failed

The stack cannot return to a good state. For example, a dependent resource cannot return to its original state, which causes a failure. Contact AWS customer support (p. 604) to fix the stack.

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. 569).
- 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. 75) 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 instance 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 instance, see NAT Instances 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. 77).

  > **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. 601) 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.

Document History

The following table describes the important changes to the documentation since the last release of AWS CloudFormation.

Change	API Version	Description	Release Date
Amazon Route 53 update	2010-05-15	You can now provision and manage Amazon Route 53 hosted zones (p. 439), health checks (p. 438), failover record sets (p. 440), and geolocation record sets (p. 523).	November 06, 2014
Auto Scaling rolling update enhancement	2010-05-15	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. 543).	November 06, 2014
Fn:GetAtt default VPC values	2010-05-15	Given a VPC ID, you can retrieve the default security group and network ACL for that VPC. For more information, see Fn::GetAtt (p. 558).	November 06, 2014
AWS-specific parameter types	2010-05-15	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. 116).	November 06, 2014
CreationPolicy attribute	2010-05-15	With the CreationPolicy attribute, you can instruct AWS CloudFormation to wait until applications are ready on Amazon 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. 537).	November 06, 2014
Amazon CloudFront forwarded values	2010-05-15	For cache behaviors, you can forward headers to the origin. See CloudFront ForwardedValues (p. 482).	September 29, 2014

Change	API Version	Description	Release Date
AWS OpsWorks update	2010-05-15	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. 409).	September 29, 2014
Elastic Load Balancing tagging support	2010-05-15	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. 376).	September 29, 2014
Amazon Simple Notification Service topic policy	2010-05-15	You can now update Amazon SNS topic policies. For more information, see AWS::SNS::TopicPolicy (p. 457).	September 29, 2014
Amazon RDS database instance update	2010-05-15	You can specify whether a database instance is Internet-facing by using the `PubliclyAccessible` property in the AWS::RDS::DBInstance (p. 423) resource.	September 05, 2014
UpdatePolicy Attribute update	2010-05-15	You can specify an update policy for an Auto Scaling group that has an associated scheduled action. For more information, see UpdatePolicy (p. 543).	September 05, 2014
Amazon CloudWatch support	2010-05-15	You can use AWS CloudFormation to provision and manage CloudWatch Logs log groups and metric filters. For more information, see AWS::Logs::LogGroup (p. 397) or AWS::Logs::MetricFilter (p. 398).	July 10, 2014
Amazon CloudFront distribution configuration update	2010-05-15	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. 283).	June 17, 2014
Amazon EC2 instance update	2010-05-15	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. 302).	June 17, 2014
Amazon EBS volume update	2010-05-15	You can use encrypted Amazon EBS volumes with supported instance types. For more information, see AWS::EC2::Volume (p. 337).	June 17, 2014

Change	API Version	Description	Release Date
Amazon VPC peering	2010-05-15	You can use AWS CloudFormation to create a VPC peering connection, which establishes a network connection between two VPCs. For more information, see AWS::EC2::VPCPeeringConnection (p. 346).	June 17, 2014
Auto Scaling group update	2010-05-15	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. 245).	June 17, 2014
AWS CloudTrail	2010-05-15	AWS CloudFormation supports AWS CloudTrail, which an capture API calls made from your AWS account and where to publish the logs at a location you designate. For more information, see AWS::CloudTrail::Trail (p. 284).	June 17, 2014
Update stack enhancements	2010-05-15	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. 88).	May 12, 2014
Amazon Kinesis	2010-05-15	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. 396).	May 06, 2014
Amazon S3	2010-05-15	AWS CloudFormation supports additional Amazon 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 what 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. 446).	May 05, 2014
Auto Scaling	2010-05-15	AWS CloudFormation supports metrics collection for an Auto Scaling group. For more information, see AWS::AutoScaling::AutoScalingGroup (p. 245).	May 05, 2014

Change	API Version	Description	Release Date
`Fn::If` update	2010-05-15	You can use the `Fn::If` intrinsic function in the output section of a template. For more information, see Condition Functions (p. 547).	May 05, 2014
API logging with AWS CloudTrail	2010-05-15	You can use AWS CloudTrail to log AWS CloudFormation requests. With AWS 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. 596).	April 02, 2014
Elastic Load Balancing update	2010-05-15	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. 376).	March 20, 2014
AWS OpsWorks support	2010-05-15	You can use AWS CloudFormation to provision and manage AWS OpsWorks stacks. For more information, see AWS::OpsWorks::Stack (p. 409) or AWS OpsWorks Snippets (p. 190).	March 03, 2014
Limit increase	2010-05-15	You can specify template sizes up to 460,800 bytes in Amazon S3.	February 18, 2014
Amazon Redshift support	2010-05-15	You can use AWS CloudFormation to provision and manage Amazon Redshift clusters. For more information, see Amazon Redshift Snippets (p. 193) or AWS::Redshift::Cluster (p. 413).	February 10, 2014
Amazon S3 buckets and bucket policies update	2010-05-15	You can update some properties of the Amazon S3 bucket and bucket policy resources. For more information, see AWS::S3::Bucket (p. 446) or AWS::S3::BucketPolicy (p. 453).	February 10, 2014
AWS Elastic Beanstalk environments and application versions update	2010-05-15	You can update AWS Elastic Beanstalk environment configurations and application versions. For more information, see AWS::ElasticBeanstalk::Environment (p. 373), AWS::ElasticBeanstalk::ConfigurationTemplate (p. 371), or AWS::ElasticBeanstalk::ApplicationVersion (p. 369).	February 10, 2014
Amazon SQS update	2010-05-15	You can specify a dead letter queue for an Amazon SQS queue. For more information, see AWS::SQS::Queue (p. 458).	January 29, 2014
Auto Scaling scheduled actions	2010-05-15	You can scale the number of Amazon 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. 259).	January 27, 2014
DynamoDB secondary indexes	2010-05-15	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. 291).	January 27, 2014

Change	API Version	Description	Release Date
Auto Scaling update	2010-05-15	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. 245) or AWS::AutoScaling::LaunchConfiguration (p. 251).	January 02, 2014
Amazon SQS update	2010-05-15	You can update Amazon SQS queues and specify additional properties. For more information, see AWS::SQS::Queue (p. 458).	January 02, 2014
Limit increases	2010-05-15	You can specify up to 60 parameters and 60 outputs in your AWS CloudFormation templates	January 02, 2014
New console	2010-05-15	The new AWS CloudFormation console adds features like auto-refreshing stack events and alphabetical ordering of stack parameters.	December 19, 2013
Cross-zone load balancing	2010-05-15	With cross-zone load balancing, you can route traffic to back-end instances across all Avalibility Zones. For more information, see AWS::ElasticLoadBalancing::LoadBalancer (p. 376).	December 19, 2013
AWS Elastic Beanstalk environment tiers	2010-05-15	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. 373).	December 19, 2013
Resource names	2010-05-15	You can assign names (physical IDs) to the following resources: • ElastiCache Clusters • Elastic Load Balancing load balancers • Amazon Relational Database Service DB instances For more information, see Name Type (p. 514).	December 19, 2013
VPN support	2010-05-15	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. 358).	November 22, 2013
Conditionally create resources and assign properties	2010-05-15	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. 547).	November 08, 2013

Change	API Version	Description	Release Date
Prevent accidental updates to stack resources	2010-05-15	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. 96).	November 08, 2013
Name resources	2010-05-15	Instead of using AWS CloudFormation-generated physical IDs, you can assign names to certain resources. The following AWS CloudFormation resources support naming: • Amazon CloudWatch alarms • Amazon DynamoDB tables • AWS Elastic Beanstalk applications and environments • Amazon S3 buckets • Amazon SNS topics • Amazon SQS queues For more information, see Name Type (p. 514).	November 08, 2013
Assign custom resource types	2010-05-15	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. 265).	November 08, 2013
Add pseudo parameter	2010-05-15	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. 568).	November 08, 2013
Specify stacks in IAM policies	2010-05-15	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. 65).	November 08, 2013
Federation support	2010-05-15	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 Amazon 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. 65).	October 14, 2013

Change	API Version	Description	Release Date
Amazon RDS read replica support	2010-05-15	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. 423) resource.	September 24, 2013
Associate public IP address with instances in Auto Scaling group.	2010-05-15	You can now associate public IP addresses with instances in an Auto Scaling group. For more information, see AWS::AutoScaling::LaunchConfiguration (p. 251).	September 19, 2013
Additional VPC support.	2010-05-15	AWS CloudFormation added 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. 313). 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. 341). • You can specify a static route between a virtual private gateway to your VPN gateway. For more information, see AWS::EC2::VPNConnectionRoute (p. 356).	September 17, 2013
Redis and VPC security groups support for Amazon ElastiCache.	2010-05-15	You can now specify Redis as the cache engine for an ElastiCache cluster. You can also now assign VPC security groups to ElastiCache clusters. For more information, see AWS::ElastiCache::CacheCluster (p. 360).	September 3, 2013
Parallel stack creation, update and deletion, and nested stack updates.	2010-05-15	CloudFormation now creates, updates, and deletes resources in parallel, improving the operations' performance. If you update a top-level template, CloudFormation automatically updates any nested stacks that have changed. For more information, see AWS CloudFormation Stacks Updates (p. 88).	August 12, 2013
VPC security groups can now be set in AWS RDS instances	2010-05-15	You can now assign VPC security groups to an Amazon RDS instance with AWS CloudFormation. For more information, see the VPCSecurityGroups (p. 429) property in AWS::RDS::DBInstance (p. 423).	February 28, 2013
Rolling Deployments for Auto Scaling Groups	2010-05-15	AWS CloudFormation now supports update policies on autoscaling groups, which describe how instances in the autoscaling 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. 543).	February 20, 2013

Change	API Version	Description	Release Date
Cancel and Rollback Action for Stack Updates	2010-05-15	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. 95) • `aws cloudformation cancel-update-stack` • CancelUpdateStack in the *AWS CloudFormation API Reference*	February 20, 2013
EBS-Optimized Instances for Auto Scaling Groups	2010-05-15	You can now provision EBS-optimized instances in auto scaling groups for dedicated throughput to Amazon EBS in autoscaled instances. The implementation is similar to that of the previously released support for optimized EBS EC2 instances. For more information, see the new *`EbsOptimized`* property in AWS::AutoScaling::LaunchConfiguration (p. 251).	February 20, 2013
New Documentation	2010-05-15	AWS::EC2::Instance (p. 302) 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. 490) • Amazon Elastic Block Store Block Device Property (p. 492)	December 21, 2012
New Documentation	2010-05-15	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. 77)	December 21, 2012
New Documentation	2010-05-15	Custom resources are special AWS CloudFormation resources that provide a way for a template developer to include non-AWS resources in an AWS CloudFormation stack. The custom resource provider can be either a template developer or a separate third-party resource provider. Information about custom resources is provided in the following topics: • AWS CloudFormation Custom Resource Walkthrough (p. 47) • AWS::CloudFormation::CustomResource (p. 265) • Custom Resource Reference (p. 585)	November 15, 2012

