AWS Config

Developer Guide

AWS Config: Developer Guide

Table of Contents

What Is AWS Config?

AWS Config provides a detailed view of the configuration of AWS resources in your AWS account. This includes how the resources are related to one another and how they were configured in the past so that you can see how the configurations and relationships change over time.

An AWS *resource* is an entity you can work with in AWS, such as an Amazon Elastic Compute Cloud (EC2) instance, an Amazon Elastic Block Store (EBS) volume, a security group, or an Amazon Virtual Private Cloud (VPC), for example. For a complete list of AWS resources supported by AWS Config, see Supported AWS Resource Types (p. 6).

With AWS Config, you can do the following:

- Evaluate your AWS resource configurations for desired settings.
- Get a snapshot of the current configurations of the supported resources that are associated with your AWS account.
- Retrieve configurations of one or more resources that exist in your account.
- Retrieve historical configurations of one or more resources.
- Receive a notification whenever a resource is created, modified, or deleted.
- View relationships between resources. For example, you might want to find all resources that use a particular security group.

Ways to Use AWS Config

When you run your applications on AWS, you usually use AWS resources, which you must create and manage collectively. As the demand for your application keeps growing, so does your need to keep track of your AWS resources. AWS Config is designed to help you oversee your application resources in the following scenarios:

Resource Administration

To exercise better governance over your resource configurations and to detect resource misconfigurations, you need fine-grained visibility into what resources exist and how these resources are configured at any time. You can use AWS Config to notify you whenever resources are created, modified, or deleted without having to monitor these changes by polling the calls made to each resource.

You can use AWS Config rules to evaluate the configuration settings of your AWS resources. When AWS Config detects that a resource violates the conditions in one of your rules, AWS Config flags the resource as noncompliant and sends a notification. AWS Config continuously evaluates your resources as they are created, changed, or deleted.

Auditing and Compliance

You might be working with data that requires frequent audits to ensure compliance with internal policies and best practices. To demonstrate compliance, you need access to the historical configurations of your resources. This information is provided by AWS Config.

Managing and Troubleshooting Configuration Changes

When you use multiple AWS resources that depend on one another, a change in the configuration of one resource might have unintended consequences on related resources. With AWS Config, you can view how the resource you intend to modify is related to other resources and assess the impact of your change.

You can also use the historical configurations of your resources provided by AWS Config to troubleshoot issues and to access the last known good configuration of a problem resource.

Security Analysis

To analyze potential security weaknesses, you need detailed historical information about your AWS resource configurations, such as the AWS Identity and Access Management (IAM) permissions that are granted to your users, or the Amazon EC2 security group rules that control access to your resources.

You can use AWS Config to view the IAM policy that was assigned to an IAM user, group, or role at any time in which AWS Config was recording. This information can help you determine the permissions that belonged to a user at a specific time: for example, you can view whether the user `John Doe` had permission to modify Amazon VPC settings on Jan 1, 2015.

You can also use AWS Config to view the configuration of your EC2 security groups, including the port rules that were open at a specific time. This information can help you determine whether a security group blocked incoming TCP traffic to a specific port.

Concepts

Understanding the basic components of AWS Config will help you get the most out of this service.

Topics

AWS Config Rules

An AWS Config rule represents your desired configuration settings for specific AWS resources or for an entire AWS account. AWS Config provides customizable, predefined rules to help you get started. You can also create custom rules. While AWS Config continuously tracks your resource configuration changes, it checks whether these changes violate any of the conditions in your rules. If a resource violates a rule, AWS Config flags the resource and the rule as noncompliant, and AWS Config notifies you through Amazon SNS. For more information, see Evaluating Resources With AWS Config Rules (p. 23).

AWS Resources

AWS resources are entities that you create and manage using the AWS Management Console, the AWS Command Line Interface (CLI), the AWS SDKs, or AWS partner tools. Examples of AWS resources include Amazon EC2 instances, security groups, Amazon VPCs, and Amazon Elastic Block Store. AWS Config refers to each resource using its unique identifier, such as the resource ID or an Amazon Resource Name (ARN). For details, see Supported AWS Resource Types (p. 6).

Configuration Items

A *configuration item* represents a point-in-time view of the various attributes of a supported AWS resource that exists in your account. The components of a configuration item include metadata, attributes, relationships, current configuration, and related events. For details, see Components of a Configuration Item (p. 7).

Resource Relationship

AWS Config discovers AWS resources in your account and then creates a map of relationships between AWS resources. For example, a relationship might include an Amazon EBS volume `vol-123ab45d` attached to an Amazon EC2 instance `i-a1b2c3d4` that is associated with security group `sg-ef678hk`. For details, see Supported Resource Relationships (p. 8).

Configuration Snapshot

A configuration snapshot is a collection of the configuration items for the supported resources that exist in your account. This configuration snapshot is a complete picture of the resources that are being recorded and their configurations. The configuration snapshot can be a useful tool for validating your configuration. For example, you may want to examine the configuration snapshot regularly for resources that are configured incorrectly or that potentially should not exist. The configuration snapshot is available in multiple formats. You can have the configuration snapshot delivered to an Amazon Simple Storage Service (Amazon S3) bucket that you specify. Additionally, you can select a point in time in the AWS Config console and navigate through the snapshot of configuration items using the relationships between the resources.

Configuration Stream

A configuration stream is an automatically updated list of all configuration items for the resources that AWS Config is recording. Every time a resource is created, modified, or deleted, AWS Config creates a configuration item and adds to the configuration stream. The configuration stream works by using an Amazon Simple Notification Service (Amazon SNS) topic of your choice. The configuration stream is helpful for observing configuration changes as they occur so that you can spot potential problems, generating notifications if certain resources are changed, or updating external systems that need to reflect the configuration of your AWS resources.

Configuration History

A configuration history is a collection of the configuration items for a given resource over any time period. A configuration history can help you answer questions about, for example, when the resource was first created, how the resource has been configured over the last month, and what configuration changes were introduced yesterday at 9 AM. The configuration history is available to you in multiple formats. AWS Config automatically delivers a configuration history file for each resource type that is being recorded to an Amazon S3 bucket that you specify. You can select a given resource in the AWS Config console and navigate to all previous configuration items for that resource using the timeline. Additionally, you can access the historical configuration items for a resource from the API.

Configuration Recorder

The configuration recorder stores the configurations of the supported resources in your account as configuration items. You must first create and then start the configuration recorder before you can start recording. You can stop and restart the configuration recorder at any time. By default, the configuration recorder records all supported resources in the region where AWS Config is running. You can create a customized configuration recorder that records only the resource types that you specify.

For more information about how to record only specific resources, see Selecting Which Resources AWS Config Records (p. 76).

If you use the AWS Management Console or the CLI to turn on the service, AWS Config automatically creates and starts a configuration recorder for you.

How Does AWS Config Work?

When you turn on AWS Config, it first discovers the supported AWS resources that exist in your account and generates a configuration item (p. 3) for each resource.

AWS Config also generates configuration items when the configuration of a resource changes, and it maintains historical records of the configuration items of your resources from the time you start the configuration recorder. By default, AWS Config creates configuration items for every supported resource in the region. If you don't want AWS Config to create configuration items for all supported resources, you can specify the resource types that you want it to track.

AWS Config keeps track of all changes to your resources by invoking the Describe or the List API call for each resource in your account. The service uses those same API calls to capture configuration details for all related resources.

For example, removing an egress rule from a VPC security group causes AWS Config to invoke a Describe API call on the security group. AWS Config then invokes a Describe API call on all of the instances associated with the security group. The updated configurations of the security group (the resource) and of each instance (the related resources) are recorded as configuration items and delivered in a configuration stream to an Amazon Simple Storage Service (Amazon S3) bucket.

AWS Config also tracks the configuration changes that were not initiated by the API. AWS Config examines the resource configurations periodically and generates configuration items for the configurations that have changed.

If you are using AWS Config rules, AWS Config continuously evaluates your AWS resource configurations for desired settings. Depending on the rule, AWS Config will evaluate your resources either in response to configuration changes or periodically. Each rule is associated with an AWS Lambda function, which contains the evaluation logic for the rule. When AWS Config evaluates your resources, it invokes the rule's AWS Lambda function. The function returns the compliance status of the evaluated resources. If a resource violates the conditions of a rule, AWS Config flags the resource and the rule as noncompliant. When the compliance status of a resource changes, AWS Config sends a notification to your Amazon SNS topic.

Deliver Configuration Items

AWS Config can deliver configuration items through one of the following channels:

Amazon S3 Bucket

AWS Config tracks changes in the configuration of your AWS resources, and it regularly sends updated configuration details to an S3 bucket that you specify. For each resource type that AWS Config records, it sends a *configuration history file* every six hours. Each configuration history file contains details about the resources that changed in that six-hour period. Each file includes resources of one type, such as Amazon EC2 instances or Amazon EBS volumes. If no configuration changes occur, AWS Config does not send a file.

AWS Config sends a *configuration snapshot* to your S3 bucket when you use the deliver-config-snapshot command with the AWS CLI, or when you use the DeliverConfigSnapshot action with the AWS Config API. A configuration snapshot contains configuration details for all resources that AWS Config records in your AWS account. Both the configuration history file and configuration snapshot are in JSON format.

Amazon SNS Topic

An Amazon Simple Notification Service (Amazon SNS) topic is a communication channel that Amazon SNS uses to deliver messages (or *notifications*) to subscribing endpoints such as an email address or clients such as an Amazon Simple Queue Service queue. Other types of Amazon SNS notifications include push notification messages to apps on mobile phones, Short Message Service (SMS) notifications to SMS-enabled mobile phones and smart phones, and HTTP POST requests. For best results, use Amazon SQS as the notification endpoint for the SNS topic and then process the information in the notification programmatically.

AWS Config uses the Amazon SNS topic that you specify to send you notifications. The type of notification that you are receiving is indicated by the value for the `messageType` key in the message body, as in the following example:

```
"messageType": "ConfigurationHistoryDeliveryCompleted"
```

The notifications can be any of the following message types:

`ComplianceChangeNotification`
: The compliance type of a resource that AWS Config evaluates has changed. The compliance type indicates whether the resource complies with a specific AWS Config rule, and it is represented by the `ComplianceType` key in the message. The message includes `newEvaluationResult` and `oldEvaluationResult` objects for comparison.

`ConfigurationSnapshotDeliveryStarted`
: AWS Config started delivering the configuration snapshot to your S3 bucket. The name of the S3 bucket is provided for the `s3Bucket` key in the message.

`ConfigurationSnapshotDeliveryCompleted`
: AWS Config successfully delivered the configuration snapshot to your S3 bucket.

`ConfigurationSnapshotDeliveryFailed`
: AWS Config failed to deliver the configuration snapshot to your S3 bucket.

`ConfigurationHistoryDeliveryCompleted`
: AWS Config successfully delivered the configuration history to your S3 bucket.

`ConfigurationItemChangeNotification`
: A resource has been created, deleted, or changed in configuration. This message includes the details of the configuration item that AWS Config creates for this change, and it includes the type of change. These notification are delivered within minutes of a change and are collectively known as the configuration stream.

For more information about Amazon SNS, see What is Amazon SNS?

Supported Resources, Configuration Items, and Relationships

AWS Config supports the following AWS resources, configuration items, and resource relationships.

Topics

Supported AWS Resource Types

AWS Config supports the following AWS resources.

AWS Service	Resource Type	`resourceType` Value
Amazon Elastic Block Store	Amazon EBS volume	`AWS::EC2::Volume`
Amazon Elastic Compute Cloud	EC2 Dedicated host[1]	`AWS::EC2::Host`
	EC2 Elastic IP (VPC only)	`AWS::EC2::EIP`
	EC2 instance	`AWS::EC2::Instance`
	EC2 network interface	`AWS::EC2::NetworkInterface`
	EC2 security group	`AWS::EC2::SecurityGroup`
Amazon Virtual Private Cloud	Customer gateway	`AWS::EC2::CustomerGateway`
	Internet gateway	`AWS::EC2::InternetGateway`
	Network access control list (ACL)	`AWS::EC2::NetworkAcl`
	Route table	`AWS::EC2::RouteTable`
	Subnet	`AWS::EC2::Subnet`
	Virtual private cloud (VPC)	`AWS::EC2::VPC`
	VPN connection	`AWS::EC2::VPNConnection`
	VPN gateway	`AWS::EC2::VPNGateway`
AWS CloudTrail	Trail	`AWS::CloudTrail::Trail`

AWS Service	Resource Type	`resourceType` Value
AWS Identity and Access Management[2]	IAM user[3]	`AWS::IAM::User`
	IAM group[3]	`AWS::IAM::Group`
	IAM role[3]	`AWS::IAM::Role`
	IAM customer managed policy	`AWS::IAM::Policy`

Notes

1. AWS Config records the configuration details of Dedicated hosts and the instances that you launch on them. As a result, you can use AWS Config as a data source when you report compliance with your server-bound software licenses. For example, you can view the configuration history of an instance and determine which Amazon Machine Image (AMI) it is based on. Then, you can look up the configuration history of the host, which includes details such as the numbers of sockets and cores, to verify that the host complies with the license requirements of the AMI. For more information, see Tracking Configuration Changes withAWS Config in the *Amazon EC2 User Guide for Linux Instances*.
2. AWS Identity and Access Management (IAM) resources are *global resources*. Global resources are not tied to an individual region and can be used in all regions. The configuration details for a global resource are the same in all regions. For more information, see Selecting Which Resources AWS Config Records (p. 76).
3. AWS Config includes inline policies with the configuration details that it records.

Components of a Configuration Item

A configuration item consists of the following components.

Component	Description	Contains
Metadata	Information about this configuration item	• Version ID • Configuration item ID • Time when the configuration item was captured • Status of the configuration item indicating whether the item was captured successfully • State ID indicating the ordering of the configuration items of a resource • A unique MD5Hash representing the state of a configuration item that can be used to compare two states of two or more configuration items of the same resource

Component	Description	Contains
Attributes	Resource attributes	• Resource ID • List of key–value tags for this resource • Resource type; see Supported AWS Resource Types (p. 6) • Amazon Resource Name (ARN) • Availability Zone that contains this resource, if applicable • Time the resource was created
Relationships	How the resource is related to other resources associated with the account	Description of the relationship, such as Amazon EBS volume `vol-1234567` is attached to an Amazon EC2 instance `i-a1b2c3d4`
Current configuration	Information returned through a call to the Describe or List API of the resource	For example, `DescribeVolumes` API returns the following information about the volume: • Availability Zone the volume is in • Time the volume was attached • ID of the EC2 instance it is attached to • Current status of the volume • State of DeleteOnTermination flag • Device the volume is attached to • Type of volume, such as `gp2`, `io1`, or `standard`
Related events	The AWS CloudTrail events that is related to the current configuration of the resource	CloudTrail event ID

A configuration item does not contain information about the contents of your resource. For example, the configuration item for an Amazon EC2 instance does not contain information about the operating system, user software, and various other parameters. A configuration item relationship does not include network flow or data flow dependencies. Configuration items cannot be customized to represent your application architecture.

Supported Resource Relationships

AWS Config supports the following relationships between different resources.

Resource	Relationship	Related Resource
Amazon EBS volume	is attached to	EC2 instance

Resource	Relationship	Related Resource
Customer gateway	is attached to	VPN connection
EC2 Dedicated host	contains	EC2 instance
EC2 Elastic IP (EIP)	is attached to	EC2 instance
		Network interface
EC2 instance	contains	EC2 network interface
	is associated with	EC2 security group
	is attached to	Amazon EBS volume
		EC2 Elastic IP (EIP)
	is contained in	EC2 Dedicated host
		Route table
		Subnet
		Virtual private cloud (VPC)
EC2 network interface	is associated with	EC2 security group
	is attached to	EC2 Elastic IP (EIP)
		EC2 instance
	is contained in	Route table
		Subnet
		Virtual private cloud (VPC)
EC2 security group	is associated with	EC2 instance
		EC2 network interface
		VPC

Resource	Relationship	Related Resource
IAM user	is attached to	IAM group
	is attached to	IAM customer managed policy
IAM group	contains	IAM user
	is attached to	IAM customer managed policy
IAM role	is attached to	IAM customer managed policy
IAM customer managed policy	is attached to	IAM user
	is attached to	IAM group
	is attached to	IAM role
Internet gateway	is attached to	Virtual private cloud (VPC)
Network ACL	is attached to	Subnet
	is contained in	Virtual private cloud (VPC)
Route table	contains	EC2 instance
		EC2 network interface
		Subnet
		VPN gateway
	is contained in	Virtual private cloud (VPC)

Resource	Relationship	Related Resource
Subnet	contains	EC2 instance
	contains	EC2 network interface
	is attached to	Network ACL
	is contained in	Route table
		Virtual private cloud (VPC)
Virtual private cloud (VPC)	contains	EC2 instance
		EC2 network interface
		Network ACL
		Route table
		Subnet
	is associated with	Security group
	is attached to	Internet gateway
		VPN gateway
VPN connection	is attached to	Customer gateway
		VPN gateway
VPN gateway	is attached to	Virtual private cloud (VPC)
		VPN connection
	is contained in	Route table

Setting Up

Before you can use AWS Config, you'll need to sign up for an Amazon Web Services (AWS) account. After signing up, you'll need to choose whether you want to access AWS Config through the console, the AWS CLI, or the AWS SDKs. The Getting Started With AWS Config (p. 15) section of this guide walks you through accessing AWS Config from the console or the AWS CLI.

Topics

Sign Up for Amazon Web Services (AWS)

When you sign up for AWS, your account automatically has access to all AWS services. You pay only for the services that you use. If you already have an AWS account, skip the following sign-in procedure.

If you do not have an AWS account, use the following procedure to create one.

To sign up for AWS

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

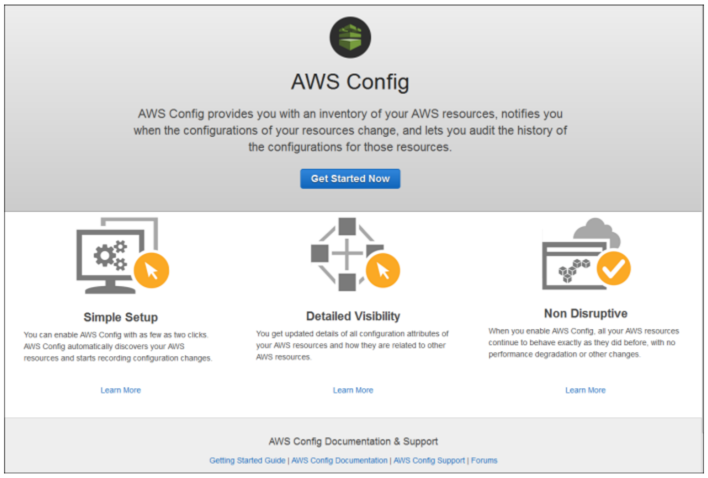
Open the AWS Config Console

The AWS Management Console is a point-and-click web-based interface from which you can access the AWS Config console and other AWS services. You can use the console to make API requests to AWS Config and other AWS APIs.

To open the AWS Config console

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.

2. If this is the first time you have opened the AWS Config console or you have not yet started using the service, your AWS Config console page might look something like this:



AWS Config

AWS Config provides you with an inventory of your AWS resources, notifies you when the configurations of your resources change, and lets you audit the history of the configurations for those resources.

Simple Setup

You can enable AWS Config with as few as two clicks. AWS Config automatically discovers your AWS resources and starts recording configuration changes.

Detailed Visibility

You get updated details of all configuration attributes of your AWS resources and how they are related to other AWS resources.

Non Disruptive

When you enable AWS Config, all your AWS resources continue to behave exactly as they did before, with no performance degradation or other changes.

AWS Config Documentation & Support

Getting Started Guide | AWS Config Documentation | AWS Config Support | Forums

Use the AWS Command Line Interface

Amazon Web Services (AWS) provides a command line interface (CLI) that supports the broader set of AWS services, including AWS Config. You can use the AWS CLI to control and automate the services from AWS.

For more information about the AWS CLI and for instructions on installing the AWS CLI tools, step through the following sections in the *AWS Command Line Interface User Guide*.

- What Is the AWS Command Line Interface? introduces the AWS CLI.
- Getting Set Up with the AWS Command Line Interface explains how to install and configure the AWS CLI.

AWS Software Development Kits

An AWS software development kit (SDK) can make it easier for developers to build applications that tap into cost-effective, scalable, and reliable AWS infrastructure services. With AWS SDKs, you can get started in minutes with a single, downloadable package that includes the library, code samples, and reference documentation. You can access AWS Config programmatically using the SDKs in Java, .NET, Python, or Ruby. The following table lists the available SDKs and third-party libraries you can use to access AWS Config programmatically.