Change	API Version	Description	Release Date
Updated Documentation	2010-05-15	AWS CloudFormation now supports specifying provisioned I/O operations per second (IOPS) for Amazon RDS instances. You can set this value from 1000–10,000 in 1000 IOPS increments by using the new Iops (p. 427) property in AWS::RDS::DBInstance (p. 423). For more information about specifying IOPS for RDS instances, see Provisioned IOPS in the *Amazon Relational Database Service User Guide*.	November 15, 2012
New and Updated Documentation	2010-05-15	Reorganization of topics 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 to the documentation, including new guides and updated reference topics: • New topic in Using the Console: Setting Stack Options (p. 75). • New information about tags in the *AWS CloudFormation API reference*: CreateStack, Stack, and Tag. New information about working with Windows Stacks (p. 105): • Microsoft Windows Amazon Machine Images (AMIs) and AWS CloudFormation Templates (p. 106) • Bootstrapping AWS CloudFormation Windows Stacks (p. 106) • Accessing AWS CloudFormation Windows Instances (p. 110) New topic: Using Regular Expressions in AWS CloudFormation Templates (p. 242).	August 27, 2012

Change	API Version	Description	Release Date
New Feature	2010-05-15	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 can be downloaded that demonstrate new VPC features: 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. 322) AWS::EC2::SecurityGroupIngress (p. 328) AWS::EC2::SecurityGroupEgress (p. 325) AWS::EC2::Instance (p. 302) AWS::AutoScaling::AutoScalingGroup (p. 245) AWS::EC2::EIP (p. 299) AWS::EC2::EIPAssociation (p. 301) AWS::ElasticLoadBalancing::LoadBalancer (p. 376) New resource types have been added to the documentation: AWS::EC2::VPC (p. 341) AWS::EC2::InternetGateway (p. 309) AWS::EC2::DHCPOptions (p. 297) AWS::EC2::DHCPOptions (p. 321) AWS::EC2::RouteTable (p. 318) AWS::EC2::NetworkAcl (p. 310) AWS::EC2::NetworkAclEntry (p. 311) AWS::EC2::Subnet (p. 332) AWS::EC2::VPNGateway (p. 357) AWS::EC2::CustomerGateway (p. 295)	April 25, 2012
New Feature	2010-05-15	AWS CloudFormation now allows you to add or remove elements from a stack when updating it. AWS CloudFormation Stacks Updates (p. 88) has been updated, and a new section has been added to the walkthrough: Change the Stack's Resources (p. 38), which describes how to add and remove resources when updating the stack.	April 13, 2012

Change	API Version	Description	Release Date
New Feature	2010-05-15	AWS CloudFormation now provides support for resources in an existing Amazon Virtual Private Cloud (VPC). With this release, you can: • Launch an EC2 Dedicated Instance into an existing VPC. For more information, see AWS::EC2::Instance (p. 302). • Set the SourceDestCheck attribute of an Amazon EC2 instance that resides in an existing VPC. For more information, see AWS::EC2::Instance (p. 302) • Create Amazon Elastic IP Addresses in an existing VPC. For more information, see AWS::EC2::EIP (p. 299) • Use CloudFormation to create VPC security groups and ingress/egress rules in an existing VPC. For more information, see AWS::EC2::SecurityGroup (p. 322). • 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. 245). • Attach an Elastic Load Balancing LoadBalancer to a VPC subnet and create security groups for the LoadBalancer. For more information, see AWS::ElasticLoadBalancing::LoadBalancer (p. 376). • Create an RDS instance in an existing VPC. For more information, see AWS::RDS::DBInstance (p. 423).	February 2, 2012
New Feature	2010-05-15	You can now update properties for the following resources in an existing stack: • AWS::EC2::SecurityGroupIngress (p. 328) • AWS::EC2::SecurityGroupEgress (p. 325) • AWS::EC2::EIPAssociation (p. 301) • AWS::RDS::DBSubnetGroup (p. 434) • AWS::RDS::DBSecurityGroup (p. 435) • AWS::RDS::DBSecurityGroupIngress (p. 437) • AWS::Route53::RecordSetGroup (p. 444) For the full list of updateable resources and details about things to consider when updating a stack, see AWS CloudFormation Stacks Updates (p. 88).	February 2, 2012
Restructured Guide	2010-05-15	Reorganized existing sections into new sections: Working with AWS CloudFormation Templates (p. 114) and **Managing Stacks**. Moved Template Reference (p. 243) to the top level of the Table of Contents. Moved Estimating the Cost of Your AWS CloudFormation Stack (p. 76) to the Getting Started section.	February 2, 2012

Change	API Version	Description	Release Date
New Content	2010-05-15	Added three new sections: • Walkthrough: Updating a Stack (p. 24) is a tutorial that walks through the process of updating a LAMP stack. • Deploying Applications on Amazon EC2 with AWS CloudFormation (p. 224) describes how to use AWS CloudFormation helper scripts to deploy applications using metadata stored in your template. • CloudFormation Helper Scripts Reference (p. 569) provides reference material for the AWS CloudFormation helper scripts (cfn-init, cfn-get-metadata, cfn-signal, and cfn-hup).	February 2, 2012
New Feature	2010-05-15	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. 80).	May 26, 2011
New Features	2010-05-15	The aws cloudformation describe-stack-resources and aws cloudformation get-template commands now enable you to get information from stacks which have been deleted for 90 days after they have been deleted. For more information, see Listing Resources (p. 85) and Retrieving a Template (p. 86).	May 26, 2011
New Link	2010-05-15	AWS CloudFormation endpoint information is now located in the Amazon Web Services General Reference. For more information, go to Regions and Endpoints in Amazon Web Services General Reference.	March 1, 2011
Initial Release	2010-05-15	This is the initial public release of AWS CloudFormation.	February 25, 2011

AWS Glossary

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

Numbers and Symbols

100-continue

A method that enables a client to see if a server can accept a request before actually sending it. For large PUT requests, this method can save both time and bandwidth charges.

A

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

access control list

A document that defines who can access a particular bucket or object. Each bucket and object in Amazon S3 has an ACL. The document defines what each type of user can do, such as write and read permissions.

access identifiers

See credentials.

access key ID

A unique identifier that's associated with a secret access key (p. 642); the access key ID and secret access key are used together to sign programmatic AWS requests cryptographically.

access key rotation

A method to increase security by changing the AWS access key ID. This method enables you to retire an old key at your discretion.

access policy language

A language for writing documents (that is, *policies*) that specify who can access a particular AWS resource and under what conditions.

account

The AWS account associated with a particular AWS login ID and password.

IAM: The AWS account that centrally controls all the resources created under its umbrella and pays for all AWS activity for those resources. The AWS account has permission to do anything and everything with all the AWS account resources. This is in contrast to the user (p. 646).

account activity

A web page showing your month-to-date AWS usage and costs. The account activity page is located at http://aws.amazon.com/account-activity/.

action

An API function. Also called *operation* or *call*. The activity the principal (p. 637) has permission to perform. The action is B in the statement "A has permission to do B to C where D applies." For example, Jane sends a request to Amazon SQS with Action=ReceiveMessage.

Amazon CloudWatch: The response initiated by the change in an alarm's state: for example, from OK to ALARM. The state change may be triggered by a metric reaching the alarm threshold, or by a SetAlarmState request. Each alarm can have one or more actions assigned to each state. Actions are performed once each time the alarm changes to a state that has an action assigned, such as an Amazon Simple Notification Service notification, an Auto Scaling policy execution or an Amazon EC2 instance stop/terminate action.

activate URL

The location where your customers can generate a new activation key (p. 618) for your product, should you request it. The activate URL is http://www.amazon.com/dp-activate.

activation

The process your product goes through to prepare itself for the customer's use. During activation, your product obtains the required credentials for the customer.

activation key

An encoded string that represents the relationship between a customer and a Amazon DevPay product the customer has purchased. AWS generates this value when the customer completes the purchase of the product. You use the key to obtain credentials related to the customer and product.

active authorization

When you inform customers of a price change for your product, they have to take action to accept the price change. If they don't take the action to accept the price change, their access to your product is canceled when the price change takes effect.

active trusted signers

A list showing each of the trusted signers you've specified and the IDs of the corresponding active key pairs that CloudFront is aware of. To be able to create working signed URLs, a trusted signer must appear in this list with at least one key pair ID.

administrative suspension

Auto Scaling might suspend processes for Auto Scaling group (p. 621) that repeatedly fail to launch instances. Auto Scaling groups that most commonly experience administrative suspension have zero running instances, have been trying to launch instances for more than 24 hours, and have not succeeded in that time.

alarm

An item that watches a single metric over a specified time period, and triggers an Amazon SNS topic (p. 646) or an Auto Scaling policy (p. 636) if the value of the metric crosses a threshold value over a predetermined number of time periods.

allow

One of two possible outcomes (the other is deny (p. 625)) when an IAM access policy (p. 636) is evaluated. When a user makes a request to AWS, AWS evaluates the request based on all permissions that apply to the user and then returns either allow or deny.

Amazon CloudFront

An AWS content delivery service that helps you improve the performance, reliability, and availability of your websites and applications.
See Also http://aws.amazon.com/cloudfront.

Amazon CloudSearch

A fully-managed service in the AWS cloud that makes it easy to set up, manage, and scale a search solution for your website or application.

Amazon CloudWatch

A web service that enables you to monitor and manage various metrics, and configure alarm actions based on data from those metrics.
See Also http://aws.amazon.com/cloudwatch.

Amazon DevPay

An easy-to-use online billing and account management service that makes it easy for you to sell an Amazon EC2 AMI or an application built on Amazon S3.
See Also http://aws.amazon.com/devpay.

Amazon Elastic Block Store

A service that provides block level storage volumes for use with EC2 instances.
See Also http://aws.amazon.com/ebs.

Amazon EBS-backed AMI

Instances launched from this type of AMI use an Amazon EBS volume as their root device. Compare this with instances launched from instance store-backed AMIs, which use the instance store as the root device.

Amazon Elastic Compute Cloud

A web service that enables you to launch and manage Linux/UNIX and Windows server instances in Amazon's data centers.
See Also http://aws.amazon.com/ec2.

Amazon EC2 VM Import Connector

See http://aws.amazon.com/ec2/vm-import.

Amazon Elastic MapReduce

A web service that makes it easy to process large amounts of data efficiently. Amazon EMR uses Hadoop processing combined with several AWS products to do such tasks as web indexing, data mining, log file analysis, machine learning, scientific simulation, and data warehousing.
See Also http://aws.amazon.com/elasticmapreduce.

Amazon Machine Image

An encrypted machine image stored in Amazon Elastic Block Store (p. 619) or Amazon Simple Storage Service. AMIs are like a template of a computer's root drive. They contain the operating system and can also include software and layers of your application, such as database servers, middleware, web servers, and so on.

Mechanical Turk

Provides an on-demand, scalable, human workforce to complete jobs that humans can do better than computers.
See Also http://aws.amazon.com/mturk.

Amazon Relational Database Service

A web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.
See Also http://aws.amazon.com/rds.

Amazon Resource Name

A standardized way to refer to an AWS resource. For example: arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Bob.

Amazon Route 53

A web service you can use to create a new DNS service or to migrate your existing DNS service to the cloud.
See Also http://aws.amazon.com/route53.

Amazon S3

See Amazon Simple Storage Service.

Amazon S3-Backed AMI

See instance store-backed AMI.

Amazon Simple Email Service

An easy-to-use, cost-effective email solution for applications.
See Also http://aws.amazon.com/ses.

Amazon Simple Notification Service

A web service that enables applications, end-users, and devices to instantly send and receive notifications from the cloud.
See Also http://aws.amazon.com/sns.

Amazon Simple Queue Service

Reliable and scalable hosted queues for storing messages as they travel between computers.

See Also http://aws.amazon.com/sqs.

Amazon Simple Storage Service

Storage for the internet. You can use it to store and retrieve any amount of data at any time, from anywhere on the web.
See Also http://aws.amazon.com/s3.

Amazon SimpleDB

A highly-available, scalable, and flexible non-relational data store that enables you to store and query data items using web service requests.
See Also http://aws.amazon.com/simpledb.

Amazon Virtual Private Cloud

A web service that enables you to create a virtual network for your AWS resources.
See Also http://aws.amazon.com/vpc.

Amazon Web Services

An infrastructure web services platform in the cloud for companies of all sizes.
See Also http://aws.amazon.com.

AMI

See Amazon Machine Image.

application

A logical collection of AWS Elastic Beanstalk components, including environments, versions, and environment configurations. An application is conceptually similar to a folder.

analysis scheme

Language-specific Amazon CloudSearch text analysis options that are applied to a text field to control stemming and configure stopwords and synonyms.

Application Billing

The location where your customers manage the Amazon DevPay products they've purchased. This is the URL http://www.amazon.com/dp-applications.

application version

A specific, labeled iteration of an application that represents a functionally consistent set of deployable application code. A version points to an Amazon S3 object (a JAVA WAR file) that contains the application code.

approval

If a Worker's response satisfies your Human Intelligence Task (p. 630), you approve the assignment. When you approve an assignment Mechanical Turk transfers the HIT reward from your Mechanical Turk account to the Worker's Amazon Payments account.

ARN

See Amazon Resource Name.

assignment

When a worker (p. 648) finds a Human Intelligence Task (p. 630) (HIT) to complete, the worker accepts the HIT. Mechanical Turk creates an *assignment* to track the work to completion and store the answer the worker submits. The assignment belongs exclusively to the worker who accepted it and guarantees that the worker can submit results and be eligible for a reward—up until the HIT or assignment expires.

asynchronous bounce

A type of bounce (p. 622) that occurs when a receiver (p. 639) initially accepts an email message for delivery and then subsequently fails to deliver it.

attribute

Similar to a column on a spreadsheet, an attribute represents a data category. In Amazon SimpleDB, an attribute has a name (such as *color*), which has a value (such as *blue*) when applied to a data item.

authentication

The process of proving your identity to a system.

Auto Scaling

A web service designed to launch or terminate instance (p. 630)s automatically based on user-defined policies, schedules, and health checks.
See Also http://aws.amazon.com/autoscaling.

Auto Scaling group

A representation of multiple Amazon Elastic Compute Cloud (p. 619) instance (p. 630)s that share similar characteristics, and that are treated as a logical grouping for the purposes of instance scaling and management.

Availability Zone

A distinct location within a region (p. 639) that is insulated from failures in other Availability Zones, and provides inexpensive, low-latency network connectivity to other Availability Zones in the same region.

AWS

See Amazon Web Services.

AWS CloudFormation

A service for writing or changing templates that create and delete related AWS resources together as a unit.
See Also http://aws.amazon.com/cloudformation.

AWS Consolidated Billing

A billing option that lets you get a single bill for multiple AWS accounts.
See Also http://aws.amazon.com/consolidated-billing.

AWS Elastic Beanstalk

See Also http://aws.amazon.com/elasticbeanstalk.

AWS Import/Export

A service for transferring large amounts of data between AWS and portable storage devices.
See Also http://aws.amazon.com/importexport.

AWS Identity and Access Management

A web service that enables Amazon Web Services (p. 620) customers to manage users and user permissions within AWS.
See Also http://aws.amazon.com/iam.

AWS Management Console

A graphical interface to manage compute, storage, and other cloud resources.
See Also http://aws.amazon.com/console.

AWS Multi-Factor Authentication

An optional AWS account security feature. Once you enable AWS MFA, you must provide a six-digit, single-use code in addition to your sign-in credentials whenever you access secure AWS web site pages or the AWS Management Console. You get this single-use code from an authentication device that you keep in your physical possession.
See Also http://aws.amazon.com/mfa/.

AWS Resources

See resource.

AWS VPN CloudHub

Enables secure communication between branch offices using a simple hub-and-spoke model, with or without a VPC.

B

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

basic monitoring

Monitoring of AWS-provided metrics derived at a 5-minute frequency.

batch

See document batch.

BGP ASN

Border Gateway Protocol Autonomous System Number. A unique identifier for a network, for use in BGP routing. Amazon EC2 supports all 2-byte ASN numbers in the range of 1 - 65334, with the exception of 7224, which is reserved.

blacklist

A list of IP addresses, email addresses, or domains that an Internet Service Provider (p. 631) suspects to be the source of spam (p. 643). The ISP blocks incoming emails from these addresses or domains.

block
A data set. Amazon EMR breaks large amounts of data into subsets. Each subset is called a data block. Amazon EMR assigns an ID to each block and uses a hash table to keep track of block processing.

block device
A storage device that supports reading and (optionally) writing data in fixed-size blocks, sectors, or clusters.

block device mapping
A mapping structure for every AMI and instance that specifies the block devices attached to the instance.

bootstrap action
A user-specified default or custom action that runs a script or an application on all nodes of a job flow before Hadoop starts.

Border Gateway Protocol Autonomous System Number
See BGP ASN.

bounce
A failed email delivery attempt.

breach
The condition in which a user-set threshold (upper or lower boundary) is passed. If the duration of the breach is significant, as set by a breach duration parameter, it can possibly start a scaling activity (p. 641).

bucket
A container for objects stored in Amazon S3. Every object is contained in a bucket. For example, if the object named `photos/puppy.jpg` is stored in the `johnsmith` bucket, then authorized users can access the object with the URL `http://johnsmith.s3.amazonaws.com/photos/puppy.jpg`.

bucket owner
Just as Amazon is the only owner of the domain name Amazon.com, only one person or organization can own a bucket in Amazon S3.

bundling
A commonly used term for creating an Amazon Machine Image (p. 619). It specifically refers to creating instance store-backed AMIs.

C

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

cache cluster
A logical cache distributed over multiple cache node (p. 622)s. A cache cluster can be set up with a specific number of cache nodes.

cache cluster identifier
Customer-supplied identifier for the cache cluster that must be unique for that customer in an AWS region.

cache engine version
The version of the Memcached service that is running on the cache node.

cache node
A fixed-size chunk of secure, network-attached RAM. Each cache node runs an instance of the Memcached service, and has its own DNS name and port. Multiple types of cache nodes are supported, each with varying amounts of associated memory.

cache node type
EC2 instance type used to run the cache node.

cache parameter group
A container for cache engine parameter values that can be applied to one or more cache clusters.

cache security group

A group maintained by ElastiCache that combines ingress authorizations to cache nodes for hosts belonging to Amazon EC2 security groups specified through the console or the API or command line tools.

canned access policy

A standard access control policy that you can apply to a bucket or object. Options include: private, public-read, public-read-write, and authenticated-read.

canonicalization

The process of converting data into a standard format that a service such as Amazon S3 can recognize.

capacity

Each Auto Scaling group (p. 621) is defined with a minimum and maximum compute size. The amount of available compute size at any time is the current capacity.
A scaling activity (p. 641) increases or decreases the capacity—within the defined minimum and maximum values.

Cascading

Cascading is an open-source Java library that provides a query API, a query planner, and a job scheduler for creating and running Hadoop MapReduce applications. Applications developed with Cascading are compiled and packaged into standard Hadoop-compatible JAR files similar to other native Hadoop applications.

certificate

A credential that some AWS products use to authenticate AWS accounts and users. Also known as an X.509 certificate. The certificate is paired with a private key.

chargeable resources

Features or services whose use incurs fees. Although some AWS products are free, others include charges. For example, in an AWS CloudFormation stack (p. 643), AWS resources that have been created incur charges. The amount charged depends on the usage load. Use the Amazon Web Services Simple Monthly Calculator at http://calculator.s3.amazonaws.com/calc5.html to estimate your cost prior to creating instances, stacks, or other resources.

CIDR block

Classless Inter-Domain Routing. An Internet protocol address allocation and route aggregation methodology.
See Also http://en.wikipedia.org/wiki/CIDR_notation.

CloudHub

See AWS VPN CloudHub.

cluster compute instance

A type of instance (p. 630) that provides a great amount of CPU power coupled with increased networking performance, making it well suited for High Performance Compute (HPC) applications and other demanding network-bound applications.

cluster placement group

A logical cluster compute instance (p. 623) grouping to provide lower latency and high-bandwidth connectivity between the instances.