Type of Access	Description
AWS SDKs	AWS provides the following SDKs: • AWS SDK for Java Documentation • AWS SDK for .NET Documentation • AWS SDK for Python (boto) Documentation • AWS SDK for Ruby Documentation

Type of Access	Description
Third-party libraries	Developers in the AWS developer community also provide their own libraries, which you can find at the following AWS developer centers: • AWS Java Developer Center • AWS Python Developer Center • AWS Ruby Developer Center • AWS Windows and .NET Developer Center

Getting Started With AWS Config

You can get started with AWS Config by using either the AWS Management Console or the AWS CLI. Use the console for a quick, streamlined process and use the CLI if you are comfortable controlling AWS offerings programmatically.

You will perform the following tasks whether you use the console or the CLI:

- Set up an Amazon S3 bucket to receive a configuration snapshot on request and configuration history.
- Set up an Amazon SNS topic to send configuration stream notifications.
- Grant AWS Config the permissions it needs to access the Amazon S3 bucket and the SNS topic

In AWS Config, both the Amazon S3 bucket and the Amazon SNS topic are referred to as *delivery channels*.

Topics

Set Up AWS Config Using the Console

This topic describes how to use the AWS Management Console to get started with AWS Config. In just a few steps, you can set up Amazon SNS to notify you of configuration changes and designate an Amazon S3 bucket to receive snapshots and configuration history information.

> **Note**
> If you are using AWS Config in a region that supports AWS Config rules, follow the process in Setting Up AWS Config Rules (p. 25). For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.

To set up AWS Config

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. On the **Resource inventory** page, choose the gear icon (⚙) to see the **Set Up AWS Config** page.
3. In the **Resource types to record** section, specify which types of AWS resources you want AWS Config to record:

- **All resources** – AWS Config records all supported resources with the following options:
  - **Record all resources supported in this region** – AWS Config records configuration changes for every supported type of regional resource. When AWS Config adds support for a new type of regional resource, it automatically starts recording resources of that type.
  - **Include global resources** – AWS Config includes supported types of global resources with the resources that it records. When AWS Config adds support for a new type of global resource, it automatically starts recording resources of that type.
- **Specific types** – AWS Config records configuration changes for only those types of AWS resources that you specify.

For more information about these options, see Selecting Which Resources AWS Config Records (p. 76).

4. Under **Amazon S3 Bucket**, choose the Amazon S3 bucket to which AWS Config sends configuration history and configuration snapshot files:

   - **Create a new bucket** – For **Bucket Name**, type a name for your Amazon S3 bucket.

     The name that you type must be unique across all existing bucket names in Amazon S3. One way to help ensure uniqueness is to include a prefix; for example, the name of your organization. You cannot change the name of a bucket after it is created. For more information, see Bucket Restrictions and Limitations in the *Amazon Simple Storage Service Developer Guide*.
   - **Choose a bucket from my account** – For **Bucket Name**, select your preferred bucket.
   - **Choose a bucket from another account** – For **Bucket Name**, type the name of your bucket.

     If you choose a bucket from another account, that bucket must have policies that grant access permissions to AWS Config. For more information, see Permissions for the Amazon S3 Bucket (p. 82).

5. Under **Amazon SNS Topic**, choose whether AWS Config will stream information by selecting the **Stream configuration changes and notifications to an Amazon SNS topic** option.
6. If you chose to have AWS Config stream to an Amazon SNS topic, choose the target topic:

   - **Create a new topic** – For **Topic Name**, type a name for your SNS topic.
   - **Choose a topic from your account** – For **Topic Name**, select your preferred topic.
   - **Choose a topic from another account** – For **Topic ARN**, type the Amazon Resource Name (ARN) of the topic.

     If you choose a topic from another account, the topic must have policies that grant access permissions to AWS Config. For more information, see Permissions for the Amazon SNS Topic (p. 84).

     Note
     The Amazon SNS topic must exist in the same region as the region in which you set up AWS Config.
7. Choose **Continue**.
8. In the **AWS Config is requesting permissions to read your resources' configuration** page, choose **Allow**.

   (Optional) Choose **View Details** to view the IAM role that AWS Config created for you. In the **Role Summary** pane, choose **View Policy Document** to view the permissions granted to AWS Config. For more information about role creation, see Creating a Role to Delegate Permissions to an AWS Service.

For information about looking up the existing resources in your account and understanding the configurations of your resources, see Viewing AWS Resource Configurations and History (p. 50).

If you chose to have AWS Config stream information to an Amazon SNS topic, you can receive notifications by email. For more information, see Monitoring AWS Config Resource Changes by Email (p. 56). You can also use Amazon Simple Queue Service to monitor AWS resources programmatically. For more information, see Using Amazon SQS to Monitor AWS Resource Changes (p. 70).

Set Up AWS Config Using the AWS CLI

You can use the `subscribe` command to have AWS Config start recording configurations of all supported AWS resources in your account. The `subscribe` command creates a configuration recorder, a delivery channel using a specified Amazon S3 bucket and Amazon SNS topic, and starts recording the configuration items. You are limited to only one configuration recorder and one delivery channel per account in the region with your account's resources.

Before AWS Config can start delivering configuration items to your delivery channels, you must give AWS Config permissions to access those channels. To do that, you use an AWS Identity and Access Management (IAM) role to define a set of permissions. AWS Config assumes the role that you assign to it to make read or write requests to the delivery channel. For more information on IAM roles, see Roles (Delegation and Federation) in the *AWS Identity and Access Management User Guide*.

The `subscribe` command uses the following options:

`--s3-bucket`
: Specify the name of an Amazon S3 bucket existing in your account or existing in another account.

`--sns-topic`
: Specify the Amazon Resource Name (ARN) of an SNS topic existing in your account or existing in another account.

`--iam-role`
: Specify the Amazon Resource Name (ARN) of an existing IAM Role.

  The specified IAM role must have policies attached that grant AWS Config permissions to deliver configuration items to the Amazon S3 bucket and the Amazon SNS topic, and the role must grant permissions to the `Describe` APIs of the supported AWS resources.

Topics

Prerequisites for Setting Up AWS Config

This section walks you through the process for creating an Amazon S3 bucket, an Amazon SNS topic, and an IAM role with attached policies.

Topics

Create an Amazon S3 Bucket

If you already have an Amazon S3 bucket in your account and want to use it, skip this step and go to Create an Amazon SNS Topic (p. 18).

You can also use an Amazon S3 bucket from a different account, but in that case you might need to create a policy for the bucket that grants access permissions to AWS Config. For information on granting permissions to an Amazon S3 bucket, see Permissions for the Amazon S3 Bucket (p. 82), and then go to Create an Amazon SNS Topic (p. 18).

Follow the steps to create an Amazon S3 bucket.

To create an Amazon S3 bucket

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose **Actions** and then choose **Create Bucket**.
3. For the **Bucket Name:**, type a name for your Amazon S3 bucket, such as `my-config-bucket`.

   Note
   Make sure the bucket name you choose is unique across all existing bucket names in Amazon S3. You cannot change the name of a bucket after it is created. For more information on bucket naming rules and conventions, see Bucket restrictions and Limitations in the *Amazon Simple Storage Service Developer Guide*.
4. Choose **Create**.

Create an Amazon SNS Topic

If you already have an Amazon SNS topic in your account and want to use it, skip this step and go to Create an IAM Role (p. 19).

You can also use an Amazon SNS topic in a different account, but in that case you might need to create a policy for topic that grants access permissions to AWS Config. For information on granting permissions to an Amazon SNS topic, see Permissions for the Amazon SNS Topic (p. 84) and then go to Create an IAM Role (p. 19).

Follow the steps to create an Amazon SNS topic.

To create an Amazon SNS topic

1. Sign in to the AWS Management Console and open the Amazon SNS console at https://console.aws.amazon.com/sns/.
2. Choose **Create New Topic**.
3. For **Topic Name**, type a name for your SNS topic, such as `my-config-notice`.
4. Choose **Create Topic**.

   The new topic appears in the **Topic Details** page. Copy the **Topic ARN** (Amazon Resource Name) for the next task.

To receive notifications from AWS Config, you must subscribe an email address to the topic.

To subscribe an email address to the SNS topic

1. In the Amazon SNS console, choose **Subscriptions** in the navigation pane.

2. On the **Subscriptions** page, choose **Create Subscription**.
3. For **Topic ARN**, paste the topic ARN you copied in the previous task.
4. For **Protocol**, select **Email**.
5. For **Endpoint**, type an email address that you can use to receive the notification. Then choose **Subscribe**.
6. Go to your email application and open the message from **AWS Notifications**. Choose the link to confirm your subscription.

   Your web browser displays a confirmation response from Amazon SNS. Amazon SNS is now configured to receive notifications and send the notification as an email to the specified email address.

Create an IAM Role

You can use the IAM console to create an IAM role that grants AWS Config permissions to access your Amazon S3 bucket, access your Amazon SNS topic, and get configuration details for supported AWS resources. After you create the IAM role, you will create and attach policies to the role.

To create an IAM role

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the IAM console, choose **Roles** in the navigation pane, and choose **Create New Role**.
3. For **Role Name**, type a name that describes the purpose of this role. Role names must be unique within your AWS account. Because various entities might reference the role, you cannot edit the name of the role after you create it.

   Choose **Next Step**.
4. Choose **AWS Service Roles**, and then choose **Select** for **AWS Config** .
5. On the **Attach Policy** page, select **AWSConfigRole**. This AWS managed policy grants AWS Config permission to get configuration details for supported AWS resources. Then, choose **Next Step**.
6. On the **Review** page, review the details about your role, and choose **Create Role**.
7. On the **Roles** page, choose the role that you created to open its details page.

You will expand the permissions in the role by creating inline policies that allow AWS Config to access your Amazon S3 bucket and your Amazon SNS topic.

To create an inline policy that grants AWS Config permission to access your Amazon S3 bucket

1. In the **Permissions** section, expand the **Inline Polices** section, and choose **click here**.
2. Choose **Custom Policy**, and choose **Select**.
3. For **Policy Name**, type a name for your inline policy.
4. Copy the example Amazon S3 bucket policy in Creating IAM Role Policies (p. 81) and paste it in the **Policy Document** editor.

   Important
   Before you proceed to the next step, replace the following values in the policy. If you do not replace the values, your policy will fail.

   - *`myBucketName`* – Replace with the name of your Amazon S3 bucket.
   - *`prefix`* – Replace with your own prefix or leave blank by removing the trailing '/'.
   - *`myAccountID-WithoutHyphens`* – Replace with your AWS account ID.

5. Choose **Apply Policy**.

To create an inline policy that grants AWS Config permissions to deliver notifications to your Amazon SNS topic

1. In the **Permissions** section, expand the **Inline Polices** section, and choose **click here**.
2. Choose **Custom Policy**, and choose **Select**.
3. For **Policy Name**, type a name for your inline policy.
4. Copy the Amazon SNS topic example policy in Creating IAM Role Policies (p. 81) and paste it in the **Policy Document** editor.

   Important
   Before you proceed to the next step, replace *`arn:aws:sns:region:account-id:myTopic`* with the ARN you saved when you created your Amazon SNS topic.
5. Choose **Apply Policy**.

Turn On AWS Config

You can use the AWS CLI to turn on AWS Config. All it takes is the `subscribe` command and a few additional parameters.

To turn on AWS Config from the command line

- At the command line, type `subscribe` with the following parameters:
  - `--s3-bucket` *`yourS3bucketname`*.
  - `--sns-topic` *`yourSNStopicARN`*.
  - `--iam-role` *`yourIAMroleARN`*.

Your command should look like the following example:

```
$ aws configservice subscribe --s3-bucket my-config-bucket --sns-topic
arn:aws:sns:us-east-1:012345678912:my-config-notice --iam-role
arn:aws:iam::012345678912:role/myConfigRole
```

After you run the `subscribe` command, AWS Config records all supported resources that it discovers in the region. If you don't want AWS Config to record all supported resources, you can specify the types of resources that it records by updating the configuration recorder to use a recording group. To learn how to create a recording group, see Selecting Which Resources AWS Config Records (p. 76).

Verify that AWS Config Is On

Once you have turned on AWS Config, you can use AWS CLI commands to verify that the AWS Config is running and that the `subscribe` command has created a configuration recorder and a delivery channel. You can also confirm that AWS Config has started recording and delivering configurations to the delivery channel.

Topics

- Verify that AWS Config has started recording (p. 21)

Verify that the Delivery Channel Is Created

Use the `describe-delivery-channels` command to verify that your Amazon S3 bucket and Amazon SNS topic is configured.

```
$ aws configservice describe-delivery-channels
{
    "DeliveryChannels": [
        {
            "snsTopicARN": "arn:aws:sns:us-west-2:0123456789012:my-config-no
tice",
            "name": "default",
            "s3BucketName": "my-config-bucket"
        }
    ]
}
```

When you use the CLI, the service API, or the SDKs to configure your delivery channel and do not specify a name, AWS Config automatically assigns the name "`default`".

Verify that the Configuration Recorder Is Created

Use the `describe-configuration-recorders` command to verify that a configuration recorder is created and that the configuration recorder has assumed an IAM role. For more information, see Create an IAM Role (p. 19).

```
$ aws configservice describe-configuration-recorders
{
    "ConfigurationRecorders": [
        {
            "roleARN": "arn:aws:iam::012345678912:role/myConfigRole",
            "name": "default"
        }
    ]
}
```

Any configuration recorder you create using the CLI, the service API, or the SDKs has the name "`default`". You cannot change this name.

Verify that AWS Config has started recording

Use the `describe-configuration-recorder-status` command to verify that the AWS Config has started recording the configurations of the supported AWS resources existing in your account. The recorded configurations are delivered to the specified delivery channel.

```
$ aws configservice describe-configuration-recorder-status
{
    "ConfigurationRecordersStatus": [
        {
            "name": "default",
            "lastStatus": "SUCCESS",
            "lastStopTime": 1414511624.914,
            "lastStartTime": 1414708460.276,
            "recording": true,
            "lastStatusChangeTime": 1414816537.148,
            "lastErrorMessage": "NA",
            "lastErrorCode": "400"
        }
    ]
}
```

The value `true` in the `recording` field confirms that the configuration recorder has started recording configurations of all your resources. AWS Config uses UTC format (GMT - 8:00) to record the time.

For information about looking up the resources existing in your account and understanding the configurations of your resources, see Viewing AWS Resource Configurations and History (p. 50).

Evaluating Resources With AWS Config Rules

You can use AWS Config to evaluate the configuration settings of your AWS resources. You do this by creating AWS Config rules, which represent your ideal configuration settings. AWS Config provides customizable, predefined rules to help you get started. You can also create your own custom rules from scratch. While AWS Config continuously tracks the configuration changes that occur among your resources, it checks whether these changes violate any of the conditions in your rules. If a resource does violate a rule, AWS Config flags the resource and the rule as noncompliant.

The AWS Config console shows you the compliance status of your rules and resources. You can use the console to assess how your AWS resources comply overall with your desired configurations, and you can learn which specific resources are noncompliant and which configuration attributes are the cause. You can also use the AWS CLI, the AWS Config API, and AWS SDKs to make requests to the AWS Config service for compliance information.

By using AWS Config to evaluate your resource configurations, you can more easily assess how well your resource configurations comply with internal practices, industry guidelines, and regulations.

To see which regions support AWS Config rules, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.

Topics

AWS Config Rules

An AWS Config rule represents your desired configuration settings for certain AWS resources or for an entire AWS account. While AWS Config captures ongoing configuration changes as configuration items, it evaluates whether each configuration change complies with your rules.

Depending on the rule, the evaluations are triggered by configuration changes or periodically.

Evaluations Triggered by Configuration Changes

Evaluations that are triggered by configuration changes run when AWS Config detects that an individual resource was created, changed, or deleted. If the resource violates a rule, AWS Config flags the resource and the violated rule as noncompliant. For example, an evaluation can detect that an Elastic IP address is not associated with an EC2 instance, or that an EC2 volume is not encrypted. AWS Config can run these evaluations on any resource in your recording group.

Scope

For evaluations that are triggered by resource configuration changes, you can define which resources trigger the evaluation by defining the rule's *scope*. The scope can include one or more resource types, a combination of a resource type and a resource ID, or a combination of a tag key and value. When AWS Config detects a configuration change for any resource that matches the rule's scope, the evaluation is triggered. Specify a scope to constrain which resources trigger an evaluation for a rule. Otherwise, evaluations for the rule are triggered when any recorded resource changes.

Periodic Evaluations

Periodic evaluations run when AWS Config delivers a configuration snapshot to your Amazon S3 bucket, which happens at a frequency of your choice. AWS Config evaluates the content of the snapshot against your periodic rules. Unlike evaluations that are triggered by configuration changes, periodic evaluations have no scope and are run on every configuration item in the snapshot. The snapshot includes configuration items for every resource in your recording group. If any condition in the snapshot violates a rule for periodic evaluations, AWS Config flags the account and the rule as noncompliant. Periodic evaluations do not flag individual resources as compliant or noncompliant.

Because they are global, it's a good idea to use periodic evaluations for conditions that you want to be true for an entire AWS account, as opposed to an individual resource. For example, a periodic evaluation can check whether the number of EC2 volumes in an account stays within a desired total, or whether an account uses AWS CloudTrail for logging.

AWS Managed Rules

AWS managed rules are customizable, predefined rules, which AWS Config provides to help you start ongoing evaluations for common needs. You can use the AWS Config console to select one of these rules, customize it for your needs, and activate it.

For a list of managed rules that AWS Config provides, and for instructions to use them, see Using AWS Managed Config Rules (p. 27).

Customer Managed Rules

Customer managed rules are custom rules that you develop and add to AWS Config. Before you can add a customer managed rule, you must first create an AWS Lambda function that contains the evaluation logic for your rule. You associate this function with your rule, and your rule invokes the function either in response to configuration changes or periodically. The function then evaluates whether your resources comply with your rule, and it sends its evaluation results to AWS Config.

To make it easier to create a Lambda function for a rule, the AWS Lambda console provides blueprints that you can customize, and the AWS Config Rules GitHub repository provides sample functions that are developed and contributed by AWS Config users.

For more information, see Developing Custom Rules for AWS Config (p. 30).

Setting Up AWS Config Rules

Before you can use AWS Config to create rules and evaluate your AWS resource configurations, you must set up AWS Config with required settings and permissions. The first time you use the AWS Config console in a region that supports rules, AWS Config guides you through the setup process.

Important
As part of the set-up process, you will update or create an IAM role that is assigned to AWS Config. By updating or creating the IAM role, you attach a policy that includes the `config:Put*` IAM permission, which allows AWS Config to provide results after it evaluates your resources against your rules.

To set up AWS Config rules

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. On the **Settings** page, complete the following steps:

   a. In the **Resource types to record** section, specify which types of AWS resources you want AWS Config to record:

      - **All resources** – AWS Config records all supported resources with the following options:
        - **Record all resources supported in this region** – AWS Config records configuration changes for every supported type of regional resource. When AWS Config adds support for a new type of regional resource, it automatically starts recording resources of that type.
        - **Include global resources** – AWS Config includes supported types of global resources with the resources that it records. When AWS Config adds support for a new type of global resource, it automatically starts recording resources of that type.
      - **Specific types** – AWS Config records configuration changes for only those types of AWS resources that you specify.

      For more information about these options, see Selecting Which Resources AWS Config Records (p. 76).

   b. Under **Amazon S3 Bucket**, choose the Amazon S3 bucket to which AWS Config sends configuration history and configuration snapshot files:

      - **Create a new bucket** – For **Bucket Name**, type a name for your Amazon S3 bucket.

The name that you type must be unique across all existing bucket names in Amazon S3. One way to help ensure uniqueness is to include a prefix; for example, the name of your organization. You cannot change the name of a bucket after it is created. For more information, see Bucket Restrictions and Limitations in the *Amazon Simple Storage Service Developer Guide*.

- **Choose a bucket from your account** – For **Bucket Name**, select your preferred bucket.
- **Choose a bucket from another account** – For **Bucket Name**, type the name of your bucket.

  If you choose a bucket from another account, that bucket must have policies that grant access permissions to AWS Config. For more information, see Permissions for the Amazon S3 Bucket (p. 82).

c. Under **Amazon SNS Topic**, choose whether AWS Config will stream information by selecting the **Stream configuration changes and notifications to an Amazon SNS topic** option.

d. If you chose to have AWS Config stream to an Amazon SNS topic, choose the target topic:

- **Create a new topic** – For **Topic Name**, type a name for your SNS topic.
- **Choose a topic from your account** – For **Topic Name**, select your preferred topic.
- **Choose a topic from another account** – For **Topic ARN**, type the Amazon Resource Name (ARN) of the topic.