CNAME

Canonical Name Record. A type of resource record in the Domain Name System (DNS) that specifies that the domain name is an alias of another, canonical domain name. More simply, it is an entry in a DNS table that lets you alias one fully qualified domain name to another.

complaint

The event in which a recipient (p. 639) who does not want to receive an email message clicks "Mark as Spam" within the email client, and the Internet Service Provider (p. 631) sends a notification to Amazon SES.

compound query

A search request that specifies multiple search criteria using the Amazon CloudSearch structured search syntax.

condition

Any restriction or detail about a permission. The condition is *D* in the statement "A has permission to do B to C where D applies."

conditional parameter

See mapping.

configuration API

The Amazon CloudSearch API that you use to create, configure, and manage search domains.

configuration template

A series of key–value pairs that define parameters for various AWS products so that AWS Elastic Beanstalk can provision them for an environment.

confirmation email

The email Amazon Payments sends to your customers to notify them that a price change you scheduled has taken effect.

consistency model

The method a service uses to achieve high availability. For example, it could involve replicating data across multiple servers in a data center.
See Also eventual consistency.

consistent read

When data is written or updated successfully, all copies of the data are updated in all AWS regions. However, it takes time for the data to propagate to all storage locations. A consistent read returns a result that reflects any writes that received a successful response before the read request—regardless of the region. By contrast, an eventually consistent read returns data from only one region and might not show the most recent write information.
See Also eventual consistency.

console

See AWS Management Console.

Consolidated Billing

See AWS Consolidated Billing.

cooldown period

Amount of time during which Auto Scaling does not allow the desired size of the Auto Scaling group (p. 621) to be changed by any other notification from a CloudWatch alarm (p. 618).

core node

An EC2 instance (p. 626) that runs Hadoop map and reduce tasks and stores data using the Hadoop Distributed File System (HDFS). Core nodes are managed by the master node (p. 634), which assigns Hadoop tasks to nodes and monitors their status. The EC2 instances you assign as core nodes are capacity that must be allotted for the entire job flow run. Because core nodes store data, you can't remove them from a job flow. However, you can add more core nodes to a running job flow.

Core nodes run both the DataNodes and TaskTracker Hadoop daemons.

corpus

A collection of data that you want to search.

credentials

Also called *access credentials* or *security credentials*. In authentication and authorization, a system uses credentials to identify who is making a call and whether to allow the requested access. In AWS, these credentials are typically the access key ID (p. 617) and the secret access key (p. 642).

customer gateway

A router or software application on your side of a VPN tunnel that is managed by Amazon VPC. The internal interfaces of the customer gateway are attached to one or more devices in your home network. The external interface is attached to the VPG (p. 647) across the VPN tunnel.

D

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

dashboard

See service health dashboard.

database engine

The database software and version running on the DB instance (p. 625).

database name

The name of a database hosted in a DB instance (p. 625). A DB instance can host multiple databases, but databases hosted by the same DB instance must each have a unique name within that instance.

DB compute class

Size of the database compute platform used to run the instance.

DB instance

An isolated database environment running in the cloud. A DB instance can contain multiple user-created databases.

DB instance identifier

User-supplied identifier for the DB instance. The identifier must be unique for that user in an AWS region (p. 639).

DB parameter group

A container for database engine parameter values that apply to one or more DB instance (p. 625)s.

DB security group

A method that controls access to the DB instance (p. 625). By default, network access is turned off to DB instances. After ingress is configured for a security group, the same rules apply to all DB instances associated with that group.

DB snapshot

A user-initiated point backup of a DB instance.

Dedicated Instance

An instance that is physically isolated at the host hardware level and launched within a VPC.

Dedicated Reserved Instance

An option you purchase to guarantee that sufficient capacity will be available to launch Dedicated Instances into a VPC.

delete marker

An object with a key and version ID, but without content. Amazon S3 inserts delete markers automatically into versioned buckets when an object is deleted.

deliverability

The likelihood that an email message will arrive at its intended destination.

deliveries

The number of emails, sent through Amazon SES, that were accepted by an Internet Service Provider (p. 631) for delivery to recipient (p. 639)s over a period of time.

deny

The result of a policy statement that includes deny as the effect, so that a specific action or actions are expressly forbidden for a user, group, or role. Explicit deny take precedence over explicit allow (p. 618).

detailed monitoring

Monitoring of AWS-provided metrics derived at a 1-minute frequency.

Description property

A property added to parameters, resources, resource properties, mappings, and outputs, to help you to document AWS CloudFormation template elements.

DevPay Activity

The location (Amazon DevPay Activity) where you manage the Amazon DevPay products you've created.

dimension

A name/value pair (for example, InstanceType=m1.small, or EngineName=mysql), that contains additional information to identify a metric.

discussion forums

A place where AWS users can post technical questions and feedback to help accelerate their development efforts and to engage with the AWS community. The discussion forums are located at http://aws.amazon.com/forums/.

distributed cache

A Hadoop feature that allow you to transfer files from a distributed file system to the local file system. It can distribute data and text files as well as more complex types such as archives and JARs.

distribution

A link between an origin server (such as an Amazon S3 bucket) and a domain name, which CloudFront automatically assigns. Through this link, CloudFront identifies the object you have stored in your origin server (p. 635).

DKIM

DomainKeys Identified Mail. A standard that email senders use to sign their messages. ISPs use those signatures to verify that messages are legitimate. For more information, see http://www.dkim.org.

DNS

See Domain Name System (DNS).

document

Represents an item that can be returned as a search result in Amazon CloudSearch. Each document has a collection of fields that contain the data that can be searched or returned. The value of a field can be either a string or a number. Each document must have a unique ID and at least one field.

document batch

A collection of add and delete document operations for Amazon CloudSearch. You use the document service API to submit batches to update the data in your search domain.

document service API

The Amazon CloudSearch API that you use to submit document batches to update the data in a search domain.

document service endpoint

The URL that you connect to when sending document updates to an Amazon CloudSearch domain. Each search domain has a unique document service endpoint that remains the same for the life of the domain.

domain

All Amazon SimpleDB information is stored in domains. Domains are like tables that contain similar data. You can execute queries against a domain, but cannot execute joins between domains.
See Also search domain.

Domain Name System (DNS)

A distributed naming system that associates network information with human-readable domain names on the Internet.

Donation button

An HTML-coded button to provide an easy and secure way for US-based, IRS-certified 501(c)3 nonprofit organizations to solicit donations.

E

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

EBS

See Amazon Elastic Block Store.

EC2

See Amazon Elastic Compute Cloud.

EC2 compute unit

An AWS standard for compute CPU and memory. This measure enables you to evaluate the CPU capacity of different EC2 instance types.

EC2 instance

In Amazon EC2, this is simply an instance. Other AWS services use the term EC2 instance to distinguish these instances from other types of instances they support.

edge location

A site that CloudFront uses to cache copies of your content for faster delivery to users at any location.

Elastic Block Store

See Amazon Elastic Block Store.

Elastic IP address

A fixed (static) IP address that you have allocated in Amazon EC2 or Amazon VPC and then attached to an instance. Elastic IP addresses are associated with your account, not a specific instance. They are *elastic* because you can easily allocate, attach, detach, and free them as your needs change. Unlike traditional static IP addresses, Elastic IP addresses allow you to mask instance or Availability Zone failures by rapidly remapping your public IP addresses to another instance.

Elastic Load Balancing

A web service that improves an application's availability by distributing incoming traffic between two or more EC2 instance (p. 626)s.
See Also http://aws.amazon.com/elasticloadbalancing.

elastic network interface

An additional network interface that can be attached to an instance (p. 630). ENIs include a primary private IP address, one or more secondary private IP addresses, an elastic IP address (optional), a MAC address, membership in specified security groups, a description, and a source/destination check flag. You can create an ENI, attach it to an instance, detach it from an instance, and attach it to another instance.

endpoint

A URL that identifies a host and port as the entry point for a web service. Every web service request contains an endpoint. Most AWS products provide regional endpoints to enable faster connectivity. For more information, see Regions and Endpoints in the *Amazon Web Services General Reference*

ElastiCache: The DNS name of a cache node (p. 622).

Amazon RDS: The DNS name of a DB instance (p. 625).

AWS CloudFormation: The DNS name or IP address of the server that receives an HTTP request.

endpoint port

ElastiCache: The port number used by a cache node (p. 622).

Amazon RDS: The port number used by a DB instance (p. 625).

environment

A specific running instance of an application (p. 620). The application has a CNAME and includes an application version and a customizable configuration (which is inherited from the default container type).

environment configuration

A collection of parameters and settings that define how an environment and its associated resources behave.

ephemeral store

See instance store.

epoch

The date from which time is measured. For most Unix environments, the epoch is January 1, 1970.

eventual consistency

The method through which AWS products achieve high availability, which involves replicating data across multiple servers in Amazon's data centers. When data is written or updated and "Success" is returned, all copies of the data are updated. However, it takes time for the data to propagate to all storage locations. The data will eventually be consistent, but an immediate read might not show the change. Consistency is usually reached within seconds, but a high system load might increase this time.

eventually consistent read

See consistent read.

eviction

An *eviction* occurs when CloudFront deletes an object from an edge location (p. 626) before its expiration time. If an object in an edge location isn't frequently requested, CloudFront might evict the object (remove the object before its expiration date) to make room for objects that are more popular.

expiration

Expiration occurs when CloudFront stops serving an object from an edge location (p. 626). The next time the edge location needs to serve that object, CloudFront gets a new copy from the origin server (p. 635).

explicit launch permission

An Amazon Machine Image (p. 619) launch permission granted to a specific AWS account.

exponential backoff

A strategy that incrementally increases the wait between retry attempts in order to reduce the load on the system and increase the likelihood that repeated requests will succeed. For example, client applications might wait up to 400 milliseconds before attempting the first retry, up to 1600 milliseconds before the second, up to 6400 milliseconds (6.4 seconds) before the third, and so on.

expression

A numeric expression that you can use to control how search hits are sorted. You can construct Amazon CloudSearch expressions using numeric fields, other rank expressions, a document's default relevance _score, and standard numeric operators and functions. When you use the `sort` option to specify an expression in a search request, the expression is evaluated for each search hit and the hits are listed according to their expression values.

F

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

facet

An Amazon CloudSearch index field that represents a category that you want to use to refine and filter search results.

facet enabled

An Amazon CloudSearch index field option that enables facet information to be calculated for the field.

FBL

See feedback loop.

federated identity management

Allows individuals to sign in to different networks or services, using the same group or personal credentials to access data across all networks. With identity federation in AWS, external identities (federated users) are granted secure access to resources in an AWS account without having to create IAM users. These external identities can come from a corporate identity store (such as LDAP or Windows Active Directory) or from a third party (such as Login with Amazon, Facebook, or Google). AWS federation also supports SAML 2.0.

federated user

See federated identity management.

feedback loop

The mechanism by which a mailbox provider (for example, an Internet Service Provider (p. 631)) forwards a recipient (p. 639)'s complaint (p. 623) back to the sender (p. 642).

field weight

The relative importance of a text field in a search index. Field weights control how much matches in particular text fields affect a document's relevance _score.

filter

A criterion you specify to limit the results when you list or describe your Amazon EC2 resources.

filter query

A way to filter search results without affecting how the results are scored and sorted. Specified with the Amazon CloudSearch `fq` parameter.

FIM

See federated identity management.

format version

See template format version.

forums

See discussion forums.

function

See intrinsic function.

fuzzy search

A simple search query that uses approximate string matching (fuzzy matching) to correct for typographical errors and misspellings.

G

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

geospatial search

A search query that uses locations specified as a latitude and longitude to determine matches and sort the results.

gibibyte

A contraction of giga binary byte, a gibibyte is 2^30 bytes or 1,073,741,824 bytes. A gigabyte is 10^9 or 1,000,000,000 bytes.

H

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

Hadoop

See http://hadoop.apache.org.

hard bounce

A persistent email delivery failure such as "mailbox does not exist."

hardware VPN

A hardware-based IPsec VPN connection over the Internet.

HDFS

Hadoop Distributed File System. The HDFS file system stores large files across multiple machines. It achieves reliability by replicating the data across multiple hosts, and hence does not require RAID storage on hosts.

health check

A system call to check on the health status of each instance in an Auto Scaling group.

high-quality email

Email that recipients find valuable and want to receive. Value means different things to different recipients and can come in the form of offers, order confirmations, receipts, newsletters, etc.

highlights

Excerpts returned with Amazon CloudSearch results that show where the search terms appear within the text of the matching documents.

highlight enabled

An Amazon CloudSearch index field option that enables matches within the field to be highlighted.

hit

A document that matches the criteria specified in a search request. Also referred to as a *search result*.

HIT

See Human Intelligence Task.

Hive

An open source, data warehouse and analytic package that runs on top of Hadoop. Hive scripts use an SQL-like language called Hive QL (query language) that abstracts the MapReduce programming model and supports typical data warehouse interactions.

HMAC

Hash-based Message Authentication Code. A specific construction for calculating a message authentication code (MAC) involving a cryptographic hash function in combination with a secret key. You can use it to verify both the data integrity and the authenticity of a message at the same time. AWS calculates the HMAC using a standard, cryptographic hash algorithm, such as SHA-256.

hosted zone

A collection of resource record sets that Amazon Route 53 hosts. Like a traditional DNS zone file, a hosted zone represents a collection of records that are managed together under a single domain name.

Human Intelligence Task

A task that a Requester (p. 640) submits to Mechanical Turk for workers (p. 648) to perform. A HIT represents a single, self-contained task, for example, "Identify the car color in the photo." HITs contain all of the information a worker needs to answer a question, including the kinds of answers you would consider valid.

HVM virtualization

Hardware Virtual Machine virtualization. Allows the guest VM to run as though it is on a native hardware platform, except that it still uses paravirtual (PV) network and storage drivers for improved performance.
See Also PV virtualization.

I

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

image

See Amazon Machine Image.

import/export station

A machine that uploads or downloads your data to, or from, Amazon S3.

import log

A report that contains details about how AWS Import/Export processed your data.

index

See search index.

index field

A name-value pair that is included in an Amazon CloudSearch domain's index. An index field can contain text or numeric data, dates, or a location.

indexing options

Configuration settings that define an Amazon CloudSearch domain's index fields, how document data is mapped to those index fields, and how the index fields can be used.

instance

A copy of an Amazon Machine Image running as a virtual server in the AWS cloud.

instance family

A general instance type (p. 631) grouping using either storage or CPU capacity.

instance group

A Hadoop cluster contains one master instance group that contains one master node (p. 634), a core instance group containing one or more core node (p. 624) and an optional task node (p. 645) instance group, which can contain any number of task nodes.

instance store

Disk storage that is physically attached to the host computer for an EC2 instance, and therefore has the same lifespan as the instance. When the instance terminates, you lose any data in the instance store.

instance store-backed AMI

Instances launched from this type of AMI use an instance store volume as the root device. Compare this with instances launched from Amazon EBS-backed AMIs, which use an Amazon EBS volume as the root device.

instance type

A specification that defines the memory, CPU, storage capacity, and hourly cost for an instance. Some instance types are designed for standard applications, whereas others are designed for CPU-intensive, memory-intensive applications, and so on.

Internet gateway

Connects a network to the Internet. You can route traffic for IP addresses outside your VPC (p. 647) to the Internet gateway.

Internet Service Provider

A company that provides subscribers with access to the Internet. Many ISPs are also mailbox provider (p. 633)s. Mailbox providers are sometimes referred to as ISPs, even if they only provide mailbox services.

intrinsic function

A special action in a template that assigns values to properties not available until runtime. These functions follow the format *Fn::Attribute*, such as `Fn::GetAtt`. Arguments for intrinsic functions can be parameters, pseudo parameters, or the output of other intrinsic functions.

IP address

All EC2 instances are assigned two IP addresses at launch, which are directly mapped to each other through network address translation (NAT): a private IP address (following RFC 1918) and a public IP address. Instances launched in a VPC are assigned only a private IP address. Instances launched in your default VPC are assigned both a private IP address and a public IP address.

ISP

See Internet Service Provider.

issuer

The issuer is the person who writes a policy to grant permissions to a resource. The issuer (by definition) is always the resource owner. AWS does not permit Amazon SQS users to create policies for resources they don't own. If John is the resource owner, AWS authenticates John's identity when he submits the policy he's written to grant permissions for that resource.

item

Similar to rows on a spreadsheet, items represent individual objects that contain one or more value-attribute pairs.

item name

An identifier for an item. The identifier must be unique within the domain (p. 626).

J

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

job flow

A job flow specifies the complete processing of the data. It's comprised of one or more steps, which specify all of the functions to be performed on the data.

job ID

A five-character, alphanumeric string that uniquely identifies a storage device in your shipment. AWS issues the job ID in response to a `CREATE JOB` email command.

job prefix

The AWS Import/Export process generates a log file. The log file name always ends with the phrase *import-log-* followed by your Job ID. There is a remote chance that you already have an object with this name. To avoid a key collision, you can add an optional prefix to the log file.
See Also key prefix.

JSON

JavaScript Object Notation. A lightweight data-interchange format. For information about JSON, see http://www.json.org/.

junk folder

The location where email messages that various filters determine to be of lesser value are collected so that they do not arrive in the recipient (p. 639)'s inbox, but are still accessible to the recipient. This is also referred to as a spam (p. 643) or bulk folder.

K

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

key

A credential that identifies an AWS account or user to AWS (such as the AWS secret access key (p. 642)).

Amazon S3, Amazon EMR: The unique identifier for an object in a bucket. Every object in a bucket has exactly one key. Because a bucket and key together uniquely identify each object, you can think of Amazon S3 as a basic data map between the *bucket* + *key*, and the object itself. You can uniquely address every object in Amazon S3 through the combination of the web service endpoint, bucket name, and key, for example: `http://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl`, where `doc` is the name of the bucket, and `2006-03-01/AmazonS3.wsdl` is the key.

AWS Import/Export: The name of an object in Amazon S3. It is a sequence of Unicode characters whose UTF-8 encoding cannot exceed 1024 bytes. If a key, for example, logPrefix + import-log-JOBID, is longer than 1024 bytes, AWS Elastic Beanstalk returns an `InvalidManifestField` error.

IAM: In the context of writing a policy (p. 636): A specific characteristic that is the basis for restricting access (such as the current time, or the IP address of the requester).

key pair

A set of security credentials you use to prove your identity electronically. A key pair consists of a private key and a public key.

key prefix

A logical grouping of the objects in a bucket (p. 622). The prefix value is similar to a directory name that enables you to store similar data under the same directory in a bucket.

L

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

launch configuration

A set of descriptive parameters used to create new EC2 instances in an Auto Scaling activity.

A template that an Auto Scaling group (p. 621) uses to launch new EC2 instances. The launch configuration contains information such as the Amazon Machine Image (p. 619) ID, the instance type, key pairs, security groups, and block device mappings, among other configuration settings.

launch permission

An Amazon Machine Image (p. 619) (AMI) attribute that allows users to launch an AMI.

lifecycle

The lifecycle state of the EC2 instance (p. 626) contained in an AutoScalingGroup. EC2 instances progress through several states over their lifespan; these include *Pending*, *InService*, *Terminating* and *Terminated*.

load balancer

A load balancer is a combination of a DNS name and a set of ports, which together provide a destination for all requests intended for your application. A load balancer can distribute traffic to multiple application instances across every Availability Zone (p. 621) within a region (p. 639). Load balancers can span multiple Availability Zones within an Amazon EC2 region, but they cannot span multiple regions.

logical name

A case-sensitive unique string within an AWS CloudFormation template that identifies a resource (p. 640), mapping (p. 633), parameter, or output. In an AWS CloudFormation template, each parameter, resource, property, mapping, and output must be declared with a unique logical name. You use the logical name when dereferencing these items using the `Ref` function.

M

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

machine utilization

The amount of machine capacity used to complete a particular request (for example SELECT, GET, PUT, and so on), normalized to the hourly capacity of a standard processor. Machine utilization is measured in machine hour increments.

Mail Transfer Agent (MTA)

Software that transports email messages from one computer to another by using a client-server architecture.

mailbox provider

An organization that provides email mailbox hosting services. Mailbox providers are sometimes referred to as Internet Service Provider (p. 631)s, even if they only provide mailbox services.

mailbox simulator

A set of email addresses that you can use to test an Amazon SES-based email sending application without sending messages to actual recipients. Each email address represents a specific scenario (such as a bounce or complaint) and generates a typical response that is specific to the scenario.

main route table

The default route table that any new VPC subnet uses for routing. You can associate a subnet with a different route table of your choice. You can also change which route table is the main route table.

manifest

When sending a *create job* request for an import or export operation you describe your job in a text file called a manifest. The manifest file is a YAML-formatted file that specifies how to transfer data between your storage device and the AWS cloud.