  If you choose a topic from another account, the topic must have policies that grant access permissions to AWS Config. For more information, see Permissions for the Amazon SNS Topic (p. 84).

Note
The Amazon SNS topic must exist in the same region as the region in which you set up AWS Config.

e. Under **AWS Config role**, choose the IAM role that grants AWS Config permission to record configuration information and send this information to Amazon S3 and Amazon SNS:

- **Create a role** – AWS Config creates a role that has the required permissions. For **Role name**, you can customize the name that AWS Config creates.
- **Choose a role from your account** – For **Role name**, select an IAM role in your account that grants the required permissions. For the required permissions, see Permissions for the AWS Config IAM Role (p. 80).

f. Choose **Next**.

4. On the **AWS Config rules** page, select any of the example rules and choose **Next**. AWS Config evaluates your AWS resources against the rules when setup is complete. Remember that AWS Config can evaluate only those resources that it is recording. You can specify which resources AWS Config records in the next step.
5. On the **Review** page, verify your setup details, and choose **Confirm**.

AWS Config shows the **Rules** page, which shows your rules and their current compliance results in the table. The result for each rule will be **Evaluating...** until AWS Config finishes evaluating your resources against the rule. You can update the results with the refresh button.

Using AWS Managed Config Rules

AWS Config provides *AWS managed rules*, which are predefined, customizable rules that AWS Config uses to evaluate whether your AWS resources comply with common best practices. For example, you could use a managed rule to quickly start assessing whether the your Amazon Elastic Block Store (Amazon EBS) volumes are encrypted or whether specific tags are applied to your resources. You can set up and activate these rules without writing the code to create an AWS Lambda function, which is otherwise required if you want to create your own custom rules. The AWS Config console guides you through the process of configuring and activating a managed rule. You can also use the AWS Command Line Interface or AWS Config API to pass the JSON code that defines your configuration of a managed rule.

You can customize the behavior of a managed rule to suit your needs. For example, you can define the rule's scope to constrain which resources trigger an evaluation for the rule, such as EC2 instances or volumes. You can customize the rule's parameters to define attributes that your resources must have to comply with the rule. For example, you can customize a parameter to specify that your security group should block incoming traffic to a specific port number.

After you activate a rule, AWS Config compares your resources to the conditions of the rule. After this initial evaluation, AWS Config continues to run evaluations each time one is triggered. The evaluation trigger is defined as part of the rule, and it can be one of the following types:

- **Configuration changes** – AWS Config triggers the evaluation when any resource that matches the rule's scope changes in configuration. The evaluation runs after AWS Config sends a configuration item change notification.
- **Periodic** – The evaluation is triggered each time AWS Config sends a configuration snapshot, which it does at an interval of your choice. AWS Config evaluates the contents of the snapshot according to the rule.

The AWS Config console shows which resources comply with the rule and which rules are being followed. For more information, see Viewing Configuration Compliance with AWS Config (p. 43).

AWS Managed Rules

AWS Config provides the following AWS managed rules.

Identifier	Description	Trigger Type	Parameters
`CLOUD_TRAIL_EN-ABLED`	Checks whether AWS CloudTrail is enabled in your AWS account. Optionally, you can specify which S3 bucket, SNS topic, and Amazon CloudWatch Logs ARN to use.	Periodic	• *`s3BucketName`* – The name of the S3 bucket for AWS CloudTrail to deliver log files to. • *`snsTopicArn`* – The ARN of the SNS topic for AWS CloudTrail to use for notifications. • *`cloudWatchLogs-LogGroupArn`* – The ARN of the Amazon CloudWatch log group for AWS CloudTrail to send data to.

Identifier	Description	Trigger Type	Parameters
`EIP_ATTACHED`	Checks whether Elastic IP addresses that are allocated for use in a VPC are attached to EC2 instances. Results might take up to 6 hours to become available after an evaluation occurs.	Configuration changes	
`ENCRYPTED_VOLUMES`	Checks whether EBS volumes that are in an attached state are encrypted. Optionally, you can specify the ID of a KMS key to use to encrypt the volume.	Configuration changes	• *`kmsId`* – The ID of the KMS key to use to encrypt the volume.
`INCOMING_SSH_DISABLED`	Checks whether security groups that are in use disallow unrestricted incoming SSH traffic.	Configuration changes	
`INSTANCES_IN_VPC`	Checks whether your EC2 instances belong to a virtual private cloud (VPC). Optionally, you can specify the VPC ID to associate with your instances.	Configuration changes	• *`vpcId`* –The ID of the VPC to contain these instances.
`REQUIRED_TAGS`	Checks whether your resources have all of the tags you specify; for example, you can check whether the `CostCenter` tag is present on your EC2 instances. Results might take up to 6 hours to become available after an evaluation occurs.	Configuration changes	• *`tag1key`* – Key of the required tag. • *`tag1value`* – Optional value of the required tag.
`RESTRICTED_INCOMING_TRAFFIC`	Checks whether security groups that are in use disallow unrestricted incoming TCP traffic to the specified ports.	Configuration changes	• *`blockedPort1`* – Blocked TCP port number. • *`blockedPort2`* – Blocked TCP port number.

Using AWS Managed Rules

You can set up and activate AWS managed rules from the AWS Management Console, AWS CLI, or AWS Config API.

To set up and activate an AWS managed rule (Console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. In the left navigation, choose **Rules**.
4. On the **Rules** page, choose **Add rule**.
5. On the **Add rule** page, choose the rule that you want to use.
6. On the **Configure rule** page, configure the rule by completing the following steps:

   a. For **Name**, type a unique name for the rule.

   b. If the **Trigger type** for your rule is **Configuration changes**, specify when AWS Config triggers an evaluation by choosing one of the following options for **Scope of changes**:

      - **Resources** – When any resource that matches the specified resource type, or the type plus identifier, is created, changed, or deleted.
      - **Tags** – When any resource with the specified tag is created, changed, or deleted.
      - **All changes** – When any resource recorded by AWS Config is created, changed, or deleted.

   c. If the **Trigger type** for your rule is **Periodic**, choose how often your rule evaluates your resources for **Frequency**. AWS Config delivers a configuration snapshot at the frequency that you choose. The snapshot is delivered to the Amazon S3 bucket that you specified when you set up AWS Config. After AWS Config delivers your snapshot, it evaluates the contents of the snapshot according to your rule.

   d. If your rule includes parameters in the **Rule parameters** section, you can customize the values for the provided keys. A parameter is an attribute that your resources must have before they are considered compliant with the rule.

7. Choose **Save**. Your new rule displays on the **Rules** page.

   Compliance will display **Evaluating...** until AWS Config has evaluation results for your rule. A summary of the results appears after several minutes. You can update the results with the refresh button.

   If the rule or function is not working as expected, you might see one of the following for **Compliance**:

   - **No results reported** - The rule was executed but did not apply to any of the AWS resources within its scope, or all such resources were deleted. Verify that the scope includes **Instance** for **Resources**, and try again.
   - **No resources in scope** - AWS Config cannot evaluate your recorded AWS resources against this rule because none of your resources are within the rule’s scope. You can choose which resources AWS Config records on the **Settings** page.
   - **Evaluations failed** - For information that can help you determine the problem, choose the rule name to open its details page and see the error message.

To set up and activate an AWS managed rule (AWS CLI)

- Use the `put-config-rule` command.

To set up and activate an AWS managed rule (AWS Config API)

- Use the `PutConfigRule` action.

Developing Custom Rules for AWS Config

You can create your own customer managed rules, which are custom AWS Config rules that you develop from scratch. You associate each custom rule with an AWS Lambda function, which contains the logic that evaluates whether your AWS resources comply with the rule.

The exercise in Getting Started with Custom Rules (p. 30) guides you through creating a custom rule for the first time. It includes an example function that you can add to AWS Lambda with no modification.

To learn in depth about how AWS Lambda functions work and how to develop them, see the AWS Lambda Developer Guide.

Topics

Getting Started with Custom Rules

This procedure guides you through the process of creating a customer managed rule that evaluates whether each of your EC2 instances is the t2.micro type. AWS Config will run event-based evaluations for this rule, meaning it will check your instance configurations each time AWS Config detects a configuration change in an instance. AWS Config will flag t2.micro instances as compliant and all other instances as noncompliant. The compliance status will appear in the AWS Config console.

To have the best outcome with this procedure, your should have one or more EC2 instances in your AWS account. Your instances should include a combination of at least one t2.micro instance and other types.

To create this rule, first, you will create an AWS Lambda function by customizing a blueprint in the AWS Lambda console. Then, you will create a custom rule in AWS Config, and you will associate the rule with the function.

To create the AWS Lambda function for your custom rule

1. Sign in to the AWS Management Console and open the AWS Lambda console at https://console.aws.amazon.com/lambda/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. In the AWS Lambda console, choose **Create a Lambda function**.
4. On the **Select blueprint** page, for **filter**, type **config-rule-change-triggered**. Select the blueprint in the filter results.
5. On the **Configure function** page, complete the following steps:

   a. For **Name**, type **`InstanceTypeCheck`**.

b. For **Runtime**, keep **Node.js**.
c. For **Code entry type**, keep **Edit code inline**. The Node.js code for your function is provided in the code editor. For this procedure, you do not need to change the code.
d. For **Handler**, keep **`index.handler`**.
e. For **Role**, choose **AWS Config role**.
f. On the **AWS Lambda requires access to your resources** page, choose **Allow**. The role is created, and it includes a trust relationship with AWS Lambda.
g. On the **Configure function** page, choose **Next**.
h. On the **Review page**, verify the details about your function, and choose **Create function**. The AWS Lambda console displays your function.

6. To verify that your function is set up correctly, test it by completing the following steps:

   a. Choose **Actions**, **Configure test event**.
   b. In the **Input test event** window, replace the contents in the editor with the following example event:

```
{
  "invokingEvent": "{\"configurationItem\":{\"configurationItemCapture
Time\":\"2015-09-25T04:05:35.693Z\",\"configurationItem
Status\":\"OK\",\"resourceId\":\"resourceId\",\"resource
Type\":\"AWS::EC2::Instance\",\"tags\":{},\"relationships\":[],\"config
uration\":{\"instanceType\":\"t2.micro\"}}}",
  "ruleParameters": "{\"desiredInstanceType\":\"t2.micro\"}",
  "resultToken": "38400000-8cf0-11bd-b23e-10b96e4ef00d",
  "eventLeftScope": false
}
```

   c. Choose **Save and test**. AWS Lambda tests your function with the example event. If your function is working as expected, an error message similar to the following appears under **Execution result**:

```
{
  "errorMessage": "Result Token provided is invalid",
  "errorType": "InvalidResultTokenException",
. . .
```

   The `InvalidResultTokenException` is expected because your function runs successfully only when it receives a *result token* from AWS Config. The result token identifies the AWS Config rule and the event that caused the evaluation, and the result token associates an evaluation with a rule. This exception indicates that your function has the permission it needs to send results to AWS Config. Otherwise, the following error message appears: `not authorized to perform: config:PutEvaluations`. If this error occurs, update the role that you assigned to your function to allow the `config:PutEvaluations` action, and test your function again.

To add your custom rule to AWS Config

1. Open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to the same region in which you created the AWS Lambda function for your custom rule.
3. On the **Rules** page, choose **Add rule**.

4. On the **Add rule** page, choose **Add custom rule**.
5. On the **Configure rule** page, complete the following steps:

   a. For **Name**, type **`InstanceTypesAreT2micro`**.
   b. For **Description**, type **`Evaluates whether EC2 instances are the t2.micro type`**.
   c. For **AWS Lambda function ARN**, specify the ARN that AWS Lambda assigned to your function.

      Note
      The ARN that you specify in this step must not include the `$LATEST` qualifier. You can specify an ARN without a version qualifier or with any qualifier besides `$LATEST`. AWS Lambda supports function versioning, and each version is assigned an ARN with a qualifier. AWS Lambda uses the `$LATEST` qualifier for the latest version.

   d. For **Trigger type**, choose **Configuration changes**.
   e. For **Scope of changes**, choose **Resources**.
   f. For **Resources**, choose **Instance**.
   g. In the **Rule parameters** section, you must specify the rule parameter that your AWS Lambda function evaluates and the desired value. The function for this procedure evaluates the `desiredInstanceType` parameter.

      For **Key**, type **`desiredInstanceType`**. For **Value**, type **`t2.micro`**.

6. Choose **Save**. Your new rule displays on the **Rules** page.

   Compliance will display **Evaluating...** until AWS Config receives evaluation results from your AWS Lambda function. If the rule and the function are working as expected, a summary of the results appears after several minutes. For example, a result of **2 noncompliant resource(s)** indicates that 2 of your instances are not t2.micro instances, and a result of **Compliant** indicates that all instances are t2.micro. You can update the results with the refresh button.

   If the rule or function is not working as expected, you might see one of the following for **Compliance**:

   - **No results reported** - The rule was executed but did not apply to any of the AWS resources within its scope, or all such resources were deleted. Verify that the scope includes **Instance** for **Resources**, and try again.
   - **No resources in scope** - AWS Config cannot evaluate your recorded AWS resources against this rule because none of your resources are within the rule's scope. Verify that AWS Config is recording EC2 instances. You can choose which resources AWS Config records on the **Settings** page.
   - **Evaluations failed** - For information that can help you determine the problem, choose the rule name to open its details page and see the error message.

If your rule works correctly and AWS Config provides evaluation results, you can learn which conditions affect the compliance status of your rule. You can learn which resources, if any, are noncompliant, and why. For more information, see Viewing Configuration Compliance with AWS Config (p. 43).

Developing a Custom Rule for AWS Config

Complete the following procedure to create a customer managed rule, which is a custom rule that you develop and add to AWS Config. To create a custom rule, you first create an AWS Lambda function, which contains the evaluation logic for the rule. Then you associate the function with a custom rule that you create in AWS Config.

Creating an AWS Lambda Function for a Custom Config Rule

A *Lambda function* is custom code that you upload to AWS Lambda, and it is invoked by events that are published to it by an event source. If the Lambda function is associated with a Config rule, AWS Config invokes it when the rule's trigger occurs. The Lambda function then evaluates the configuration information that is sent by AWS Config, and it returns the evaluation results. For more information about Lambda functions, see Function and Event Sources in the *AWS Lambda Developer Guide*.

You can use a programming language that is supported by AWS Lambda to create a Lambda function for a custom rule. To make this task easier, you can customize an AWS Lambda blueprint or reuse a sample function from the AWS Config Rules GitHub repository.

AWS Lambda blueprints

The AWS Lambda console provides sample functions, or *blueprints*, which you can customize by adding your own evaluation logic. When you create a function, you can choose one of the following blueprints for triggered or periodic rules:

- **config-rule-change-triggered** – Triggered when your AWS resource configurations change.
- **config-rule-periodic** – Triggered when AWS Config delivers a configuration snapshot to S3, which happens at a periodic frequency that you control.

AWS Config Rules GitHub repository

A public repository of sample functions for custom rules is available on GitHub, a web-based code hosting and sharing service. The sample functions are developed and contributed by AWS Config users. If you want to use a sample, you can copy its code into a new AWS Lambda function. To view the repository, see https://github.com/awslabs/aws-config-rules/.

To create the function for your custom rule

1. Sign in to the AWS Management Console and open the AWS Lambda console at https://console.aws.amazon.com/lambda/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. Choose **Create a Lambda function**.
4. On the **Select blueprint** page, you can choose one of the blueprint functions for AWS Config rules as a starting point, or you can proceed without a blueprint by choosing **Skip**.
5. On the **Configure function** page, type a name and description.
6. For **Runtime**, choose the programming language in which your function is written.
7. For **Code entry type**, choose your preferred entry type. If you are using a blueprint, keep **Edit code inline**.
8. Provide your code using the method required by the code entry type that you selected. If you are using a blueprint, the function code is provided in the code editor, and you can customize it to include your own evaluation logic. Your code can evaluate the event data that AWS Config provides when it invokes your function:

   - For functions that are based on the **config-rule-change-triggered** blueprint, or for any function that is triggered by configuration changes, the event data is the `ConfigurationItem` object for the AWS resource that changed in configuration.
   - For functions that are based on the **config-rule-periodic** blueprint, or for any function that is triggered periodically, the event data is a configuration snapshot, which contains configuration items for all of the resources that AWS Config records in your AWS account.

- For both types of functions, AWS Config also passes rule parameters in JSON format. You can define which rule parameters are passed when you create the custom rule in AWS Config.

9. For **Handler**, specify the handler for your function. If you are using a blueprint, keep the default value.
10. For **Role**, choose **AWS Config role**.
11. On the **AWS Lambda requires access to your resources** page, choose **Allow**. The role is created, and it includes a trust relationship with AWS Lambda.
12. On the **Configure function** page, choose **Next**.
13. On the **Review page**, verify the details about your function, and choose **Create function**.

Creating a Custom Rule in AWS Config

Use AWS Config to create a custom rule and associate the rule with a Lambda function.

To create a custom rule

1. Open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to the same region in which you created the AWS Lambda function for your custom rule.
3. On the **Rules** page, choose **Add rule**.
4. On the **Add rule** page, choose **Add custom rule**.
5. On the **Configure rule** page, type a name and description.
6. For **AWS Lambda function ARN**, specify the ARN that AWS Lambda assigned to your function.

   Note
   The ARN that you specify in this step must not include the `$LATEST` qualifier. You can specify an ARN without a version qualifier or with any qualifier besides `$LATEST`. AWS Lambda supports function versioning, and each version is assigned an ARN with a qualifier. AWS Lambda uses the `$LATEST` qualifier for the latest version.
7. Choose the **Trigger type**:

   - **Configuration changes** – AWS Config invokes your Lambda function when it detects a configuration change.
   - **Periodic** – AWS Config recurringly invokes your Lambda function when it delivers a configuration snapshot.

8. If the **Trigger type** for your rule is **Configuration changes**, specify when AWS Config invokes your Lambda function by choosing one of the following options for **Scope of changes**:

   - **Resources** – When any resource that matches the specified resource type, or the type plus identifier, is created, changed, or deleted.
   - **Tags** – When any resource with the specified tag is created, changed, or deleted.
   - **All changes** – When any resource recorded by AWS Config is created, changed, or deleted.

9. If the **Trigger type** for your rule is **Periodic**, use the **Frequency** options to choose how often AWS Config invokes your Lambda function. AWS Config delivers a configuration snapshot at the chosen frequency to the Amazon S3 bucket that you specified when you set up AWS Config. After AWS Config delivers your snapshot, it invokes the function to evaluate the contents of the snapshot.
10. In the **Rule parameters** section, specify any rule parameters that your AWS Lambda function evaluates and the desired value.
11. Choose **Save**. Your new rule displays on the **Rules** page.

Compliance will display **Evaluating...** until AWS Config receives evaluation results from your AWS Lambda function. If the rule and the function are working as expected, a summary of results appears after several minutes. You can update the results with the refresh button.

If the rule or function is not working as expected, you might see one of the following for **Compliance**:

- **No results reported** - The rule was executed but did not apply to any of the AWS resources within its scope, or all such resources were deleted.
- **No resources in scope** - AWS Config cannot evaluate your recorded AWS resources against this rule because none of your resources are within the rule's scope. You can choose which resources AWS Config records on the **Settings** page.
- **Evaluations failed** - For information that can help you determine the problem, choose the rule name to open its details page and see the error message.

Example AWS Lambda Functions and Events for AWS Config Rules

Each custom Config rule is associated with an AWS Lambda *function*, which is custom code that contains the evaluation logic for the rule. When the trigger for a Config rule occurs (for example, when AWS Config detects a configuration change), AWS Config invokes the rule's Lambda function by publishing an *event*, which is a JSON object that provides the configuration data that the function evaluates.

For more information about functions and events in AWS Lambda, see Function and Event Sources in the *AWS Lambda Developer Guide*.

Topics

Example AWS Lambda Functions for AWS Config Rules (Node.js)

AWS Lambda executes functions in response to events that are published by AWS services. The function for a custom Config rule receives an event that is published by AWS Config, and the function then evaluates whether the configuration data provided by the event complies with the rule. The operations in a function for a Config rule differ depending on whether it performs an evaluation that is triggered by configuration changes or triggered periodically.

For information about common patterns within AWS Lambda functions, see Programming Model in the *AWS Lambda Developer Guide*.