MapReduce

See http://hadoop.apache.org/docs/r1.2.0/mapred_tutorial.html.

mapper

An executable that splits the raw data into key/value pairs. The reducer uses the output of the mapper, called the *intermediate results*, as its input.

mapping

A way to add conditional parameter values to an AWS CloudFormation template. You specify mappings in the template's optional Mappings section and retrieve the desired value using the *`FN::FindInMap`* function.

marker

See pagination.

master node

A process running on an Amazon Machine Image (p. 619) that keeps track of the work its core and task nodes complete.

maximum price

The maximum price you will pay to launch one or more Spot Instances. If your maximum price exceeds the current Spot Price (p. 643) and your restrictions are met, Amazon EC2 launches instances on your behalf.

maximum send rate

The maximum number of emails that you can send per second using Amazon SES.

member resources

See resource.

message ID

Amazon SES: A unique identifier that is assigned to every email message that is sent.

Amazon SQS: The identifier returned when you send a message to a queue.

metadata

Amazon S3, Amazon EMR: A set of name/value pairs that describe the object. These include default metadata such as the date last modified and standard HTTP metadata such as Content-Type. Users can also specify custom metadata at the time they store an object.
Amazon EC2: Data about an EC2 instance (p. 626) that the instance can retrieve to determine things about itself, such as, the instance type, the IP address, and so on.

metric

An element of time-series data defined by a unique combination of exactly one namespace, exactly one metric name, and between zero and ten dimensions. Metrics and the statistics derived from them are the basis of Amazon CloudWatch.

metric name

The primary identifier of a metric, used in combination with a namespace and optional dimensions.

MFA

See AWS Multi-Factor Authentication.

micro instance

A type of EC2 instance (p. 626) that is more economical to use if you have occasional bursts of high CPU activity.

MIME

See Multipurpose Internet Mail Extensions (MIME).

MTA

See Mail Transfer Agent (MTA).

Multi-AZ deployment

A primary DB instance (p. 625) that has a synchronous standby replica in a different Availability Zone (p. 621). The primary DB instance is synchronously replicated across Availability Zones to the standby replica.

Multi-Factor Authentication

See AWS Multi-Factor Authentication.

multi-valued attribute

An attribute with more than one value.

multipart upload

A feature that allows you to upload a single object as a set of parts.

Multipurpose Internet Mail Extensions (MIME)

An Internet standard that extends the email protocol to include non-ASCII text and non-text elements like attachments.

Multitool

A Cascading (p. 623) application that provides a simple command-line interface for managing large datasets.

N

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

namespace
An abstract container that provides context for the items (names, or technical terms, or words) it holds, and allows disambiguation of homonym items residing in different namespaces.

NAT
Network address translation.

NAT instance
An instance that is configured to perform NAT (p. 635) in a VPC. A NAT instance enables private instances in the VPC to initiate Internet-bound traffic without being directly reachable from the Internet.

network ACL
An optional layer of security that acts as a firewall for controlling traffic in and out of a subnet. You can associate multiple subnets with a single network ACL, but a subnet can be associated with only one network ACL at a time.

node
After an Amazon Machine Image (p. 619) is launched, the resulting running system is referred to as a node. All instances based on the same AMI are identical at start-up. Any information about the node is lost when the node terminates or fails.

NoEcho
A property of AWS CloudFormation parameters that will prevent the otherwise default reporting of names and values of a template parameter. Declaring the *`NoEcho`* property causes the parameter value to be masked with asterisks in the report by the `cfn-describe-stacks` command.

notification email
The email Amazon Payments sends to your customers to notify them of an upcoming price change you've scheduled.

null object
A null object is one whose version ID is null. Amazon S3 adds a null object to a bucket when versioning (p. 647) for that bucket is suspended. It is possible to have only one null object for each key in a bucket.

O

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

object
Amazon S3: The fundamental entity type stored in Amazon S3. Objects consist of object data and metadata. The data portion is opaque to Amazon S3.

CloudFront: Any entity that can be served either over HTTP or a version of RTMP.

on-demand instance
An Amazon EC2 pricing option that charges you for compute capacity by the hour with no long-term commitment.

operation
An API function. Also called an *action*.

origin access identity
Also called OAI. A virtual identity you use when giving your distribution permission to fetch a private object from your origin server (Amazon S3 bucket).

origin server
The Amazon S3 bucket or custom origin containing the definitive original version of the content you deliver through CloudFront.

P

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

pagination

Some APIs that return a potentially large list of records can return a subset by using a value to set the maximum number of returned records. They then provide a marker, which identifies the last record returned so that in a subsequent call, the user can get the next sequence of records.

paid AMI

An Amazon Machine Image (AMI) that you sell to other Amazon EC2 users using Amazon DevPay.

paravirtual virtualization

See PV virtualization.

part

In a multipart upload request, each part is a contiguous portion of the object's data.

passive authorization

A passive authorization happens when you inform customers at least 14 days in advance of a price change for your product, and they don't take any action; the price change is accepted automatically.

PAT

Port address translation.

period

See sampling period.

permission

A statement within a policy (p. 636) that allows or denies access to a particular resource. You can state any permission like this: "A has permission to do B to C." For example, Jane (A) has permission to read messages (B) from John's Amazon SQS queue (C). Whenever Jane sends a request to Amazon SQS to use John's queue, the service checks to see if she has permission and if the request satisfies the conditions John set forth in the permission.

persistent identifier

Also called PID. An encoded string that represents the relationship between a customer and the owner of Amazon DevPay products. After a customer purchases one of your products, you can use the PID to confirm the status of the customer's subscription to the product.

persistent storage

A long-term data storage solution. Options within AWS are: Amazon S3, Amazon EBS, and Amazon SimpleDB.

physical name

A unique label AWS CloudFormation assigns to each resource when creating a stack (p. 643). Some AWS CloudFormation commands accept the physical name as a value with the `--physical-name` parameter.

Pig

An open-source Apache library that runs on top of Hadoop. The library takes SQL-like commands written in a language called Pig Latin and converts those commands into MapReduce job flows.

policy

A document defining permissions that apply to a user, group, or role; the permissions in turn determine what users can do in AWS. A policy typically allow (p. 618)s access to specific actions, and can optionally grant that the actions are allowed for specific resources, like EC2 instances, S3 buckets, and so on. Policies can also explicitly deny (p. 625) access.

Auto Scaling: An object that stores the information needed to launch or terminate instances for an Auto Scaling group. Executing the policy causes instances to be launched or terminated. You can configure an alarm (p. 618) to invoke an Auto Scaling policy.

pre-signed URL

A URL that uses query string authentication (p. 638).

prefix

See job prefix.

Premium Support

A one-on-one, fast-response support channel that AWS customers can subscribe to for support for AWS infrastructure services.
See Also https://aws.amazon.com/premiumsupport/.

principal

The user, service, or account that receives permissions that are defined in a policy (p. 636). The principal is A in the statement "A has permission to do B to C."

private IP address

All EC2 instances are assigned two IP addresses at launch, which are directly mapped to each other through Network Address Translation (NAT): a private address (following RFC 1918) and a public address. *Exception:* Instances launched in Amazon VPC are assigned only a private IP address.

private subnet

A VPC subnet whose instances cannot be reached from the Internet.

product activation

See activation.

product code

The product code is an eight-character string that identifies your registered product to AWS.

product identification token

See product token.

product token

The product token is a long encoded string that identifies the product to AWS. You might also see the product token referred to as the *product identification token*.

properties

See resource property.

property rule

A JSON (p. 632)-compliant markup standard for declaring properties, mappings, and output values in an AWS CloudFormation template.

Provisioned IOPS

A storage option designed to deliver fast, predictable, and consistent I/O performance. When you specify an IOPS rate while creating a DB instance, Amazon RDS provisions that IOPS rate for the lifetime of the DB instance.

pseudo parameter

A predefined setting, such as `AWS:StackName` that can be used in AWS CloudFormation templates without having to declare them. You can use pseudo parameters anywhere you can use a regular parameter.

public AMI

An Amazon Machine Image (p. 619) that all AWS accounts have permission to launch.

public data set

A large set of public data that can be seamlessly integrated into AWS cloud-based applications. Amazon stores public data sets at no charge to the community and, like all AWS services, users pay only for the compute and storage they use for their own applications. These data sets currently include data from the Human Genome Project, the U.S. Census, Wikipedia, and other sources.
See Also http://aws.amazon.com/publicdatasets.

public IP address

All EC2 instances are assigned two IP addresses at launch, which are directly mapped to each other through Network Address Translation (NAT): a private address (following RFC 1918) and a public address. *Exception:* Instances launched in Amazon VPC are assigned only a private IP address.

public subnet

A subnet whose instances can be reached from the Internet.

purchase URL

The URL your customers use to purchase your product. When you advertise your product, you provide the purchase URL as the sign-up link for customers to use.

PV virtualization

Paravirtual virtualization. Allows guest VMs to run on host systems that do not have special support extensions for full hardware and CPU virtualization. Because PV guests run a modified operating system that does not use hardware emulation, they cannot provide hardware-related features such as enhanced networking or GPU support.
See Also HVM virtualization.

Q

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

Qualification

A property associated with a worker (p. 648) that represents that worker's skill, ability, or reputation. A Requester (p. 640) can use Qualifications to control which workers can perform HITs. Each worker can have multiple Qualifications.

Qualification requirements

A Human Intelligence Task (p. 630) can have Qualification requirements that a worker's Qualifications (q.v.) must meet before the worker can accept that HIT.

Qualification test

A form, similar to a HIT, containing a set of questions that the worker must complete successfully to receive a particular Qualification (p. 638).

Qualification type

Just as each worker (p. 648) has one or more Qualification (p. 638), each Human Intelligence Task (p. 630) has one or more Qualification type. These types specify what Qualifications the worker must have.

Query

A type of HTTP-based request interface that generally uses only the GET or POST HTTP method and a query string with parameters.
See Also REST, REST-Query.

query string authentication

An AWS feature that lets you place the authentication information in the HTTP request query string instead of in the Authorization header. For example: with Amazon DevPay, query string authentication enables your product to give anyone easy, URL-based access to objects in the customer's bucket.

queue

A sequence of messages or jobs held in temporary storage awaiting transmission or processing.

queue URL

A URL that uniquely identifies a queue.

quota

Amazon RDS: The maximum number of DB instance (p. 625)s and available storage you can use.

ElastiCache: The maximum number of the following items:

- The number of cache clusters for each AWS account
- The number of cache nodes per cache cluster
- The total number of cache nodes per AWS account across all cache clusters created by that AWS account

R

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

range GET

A range GET specifies a byte range of data to get for a download. If an object is large, you can break up a download into smaller units by sending multiple range GET requests that each specify a different byte range to GET.

raw email

A type of *sendmail* request that allows you to specify the email headers and MIME types.

RDS

See Amazon Relational Database Service.

read replica

An active copy of another DB instance. Any updates to the data on the source DB instance are replicated to the read replica DB instance using the built-in replication feature of MySQL 5.1.

receipt handle

An identifier you get when you receive a message from the queue. This identifier is required to delete a message from the queue or when changing a message's visibility timeout.

receiver

The entity that consists of the network systems, software, and policies that manage email delivery for a recipient (p. 639).

recipient

Amazon SES: The person or entity receiving an email message. For example, a person named in the "To" field of a message.

redirect URL

The page on your own website that you want customers to see at the end of the purchase process for your product. You provide the URL when you register the product with Amazon DevPay.

reducer

An executable in the MapReduce process that uses the intermediate results from the mapper and processes them into the final output.

reference

A means of inserting a property from one AWS resource into another. For example, you could insert an Amazon EC2 security group property into an Amazon RDS resource.

region

A named set of AWS resources in the same geographical area. A region comprises at least two Availability Zones.

reply path

The email address to which an email reply is sent. This is different from the return path (p. 640).

reputation

1. An Amazon SES metric, based on factors that might include bounces, complaints, and other metrics, regarding whether or not a customer is sending high-quality emails.

2. A measure of confidence, as judged by an Internet Service Provider (p. 631) or other entity that an IP address that they are receiving emails from is not the source of spam (p. 643).

requester

The person (or application) that sends a request to AWS to perform a specific action. When AWS receives a request, it first evaluates the requester's permissions to determine whether the requester is allowed to perform the request action (if applicable, for the requested resource).
See Also Requester.

Requester

(Note capitalization) A company, organization, or person that creates and submits tasks (a Human Intelligence Task (p. 630)) to Mechanical Turk; for workers (p. 648) to perform.

Requester Pays

An Amazon S3 feature that allows a bucket owner (p. 622) to specify that anyone who requests access to objects in a particular bucket must pay the data transfer and request costs.

reservation

A collection of EC2 instances started as part of the same launch request. Not to be confused with a Reserved Instance (p. 640).

Reserved Instance

A pricing option that lets you make a low, one-time payment for each instance to reserve and receive a significant discount on the hourly usage charge for that instance.

Reserved Instance Marketplace

Matches sellers who have reserved capacity that they no longer need with buyers who are looking to purchase additional capacity. Reserved Instances that you purchase from third-party sellers will have less than a full standard term remaining and can be sold at different upfront prices. The usage or reoccurring fees will remain the same as the fees set when the Reserved Instances were originally purchased. Full standard terms for Reserved Instances available from AWS run for one year or three years.

resource

1. An entity that users can work with in AWS, such as an EC2 instance, a DynamoDB table, an IAM user, an AWS OpsWorks stack, and so on.
2. Tools, code, and documents that AWS provides to support users.
3. A required element of an AWS CloudFormation stack (p. 643). Each stack contains at least one resource, such as an Auto Scaling LaunchConfiguration. All resources in a stack must be created successfully for the stack to be created.

resource property

A value required when including an AWS resource in an AWS CloudFormation stack (p. 643). Each resource may have one or more properties associated with it. For example, an `AWS::EC2::Instance` resource may have a `UserData` property. In an AWS CloudFormation template, resources must declare a properties section, even if the resource has no properties.

resource record

Also called *resource record set*. Standard DNS terminology.
See Also http://en.wikipedia.org/wiki/Domain_Name_System.

REST

A type of HTTP-based request interface that generally uses only the GET or POST HTTP method and a query string with parameters. Sometimes known as Query. In some implementations of a REST interface, other HTTP verbs besides GET and POST are used.

REST-Query

Also known as Query or HTTP Query. This is a type of HTTP request that generally uses only the GET or POST HTTP method and a query string with parameters. Compare this with REST, which is a type of HTTP request that uses any HTTP method (GET, DELETE, POST, etc.), a resource, HTTP headers, and possibly a query string with parameters.

return enabled

An Amazon CloudSearch index field option that enables the field's values to be returned in the search results.

return path

The email address to which bounced emails are returned. The return path is specified in the header of the original email. This is different from the reply path (p. 639).

reward

The money a Requester (p. 640) pays a worker (p. 648) for satisfactory work done on the Requester's Human Intelligence Task (p. 630)s.

rollback

A return to a previous state that follows the failure to create an object, such as AWS CloudFormation stack (p. 643). All resources associated with the failure are deleted during the rollback. For AWS CloudFormation, you can override this behavior using the `--disable-rollback` option on the command line.

root device volume

Contains the image used to boot the instance. If you launched the instance from an AMI backed by instance store, this is an instance store volume created from a template stored in Amazon S3. If you launched the instance from an AMI backed by Amazon EBS, this is an Amazon EBS volume created from an Amazon EBS snapshot.

route table

A set of routing rules that controls the traffic leaving any subnet that is associated with the route table. You can associate multiple subnets with a single route table, but a subnet can be associated with only one route table at a time.

S

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

sampling period

A defined duration of time, such as one minute, over which CloudWatch computes a statistic (p. 643).

sandbox

A testing location where you can test the functionality of your application without affecting production, incurring charges, or purchasing products.

Amazon SES: An Amazon SES environment that is designed for developers to test and evaluate the service. In the sandbox, you have full access to the Amazon SES API, but you can only send messages to verified email addresses and the mailbox simulator. To get out of the sandbox, you need to apply for production access. Accounts in the sandbox also have lower sending limits (p. 642) than production accounts.

scaling activity

A process that changes the size, configuration, or makeup of an Auto Scaling group (p. 621) by launching or terminating instances. For more information, see Auto Scaling Concepts in the Auto Scaling Developer Guide.

search API

The Amazon CloudSearch API that you use to submit search requests to a search domain.

search domain

Encapsulates your searchable data and the search instances that handle your search requests. You typically set up a separate Amazon CloudSearch domain for each different collection of data that you want to search.

search domain configuration

An Amazon CloudSearch domain's indexing options, analysis schemes, expressions, suggesters, access policies, and scaling and availability options.

search enabled

An Amazon CloudSearch index field option that enables the field data to be searched.

search endpoint

The URL that you connect to when sending search requests to a search domain. Each Amazon CloudSearch domain has a unique search endpoint that remains the same for the life of the domain.

search index

A representation of your searchable data that facilitates fast and accurate data retrieval.

search instance

A compute resource that indexes your data and processes search requests. An Amazon CloudSearch domain has one or more search instances, each with a finite amount of RAM and CPU resources. As your data volume grows, more search instances or larger search instances are deployed to contain your indexed data. When necessary, your index is automatically partitioned across multiple search instances. As your request volume or complexity increases, each search partition is automatically replicated to provide additional processing capacity.

search request

A request that is sent to an Amazon CloudSearch domain's search endpoint to retrieve documents from the index that match particular search criteria.

search result

A document that matches a search request. Also referred to as a *search hit*.

secret access key

A key that is used in conjunction with the access key ID (p. 617) to cryptographically sign programmatic AWS requests. Signing a request identifies the sender and prevents the request from being altered. You can generate secret access keys for your AWS account, individual IAM users, and temporary sessions.

security group

A named set of allowed inbound network connections for an instance. (Security groups in Amazon VPC also include support for outbound connections.) Each security group consists of a list of protocols, ports, and IP address ranges. A security group can apply to multiple instances, and multiple groups can regulate a single instance.

sender

The person or entity sending an email message.

Sender ID

A Microsoft-controlled version of SPF. An email authentication and anti-spoofing system. For more information about Sender ID, go to http://wikipedia.org/wiki/Sender_ID.

sending limits

The sending quota (p. 642) and maximum send rate (p. 634) that are associated with every Amazon SES account.

sending quota

The maximum number of emails that you can send using Amazon SES in a 24-hour period.

service endpoint

See endpoint.

service health dashboard

A web page showing up-to-the-minute information about AWS service availability. The dashboard is located at http://status.aws.amazon.com.

SHA

Secure Hash Algorithm. SHA1 is an earlier version of the algorithm, which AWS has deprecated in favor of SHA256.

shared AMI

An Amazon Machine Image (p. 619) that a developer builds and makes available for others to use.

shutdown action

A predefined bootstrap action that launches a script that executes a series of commands in parallel before terminating the job flow.

signature

Refers to a *digital signature*, which is a mathematical way to confirm the authenticity of a digital message. AWS uses signatures to authenticate the requests you send to our web services. For more information, to http://aws.amazon.com/security.

SIGNATURE file

A file you copy to the root directory of your storage device. The file contains a job ID, manifest file, and a signature.

Simple Mail Transfer Protocol

See SMTP.

Single-AZ DB Instance

A standard (non-Multi-AZ) DB instance (p. 625) that is deployed in one Availability Zone (p. 621), without a standby replica in another Availability Zone.
See Also Multi-AZ deployment.

single-valued attribute

An attribute with one value.

sloppy phrase search

A search for a phrase that specifies how close the terms must be to one another to be considered a match.

SMTP

Simple Mail Transfer Protocol. The standard that is used to exchange email messages between internet hosts for the purpose of routing and delivery.

snapshot

Amazon Elastic Block Store (p. 619) creates *snapshots* or backups of your volumes and stores them in Amazon S3. You can use these snapshots as the starting point for new Amazon EBS volumes or to protect your data for long-term durability.

soft bounce

A temporary email delivery failure such as "mailbox full."

software VPN

A software appliance-based VPN connection over the Internet.

sort enabled

An Amazon CloudSearch index field option that enables a field to be used to sort the search results.

source/destination checking

A security measure to verify that an EC2 instance is the origin of all traffic that it sends and the ultimate destination of all traffic that it receives, that is, that the instance is not relaying traffic. Source/destination checking is enabled by default. For instances that function as gateways, such as VPC NAT instances, source/destination checking must be disabled.

spam

Unsolicited bulk email.

spamtrap

An email address that is set up by an anti-spam (p. 643) entity, not for correspondence, but to monitor unsolicited email. This is also called a *honeypot*.