Example Function for Evaluations Triggered by Configuration Changes

AWS Config will invoke a function like the following example when it detects a configuration change for a resource that is within a custom rule's scope.

If you use the AWS Config console to create a rule that is associated with a function like this example, choose **Configuration changes** as the trigger type. If you use the AWS Config API or AWS CLI to create the rule, set the `MessageType` attribute to `ConfigurationItemChangeNotification`.

This example evaluates an Amazon EC2 instance and checks whether its instance type matches a specified value (for example, t2.micro).

```
var aws  = require('aws-sdk'), // Loads the AWS SDK for JavaScript.
    config = new aws.ConfigService(), // Constructs a service object to use the
 aws.ConfigService class.
    COMPLIANCE_STATES = {
        COMPLAINT: 'COMPLIANT',
        NON_COMPLIANT: 'NON_COMPLIANT',
        NOT_APPLICABLE: 'NOT_APPLICABLE'
    };

// Receives the event and context from AWS Lambda. You can copy this handler
and use it in your own
// code with little or no modification.
exports.handler = function(event, context, callback) {
    // Parses the invokingEvent and ruleParameters values, which contain JSON
objects passed as strings.
    var invokingEvent = JSON.parse(event.invokingEvent),
        ruleParameters = JSON.parse(event.ruleParameters),
        compliance = COMPLIANCE_STATES.NOT_APPLICABLE,
        putEvaluationsRequest;

    if (isApplicable(invokingEvent.configurationItem, event)) {
        compliance = evaluateCompliance(invokingEvent.configurationItem, rule
Parameters);
    }

    // Initializes the request that contains the evaluation results.
    putEvaluationsRequest = {
        Evaluations: [
            {
                // Applies the evaluation result to the resource published in
the event.
              ComplianceResourceType: invokingEvent.configurationItem.resource
Type,
              ComplianceResourceId: invokingEvent.configurationItem.resourceId,

                ComplianceType: compliance,
                OrderingTimestamp: invokingEvent.configurationItem.configura
tionItemCaptureTime
            }
        ],
        ResultToken: event.resultToken
    };

    // Sends the evaluation results to AWS Config.
    config.putEvaluations(putEvaluationsRequest, function (err, data) {
        if (err) {
            callback(err, null);
        } else {
            if(data.FailedEvaluations.length > 0) {
                // Ends the function execution if any evaluation results are
not successfully reported.
                callback(null, JSON.stringify(data));
            } else {
                callback(null, data);
            }
        }
    });
};
```

```
// Checks whether the resource has been deleted or is out of scope. If so, the
 evaluation is reported as
// 'NOT_APPLICABLE'. You can copy this function and use it in your own code
with little or no modification.
function isApplicable(configurationItem, event) {
    var status = configurationItem.configurationItemStatus,
        eventLeftScope = event.eventLeftScope;
    return (status === 'OK' || status === 'ResourceDiscovered') && eventLeftScope
 === false;
}

// Evaluates the resource and returns the compliance value to the handler.
function evaluateCompliance(configurationItem, ruleParameters) {
    // Designates the resources as not applicable if it is not an EC2 instance.

    if(configurationItem.resourceType !== 'AWS::EC2::Instance') {
        return COMPLIANCE_STATES.NOT_APPLICABLE;
    }
    // Designates the resources as compliant if it is an EC2 instance of the
desired type.
    if(configurationItem.configuration.instanceType === ruleParameters.desired
InstanceType) {
        return COMPLIANCE_STATES.COMPLAINT;
    }
    return COMPLIANCE_STATES.NON_COMPLIANT;
}
```

Function Operations

The function performs the following operations at runtime:

1. The function runs when AWS Lambda passes the `event` object to the `handler` function. AWS Lambda also passes a `context` object, which contains information and methods that the function can use while it runs. In this example, the function accepts the optional `callback` parameter, which it uses to return information to the caller.
2. The handler calls the `isApplicable` function to determine whether the event should be evaluated. If the function determines that the resource is deleted or outside of the rule's scope, it returns a compliance value of `NOT_APPLICABLE`, and the evaluation is skipped.
3. The handler calls the `evaluateCompliance` function to get the compliance result. The handler passes the `configurationItem` and `ruleParameters` objects that AWS Config published in the event.

   In this example, the function tests whether the resource being evaluated is an EC2 instance. If the resource is not an EC2 instance, the function returns a compliance value of `NOT_APPLICABLE`.

   Tip
   Instead of writing code to test whether the resource is a certain type, you can constrain the Config rule's scope to include only resources of that type. In the example function, it would be unnecessary to test whether the resource is an EC2 instance if the rule's scope included only the EC2 instance resource type.

   The function tests whether the `instanceType` attribute in the configuration item is equal to the `desiredInstanceType` parameter value. If the values are equal, the function returns `COMPLIANT`, and if the values are not equal, the function returns `NON_COMPLIANT`.
4. The handler prepares to send the evaluation results to AWS Config by initializing the `putEvaluationsRequest` object. This object includes the `Evaluations` parameter, which identifies the compliance result, the type, and the ID of the resource that was evaluated. You can use the

   `Evaluations` parameter to apply the result to any resource type that is supported by AWS Config. The `putEvaluationsRequest` object also includes the result token from the event, which identifies the rule and the event for AWS Config.
5. The handler sends the evaluation results to AWS Config by passing the object to the `putEvaluations` method of the `aws.ConfigService` class.

Example Function for Periodic Evaluations

AWS Config will invoke a function like the following example for periodic evaluations. Periodic evaluations occur when AWS Config creates configuration snapshots and delivers them to the Amazon S3 bucket that is assigned to the AWS Config delivery channel.

If you use the AWS Config console to create a rule that is associated with a function like this example, choose **Periodic** as the trigger type. If you use the AWS Config API or AWS CLI to create the rule, set the `MessageType` attribute to `ConfigurationSnapshotDeliveryCompleted`.

This example evaluates a configuration snapshot and checks whether the total number of a specified resource exceeds a specified maximum.

```
var aws  = require('aws-sdk'), // Loads the AWS SDK for JavaScript.
    s3 = new aws.S3(), // Constructs a service object to use the aws.S3 class.

    zlib = require('zlib'), // Loads the zlib module for uncompressing data
from S3.
    config = new aws.ConfigService(), // Constructs a service object to use the
 aws.ConfigService class.
    COMPLIANCE_STATES = {
        COMPLAINT: 'COMPLIANT',
        NON_COMPLIANT: 'NON_COMPLIANT',
        NOT_APPLICABLE: 'NOT_APPLICABLE'
    };

// Receives the event and context from AWS Lambda. You can copy this handler
and use it in your own
// code with little or no modification.
exports.handler = function(event, context, callback) {
    // Parses the invokingEvent and ruleParameters values, which contain JSON
objects passed as strings.
    var invokingEvent = JSON.parse(event.invokingEvent),
        ruleParameters = JSON.parse(event.ruleParameters),
        // Stores the key and bucket name required to access the configuration
 snapshot in Amazon S3.
        s3key = invokingEvent.s3ObjectKey,
        s3bucket = invokingEvent.s3Bucket;
    readSnapshot(s3, s3key, s3bucket, function(err, snapshot) {
        var putEvaluationsRequest;
        if (err === null) {
            // Initializes the request that contains the evaluation results.
            putEvaluationsRequest = {
                Evaluations: [
                    {
                        // Applies the evaluation result to the AWS account
published in the event.
                        ComplianceResourceType: 'AWS::::Account',
                        ComplianceResourceId: event.accountId,
                        ComplianceType: evaluateCompliance(snapshot.configura
tionItems,
                            ruleParameters),
                     OrderingTimestamp: invokingEvent.notificationCreationTime

                    }
                ],
                ResultToken: event.resultToken
            };
            // Sends the evaluation results to AWS Config.
            config.putEvaluations(putEvaluationsRequest, function (err, data)
{
                if (err) {
                    callback(err, null);
                } else {
                    if(data.FailedEvaluations.length > 0) {
                       // Fails the function if any evaluation results are not
 successfully reported
                        callback(JSON.stringify(data));
                    }
                    callback(null, data);
                }
            });
        } else {
            callback(err, null);
        }
    });
};

// Reads and parses the configuration snapshot from the S3 bucket to which it
was delivered.
function readSnapshot(s3client, s3key, s3bucket, callback) {
    var params = {
            Key: s3key,
            Bucket: s3bucket
        },
        buffer = '';
    try {
        s3client.getObject(params)
            .createReadStream()
            .pipe(zlib.createGunzip())
            .on('data', function(chunk) {
                buffer = buffer + chunk;
            })
            .on('end', function() {
                callback(null, JSON.parse(buffer));
            });
    } catch(err) {
        callback(err, null);
    }
}

// Evaluates the configuration items in the snapshot and returns the compliance
 value to the handler.
function evaluateCompliance(configurationItems, ruleParameters) {
    var applicableResourceType = ruleParameters.applicableResourceType,
        maxCount = parseInt(ruleParameters.maxCount),
        count = 0;
   // Counts the resources that match the type assigned in the rule parameters.
```

```
    configurationItems.forEach(function(configurationItem) {
        if (configurationItem.resourceType === applicableResourceType) {
            count++;
        }
    });
    // Compares the number of matching resources to the maximum assigned in the
 rule parameters.
    if (count > maxCount) {
        return COMPLIANCE_STATES.NON_COMPLIANT;
    } else {
        return COMPLIANCE_STATES.COMPLAINT;
    }
}
```

Function Operations

The function performs the following operations at runtime:

1. The function runs when AWS Lambda passes the `event` object to the `handler` function. AWS Lambda also passes a `context` object, which contains information and methods that the function can use while it runs. In this example, the function accepts the optional `callback` parameter, which it uses to return information to the caller.
2. The handler calls the `readSnapshot` function, which accesses the Amazon S3 bucket and retrieves the configuration snapshot. The configuration snapshot contains configuration items for each resource that is recorded by AWS Config. The snapshot represents the state of the account's resources at the moment that the snapshot was created. To retrieve the snapshot, the `readSnapshot` function calls the `getObject` method of the `aws.S3` class. The call contains the bucket and key names that are published in the event. The `readSnapshot` function then uses the `zlib` module to decompress the data from the bucket, and it parses the data as JSON code.
3. The handler calls the `evaluateCompliance` function to get the compliance result. The handler passes the `configurationItems` object that it retrieved from the configuration snapshot, and it passes the `ruleParameters` object that AWS Config published in the event. The function traverses the configuration items in the snapshot and counts the resources that match the type assigned to the `applicableResourceType` parameter. Then the function tests whether the number exceeds the maximum assigned to the `maxCount` parameter. If the number of resources exceeds the maximum, the function returns `NON_COMPLIANT`. If the number of resources does not exceed the maximum, the function returns `COMPLIANT`.
4. The handler prepares to send the evaluation results to AWS Config by initializing the `putEvaluationsRequest` object. This object includes the `Evaluations` parameter, which identifies the compliance result and the AWS account that was published in the event. You can use the `Evaluations` parameter to apply the result to any resource type that is supported by AWS Config. The `putEvaluationsRequest` object also includes the result token from the event, which identifies the rule and the event for AWS Config.
5. The handler sends the evaluation results to AWS Config by passing the object to the `putEvaluations` method of the `aws.ConfigService` class.

Example Events for AWS Config Rules

When the trigger for a Config rule occurs, AWS Config invokes the rule's AWS Lambda function by publishing an event. Then AWS Lambda executes the function by passing the event to the function's handler.

Example Event for Evaluations Triggered by Configuration Changes

When AWS Config detects a configuration change for a resource that is within a rule's scope, it publishes an event similar to the following example:

```
{
    "invokingEvent": "{\"configurationItem\":{\"configurationItemCapture
Time\":\"2016-02-17T01:36:34.043Z\",\"awsAccountId\":\"000000000000\",\"config
urationItemStatus\":\"OK\",\"resourceId\":\"i-
00000000\",\"ARN\":\"arn:aws:ec2:us-east-1:000000000000:instance/i-
00000000\",\"awsRegion\":\"us-east-1\",\"availabilityZone\":\"us-east-
1a\",\"resourceType\":\"AWS::EC2::Instance\",\"tags\":{\"Foo\":\"Bar\"},\"rela
tionships\":[{\"resourceId\":\"eipalloc-00000000\",\"resource
Type\":\"AWS::EC2::EIP\",\"name\":\"Is attached to ElasticIp\"}],\"configura
tion\":{\"foo\":\"bar\"}},\"messageType\":\"ConfigurationItemChangeNotifica
tion\"}",
    "ruleParameters": "{\"myParameterKey\":\"myParameterValue\"}",
    "resultToken": "myResultToken",
    "eventLeftScope": false,
    "executionRoleArn": "arn:aws:iam::012345678912:role/config-role",
   "configRuleArn": "arn:aws:config:us-east-1:012345678912:config-rule/config-
rule-0123456",
    "configRuleName": "change-triggered-config-rule",
    "configRuleId": "config-rule-0123456",
    "accountId": "012345678912",
    "version": "1.0"
}
```

Example Event for Periodic Evaluations

When AWS Config periodically delivers a configuration snapshot to Amazon S3, it publishes an event similar to the following example:

```
{
    "invokingEvent": "{\"configSnapshotId\":\"00000000-0000-0000-0000-
000000000000\",\"s3ObjectKey\":\"AWSLogs/000000000000/Config/us-east-
1/2016/2/24/ConfigSnapshot/000000000000_Config_us-east-1_ConfigSnap
shot_20160224T182319Z_00000000-0000-0000-0000-000000000000.json.gz\",\"s3Buck
et\":\"config-bucket\",\"notificationCreationTime\":\"2016-02-
24T18:23:20.328Z\",\"messageType\":\"ConfigurationSnapshotDelivery
Completed\",\"recordVersion\":\"1.1\"}",
    "ruleParameters": "{\"myParameterKey\":\"myParameterValue\"}",
    "resultToken": "myResultToken",
    "eventLeftScope": false,
    "executionRoleArn": "arn:aws:iam::012345678912:role/config-role",
   "configRuleArn": "arn:aws:config:us-east-1:012345678912:config-rule/config-
rule-0123456",
    "configRuleName": "periodic-config-rule",
    "configRuleId": "config-rule-0123456",
    "accountId": "012345678912",
    "version": "1.0"
}
```

Event Attributes

The JSON object for an AWS Config event contains the following attributes:

`invokingEvent`
: If the event is published in response to a resource configuration change, the value for this attribute is a string that contains a JSON configuration item. The configuration item represents the state of the resource at the moment that AWS Config detected the change. For an example of a configuration item, see the output produced by the `get-resource-config-history` AWS CLI command in View Configuration History (p. 53).

  If the event is published for a periodic evaluation, the value is a string that contains a JSON object that identifies a configuration snapshot. The configuration snapshot is stored in the Amazon S3 bucket that is assigned to the AWS Config delivery channel. For an example, see Example Configuration Snapshot from AWS Config (p. 62).

  For each type of event, a function must parse the string with a JSON parser to be able to evaluate its contents, as shown in the following Node.js example:

```
var invokingEvent = JSON.parse(event.invokingEvent);
```

`ruleParameters`
: Key/value pairs that the function processes as part of its evaluation logic. You define parameters when you use the AWS Config console to create a custom rule. You can also define parameters with the `InputParameters` attribute in the `PutConfigRule` AWS Config API request or the `put-config-rule` AWS CLI command.

  The JSON code for the parameters is contained within a string, so a function must parse the string with a JSON parser to be able to evaluate its contents, as shown in the following Node.js example:

```
var ruleParameters = JSON.parse(event.ruleParameters);
```

`resultToken`
: A token that the function must pass to AWS Config with the `PutEvaluations` call.

`eventLeftScope`
: A Boolean value that indicates whether the AWS resource to be evaluated has been removed from the rule's scope. If the value is `true`, the function indicates that the evaluation can be ignored by passing `NOT_APPLICABLE` as the value for the `ComplianceType` attribute in the `PutEvaluations` call.

`executionRoleArn`
: The ARN of the IAM role that is assigned to AWS Config.

`configRuleArn`
: The ARN that AWS Config assigned to the rule.

`configRuleName`
: The name that you assigned to the rule that caused AWS Config to publish the event and invoke the function.

`configRuleId`
: The ID that AWS Config assigned to the rule.

`accountId`
: The ID of the AWS account that owns the rule.

`version`
: A version number assigned by AWS. The version will increment if AWS adds attributes to AWS Config events. If a function requires an attribute that is only in events that match or exceed a specific version, then that function can check the value of this attribute.

  The current version for AWS Config events is 1.0.

Viewing Configuration Compliance with AWS Config

You can use the AWS Config console, AWS CLI, or AWS Config API to view the compliance state of your rules and resources.

To view compliance (console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. In the navigation pane, choose **Rules**. The console shows the **Rules** page, which lists your rules and the compliance status of each.
4. Choose a rule to view its **Rule details** page. This page shows the rule's configuration, its status, and any AWS resources that do not comply with it.
5. If the **Rule details** shows any noncompliant resources, choose the **Config timeline** icon () for a resource to see its configuration timeline page. The page shows the configuration settings that AWS Config captured when it detected that the resource was noncompliant. This information can help you determine why the resource fails to comply with the rule. For more information, see Viewing Configuration Details in the AWS Config Console (p. 52).

You can also view the compliance of your resources by looking them up on the **Resource inventory** page. For more information, see Looking Up Resources That Are Discovered by AWS Config (p. 50).

To view compliance (AWS CLI)

To view compliance, use any of the following CLI commands:

- To see the compliance state of each of your rules, use the `describe-compliance-by-config-rule` command, as shown in the following example:

```
$ aws configservice describe-compliance-by-config-rule
{
    "ComplianceByConfigRules": [
        {
            "Compliance": {
                "ComplianceContributorCount": {
                    "CappedCount": 2,
                    "CapExceeded": false
                },
                "ComplianceType": "NON_COMPLIANT"
            },
            "ConfigRuleName": "instances-in-vpc"
        },
        {
            "Compliance": {
                "ComplianceType": "COMPLIANT"
            },
            "ConfigRuleName": "restricted-common-ports"
        },
...
```

For each rule that has a compliance type of `NON_COMPLIANT`, AWS Config returns the number of noncompliant resources for the `CappedCount` parameter.

- To see the compliance state of each resource that AWS Config evaluates for a specific rule, use the `get-compliance-details-by-config-rule` command, as shown in the following example:

```
$ aws configservice get-compliance-details-by-config-rule --config-rule-name
 ConfigRuleName{
    "EvaluationResults": [
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1443610576.349,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-nnnnnnnn",
                    "ConfigRuleName": "ConfigRuleName"
                }
            },
            "ResultRecordedTime": 1443751424.969,
            "ConfigRuleInvokedTime": 1443751421.208,
            "ComplianceType": "COMPLIANT"
        },
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1443610576.349,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-nnnnnnnn",
                    "ConfigRuleName": "ConfigRuleName"
                }
            },
            "ResultRecordedTime": 1443751425.083,
            "ConfigRuleInvokedTime": 1443751421.301,
            "ComplianceType": "NON_COMPLIANT"
        },
...
```

- To see the compliance state for each AWS resource of a specific type, use the `describe-compliance-by-resource` command, as shown in the following example:

```
$ aws configservice describe-compliance-by-resource --resource-type
AWS::EC2::Instance
{
    "ComplianceByResources": [
        {
            "ResourceType": "AWS::EC2::Instance",
            "ResourceId": "i-nnnnnnnn",
            "Compliance": {
                "ComplianceContributorCount": {
                    "CappedCount": 1,
                    "CapExceeded": false
                },
                "ComplianceType": "NON_COMPLIANT"
            }
        },
        {
            "ResourceType": "AWS::EC2::Instance",
            "ResourceId": "i-nnnnnnnn",
            "Compliance": {
                "ComplianceType": "COMPLIANT"
            }
        },
...
```

- To see the compliance details of an individual AWS resource, use the `get-compliance-details-by-resource` command.

```
$ aws configservice get-compliance-details-by-resource --resource-type
AWS::EC2::Instance --resource-id i-nnnnnnnn
{
    "EvaluationResults": [
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1443610576.349,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-nnnnnnnn",
                    "ConfigRuleName": "instances-in-vpc"
                }
            },
            "ResultRecordedTime": 1443751425.083,
            "ConfigRuleInvokedTime": 1443751421.301,
            "ComplianceType": "NON_COMPLIANT"
        }
    ]
}
```

To view compliance (AWS Config API)

To view compliance, use any of the following API actions:

- To see the compliance state of each of your rules, use the `DescribeComplianceByConfigRule` action.
- To see the compliance state of each resource that AWS Config evaluates for a specific rule, use the `GetComplianceDetailsByConfigRule` action.
- To see the compliance state for each AWS resource of a specific type, use the `DescribeComplianceByResource` action.
- To see the compliance details of an individual AWS resource, use the `GetComplianceDetailsByResource` action. The details include which AWS Config rules evaluated the resource, when each rule last evaluated it, and whether the resource complies with each rule.

Managing Your AWS Config Rules

You can use the AWS Config console, AWS CLI, and AWS Config API to view, update, and delete your AWS Config rules.

Viewing Your Rules

As your rules grow in number, it can be helpful to see a list of the rules associated with your account.

To view your rules (console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. In the navigation pane, choose **Rules**. The **Rules** page shows your rules and the compliance status for each. You can choose a rule to view its details.

To view your rules (AWS CLI)

- Use the `describe-config-rules` command:

```
$ aws configservice describe-config-rules
```

  AWS Config returns the details for all of your rules.

To view your rules (AWS Config API)

- Use the `DescribeConfigRules` action.

Updating a Rule

Occasionally you may need to change how a rule works, for example, to broaden or narrow its scope.

To update a rule (console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. In the navigation pane, choose **Rules**.
4. Choose the **Edit rule** icon (✎) for the rule that you want to update.
5. Modify the settings on the **Config rule** page to change your rule as needed.
6. Choose **Save**.

To update a rule (AWS CLI)

1. Use the `put-config-rule` command with the `--generate-cli-skeleton` parameter to create a local JSON file that has the parameters for your rule:

```
$ aws configservice put-config-rule --generate-cli-skeleton > putConfi
gRule.json
```

2. Open the JSON file in a text editor and remove any parameters that don't need updating, with the following exceptions:

   - You must include at least one of the following parameters to identify the rule:

     `ConfigRuleName`, `ConfigRuleArn`, or `ConfigRuleId`.
   - If you are updating a customer managed rule, you must include the `Source` object and its parameters.

3. Fill in the values for the parameters that remain. If you need to reference the details of your rule, you can use the **describe-config-rules** command.

   For example, the following JSON code updates the resource types that are in the scope of a customer managed rule:

```
{
  "ConfigRule": {
    "ConfigRuleName": "ConfigRuleName",
    "Scope": {
      "ComplianceResourceTypes": [
        "AWS::EC2::Instance",
        "AWS::EC2::Volume",
        "AWS::EC2::VPC"
      ]
    },
    "Source": {
      "Owner": "CUSTOM_LAMBDA",
     "SourceIdentifier": "arn:aws:lambda:us-east-1:123456789012:function:Con
figRuleName",
      "SourceDetails": [
        {
          "EventSource": "aws.config",
          "MessageType": "ConfigurationItemChangeNotification"
        }
      ]
    }
  }
}
```

4. Use the `put-config-rule` command with the `--cli-input-json` parameter to pass your JSON configuration to AWS Config:

```
$ aws configservice put-config-rule --cli-input-json file://putConfigRule.json
```

5. To verify that you successfully updated your rule, use the **describe-config-rules** command to view the rule's configuration:

```
$ aws configservice describe-config-rules --config-rule-name ConfigRuleName
{
    "ConfigRules": [
        {
            "ConfigRuleState": "ACTIVE",
            "ConfigRuleName": "ConfigRuleName",
           "ConfigRuleArn": "arn:aws:config:us-east-1:123456789012:config-
rule/config-rule-nnnnnn",
            "Source": {
                "Owner": "CUSTOM_LAMBDA",
                "SourceIdentifier": "arn:aws:lambda:us-east-
1:123456789012:function:ConfigRuleName",
                "SourceDetails": [
                    {
                        "EventSource": "aws.config",
                      "MessageType": "ConfigurationItemChangeNotification"

                    }
                ]
            },
            "Scope": {
                "ComplianceResourceTypes": [
                    "AWS::EC2::Instance",
                    "AWS::EC2::Volume",
                    "AWS::EC2::VPC"
                ]
            },
            "ConfigRuleId": "config-rule-nnnnnn"
        }
    ]
}
```

To update a rule (AWS Config API)

- Use the `PutConfigRule` action.

Deleting a Rule

In time, some of your rules will become obsolete. When that happens, you can simply remove them.

To delete a rule (console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the AWS Management Console menu, verify that the region selector is set to a region that supports AWS Config rules. For the list of supported regions, see AWS Config Regions and Endpoints in the *Amazon Web Services General Reference*.
3. In the navigation pane, choose **Rules**.
4. Choose the **Edit rule** icon (✎) for the rule that you want to delete.
5. On the **Configure rule** page, choose **Delete rule**.
6. When prompted, choose **Delete**.

To delete a rule (AWS CLI)

- Use the `delete-config-rule` command as shown in the following example:

```
$ aws configservice delete-config-rule --config-rule-name ConfigRuleName
```

To delete a rule (AWS Config API)

- Use the `DeleteConfigRule` action.

Viewing AWS Resource Configurations and History

You can view all of the resources that AWS Config is recording in your account, the configuration changes that took place for a resource over a specified time period, and the relationships of the selected resource with all the related resources. You can follow the steps using either the AWS Config console or the AWS CLI.

Topics

Looking Up Resources That Are Discovered by AWS Config

You can use the AWS Config console, AWS CLI, and AWS Config API to look up the resources that AWS Config has taken an inventory of, or *discovered*, including deleted resources and those that AWS Config is not currently recording. AWS Config discovers only supported resource types. For a list of supported resource types, see Supported AWS Resource Types (p. 6).

When you look up resources, AWS Config lists the resources that match your search options and provides identifying information for each, including the resource type and identifier. The resource identifier might be a resource ID, such as an EC2 instance ID, or a resource name, if applicable. You can use this information to access the configuration details for a resource.

Looking Up Resources (AWS Config Console)

You can use resource types or tag information to look up resources in the AWS Config console. The options for looking up resources are provided on the **Resource inventory** page.

To look up resources

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. On the **Resource inventory** page, specify the search options for the resources that you want to find in either of the following ways:
   - Choose **Resources** and then choose one or more resource types in the list. This list only includes resource types that AWS Config supports. To narrow the results, type a resource ID or, if applicable, a resource name in the next box. If you want the results to include deleted resources, select **Include deleted resources**.
   - Choose **Tag** and type a tag key that is applied to your resources, such as **`CostCenter`**. To narrow the results, type a tag value in the next box.
3. When you have specified the search options, choose **Lookup**. The resources that match your options and that are discovered by AWS Config appear in the table of results.
4. To view the configuration details of a resource in your results, choose the resource identifier. AWS Config displays a details page where you can view how the resource's configuration changed over time, providing AWS Config was set to record it. To learn more about the information on this page, see Viewing Configuration Details in the AWS Config Console (p. 52).

Looking Up Resources (AWS CLI or AWS Config API)

You can use the AWS CLI or AWS Config API to list resources that AWS Config has discovered. You specify a resource type, and AWS Config returns a list of resource identifiers for resources of that type. To learn more about resource identifiers, see `ResourceIdentifier` in the *AWS Config API Reference*.

To look up resources (AWS CLI)

- Use the `aws configservice` `list-discovered-resources` command:

```
$ aws configservice list-discovered-resources --resource-type
"AWS::EC2::Instance"
        {
            "resourceIdentifiers": [
                {
                    "resourceType": "AWS::EC2::Instance",
                    "resourceId": "i-nnnnnnnn"
                }
            ]
        }
```

To view the configuration details of a resource that is listed in the response, use the `get-resource-config-history` command, and specify the resource type and ID. For an example of this command and the response from AWS Config, see View Configuration History (p. 53).

To look up resources (AWS Config API)

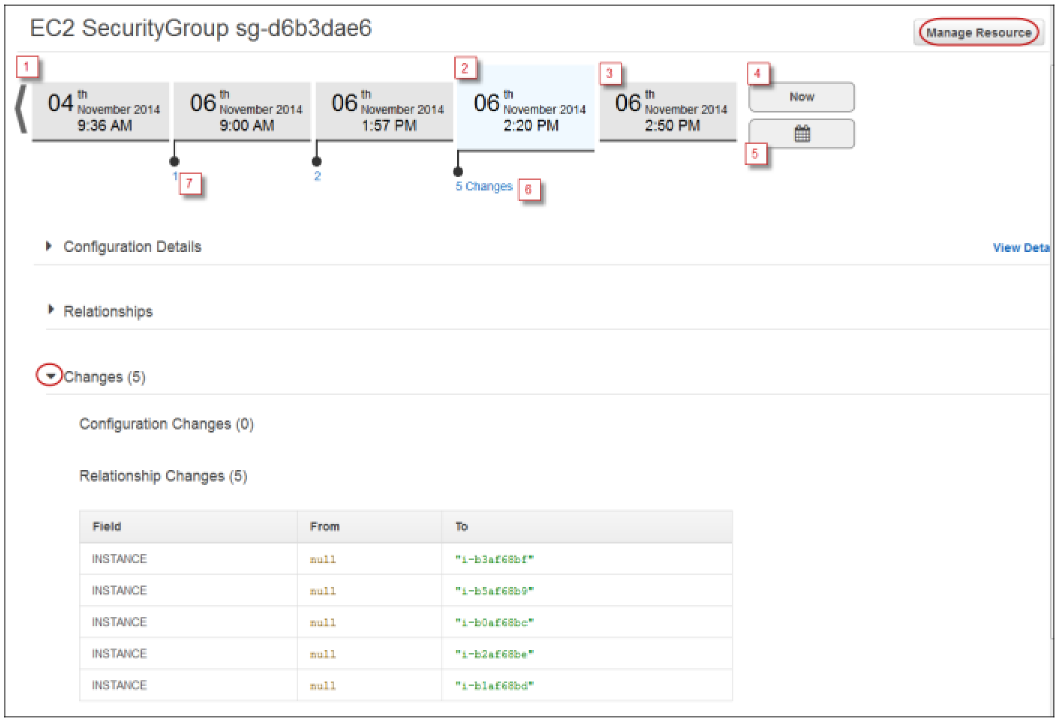
- Use the `ListDiscoveredResources` action.

To get the configuration details of a resource that is listed in the response, use the `GetResourceConfigHistory` action, and specify the resource type and ID.

Viewing Configuration Details in the AWS Config Console

When you find a resource on the **Resource inventory** page and click its resource identifier, AWS Config displays the resource's details page. The details page provides information about the configuration, relationships, and number of changes made to that resource.

The blocks at the top of the page are collectively called the *timeline*. The timeline shows the date and the time for when the recording was made.


EC2 SecurityGroup sg-d6b3dae6
Manage Resource
1
04 th November 2014 9:36 AM
06 th November 2014 9:00 AM
06 th November 2014 1:57 PM
2
06 th November 2014 2:20 PM
3
06 th November 2014 2:50 PM
4
Now
5
1
7
2
5 Changes
6
Configuration Details
View Deta
Relationships
Changes (5)

Configuration Changes (0)

Relationship Changes (5)

Field	From	To
INSTANCE	null	"i-b3af68bf"
INSTANCE	null	"i-b5af68b9"
INSTANCE	null	"i-b0af68bc"
INSTANCE	null	"i-b2af68be"
INSTANCE	null	"i-b1af68bd"

Details page features

1. Click to scroll the timeline to an earlier point in the resource's configuration history.
2. Click a timeline block to select that time period. The descriptions in the **Configuration Details**, **Relationships**, and **Changes** sections comprise the configuration item of the selected resource at the selected time period.
3. Shows the latest configuration change.
4. Click to return the timeline to the current time.
5. Click to view a configuration item by specifying a date (and, if needed, time). Then click **Apply**.

6. Click to navigate to the **Changes** section below. The numeral indicates the number of configuration changes that occurred for the resource between the selected time period and that of the previous block.

To view the configuration item details of the selected resource

1. Use the arrows at either end of the timeline to view the timeline blocks for configuration items that were recorded in other time periods.
2. Click **Configuration Details** to view the description of the selected resource.
3. Click **Relationships** to see a list of other supported resources in the account that are related to the current resource. If the **Relationships** section does not expand, the selected resource was not related to any other resource existing in your account during the selected time period.

   For more information, see Resource Relationship (p. 3).
4. If changes are indicated for the selected time period, click **Changes** to view the configuration changes made to the resource. The **Changes** section also lists the relationship changes that occurred as a result of configuration changes.
5. (Optional) Click **View Details** to view the configuration information listed in the text format. Click the arrows in the details window to see additional details.

   For more information about the entries in the details window, see Components of a Configuration Item (p. 7).
6. (Optional) Click **Manage Resource** to be taken to the console for the selected resource where you can make changes. If you make a change, go back to the AWS Config console and click **Now** to see the changes. Be aware that AWS Config can take up to 10 minutes to refresh the information it displays.

   The console also provides details pages for supported resources that you do not include in the list of resources that AWS Config records. The information that is provided on these details pages is limited, and ongoing configuration changes are not shown.

View Configuration Details Using the CLI

The configuration items that AWS Config records are delivered to the specified delivery channel on demand as a configuration snapshot and as a configuration stream. AWS Config also delivers configuration items at regular intervals to the specified delivery channel as configuration history.

You can use the AWS CLI to view the history of configuration items for each resource and to deliver and view a configuration snapshot.

Topics

View Configuration History

You can use the AWS CLI to view the history of a resource's various configurations. Use the `get-resource-config-history` command and specify the resource type and the resource ID; for example:

```
$ aws configservice get-resource-config-history --resource-type AWS::EC2::Secur
ityGroup --resource-id sg-6fbb3807
{
    "configurationItems": [
        {
            "configurationItemCaptureTime": 1414708529.9219999,
            "relationships": [
                {
                    "resourceType": "AWS::EC2::Instance",
                    "resourceId": "i-7a3b232a",
                    "relationshipName": "Is associated with Instance"
                },
                {
                    "resourceType": "AWS::EC2::Instance",
                    "resourceId": "i-8b6eb2ab",
                    "relationshipName": "Is associated with Instance"
                },
                {
                    "resourceType": "AWS::EC2::Instance",
                    "resourceId": "i-c478efe5",
                    "relationshipName": "Is associated with Instance"
                },
                {
                    "resourceType": "AWS::EC2::Instance",
                    "resourceId": "i-e4cbe38d",
                    "relationshipName": "Is associated with Instance"
                }
            ],
            "availabilityZone": "Not Applicable",
            "tags": {},
            "resourceType": "AWS::EC2::SecurityGroup",
            "resourceId": "sg-6fbb3807",
            "configurationStateId": "1",
            "relatedEvents": [],
           "arn": "arn:aws:ec2:us-east-1:012345678912:security-group/default",

            "version": "1.0",
            "configurationItemMD5Hash": "860aa81fc3869e186b2ee00bc638a01a",
           "configuration": "{\"ownerId\":\"605053316265\",\"groupName\":\"de
fault\",\"groupId\":\"sg-6fbb3807\",\"description\":\"default group\",\"ipPer
missions\":[{\"ipProtocol\":\"tcp\",\"fromPort\":80,\"toPort\":80,\"userIdGroup
Pairs\":[{\"userId\":\"amazon-elb\",\"groupName\":\"amazon-elb-sg\",\"grou
pId\":\"sg-843f59ed\"}],\"ipRanges\":[\"0.0.0.0/0\"]},{\"ipPro
tocol\":\"tcp\",\"fromPort\":0,\"toPort\":65535,\"userIdGroup
Pairs\":[{\"userId\":\"605053316265\",\"groupName\":\"default\",\"groupId\":\"sg-
6fbb3807\"}],\"ipRanges\":[]},{\"ipProtocol\":\"udp\",\"fromPort\":0,\"to
Port\":65535,\"userIdGroupPairs\":[{\"userId\":\"605053316265\",\"group
Name\":\"default\",\"groupId\":\"sg-6fbb3807\"}],\"ipRanges\":[]},{\"ipPro
tocol\":\"icmp\",\"fromPort\":-1,\"toPort\":-1,\"userIdGroup
Pairs\":[{\"userId\":\"605053316265\",\"groupName\":\"default\",\"groupId\":\"sg-
6fbb3807\"}],\"ipRanges\":[]},{\"ipProtocol\":\"tcp\",\"fromPort\":1433,\"to
Port\":1433,\"userIdGroupPairs\":[],\"ipRanges\":[\"0.0.0.0/0\"]},{\"ipPro
tocol\":\"tcp\",\"fromPort\":3389,\"toPort\":3389,\"userIdGroup
Pairs\":[],\"ipRanges\":[\"207.171.160.0/19\"]}],\"ipPermissionsEgress\":[],\"vp
cId\":null,\"tags\":[]}",
            "configurationItemStatus": "ResourceDiscovered",
            "accountId": "605053316265"
        }
    ],
    "nextToken":
     ..........
```

For detailed explanation of the response fields, see Components of a Configuration Item (p. 7) and Supported Resource Relationships (p. 8).

Deliver Configuration Snapshot

AWS Config delivers configuration items of the AWS resources that AWS Config is recording to the Amazon S3 bucket that you specified when you configured your delivery channel.

To deliver configuration snapshot

- Use the `deliver-config-snapshot` command by specifying the name assigned by AWS Config when you configured your delivery channel; for example:

```
$ aws configservice deliver-config-snapshot --delivery-channel-name default
{
    "configSnapshotId": "94ccff53-83be-42d9-996f-b4624b3c1a55"
}
```

The next step is to verify that configuration snapshot was delivered successfully to the delivery channel.

To verify delivery status

- Use the `describe-delivery-channel-status` command to verify that the AWS Config has started delivering the configurations to the specified delivery channel; for example:

```
$ aws configservice describe-delivery-channel-status
{
    "DeliveryChannelsStatus": [
        {
            "configStreamDeliveryInfo": {
                "lastStatusChangeTime": 1415138614.125,
                "lastStatus": "SUCCESS"
            },
            "configHistoryDeliveryInfo": {
                "lastSuccessfulTime": 1415148744.267,
                "lastStatus": "SUCCESS",
                "lastAttemptTime": 1415148744.267
            },
            "configSnapshotDeliveryInfo": {
                "lastSuccessfulTime": 1415333113.4159999,
                "lastStatus": "SUCCESS",
                "lastAttemptTime": 1415333113.4159999
            },
            "name": "default"
        }
    ]
}
```

  The response lists the status of all the three delivery formats that AWS Config uses to deliver configurations to your bucket and topic.

  Take a look at the `lastSuccessfulTime` field in `configSnapshotDeliveryInfo`. The time should match the time you last requested the delivery of the configuration snapshot.

Note
AWS Config uses the UTC format (GMT-08:00) to record the time.

To view the configuration snapshot in your Amazon S3 bucket

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the Amazon S3 console **All Buckets** list, click the name of your Amazon S3 bucket.
3. Click through the nested folders in your bucket until you see the `ConfigSnapshot` object with a snapshot ID that matches with the ID returned by the command. Download and open the object to view the configuration snapshot.

   The S3 bucket also contains an empty file named `ConfigWritabilityCheckFile`. AWS Config creates this file to verify that the service can successfully write to the S3 bucket.

Monitoring AWS Config Resource Changes by Email

If you have set up AWS Config to stream configuration changes and notifications to an Amazon SNS topic, you can monitor those changes by email. These emails can include configuration history or snapshot information as well as change notifications. You can also set up email filters based on the subject line or message body to look for specific changes.

To monitor resource changes by email

1. If you haven't done so already, set up AWS Config to deliver notifications to an Amazon SNS topic. For more information, see Set Up AWS Config Using the Console (p. 15) or Set Up AWS Config Using the AWS CLI (p. 17).
2. Open the Amazon SNS console at https://console.aws.amazon.com/sns/.
3. In the navigation pane of the Amazon SNS console, click **Topics**.
4. On the **Topics** page, open the Amazon SNS topic you specified when you set up AWS Config by clicking its name in the **ARN** column.
5. On the **Topic Details** page, under **Subscriptions**, click **Create subscription**.
6. In the **Create subscription** dialog box, for **Protocol**, select **Email**.
7. For **Endpoint**, type the email address where you want the notifications sent.
8. Click **Create subscription**.

   In a few moments, check your email for an email confirmation. In the meantime, the console displays **PendingConfirmation** in the **Subscription ID** column.
9. Open the email from "AWS Notifications" and click **Confirm subscription**.

   Tip
   If you want to monitor specific resources or other important changes, you can set up email filters in your email application.

Example Amazon SNS Notification and Email from AWS Config

AWS Config uses Amazon SNS to deliver notifications to subscription endpoints. These notifications provide the delivery status for configuration snapshots and configuration histories, and they provide each configuration item that AWS Config creates when the configurations of recorded AWS resources change. If you choose to have these notifications sent by email, you can use filters in your email client application based on the subject line and message body of the email.