SPF

Sender Policy Framework. A standard for authenticating email.
See Also http://www.openspf.org.

Spot Instance

A type of EC2 instance (p. 626) that you can bid on to take advantage of unused Amazon EC2 capacity.

Spot Price

The price for a Spot Instance (p. 643) at any given time. If your maximum price exceeds the current price and your restrictions are met, Amazon EC2 launches instances on your behalf.

stack

AWS CloudFormation: A collection of AWS resources you create and delete as a single unit.

AWS OpsWorks: A set of instances you manage collectively, typically because they have a common purpose such as serving PHP applications. A stack serves as a container and handles tasks that apply to the group of instances as a whole, such as managing applications and cookbooks.

station

A place at an AWS facility where we transfer your AWS Import/Export data on to, or off of, your storage device.

statistic

One of five functions of the values submitted for a given sampling period (p. 641). These functions are "Maximum", "Minimum," "Sum," "Average," and "SampleCount."

stem

The common root or substring shared by a set of related words.

stemming

The process of mapping related words to a common stem. This enables matching on variants of a word. For example, a search for "horse" could return matches for horses, horseback, and horsing, as well as horse. Amazon CloudSearch supports both dictionary based and algorithmic stemming.

step

A single function applied to the data in a job flow (p. 631). The sum of all steps comprises a job flow.

step type

The type of work done in a step. There are a limited number of step types, such as moving data from Amazon S3 to Amazon EC2 or from Amazon EC2 to Amazon S3.

sticky session

A feature of the load balancer that binds a user's session to a specific application instance so that all requests coming from the user during the session are sent to the same application instance. By contrast, a load balancer defaults to route each request independently to the application instance with the smallest load.

stopping

The process of filtering stop words from an index or search request.

stopword

A word that is not indexed and is automatically filtered out of search requests because it is either insignificant or so common that including it would result in too many matches to be useful. Stop words are language-specific.

streaming

Amazon EMR: A utility that comes with Hadoop that enables you to develop MapReduce executables in languages other than Java.

CloudFront: The ability to use a media file in real time—as it is transmitted in a steady stream from a server.

streaming distribution

A special kind of distribution (p. 626) that serves streamed media files using a Real Time Messaging Protocol (RTMP) connection.

string-to-sign

Before you calculate an HMAC signature, you first assemble the required components in a canonical order. The pre-encrypted string is the string-to-sign.

structured query

Search criteria specified using the Amazon CloudSearch structured query language. You use the structured query language to construct compound queries that use advanced search options and combine multiple search criteria using Boolean operators.

subnet

A segment of the IP address range of a VPC (p. 647) that EC2 instances can be attached to. You can create subnets to group instances according to security and operational needs.

Subscription button

An HTML-coded button that enables an easy way to charge customers a recurring fee.

suggester

Specifies an Amazon CloudSearch index field you want to use to get autocomplete suggestions and options that can enable fuzzy matches and control how suggestions are sorted.

suggestions

Documents that contain a match for the partial search string in the field designated by the suggester. Amazon CloudSearch suggestions include the document IDs and field values for each matching document. To be a match, the string must match the contents of the field starting from the beginning of the field.

supported AMI

An Amazon Machine Image (p. 619) similar to a paid AMI (p. 636), except that the owner charges for additional software or a service that customers use with their own AMIs.

synchronous bounce

A type of bounce (p. 622) that occurs while the email servers of the sender (p. 642) and receiver (p. 639) are actively communicating.

synonym

A word that is the same or nearly the same as an indexed word and that should produce the same results when specified in a search request. For example, a search for "Rocky Four" or "Rocky 4" should return the fourth *Rocky* movie. This can be done by designating that `four` and `4` are synonyms for `IV`. Synonyms are language-specific.

system Qualifications

The set of Qualifications (p. 638) that represent a worker's (p. 648) history and reputation. The Mechanical Turk system assigns these Qualifications to each worker, and continuously updates the values as they use the system.

T

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

tag

Metadata (consisting of up to 10 key/value pairs) that you can define and assign to Amazon EC2 resources.

tagging

Also called *labeling*. A way to format return path (p. 640) email addresses so that you can specify a different return path for each recipient of a message. Tagging enables you to support VERP (p. 647). For example, if Andrew manages a mailing list, he can use the return paths andrew+recipient1@example.net and andrew+recipient2@example.net so that he can determine which email bounced.

task node

An EC2 instance (p. 626) that runs Hadoop map and reduce tasks, but does not store data. Task nodes are managed by the master node (p. 634), which assigns Hadoop tasks to nodes and monitors their status. While a job flow is running you can increase and decrease the number of task nodes. Because they don't store data and can be added and removed from a job flow, you can use task nodes to manage the EC2 instance capacity your job flow uses, increasing capacity to handle peak loads and decreasing it later.

Task nodes only run a TaskTracker Hadoop daemon.

tebibyte

A contraction of tera binary byte, a tebibyte is 2^40 bytes or 1,099,511,627,776 bytes. A terabyte is 10^12 or 1,000,000,000,000 bytes.

template format version

The version of an AWS CloudFormation template design that determines the available features. If you omit the `AWSTemplateFormatVersion` section from your template, AWS CloudFormation assumes the most recent format version.

template validation

The process of confirming the use of JSON (p. 632) code in an AWS CloudFormation template. You can validate any AWS CloudFormation template using the `cfn-validate-template` command.

throttling

The means by which Amazon SES rejects your attempts to send email because you have exceeded your sending limits (p. 642).

time series data

Data provided as part of a metric. The time value is assumed to be when the value occurred. A metric is the fundamental concept for CloudWatch and represents a time-ordered set of data points. You publish metric data points into CloudWatch and later retrieve statistics about those data points as a time-series ordered data set.

time stamp

A date/time string in ISO 8601 format.

TLS

See Transport Layer Security.

tokenization

The process of splitting a stream of text into separate tokens on detectable boundaries such as whitespace and hyphens.

topic

A communication channel to send messages and subscribe to notifications. It provides an access point for publishers and subscribers to communicate with each other.

Transport Layer Security

A cryptographic protocol that provides security for communication over the Internet. Its predecessor is Secure Sockets Layer (SSL).

trusted signers

AWS accounts that the CloudFront distribution owner has given permission to create signed URLs for a distribution's content.

tuning

Selecting the number and type of AMIs (p. 619) to run a Hadoop job flow most efficiently.

tunnel

A route for transmission of private network traffic that uses the Internet to connect nodes in the private network. The tunnel uses encryption and secure protocols such as PPTP to prevent the traffic from being intercepted as it passes through public routing nodes.

U

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

unbounded

The number of potential occurrences is not limited by a set number. This value is often used when defining a data type that is a list (for example, `maxOccurs="unbounded"`), in Web Services Description Language (p. 648).

unit

Standard measurement for the values submitted to CloudWatch as metric data. Units include Seconds, Percent, Bytes, Bits, Count, Bytes/Second, Bits/Second, Count/Second, and None.

usage report

An AWS report giving details of your usage of a particular AWS service. You can generate and download usage reports from http://aws.amazon.com/usage-reports/.

user

A person or application under an account (p. 617) that needs to make API calls to AWS products. Each user has a unique name within the AWS account, and a set of security credentials not shared with other users. These credentials are separate from the AWS account's security credentials. Each user is associated with one and only one AWS account.

user token

A customer credential returned to your product during product activation (p. 618). The user token is a long, encoded string that AWS uses to identify the customer. Your product provides the customer's user token in each request for Amazon S3 the product makes on behalf of the customer. Every user token generated for a particular customer differs from the others because the creation time is one of the items making up the token value.

V

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

validation

See template validation.

value

Instances of attributes (p. 620) for an item, such as cells in a spreadsheet. An attribute might have multiple values.

value-add

The amount you charge each customer on top of the cost of the AWS services they used.

Variable Envelope Return Path

See VERP.

verification

The process of confirming that you own an email address or a domain so that you can send emails from or to it.

VERP

Variable Envelope Return Path. A way in which email sending applications can match bounced emails with the undeliverable address that caused the bounce by using a different return path (p. 640) for each recipient. VERP is typically used for mailing lists. With VERP, the recipient's email address is embedded in the address of the return path, which is where bounced emails are returned. This makes it possible to automate the processing of bounced emails without having to open the bounce messages, which may vary in content.

versioning

Every object in Amazon S3 has a key and a version ID. Objects with the same key, but different version IDs can be stored in the same bucket. Versioning is enabled at the bucket layer using PUT Bucket versioning.

virtualization

Allows multiple guest virtual machines (VM) to run on a host operating system. Guest VMs can run on one or more levels above the host hardware, depending on the type of virtualization.
See Also PV virtualization, HVM virtualization.

virtual private cloud

See VPC.

virtual private gateway

See VPG.

visibility timeout

The period of time that a message is invisible to the rest of your application after an application component gets it from the queue. During the visibility timeout, the component that received the message usually processes it, and then deletes it from the queue. This prevents multiple components from processing the same message.

VPC

Virtual private cloud. An elastic network populated by infrastructure, platform, and application services that share common security and interconnection.

VPG

Virtual private gateway. The Amazon side of a VPN connection that maintains connectivity. The internal interfaces of the virtual private gateway connect to your VPC via the VPN attachment and the external interfaces connect to the VPN connection, which leads to the customer gateway.

VPN CloudHub

See AWS VPN CloudHub.

VPN connection

Although VPN connection is a general term, we specifically mean the IPsec connection between a VPC (p. 647) and some other network, such as a corporate data center, home network, or co-location facility.

W

Numbers and Symbols (p. 617) | A (p. 617) | B (p. 621) | C (p. 622) | D (p. 624) | E (p. 626) | F (p. 628) | G (p. 629) | H (p. 629) | I (p. 630) | J (p. 631) | K (p. 632) | L (p. 632) | M (p. 633) | N (p. 635) | O (p. 635) | P (p. 636) | Q (p. 638) | R (p. 639) | S (p. 641) | T (p. 645) | U (p. 646) | V (p. 647) | W (p. 648) | X, Y, Z (p. 648)

Web Services Description Language

A language used to describe the actions that a web service can perform, along with the syntax of action requests and responses. Your SOAP or other toolkit interprets a WSDL file to provide your application access to the actions provided by the web service. For most toolkits, your application calls a service action using routines and classes provided or generated by the toolkit.

worker

A person who performs the tasks specified by a Requester (p. 640) in a Human Intelligence Task (p. 630).

X, Y, Z

No entries