The following is an example of the payload of an Amazon SNS notification that is generated when AWS Config detects that the Amazon Elastic Block Store volume vol-ce676ccc is attached to the instance with the ID i-344c463d.

```
{
    "Type": "Notification",
    "MessageId": "8b945cb0-db34-5b72-b032-1724878af488",
    "TopicArn": "arn:aws:sns:us-west-2:12345678910:example",
    "Message": {
        "MessageVersion": "1.0",
        "NotificationCreateTime": "2014-03-18T10:11:00Z",
        "messageType": "ConfigurationItemChangeNotification",
        "configurationItems": [
            {
                "configurationItemVersion": "1.0",
                "configurationItemCaptureTime": "2014-03-07T23:47:08.918Z",
                "arn": "arn:aws:us-west-2b:123456789012:volume/vol-ce676ccc",
                "resourceId": "vol-ce676ccc",
                "accountId": "12345678910",
               "configurationStateID": "3e660fdf-4e34-4f32-afeb-0ace5bf3d63a",

                "configuationItemStatus": "OK",
                "relatedEvents": [
                    "06c12a39-eb35-11de-ae07-adb69edbb1e4",
                    "c376e30d-71a2-4694-89b7-a5a04ad92281"
                ],
                "availibilityZone": "us-west-2b",
                "resourceType": "AWS::EC2::VOLUME",
                "resourceCreationTime": "2014-02-27T21:43:53.885Z",
                "tags": {},
                "relationships": [
                    {
                        "resourceId": "i-344c463d",
                        "resourceType": "AWS::EC2::INSTANCE",
                        "name": "Attached to Instance"
                    }
                ],
                "configuration": {
                    "volumeId": "vol-ce676ccc",
                    "size": 1,
                    "snapshotId": "",
                    "availabilityZone": "us-west-2b",
                    "state": "in-use",
                    "createTime": "2014-02-27T21:43:53.0885+0000",
                    "attachments": [
                        {
                            "volumeId": "vol-ce676ccc",
                            "instanceId": "i-344c463d",
                            "device": "/dev/sdf",
                            "state": "attached",
                            "attachTime": "2014-03-07T23:46:28.0000+0000",
                            "deleteOnTermination": false
                        }
                    ],
                    "tags": [],
                    "volumeType": "standard"
                }
            }
        ],
        "configurationItemDiff": {
            "changeType": "UPDATE",
            "changedProperties": {
                "Configuration.State": {
                    "previousValue": "available",
                    "updatedValue": "in-use",
                    "changeType": "UPDATE"
                },
                "Configuration.Attachments.0": {
                    "updatedValue": {
                        "VolumeId": "vol-ce676ccc",
                        "InstanceId": "i-344c463d",
                        "Device": "/dev/sdf",
                        "State": "attached",
                        "AttachTime": "FriMar0723: 46: 28UTC2014",
                        "DeleteOnTermination": "false"
                    },
                    "changeType": "CREATE"
                }
            }
        }
    },
    "Timestamp": "2014-03-07T23:47:10.001Z",
    "SignatureVersion": "1",
    "Signature": "LgfJNB5aOk/w3omqsYrv5cUFY8yvIJvO5ZZh46/KGPApk6HXRTBRlkh
jacnxIXJEWsGI9mxvMmoWPLJGYEAR5FF/+/Ro9QTmiTNcEjQ5kB8wGsR
WVrk/whAzT2lVtofc365En2T1Ncd9iSFFXfJchgBmI7EACZ28t+n2mWFgo57n6eGDvHTedslzC6Kxk
fWTfXsR6zHXzkB3XuZImktflg3iPKtvBb3Zc9iVbNsBEI4FITFWktSqqomYDjc5h0kgapIo4CtCHGK
pALW9JDmP+qZhMzEbHWpzFlEzvFl55KaZXxDbznBD1ZkqPgno/WufuxszCiMrsmV8pUNUnkU1TA==",

    "SigningCertURL": "https://sns.us-west-2.amazonaws.com/SimpleNotification
Service-e372f8ca30337fdb084e8ac449342c77.pem",
    "UnsubscribeURL": "https://sns.us-west-2.amazonaws.com/?Action=Unsub
scribe&SubscriptionArn=arn:aws:sns:us-west-2:12345678910:example:a6859fee-3638-
407c-907e-879651c9d143"
}
```

Example Email Format and Filters

If you have chosen to create an email subscription to your Amazon SNS topic, you can filter the email you receive based on information in the subject line and message body. To create a subscription for an Amazon SNS topic, see Monitoring AWS Config Resource Changes by Email (p. 56).

The subject line of an email looks like the following example:

```
[AWS Config:us-west-2] AWS::EC2::Instance i-12abcd3e Created in Account
12345678910
```

In your email client application, you can set up email filters or rules to watch for specific changes or to organize your notifications. For example, you could organize email notifications by region, resource type, resource name, or AWS account. These email filters can be useful if you are managing multiple accounts or if you have a large number of resources in your account.

The message body of an email subscription created with the **Email** protocol contains information about create, update, and delete events for your AWS resources. The following example shows an email message body created with the **Email** protocol:

```



View the Timeline for this Resource in AWS Config Management Console:
https://console.aws.amazon.com/config/home?region=us-west-2#/timeline/AWS::
EC2::Instance/i-12abcd3e

New State and Change Record:
----------------------------
{
  "configurationItemDiff": {
    "changedProperties": {},
    "changeType": "CREATE"
  },
  "configurationItem": {
    "configurationItemVersion": "1.0",
    "configurationItemCaptureTime": "2015-03-19T21:20:35.737Z",
    "configurationStateId": 1,
    "relatedEvents": [
      "4f8abc4f-6def-4g42-hi03-46j3b48k0lmn"
    ],
    "awsAccountId": "12345678910",
    "configurationItemStatus": "ResourceDiscovered",
    "resourceId": "i-92aeda5b",
    "ARN": "arn:aws:ec2:us-west-2:12345678910:instance/i-12abcd3e",
    "awsRegion": "us-west-2",
    "availabilityZone": "us-west-2c",
    "configurationStateMd5Hash": "123456789e0f930642026053208e",
    "resourceType": "AWS::EC2::Instance",
    "resourceCreationTime": "2015-03-19T21:13:05.000Z",
    "tags": {},
    "relationships": [
      {
        "resourceId": "abc-1234de56",
        "resourceType": "AWS::EC2::NetworkInterface",
        "name": "Contains NetworkInterface"
      },
      {
        "resourceId": "ab-c12defg3",
        "resourceType": "AWS::EC2::SecurityGroup",
        "name": "Is associated with SecurityGroup"
      },
      {
        "resourceId": "subnet-a1b2c3d4",
        "resourceType": "AWS::EC2::Subnet",
        "name": "Is contained in Subnet"
      },
      {
        "resourceId": "vol-a1bc234d",
        "resourceType": "AWS::EC2::Volume",
        "name": "Is attached to Volume"
      },
      {
        "resourceId": "vpc-a12bc345",
        "resourceType": "AWS::EC2::VPC",
        "name": "Is contained in Vpc"
      }
    ],
    "configuration": {
      "instanceId": "i-12abcd3e",
      "imageId": "ami-123a4567",
      "state": {
        "code": 16,
        "name": "running"
      },
      "privateDnsName": "ip-000-00-0-000.us-west-2.compute.internal",
      "publicDnsName":
"ec2-12-345-678-910.us-west-2.compute.amazonaws.com",
      "stateTransitionReason": "",
      "keyName": null,
      "amiLaunchIndex": 0,
      "productCodes": [],
      "instanceType": "t2.micro",
      "launchTime": "2015-03-19T21:13:05.000Z",
      "placement": {
        "availabilityZone": "us-west-2c",
        "groupName": "",
        "tenancy": "default"
      },
      "kernelId": null,
      "ramdiskId": null,
      "platform": null,
      "monitoring": {
        "state": "disabled"
      },
      "subnetId": "subnet-a1b2c3d4",
      "vpcId": "vpc-a12bc345",
      "privateIpAddress": "000.00.0.000",
      "publicIpAddress": "00.000.000.000",
      "stateReason": null,
      "architecture": "x86_64",
      "rootDeviceType": "ebs",
      "rootDeviceName": "/dev/abcd",
      "blockDeviceMappings": [
        {
          "deviceName": "/dev/abcd",
          "ebs": {
            "volumeId": "vol-a1bc234d",
            "status": "attached",
            "attachTime": "2015-03-19T21:13:07.000Z",
            "deleteOnTermination": true
          }
        }
      ],
      "virtualizationType": "hvm",
      "instanceLifecycle": null,
      "spotInstanceRequestId": null,
      "clientToken": "ab1234c5-6d78-910-1112-13ef14g15hi16",
      "tags": [],
      "securityGroups": [
        {
          "groupName": "default",
          "groupId": "sg-a12bcde3"
        }
      ],
      "sourceDestCheck": true,
      "hypervisor": "xen",
      "networkInterfaces": [
        {
          "networkInterfaceId": "eni-1234ab56",
          "subnetId": "subnet-a1b2c3d4",
          "vpcId": "vpc-a12bc345",
          "description": "",
          "ownerId": "12345678910",
          "status": "in-use",
          "macAddress": "1a:23:45:67:b8",
          "privateIpAddress": "000.00.0.000",
          "privateDnsName": "ip-000-00-0-000.us-west-2.compute.internal",
          "sourceDestCheck": true,
          "groups": [
            {
              "groupName": "default",
              "groupId": "sg-a12bcde3"
            }
          ],
          "attachment": {
            "attachmentId": "eni-attach-123a4b5c",
            "deviceIndex": 0,
            "status": "attached",
            "attachTime": "2015-03-19T21:13:05.000Z",
            "deleteOnTermination": true
          },
          "association": {
            "publicIp": "00.000.000.000",
            "publicDnsName":
"ec2-00-000-000-000.us-west-2.compute.amazonaws.com",
            "ipOwnerId": "amazon"
          },
          "privateIpAddresses": [
            {
              "privateIpAddress": "000.00.0.000",
              "privateDnsName":
"ip-000-00-0-000.us-west-2.compute.internal",
              "primary": true,
              "association": {
                "publicIp": "00.000.000.000",
                "publicDnsName":
"ec2-000-00-0-000.us-west-2.compute.amazonaws.com",
                "ipOwnerId": "amazon"
              }
            }
          ]
        }
      ],
      "iamInstanceProfile": null,
      "ebsOptimized": false,
      "sriovNetSupport": null
    }
  },
  "notificationCreationTime": "2015-03-19T21:20:36.808Z",
  "messageType": "ConfigurationItemChangeNotification",
  "recordVersion": "1.2"
}
```

Example Configuration Snapshot from AWS Config

AWS Config generates configuration snapshots when you invoke the DeliverConfigSnapshot action or you run the AWS CLI `deliver-config-snapshot` command. AWS Config stores configuration snapshots in the Amazon S3 bucket that you specified when you enabled AWS Config.

The following is an example of the information that AWS Config includes in a configuration snapshot. The snapshot describes the configuration for the resources that AWS Config is recording in the current region for your AWS account, and it describes the relationships between these resources.

Note
The configuration snapshot can include references to resources types and resource IDs that are not supported.

```
{
    "fileVersion": "1.0",
    "requestId": "asudf8ow-4e34-4f32-afeb-0ace5bf3trye",
    "configurationItems": [
        {
            "configurationItemVersion": "1.0",
            "resourceId": "vol-ce676ccc",
            "arn": "arn:aws:us-west-2b:123456789012:volume/vol-ce676ccc",
            "accountId": "12345678910",
            "configurationItemCaptureTime": "2014-03-07T23:47:08.918Z",
            "configurationStateID": "3e660fdf-4e34-4f32-afeb-0ace5bf3d63a",
            "configurationItemStatus": "OK",
            "relatedEvents": [
                "06c12a39-eb35-11de-ae07-adb69edbb1e4",
                "c376e30d-71a2-4694-89b7-a5a04ad92281"
            ],
            "availibilityZone": "us-west-2b",
            "resourceType": "AWS::EC2::Volume",
            "resourceCreationTime": "2014-02-27T21:43:53.885Z",
            "tags": {},
            "relationships": [
                {
                    "resourceId": "i-344c463d",
                    "resourceType": "AWS::EC2::Instance",
                    "name": "Attached to Instance"
                }
            ],
            "configuration": {
                "volumeId": "vol-ce676ccc",
                "size": 1,
                "snapshotId": "",
                "availabilityZone": "us-west-2b",
                "state": "in-use",
                "createTime": "2014-02-27T21:43:53.0885+0000",
                "attachments": [
                    {
                        "volumeId": "vol-ce676ccc",
                        "instanceId": "i-344c463d",
                        "device": "/dev/sdf",
                        "state": "attached",
                        "attachTime": "2014-03-07T23:46:28.0000+0000",
                        "deleteOnTermination": false
                    }
                ],
                "tags": [
                    {
                        "tagName": "environment",
                        "tagValue": "PROD"
                    },
                    {
                        "tagName": "name",
                        "tagValue": "DataVolume1"
                    }
                ],
                "volumeType": "standard"
            }
        },
        {
            "configurationItemVersion": "1.0",
            "resourceId": "i-344c463d",
            "accountId": "12345678910",
            "arn": "arn:aws:ec2:us-west-2b:123456789012:instance/i-344c463d",
            "configurationItemCaptureTime": "2014-03-07T23:47:09.523Z",
            "configurationStateID": "cdb571fa-ce7a-4ec5-8914-0320466a355e",
            "configurationItemStatus": "OK",
            "relatedEvents": [
                "06c12a39-eb35-11de-ae07-adb69edbb1e4",
                "c376e30d-71a2-4694-89b7-a5a04ad92281"
            ],
            "availibilityZone": "us-west-2b",
            "resourceType": "AWS::EC2::Instance",
            "resourceCreationTime": "2014-02-26T22:56:35.000Z",
            "tags": {
                "Name": "integ-test-1",
                "examplename": "examplevalue"
            },
            "relationships": [
                {
                    "resourceId": "vol-ce676ccc",
                    "resourceType": "AWS::EC2::Volume",
                    "name": "Attached Volume"
                },
                {
                    "resourceId": "vol-ef0e06ed",
                    "resourceType": "AWS::EC2::Volume",
                    "name": "Attached Volume",
                    "direction": "OUT"
                },
                {
                    "resourceId": "subnet-47b4cf2c",
                    "resourceType": "AWS::EC2::SUBNET",
                    "name": "Is contained in Subnet",
                    "direction": "IN"
                }
            ],
            "configuration": {
                "instanceId": "i-344c463d",
                "imageId": "ami-ccf297fc",
                "state": {
                    "code": 16,
                    "name": "running"
                },
               "privateDnsName": "ip-172-31-21-63.us-west-2.compute.internal",

                "publicDnsName": "ec2-54-218-4-189.us-west-2.compute.amazon
aws.com",
                "stateTransitionReason": "",
                "keyName": "configDemo",
                "amiLaunchIndex": 0,
                "productCodes": [],
                "instanceType": "t1.micro",
                "launchTime": "2014-02-26T22:56:35.0000+0000",
                "placement": {
                    "availabilityZone": "us-west-2b",
                    "groupName": "",
                    "tenancy": "default"
                },
                "kernelId": "aki-fc8f11cc",
                "monitoring": {
                    "state": "disabled"
                },
                "subnetId": "subnet-47b4cf2c",
                "vpcId": "vpc-41b4cf2a",
                "privateIpAddress": "172.31.21.63",
                "publicIpAddress": "54.218.4.189",
                "architecture": "x86_64",
                "rootDeviceType": "ebs",
                "rootDeviceName": "/dev/sda1",
                "blockDeviceMappings": [
                    {
                        "deviceName": "/dev/sda1",
                        "ebs": {
                            "volumeId": "vol-ef0e06ed",
                            "status": "attached",
                            "attachTime": "2014-02-26T22:56:38.0000+0000",
                            "deleteOnTermination": true
                        }
                    },
                    {
                        "deviceName": "/dev/sdf",
                        "ebs": {
                            "volumeId": "vol-ce676ccc",
                            "status": "attached",
                            "attachTime": "2014-03-07T23:46:28.0000+0000",
                            "deleteOnTermination": false
                        }
                    }
                ],
                "virtualizationType": "paravirtual",
                "clientToken": "aBCDe123456",
                "tags": [
                    {
                        "key": "Name",
                        "value": "integ-test-1"
                    },
                    {
                        "key": "examplekey",
                        "value": "examplevalue"
                    }
                ],
                "securityGroups": [
                    {
                        "groupName": "launch-wizard-2",
                        "groupId": "sg-892adfec"
                    }
                ],
                "sourceDestCheck": true,
                "hypervisor": "xen",
                "networkInterfaces": [
                    {
                        "networkInterfaceId": "eni-55c03d22",
                        "subnetId": "subnet-47b4cf2c",
                        "vpcId": "vpc-41b4cf2a",
                        "description": "",
                        "ownerId": "12345678910",
                        "status": "in-use",
                        "privateIpAddress": "172.31.21.63",
                      "privateDnsName": "ip-172-31-21-63.us-west-2.compute.in
ternal",
                        "sourceDestCheck": true,
                        "groups": [
                            {
                                "groupName": "launch-wizard-2",
                                "groupId": "sg-892adfec"
                            }
                        ],
                        "attachment": {
                            "attachmentId": "eni-attach-bf90c489",
                            "deviceIndex": 0,
                            "status": "attached",
                            "attachTime": "2014-02-26T22:56:35.0000+0000",
                            "deleteOnTermination": true
                        },
                        "association": {
                            "publicIp": "54.218.4.189",
                            "publicDnsName": "ec2-54-218-4-189.us-west-2.com
pute.amazonaws.com",
                            "ipOwnerId": "amazon"
                        },
                        "privateIpAddresses": [
                            {
                                "privateIpAddress": "172.31.21.63",
                                "privateDnsName": "ip-172-31-21-63.us-west-
2.compute.internal",
                                "primary": true,
                                "association": {
                                    "publicIp": "54.218.4.189",
                                   "publicDnsName": "ec2-54-218-4-189.us-west-
2.compute.amazonaws.com",
                                    "ipOwnerId": "amazon"
                                }
                            }
                        ]
                    }
                ],
                "ebsOptimized": false
            }
        }
    ]
}
```

Example Amazon EBS Configuration History from AWS Config

AWS Config generates a set of files that each represent a resource type and lists all configuration changes for the resources of that type that AWS Config is recording. AWS Config exports this resource-centric configuration history as an object in the Amazon S3 bucket that you specified when you enabled AWS Config. The configuration history file for each resource type contains the changes that were detected for the resources of that type since the last history file was delivered. History files are typically delivered every six hours.

The following is an example of the contents of the Amazon S3 object that describes the configuration history of all the Amazon Elastic Block Store volumes in the current region for your AWS account. The volumes in this account include vol-ce676ccc and vol-cia007c . Volume vol-ce676ccc had two configuration changes since the previous history file was delivered while volume vol-cia007c had one.

```
{
    "fileVersion": "1.0",
    "requestId": "asudf8ow-4e34-4f32-afeb-0ace5bf3trye",
    "configurationItems": [
        {
            "snapshotVersion": "1.0",
            "resourceId": "vol-ce676ccc",
            "arn": "arn:aws:us-west-2b:123456789012:volume/vol-ce676ccc",
            "accountId": "12345678910",
            "configurationItemCaptureTime": "2014-03-07T23:47:08.918Z",
            "configurationStateID": "3e660fdf-4e34-4f32-afeb-0ace5bf3d63a",
            "configurationItemStatus": "OK",
            "relatedEvents": [
                "06c12a39-eb35-11de-ae07-adb69edbb1e4",
                "c376e30d-71a2-4694-89b7-a5a04ad92281"
            ],
            "availibilityZone": "us-west-2b",
            "resourceType": "AWS::EC2::Volume",
            "resourceCreationTime": "2014-02-27T21:43:53.885Z",
            "tags": {},
            "relationships": [
                {
                    "resourceId": "i-344c463d",
                    "resourceType": "AWS::EC2::Instance",
                    "name": "Attached to Instance"
                }
            ],
            "configuration": {
                "volumeId": "vol-ce676ccc",
                "size": 1,
                "snapshotId": "",
                "availabilityZone": "us-west-2b",
                "state": "in-use",
                "createTime": "2014-02-27T21:43:53.0885+0000",
                "attachments": [
                    {
                        "volumeId": "vol-ce676ccc",
                        "instanceId": "i-344c463d",
                        "device": "/dev/sdf",
                        "state": "attached",
                        "attachTime": "2014-03-07T23:46:28.0000+0000",
                        "deleteOnTermination": false
                    }
                ],
                "tags": [
                    {
                        "tagName": "environment",
                        "tagValue": "PROD"
                    },
                    {
                        "tagName": "name",
                        "tagValue": "DataVolume1"
                    }
                ],
                "volumeType": "standard"
            }
        },
        {
            "configurationItemVersion": "1.0",
            "resourceId": "vol-ce676ccc",
            "arn": "arn:aws:us-west-2b:123456789012:volume/vol-ce676ccc",
            "accountId": "12345678910",
            "configurationItemCaptureTime": "2014-03-07T21:47:08.918Z",
            "configurationItemState": "3e660fdf-4e34-4f32-sseb-0ace5bf3d63a",
            "configurationItemStatus": "OK",
            "relatedEvents": [
                "06c12a39-eb35-11de-ae07-ad229edbb1e4",
                "c376e30d-71a2-4694-89b7-a5a04w292281"
            ],
            "availibilityZone": "us-west-2b",
            "resourceType": "AWS::EC2::Volume",
            "resourceCreationTime": "2014-02-27T21:43:53.885Z",
            "tags": {},
            "relationships": [
                {
                    "resourceId": "i-344c463d",
                    "resourceType": "AWS::EC2::Instance",
                    "name": "Attached to Instance"
                }
            ],
            "configuration": {
                "volumeId": "vol-ce676ccc",
                "size": 1,
                "snapshotId": "",
                "availabilityZone": "us-west-2b",
                "state": "in-use",
                "createTime": "2014-02-27T21:43:53.0885+0000",
                "attachments": [
                    {
                        "volumeId": "vol-ce676ccc",
                        "instanceId": "i-344c463d",
                        "device": "/dev/sdf",
                        "state": "attached",
                        "attachTime": "2014-03-07T23:46:28.0000+0000",
                        "deleteOnTermination": false
                    }
                ],
                "tags": [
                    {
                        "tagName": "environment",
                        "tagValue": "PROD"
                    },
                    {
                        "tagName": "name",
                        "tagValue": "DataVolume1"
                    }
                ],
                "volumeType": "standard"
            }
        },
        {
            "configurationItemVersion": "1.0",
            "resourceId": "vol-cia007c",
            "arn": "arn:aws:us-west-2b:123456789012:volume/vol-cia007c",
            "accountId": "12345678910",
            "configurationItemCaptureTime": "2014-03-07T20:47:08.918Z",
            "configurationItemState": "3e660fdf-4e34-4f88-sseb-0ace5bf3d63a",
            "configurationItemStatus": "OK",
            "relatedEvents": [
                "06c12a39-eb35-11de-ae07-adjhk8edbb1e4",
                "c376e30d-71a2-4694-89b7-a5a67u292281"
            ],
            "availibilityZone": "us-west-2b",
            "resourceType": "AWS::EC2::Volume",
            "resourceCreationTime": "2014-02-27T20:43:53.885Z",
            "tags": {},
            "relationships": [
                {
                    "resourceId": "i-344e563d",
                    "resourceType": "AWS::EC2::Instance",
                    "name": "Attached to Instance"
                }
            ],
            "configuration": {
                "volumeId": "vol-cia007c",
                "size": 1,
                "snapshotId": "",
                "availabilityZone": "us-west-2b",
                "state": "in-use",
                "createTime": "2014-02-27T20:43:53.0885+0000",
                "attachments": [
                    {
                        "volumeId": "vol-cia007c",
                        "instanceId": "i-344e563d",
                        "device": "/dev/sdf",
                        "state": "attached",
                        "attachTime": "2014-03-07T23:46:28.0000+0000",
                        "deleteOnTermination": false
                    }
                ],
                "tags": [
                    {
                        "tagName": "environment",
                        "tagValue": "PROD"
                    },
                    {
                        "tagName": "name",
                        "tagValue": "DataVolume2"
                    }
                ],
                "volumeType": "standard"
            }
        }
    ]
}
```

Using Amazon SQS to Monitor AWS Resource Changes

AWS Config uses Amazon Simple Notification Service (SNS) to send you notifications every time a supported AWS resource is created, updated, or otherwise modified as a result of user API activity. However, you might be interested in only certain resource configuration changes. For example, you might consider it critical to know when someone modifies the configuration of a security group, but not need to know every time there is a change to tags on your Amazon EC2 instances. Or, you might want to write a program that performs specific actions when specific resources are updated. For example, you might want to start a certain workflow when a security group configuration is changed. If you want to programmatically consume the data from AWS Config in these or other ways, use an Amazon Simple Queue Service queue as the notification endpoint for Amazon SNS.

Note
Notifications can also come from Amazon SNS in the form of an email, a Short Message Service (SMS) message to SMS-enabled mobile phones and smartphones, a notification message to an application on a mobile device, or a notification message to one or more HTTP or HTTPS endpoints.

You can have a single SQS queue subscribe to multiple topics, whether you have one topic per region or one topic per account per region. You must subscribe the queue to your desired SNS topic. (You can subscribe multiple queues to one SNS topic.) For more information, see Sending Amazon SNS Messages to Amazon SQS Queues.

Permissions for Amazon SQS

To use Amazon SQS with AWS Config, you must configure a policy that grants permissions to your account to perform all actions that are allowed on an SQS queue. The following example policy grants the account number 111122223333 and account number 444455556666 permission to send messages pertaining to each configuration change to the queue named arn:aws:sqs:us-east-1:444455556666:queue1.

```
{
  "Version": "2012-10-17",
  "Id": "Queue1_Policy_UUID",
  "Statement":
    {
       "Sid":"Queue1_SendMessage",
       "Effect": "Allow",
       "Principal": {
            "AWS": ["111122223333","444455556666"]
         },
        "Action": "sqs:SendMessage",
        "Resource": "arn:aws:sqs:us-east-1:444455556666:queue1"
     }
}
```

You must also create a policy that grants permissions for connections between an SNS topic and the SQS queue that subscribes to that topic. The following is an example policy that permits the SNS topic with the Amazon Resource Name (ARN) arn:aws:sns:us-east-1:111122223333:test-topic to perform any actions on the queue named arn:aws:sqs:us-east-1:111122223333:test-topic-queue.

Note
The account for the SNS topic and the SQS queue must be in the same region.

```
{
  "Version": "2012-10-17",
  "Id": "SNStoSQS",
  "Statement":
     {
        "Sid":"rule1",
        "Effect": "Allow",
        "Principal": "*",
        "Action": "sqs:*",
        "Resource": "arn:aws:sqs:us-east-1:111122223333:test-topic-queue",
        "Condition" : {
            "StringEquals" : {
            "aws:SourceArn":"arn:aws:sns:us-east-1:111122223333:test-topic"
            }
        }
     }
}
```

Each policy can include statements that cover only a single queue, not multiple queues. For information about other restrictions on Amazon SQS policies, see Special Information for Amazon SQS Policies.

Managing AWS Config

At any time, you can change the settings for your IAM role and modify or delete your delivery channel (that is, the Amazon Simple Storage Service bucket and the Amazon Simple Notification Service topic). You can start or stop the configuration recorder associated with your account, and you can customize which types of resources are recorded.

Topics

Updating Your Delivery Channel

When you configure your delivery channel for the first time, AWS Config automatically assigns the name `default` to your delivery channel.

You can specify only one delivery channel per account. If you want to create a new delivery channel, you must first delete the existing delivery channel. For more information, see Deleting a Delivery Channel (p. 74).

You can update the delivery channel by either updating both the SNS topic and the S3 bucket, or updating just one of them.

To update your SNS topic, you can specify the name of an existing SNS topic associated with your account, specify the name of an existing SNS topic associated with a different account, or create a new SNS topic. If you want to choose SNS topic in a different account, make sure that the topic has policies that grants access permissions to AWS Config. For more information, see Permissions for the Amazon SNS Topic (p. 84).

Similarly, to update your S3 bucket, you can specify the name of an existing S3 bucket associated with your account, specify the name of an existing S3 bucket associated with a different account, or create a new S3 bucket. If you want to use S3 bucket in a different account, make sure that the bucket has policies that grants access permissions to AWS Config. For more information, see Permissions for the Amazon S3 Bucket (p. 82).

Updating a Delivery Channel Using the Console

The AWS Config console does not support changing the name of your delivery channel. You can however update the S3 bucket and the SNS topic associated with your delivery channel using the console.

To update your delivery channel

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Click the gear icon to see the **Set Up AWS Config** page.
3. Follow steps 4 through 6 in Set Up AWS Config Using the Console (p. 15).

Updating a Delivery Channel Using the CLI

You can also update your S3 bucket and your SNS topic using the AWS Command Line Interface.

Use the `put-delivery-channel` command by specifying one or more of the following options to update your delivery channel. If you do not specify a name, AWS Config automatically assigns the name `default` for your delivery channel.

- `name` – Your delivery channel name
- `s3BbucketName` – Your S3 bucket name
- `snsTopicARN` – The Amazon Resource Name (ARN) of your SNS topic

Your command should look like the following example:

> **Note**
> The following example uses JSON syntax supported on a Linux, OS X, Unix, and Windows Powershell to specify the attributes. For information about specifying JSON syntax on your operating system, see Quoting Strings.

```
$ aws configservice put-delivery-channel --delivery-channel '{"name":"my-test-
dc","s3BucketName":"my-config-bucket","snsTopicARN":"arn:aws:sns:us-east-
1:012345678912:my-config-notice"}'
```

Use the `describe-delivery-channels` command to verify that the delivery channel is updated with the new values; for example:

```
$ aws configservice describe-delivery-channels
{
    "DeliveryChannels": [
        {
          "snsTopicARN": "arn:aws:sns:us-east-1:012345678912:my-config-notice",

            "name": "my-test-dc",
            "s3BucketName": "my-config-bucket"
        }
    ]
}
```

Deleting a Delivery Channel

Currently, you can specify only one delivery channel per account. If you want to create a new delivery channel, you must first delete the existing delivery channel. Currently, AWS Config console does not support deleting delivery channel.

In the CLI, use `delete-delivery-channel` command to delete your delivery channel; for example:

```
$ aws configservice delete-delivery-channel --delivery-channel-name default
```

Updating an IAM Role

You can at any time update the IAM role assumed by AWS Config. Before you update the IAM role, ensure that you have created a new role to replace the old one and have attached the policies that grant permissions to AWS Config to record configurations and deliver them to your delivery channel. In addition, make sure to copy the Amazon Resource Name (ARN) of your new IAM role. You will need it to update the IAM role. For information about creating an IAM role and attaching the required policies to the IAM role, see Create an IAM Role (p. 19).

Tip
To find the ARN of an existing IAM role, go to the IAM console at https://console.aws.amazon.com/iam/. Click **Roles** in the navigation pane. Then click the name of the desired role and find the ARN at the top of the **Summary** page.

Updating an IAM Role Using the Console

You can update your IAM role using the AWS Management Console.

To update IAM role using the console

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Click the gear icon to see the **Set Up AWS Config** page. Then click **Continue**.
3. In the **AWS Config is requesting permissions to read your resources' configuration** page, click **View Details**.
4. For **IAM Role**, select a new role that matches your requirements. Or select **Create a new IAM Role** to have AWS Config create one for you.
5. Do one of the following:
   - If you chose to create a new role, type a name for **Role Name**.
   - If you chose an existing policy, select an available policy from **Policy Name**. Or select **Create a new Role Policy**.
6. Click **Allow**.

Updating an IAM Role Using the CLI

When you configure your configuration recorder for the first time, AWS Config automatically assigns the name `default` to your configuration recorder. You cannot change the name assigned to your configuration recorder. You can however, change the IAM role associated with the configuration recorder.

Currently, AWS Config supports one configuration recorder per account. You cannot delete the configuration recorder.

To update IAM role using the CLI

Use the `put-configuration-recorder` command and specify the Amazon Resource Name (ARN) of the new role. Make sure to use `default` for the `name` option.

Your command should look like the following example:

```
$ aws configservice put-configuration-recorder --configuration-recorder
name=default,roleARN=arn:aws:iam::012345678912:role/myConfigRole
```

Stopping or Starting the Configuration Recorder

The configuration recorder records the configurations of the supported AWS resources in your account. You must create a configuration recorder before you can start recording.

If you turn on the configuration recorder from the console or the AWS CLI, AWS Config automatically creates and then starts the configuration recorder for you.

As you work with the configuration recorder, note the following:

- By default, the configuration recorder records all supported resources in the region where AWS Config is running. You can create a customized configuration recorder that records only the resource types that you specify. For more information, see Selecting Which Resources AWS Config Records (p. 76).
- You are charged service usage fees when AWS Config starts recording the configurations. You can stop and restart the configuration recorder at any time.
- You can start or stop the recorder using the console or the CLI.

Stopping Recording

You can stop recording of the configurations of your AWS resources by stopping the configuration recorder at any time. When you stop the configuration recorder, you will not be charged AWS Config usage fees. You can continue to access the configurations that are already recorded. You can start the recorder at any time.

To use the console to stop the recorder

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. On the **Resource inventory** page, under **Recording is on**, click **Turn off**.
3. In the confirmation dialog box, click **Continue**.

To use the CLI to stop the recorder

- Use the `stop-configuration-recorder` command to stop recording configuration changes; for example:

```
$ aws configservice stop-configuration-recorder --configuration-recorder-
name default
```

Starting Recording

You can restart a stopped recorder at any time. When you start a stopped recorder, AWS Config starts by taking inventory of all AWS resources existing in your account. Configurations that were recorded before you stopped the recorder continue to be accessible.

To use the console to start the recorder

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. On the **Resource inventory** page, under **Recording is off**, click **Turn on**.
3. In the confirmation dialog box, click **Continue**.

To use the CLI to start the recorder

- Use the `start-configuration-recorder` command to start recording configuration changes; for example:

```
$ aws configservice start-configuration-recorder --configuration-recorder-
name default
```

Selecting Which Resources AWS Config Records

You can customize AWS Config to record configuration changes for all supported types of resources or for only those types that are relevant to you. To learn which types of resources AWS Config can record, see Supported AWS Resource Types (p. 6).

AWS Config can record all supported resource types

By default, AWS Config records the configuration changes for all supported types of *regional resources* that AWS Config discovers in the region in which it is running. Regional resources are tied to a region and can be used only in that region. Examples of regional resources are EC2 instances and EBS volumes.

You can also have AWS Config record supported types of *global resources*. Global resources are not tied to a specific individual region and can be used in all regions. The global resource types that AWS Config supports are IAM users, groups, roles, and customer managed polices.

Important
The configuration details for a specific global resource are the same in all regions. If you customize AWS Config in multiple regions to record global resources, AWS Config creates multiple configuration items each time a global resource changes: one configuration item for each region. These configuration items will contain identical data. To prevent duplicate configuration items, you should consider customizing AWS Config in only one region to record global resources, unless you want the configuration items to be available in multiple regions.

AWS Config can record specific resource types

If you don't want AWS Config to record all resources, you can specify which types of resources it records.

If AWS Config records only specific types of resources, the information provided for recorded resources is not affected by missing data for nonrecorded resources. For example, if a recorded resource is related to a nonrecorded resource, that relationship information is provided, but only when you view details for the recorded resource.

You can stop AWS Config from recording a type of resource at any time. After AWS Config stops recording a resource, it retains the configuration information that was previously captured, and you can continue to access this information.

If some types are not being recorded, the AWS Config console provides details pages for nonrecorded resources (except for nonrecorded global resources). The details page for a nonrecorded resource provides some static information, but it provides null values for most configuration details, and it does not provide information about relationships and configuration changes.

Selecting Resources (AWS Config Console)

You can use the AWS Config console to select the types of resources that AWS Config records.

To select resources

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. On the **Resource inventory** page, choose the gear icon (⚙). The **Set Up AWS Config** page appears.
3. In the **Resource types to record** section, specify which types of AWS resources you want AWS Config to record:
   - **All resources** – AWS Config records all supported resources with the following options:
     - **Record all resources supported in this region** – AWS Config records configuration changes for every supported type of regional resource. When AWS Config adds support for a new type of regional resource, it automatically starts recording resources of that type.
     - **Include global resources** – AWS Config includes supported types of global resources with the resources that it records (for example, IAM resources). When AWS Config adds support for a new type of global resource, it automatically starts recording resources of that type.
   - **Specific types** – AWS Config records configuration changes for only those types of AWS resources that you specify.
4. Choose **Continue**.
5. In the **AWS Config is requesting permissions to read your resources' configuration** page, click **Allow**.

Selecting Resources (AWS CLI)

You can use the AWS CLI to select the types of resources that you want AWS Config to record. You do this by creating a configuration recorder, which records the types of resources that you specify in a recording group. In the recording group, you specify whether all supported types or specific types of resources are recorded.

To select all supported resources

1. Use the following `put-configuration-recorder` command:

   ```
   $ aws configservice put-configuration-recorder --configuration-recorder
   name=default,roleARN=arn:aws:iam::123456789012:role/config-role --recording-
   group allSupported=true,includeGlobalResourceTypes=true
   ```

   This command uses the following options for the `--recording-group` parameter:

   - `allSupported=true` – AWS Config records configuration changes for every supported type of *regional resource*. When AWS Config adds support for a new type of regional resource, it automatically starts recording resources of that type.
   - `includeGlobalResourceTypes=true` – AWS Config includes supported types of global resources with the resources that it records. When AWS Config adds support for a new type of global resource, it automatically starts recording resources of that type.

     Before you can set this option to `true`, you must set the `allSupported` option to `true`.

     If you do not want to include global resources, set this option to `false`, or omit it.

2. (Optional) To verify that your configuration recorder has the settings that you want, use the following `describe-configuration-recorders` command:

   ```
   $ aws configservice describe-configuration-recorders
   ```

   The following is an example response:

   ```
   {
       "ConfigurationRecorders": [
           {
               "recordingGroup": {
                   "allSupported": true,
                   "resourceTypes": [],
                   "includeGlobalResourceTypes": true
               },
               "roleARN": "arn:aws:iam::123456789012:role/config-role",
               "name": "default"
           }
       ]
   }
   ```

To select specific types of resources

1. Use the `aws configservice` `put-configuration-recorder` command, and pass one or more resource types through the `--recording-group` option, as shown in the following example:

   ```
   $ aws configservice put-configuration-recorder --configuration-recorder
   name=default,roleARN=arn:aws:iam::012345678912:role/myConfigRole --recording-
   group file://recordingGroup.json
   ```

`recordingGroup.json` is a JSON file that specifies which types of resources AWS Config will record:

```
{
  "allSupported": false,
  "includeGlobalResourceTypes": false,
  "resourceTypes": [
    "AWS::EC2::EIP",
    "AWS::EC2::Instance",
    "AWS::EC2::NetworkAcl",
    "AWS::EC2::SecurityGroup",
    "AWS::CloudTrail::Trail",
    "AWS::EC2::Volume",
    "AWS::EC2::VPC",
    "AWS::IAM::User",
    "AWS::IAM::Policy"
  ]
}
```

Before you can specify resource types for the `resourceTypes` key, you must set the `allSupported` and `includeGlobalResourceTypes` options to false or omit them.

2. (Optional) To verify that your configuration recorder has the settings that you want, use the following `describe-configuration-recorders` command:

```
$ aws configservice describe-configuration-recorders
```

The following is an example response:

```
{
    "ConfigurationRecorders": [
        {
            "recordingGroup": {
                "allSupported": false,
                "resourceTypes": [
                    "AWS::EC2::EIP",
                    "AWS::EC2::Instance",
                    "AWS::EC2::NetworkAcl",
                    "AWS::EC2::SecurityGroup",
                    "AWS::CloudTrail::Trail",
                    "AWS::EC2::Volume",
                    "AWS::EC2::VPC",
                    "AWS::IAM::User",
                    "AWS::IAM::Policy"
                ],
                "includeGlobalResourceTypes": false
            },
            "roleARN": "arn:aws:iam::123456789012:role/config-role",
            "name": "default"
        }
    ]
}
```

Permissions for AWS Config

To get the most out of AWS Config, you need to create permissions policies to attach to your IAM role, your Amazon Simple Storage Service (S3) bucket, and your Amazon Simple Notification Service (SNS) topic. A policy is a set of statements that grants AWS Config permissions. The following topics provide examples of recommended IAM policies to use with the AWS Config console and the AWS Command Line Interface.

Topics

Permissions for the AWS Config IAM Role

An AWS Identity and Access Management (IAM) role lets you define a set of permissions. AWS Config assumes the role that you assign to it to make read or write requests to the delivery channel that you specify and to get configuration details for supported AWS resources. For more information on IAM roles, see IAM Roles in the *IAM User Guide*.

AWS Config uses an IAM role for permission to access your S3 bucket and your SNS topic as well as for permissions to `Describe` or `List` APIs. If you are using the console to set up AWS Config, an IAM role with all the required permissions is automatically created for you.

If you are using the CLI to set up AWS Config or you are updating an existing IAM role, you must create separate policies for your S3 bucket and SNS topic. You also need to create a policy for the `Describe` APIs and attach it to the IAM role.

You use the following policies to attach to your IAM role.

Adding an IAM Trust Policy to your Role

You can create an IAM trust policy that enables AWS Config to assume a role and use it to track your resources. For more information about trust policies, see Assuming a Role.

Here is an example trust policy for AWS Config roles:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Creating IAM Role Policies

Follow the examples below to create policies for an S3 bucket, an SNS topic, and `Describe` APIs.

IAM Role Policy for Amazon S3 Bucket

Use this example policy as a model for granting AWS Config permissions to access your Amazon S3 bucket:

```
{
  "Version": "2012-10-17",
  "Statement":
   [

     {
       "Effect": "Allow",
       "Action": ["s3:PutObject"],
      "Resource": ["arn:aws:s3::: yourS3bucketname/prefix/AWSLogs/yourAccountID-
withoutHypens/*"],
       "Condition":
        {
          "StringLike":
            {
              "s3:x-amz-acl": "bucket-owner-full-control"
            }
        }
     },
     {
       "Effect": "Allow",
       "Action": ["s3:GetBucketAcl"],
       "Resource": "arn:aws:s3::: yourS3bucketname "
     }
  ]
  }
```

IAM Role Policy for Amazon SNS Topic

Use this example policy as a model for granting AWS Config permissions to access your SNS topic:

```
{
  "Version": "2012-10-17",
  "Statement":
   [
     {
      "Effect":"Allow",
      "Action":"sns:Publish",
      "Resource":"yourSNStopicARN"
     }
    ]
}
```

IAM Role Policy for Getting Configuration Details

To record your AWS resource configurations, AWS Config requires IAM permissions to get the configuration details about your resources. The required permissions change each time AWS Config expands the types of AWS resources that it supports. To grant a managed set of permissions, attach the AWS managed policy `AWSConfigRole` to the IAM role that you assign to AWS Config. AWS updates this policy each time AWS Config adds support for another type of AWS resource.

To attach the `AWSConfigRole` policy to an IAM role, use the `attach-role-policy` CLI command and specify the Amazon Resource Name (ARN) for `AWSConfigRole`:

```
$ aws iam attach-role-policy --role-name myConfigRole --policy-arn
arn:aws:iam::aws:policy/service-role/AWSConfigRole
```

Permissions for the Amazon S3 Bucket

By default, all Amazon S3 buckets and objects are private. Only the resource owner and the AWS account that created the bucket can access that bucket and any objects it contains. The resource owner can, however, choose to grant access permissions to other resources and users. One way to do this is to write an access policy.

If AWS Config creates an S3 bucket for you automatically (for example, if you use the AWS Config console or use the `aws config subscribe` command to set up your delivery channel) or you choose an existing S3 bucket already existing in your account, these permissions are automatically added to the S3 bucket. However, if you specify an existing S3 bucket from another account, you must ensure that the S3 bucket has the correct permissions.

Required permissions for an Amazon S3 bucket in another account

When AWS Config sends configuration information (history files and snapshots) to the Amazon S3 bucket in your account, it assumes the IAM role that you assigned when you set up AWS Config. When AWS Config sends to an S3 bucket in another account, it first attempts to use the IAM role, but this attempt fails if the access policy for the bucket does not grant `WRITE` access to the IAM role. In this event, AWS Config sends the information again, this time as the AWS Config service principal. Before the delivery can succeed, the access policy must grant `WRITE` access to the `config.amazonaws.com` principal name. AWS Config is then the owner of the objects it delivers to the S3 bucket.

Granting AWS Config access to an Amazon S3 bucket in another account

Follow these steps to add an access policy to an Amazon S3 bucket in another account. The access policy allows AWS Config to send configuration information to the bucket.

1. Sign in to the AWS Management Console using the account that has the S3 bucket.
2. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
3. Select the bucket that you want AWS Config to use to deliver configuration items, and then choose **Properties**.
4. Choose **Permissions**.
5. Choose **Edit Bucket Policy**.
6. Copy the following policy into the **Bucket Policy Editor** window:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSConfigBucketPermissionsCheck",
      "Effect": "Allow",
      "Principal": {
        "Service": [
         "config.amazonaws.com"
        ]
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::targetBucketName"
    },
    {
      "Sid": " AWSConfigBucketDelivery",
      "Effect": "Allow",
      "Principal": {
        "Service": [
         "config.amazonaws.com"
        ]
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::targetBucketName/[optional] prefix/AWS
Logs/sourceAccountID-WithoutHyphens/Config/*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    }
  ]
}
```

7. Substitute the following values in the bucket policy:

   - *targetBucketName* – The name of the Amazon S3 bucket to which AWS Config will deliver configuration items.
   - *[optional] prefix* – An optional addition to the Amazon S3 object key that helps create a folder-like organization in the bucket.

- *sourceAccountID-WithoutHyphens* – The ID of the account for which AWS Config will deliver configuration items to the target bucket.

8. Choose **Save** and then **Close**.

Permissions for the Amazon SNS Topic

AWS Config must have permissions to send notifications to an SNS topic. If you use the AWS Config console to set up your delivery channel with a new SNS topic or you choose an SNS topic that already exists in your account, these permissions are automatically added to the SNS topic. However, if you specify an existing topic from another account or you set up your delivery channel using the API, you must make sure that the topic has the correct permissions.

The following example shows the permissions that are automatically created by AWS Config for a new topic. This policy statement allows AWS Config to publish to a specified Amazon SNS topic.

If you want to use an existing SNS topic from another account or you set up your delivery channel using the API, make sure to attach the following policy to the SNS topic.

```
{
  "Id": "Policy1415489375392",
  "Statement": [
    {
      "Sid": "AWSConfigSNSPolicy20150201",
      "Action": [
        "SNS:Publish"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:sns:region:account-id:myTopic",
      "Principal": {
        "Service": [
          "config.amazonaws.com"
        ]
      }
    }
  ]
}
```

For the `Resource` key, *account-id* is the account number of the topic owner; in topics that you create, this will be your account number. You must substitute appropriate values for *region* and *myTopic*.

Recommended IAM Permissions for Using the AWS Config Console and the AWS CLI

When you give permissions to IAM users to use the AWS Config console or the AWS CLI, you can (and should) restrict their permissions to the least amount of access that they need to do their job.

In most cases, you'll want permissions that cover these common uses:

- Setting up and managing AWS Config (full-access permissions)
- Using AWS Config (read-only permissions)

Full-access permissions

Users who set up and manage AWS Config must have full-access permissions. With full-access permissions, you can perform essential setup tasks such as the following:

- Provide Amazon S3 and Amazon SNS endpoints to deliver data to
- Create the role that gets provided to AWS Config
- Turn recording on and off

Read-only permissions

Users who use AWS Config but don't need to set it up should have read-only permissions. For example, these permissions are useful for users who look up the configurations of resources at various times or who search for resources by tags.

To grant read-only access to an AWS Config user, attach the AWS managed policy `AWSConfigUserAccess` to the user's IAM user or group.

Example full-access policy

The following example policy grants full-access permissions for using AWS Config.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:AddPermission",
        "sns:CreateTopic",
        "sns:DeleteTopic",
        "sns:GetTopicAttributes",
        "sns:ListPlatformApplications",
        "sns:ListTopics",
        "sns:SetTopicAttributes"
      ],
      "Resource": "arn:aws:sns:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketNotification",
        "s3:GetBucketPolicy",
        "s3:GetBucketRequestPayment",
        "s3:GetBucketVersioning",
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:PutBucketPolicy"
      ],
      "Resource": "arn:aws:s3:::*"
   },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies",
        "iam:ListRoles",
        "iam:PassRole",
        "iam:PutRolePolicy",
        "iam:AttachRolePolicy",
        "iam:CreatePolicy",
        "iam:CreatePolicyVersion"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "config:*",
        "tag:Get*"
      ],
      "Resource": "*"
    }
  ]
}
```

Logging AWS Config API Calls with AWS CloudTrail

AWS Config is integrated with AWS CloudTrail, a service that captures API calls made by or on behalf of AWS services in your AWS account and delivers the log files to an Amazon Simple Storage Service (S3) bucket that you specify. CloudTrail captures all API calls from the AWS Config console or from the AWS Config API. Using the information collected by CloudTrail, you can determine what request was made to AWS Config, the source IP address from which the request was made, 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.

AWS Config Information in CloudTrail

When enabled in your AWS account, CloudTrail tracks API calls made to AWS Config. AWS Config records are written together with other AWS service records in a log file. CloudTrail determines when to create and write to a new file based on a time period and file size.

All of the AWS Config actions are logged and are documented in the Actions topic in the *AWS Config API Reference*. For example, calls to the `DeliverConfigSnapshot`, `DeleteDeliveryChannel`, and `DescribeDeliveryChannels` actions generate entries in the CloudTrail log files.

Every log entry contains information about who generated the request. The user identity information in the log 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, see the **userIdentity** field in the CloudTrail Event Reference.

By default, your log files are encrypted by using Amazon S3 server-side encryption (SSE). 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. For information on setting up lifecycle rules, see Managing Lifecycle Configuration in the *Amazon Simple Storage Service Console User Guide*.

If you want to take quick action upon log file delivery, you can choose to have CloudTrail publish Amazon SNS notifications when new log files are delivered. For more information, see Configuring Amazon SNS Notifications.

You can also aggregate AWS Config log files from multiple AWS regions and multiple AWS accounts into a single S3 bucket. For more information, see Aggregating CloudTrail Log Files to a Single Amazon S3 Bucket.

Understanding AWS Config Log File Entries

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

Following are examples of CloudTrail log files for all AWS Config actions.

DeleteDeliveryChannel

Following is an example CloudTrail log file for the `DeleteDeliveryChannel` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:32:57Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "DeleteDeliveryChannel",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "10.24.34.0",
      "userAgent": "aws-internal/3",
      "requestParameters": {
        "deliveryChannelName": "default"
      },
      "responseElements": null,
      "requestID": "207d695a-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "5dcff7a9-e414-411a-a43e-88d122a0ad4a",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

DeliverConfigSnapshot

Following is an example CloudTrail log file for the `DeliverConfigSnapshot` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDAABCDEFGHIJKLNMOPQ:Config-API-Test",
       "arn": "arn:aws:sts::111111111111:assumed-role/JaneDoe/Config-API-Test",

        "accountId": "111111111111",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2014-12-11T00:58:42Z"
          },
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AIDAABCDEFGHIJKLNMOPQ",
            "arn": "arn:aws:iam::111111111111:role/JaneDoe",
            "accountId": "111111111111",
            "userName": "JaneDoe"
          }
        }
      },
      "eventTime": "2014-12-11T00:58:53Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "DeliverConfigSnapshot",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "10.24.34.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": {
        "deliveryChannelName": "default"
      },
      "responseElements": {
        "configSnapshotId": "58d50f10-212d-4fa4-842e-97c614da67ce"
      },
      "requestID": "e0248561-80d0-11e4-9f1c-7739d36a3df2",
      "eventID": "3e88076c-eae1-4aa6-8990-86fe52aedbd8",
      "eventType": "AwsApiCall",
      recipientAccountId": "111111111111"
    }
```

DescribeConfigurationRecorderStatus

Following is an example CloudTrail log file for the `DescribeConfigurationRecorderStatus` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:35:44Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "DescribeConfigurationRecorderStatus",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": null,
      "responseElements": null,
      "requestID": "8442f25d-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "a675b36b-455f-4e18-a4bc-d3e01749d3f1",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

DescribeConfigurationRecorders

Following is an example CloudTrail log file for the `DescribeConfigurationRecorders` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:34:52Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "DescribeConfigurationRecorders",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": null,
      "responseElements": null,
      "requestID": "6566b55c-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "6259a9ad-889e-423b-beeb-6e1eec84a8b5",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

DescribeDeliveryChannels

Following is an example CloudTrail log file for the `DescribeDeliveryChannels` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:35:02Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "DescribeDeliveryChannels",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": null,
      "responseElements": null,
      "requestID": "6b6aee3f-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "3e15ebc5-bf39-4d2a-8b64-9392807985f1",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

GetResourceConfigHistory

Following is an example CloudTrail log file for the `GetResourceConfigHistory` action.

```

{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDAABCDEFGHIJKLNMOPQ:Config-API-Test",
       "arn": "arn:aws:sts::111111111111:assumed-role/JaneDoe/Config-API-Test",

        "accountId": "111111111111",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2014-12-11T00:58:42Z"
          },
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AIDAABCDEFGHIJKLNMOPQ",
            "arn": "arn:aws:iam::111111111111:role/JaneDoe",
            "accountId": "111111111111",
            "userName": "JaneDoe"
          }
        }
      },
      "eventTime": "2014-12-11T00:58:42Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "GetResourceConfigHistory",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "10.24.34.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": {
        "resourceId": "vpc-a12bc345",
        "resourceType": "AWS::EC2::VPC",
        "limit": 0,
        "laterTime": "Dec 11, 2014 12:58:42 AM",
        "earlierTime": "Dec 10, 2014 4:58:42 PM"
      },
      "responseElements": null,
      "requestID": "d9f3490d-80d0-11e4-9f1c-7739d36a3df2",
      "eventID": "ba9c1766-d28f-40e3-b4c6-3ffb87dd6166",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111111111111"
      }
```

PutConfigurationRecorder

Following is an example CloudTrail log file for the `PutConfigurationRecorder` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:35:23Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "PutConfigurationRecorder",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": {
        "configurationRecorder": {
          "name": "default",
          "roleARN": "arn:aws:iam::222222222222:role/config-role-pdx"
        }
      },
      "responseElements": null,
      "requestID": "779f7917-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "c91f3daa-96e8-44ee-8ddd-146ac06565a7",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

PutDeliveryChannel

Following is an example CloudTrail log file for the `PutDeliveryChannel` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:33:08Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "PutDeliveryChannel",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": {
        "deliveryChannel": {
          "name": "default",
          "s3BucketName": "config-api-test-pdx",
          "snsTopicARN": "arn:aws:sns:us-west-2:222222222222:config-api-test-
pdx"
        }
      },
      "responseElements": null,
      "requestID": "268b8d4d-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "b2db05f1-1c73-4e52-b238-db69c04e8dd4",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

StartConfigurationRecorder

Following is an example CloudTrail log file for the `StartConfigurationRecorder` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:35:34Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "StartConfigurationRecorder",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": {
        "configurationRecorderName": "default"
      },
      "responseElements": null,
      "requestID": "7e03fa6a-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "55a5507f-f306-4896-afe3-196dc078a88d",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

StopConfigurationRecorder

Following is an example CloudTrail log file for the `StopConfigurationRecorder` action.

```
{
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "JohnDoe"
      },
      "eventTime": "2014-12-11T18:35:13Z",
      "eventSource": "config.amazonaws.com",
      "eventName": "StopConfigurationRecorder",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
      "requestParameters": {
        "configurationRecorderName": "default"
      },
      "responseElements": null,
      "requestID": "716deea3-8164-11e4-ab4f-657c7ab282ab",
      "eventID": "6225a85d-1e49-41e9-bf43-3cfc5549e560",
      "eventType": "AwsApiCall",
      "recipientAccountId": "222222222222"
    }
```

AWS Config Resources

The following related resources can help you as you work with this service.

- **AWS Config** – The primary web page for information about AWS Config.
- **Pricing** – The primary web page for AWS Config pricing information.
- **Technical FAQ** – The FAQ covers questions developers have asked about AWS Config.
- **Partners** – Links to partner products that are fully integrated with AWS Config to help you visualize, monitor, and manage the data from your configuration stream, configuration snapshots, or configuration history.

- **AWS Training and Courses** – Links to role-based and specialty courses as well as self-paced labs to help sharpen your AWS skills and gain practical experience.
- **AWS Developer Tools** – Links to developer tools and resources that provide documentation, code samples, release notes, and other information to help you build innovative applications with AWS.
- **AWS Support Center** – The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.
- **AWS Support** – The primary web page for information about AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
- **Contact Us** – A central contact point for inquiries concerning AWS billing, account, events, abuse, and other issues.
- **AWS Site Terms** – Detailed information about our copyright and trademark; your account, license, and site access; and other topics.

Document History

The following table describes the important changes to the *AWS Config Developer Guide*.

- **API version**: 2014-11-12
- **Latest documentation update**: March 31, 2016

Feature	Description	Release Date
Simplified role creation and updated policies	With this update, creating an IAM role for AWS Config is simplified. This enhancement is available in regions that support Config rules. To support this enhancement, the steps in Setting Up AWS Config Rules (p. 25) are updated, the example policy in Permissions for the Amazon S3 Bucket (p. 82) is updated, and the example policy in Recommended IAM Permissions for Using the AWS Config Console and the AWS CLI (p. 84) is updated.	March 31, 2016
Example functions and events for Config rules	This update provides updated example functions in Example AWS Lambda Functions for AWS Config Rules (Node.js) (p. 35), and this update adds example events in Example Events for AWS Config Rules (p. 40).	March 29, 2016
AWS Config Rules GitHub repository	This update adds information about the AWS Config Rules GitHub repository to Evaluating Resources With AWS Config Rules (p. 23). This repository provides sample functions for custom rules that are developed and contributed by AWS Config users.	March 1, 2016

Feature	Description	Release Date
AWS Config rules	This release introduces AWS Config rules. With rules, you can use AWS Config to evaluate whether your AWS resources comply with your desired configurations. For more information, see Evaluating Resources With AWS Config Rules (p. 23).	December 18, 2015
AWS Config supports IAM resource types	With this release, you can use AWS Config to record configuration changes to your IAM users, groups, roles, and customer managed policies. For more information, see Supported Resources, Configuration Items, and Relationships (p. 6).	December 10, 2015
AWS Config supports EC2 Dedicated host	With this release, you can use AWS Config to record configuration changes to your EC2 Dedicated hosts. For more information, see Supported Resources, Configuration Items, and Relationships (p. 6).	November 23, 2015
New setup process for the rules preview	This release introduces a guided setup process for first-time users of the AWS Config rules preview. The information in Setting Up AWS Config Rules (p. 25) is updated.	October 28, 2015
New content about setting up the rules preview	This update adds Setting Up AWS Config Rules (p. 25).	October 19, 2015
Updated permissions information	This update adds information about the following AWS managed policies for AWS Config: • `AWSConfigRole` – Grants AWS Config permission to get configuration details about your resources. For more information, see IAM Role Policy for Getting Configuration Details (p. 82). • `AWSConfigUserAccess` – Grants read-only access to an AWS Config user. For more information, see Recommended IAM Permissions for Using the AWS Config Console and the AWS CLI (p. 84).	October 19, 2015

Feature	Description	Release Date
AWS Config rules preview	This release introduces the AWS Config rules preview. With rules, you can use AWS Config to evaluate whether your AWS resources comply with your desired configurations. For more information, see Evaluating Resources With AWS Config Rules (p. 23).	October 7, 2015
New and updated content	This release adds the ability to look up resources that AWS Config has discovered. For more information, see Looking Up Resources That Are Discovered by AWS Config (p. 50).	August 27, 2015
New and updated content	This release adds the ability to select which resource types AWS Config records. For more information, see Selecting Which Resources AWS Config Records (p. 76).	June 23, 2015
New and updated content	This release adds support for the following regions: Asia Pacific (Tokyo), Asia Pacific (Singapore), EU (Frankfurt), South America (São Paulo), and US West (N. California). For more information, see Regions and Endpoints.	April 6, 2015
New and updated content	This release adds support for creating an optional email subscription to your Amazon SNS topic. You can also use email filters to monitor specific resource changes. For more information, see Monitoring AWS Config Resource Changes by Email (p. 56).	March 27, 2015
New and updated content	This release supports integration with AWS CloudTrail for logging all AWS Config API activity. For more information, see Logging AWS Config API Calls with AWS CloudTrail (p. 87). This release adds support for the US West (Oregon), EU (Ireland), and Asia Pacific (Sydney) regions. This release also includes the following updates to the documentation: • Information about monitoring AWS Config configurations • Various corrections throughout the document	February 10, 2015
New guide	This release introduces AWS Config.	November 12, 2014

AWS Glossary

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