AWS Config: Developer Guide
Copyright © 2020 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

What Is AWS Config? .................................................................................................................. 1
Ways to Use AWS Config ........................................................................................................ 1
  Resource Administration ........................................................................................................ 1
  Auditing and Compliance ...................................................................................................... 1
  Managing and Troubleshooting Configuration Changes .................................................. 2
  Security Analysis ................................................................................................................ 2
AWS Config Concepts ............................................................................................................ 2
  AWS Config .......................................................................................................................... 3
  AWS Config Managed and Custom Rules .......................................................................... 4
  Multi-Account Multi-Region Data Aggregation .................................................................. 5
  Managing AWS Config ........................................................................................................ 5
  Control Access to AWS Config .......................................................................................... 6
  Partner Solutions ................................................................................................................ 6
How AWS Config Works ........................................................................................................ 6
Deliver Configuration Items .................................................................................................. 7
AWS Config Supported AWS Resource Types and Resource Relationships ....................... 9
  Amazon API Gateway .......................................................................................................... 9
  Amazon CloudFront ............................................................................................................. 9
  Amazon CloudWatch ......................................................................................................... 10
  Amazon DynamoDB ............................................................................................................ 10
  Amazon Elastic Block Store ............................................................................................... 10
  Amazon Elastic Compute Cloud ......................................................................................... 10
  Amazon Elasticsearch Service ............................................................................................. 11
  Amazon Quantum Ledger Database (QLDB) ....................................................................... 12
  Amazon Redshift ................................................................................................................ 12
  Amazon Relational Database Service .................................................................................. 12
  Amazon S3 Bucket Attributes ............................................................................................ 13
  Amazon Simple Queue Service .......................................................................................... 14
  Amazon Simple Notification Service ................................................................................. 14
  Amazon Simple Storage Service .......................................................................................... 14
  Amazon Virtual Private Cloud ............................................................................................. 14
  AWS Auto Scaling ............................................................................................................... 15
  AWS Certificate Manager .................................................................................................... 16
  AWS CloudFormation .......................................................................................................... 16
  AWS CloudTrail .................................................................................................................. 16
  AWS CodeBuild .................................................................................................................... 16
  AWS CodePipeline ............................................................................................................. 17
  AWS Elastic Beanstalk ....................................................................................................... 17
  AWS Identity and Access Management ............................................................................... 18
  AWS Key Management Service ........................................................................................... 18
  AWS Lambda Function ....................................................................................................... 18
  AWS Service Catalog .......................................................................................................... 19
  AWS Shield .......................................................................................................................... 19
  AWS Systems Manager ....................................................................................................... 19
  AWS WAF .............................................................................................................................. 20
  AWS X-Ray ............................................................................................................................ 20
  Elastic Load Balancing ....................................................................................................... 20
Getting Started ...................................................................................................................... 22
  Setting Up AWS Config (Console) .................................................................................... 22
  Setting Up AWS Config (AWS CLI) .................................................................................. 24
    Prerequisites ..................................................................................................................... 25
    Turning on AWS Config ................................................................................................. 27
    Verify that AWS Config Is On ......................................................................................... 28
  Setting Up AWS Config Rules (Console) ......................................................................... 29
# AWS Config Developer Guide

<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Viewing the AWS Config Dashboard</td>
</tr>
<tr>
<td>33</td>
<td>AWS Config</td>
</tr>
<tr>
<td>34</td>
<td>Components of a Configuration Item</td>
</tr>
<tr>
<td>35</td>
<td>Record Configurations for Third-Party Resources</td>
</tr>
<tr>
<td>35</td>
<td>Step 1: Setup Your Development Environment</td>
</tr>
<tr>
<td>35</td>
<td>Step 2: Model Your Resource</td>
</tr>
<tr>
<td>35</td>
<td>Step 3: Register Your Resource</td>
</tr>
<tr>
<td>36</td>
<td>Step 4: Publish Resource Configuration</td>
</tr>
<tr>
<td>36</td>
<td>Record and Delete a Configuration State for Third-Party Resources Using AWS CLI</td>
</tr>
<tr>
<td>37</td>
<td>Managing a Configuration State for Third-Party Resources Type Using APIs</td>
</tr>
<tr>
<td>38</td>
<td>Region Support</td>
</tr>
<tr>
<td>39</td>
<td>Viewing AWS Resource Configurations and History</td>
</tr>
<tr>
<td>40</td>
<td>Looking Up Discovered Resources</td>
</tr>
<tr>
<td>40</td>
<td>Viewing Configuration Details</td>
</tr>
<tr>
<td>41</td>
<td>Viewing Compliance History</td>
</tr>
<tr>
<td>47</td>
<td>Delivering Configuration Snapshot</td>
</tr>
<tr>
<td>51</td>
<td>Managing AWS Config</td>
</tr>
<tr>
<td>56</td>
<td>Managing the Delivery Channel</td>
</tr>
<tr>
<td>59</td>
<td>Updating the IAM Role</td>
</tr>
<tr>
<td>60</td>
<td>Managing the Configuration Recorder</td>
</tr>
<tr>
<td>62</td>
<td>Selecting Which Resources are Recorded</td>
</tr>
<tr>
<td>65</td>
<td>Recording Software Configuration for Managed Instances</td>
</tr>
<tr>
<td>66</td>
<td>Advanced Queries</td>
</tr>
<tr>
<td>73</td>
<td>Deleting data</td>
</tr>
<tr>
<td>76</td>
<td>Example Notifications</td>
</tr>
<tr>
<td>76</td>
<td>Example Configuration Item Change Notifications</td>
</tr>
<tr>
<td>84</td>
<td>Example Configuration History Delivery Notification</td>
</tr>
<tr>
<td>84</td>
<td>Example Configuration Snapshot Delivery Started Notification</td>
</tr>
<tr>
<td>85</td>
<td>Example Configuration Snapshot Delivery Notification</td>
</tr>
<tr>
<td>86</td>
<td>Example Compliance Change Notification</td>
</tr>
<tr>
<td>87</td>
<td>Example Rules Evaluation Started Notification</td>
</tr>
<tr>
<td>87</td>
<td>Example Oversized Configuration Item Change Notification</td>
</tr>
<tr>
<td>88</td>
<td>Example Delivery Failed Notification</td>
</tr>
<tr>
<td>90</td>
<td>Controlling Permissions for AWS Config</td>
</tr>
<tr>
<td>90</td>
<td>Permissions for AWS Config Administration</td>
</tr>
<tr>
<td>91</td>
<td>Creating an IAM Group and Users for AWS Config Access</td>
</tr>
<tr>
<td>91</td>
<td>Granting Full-Access Permission for AWS Config Access</td>
</tr>
<tr>
<td>92</td>
<td>Additional Resources</td>
</tr>
<tr>
<td>92</td>
<td>Custom Permissions for AWS Config Users</td>
</tr>
<tr>
<td>92</td>
<td>Read-only access</td>
</tr>
<tr>
<td>93</td>
<td>Full access</td>
</tr>
<tr>
<td>95</td>
<td>Controlling User Permissions for Actions on Multi-Account Multi-Region Data Aggregation</td>
</tr>
<tr>
<td>95</td>
<td>Additional Information</td>
</tr>
<tr>
<td>98</td>
<td>Supported Resource-Level Permissions for AWS Config Rules APIs Actions</td>
</tr>
<tr>
<td>99</td>
<td>Permissions for the IAM Role</td>
</tr>
<tr>
<td>100</td>
<td>Creating IAM Role Policies</td>
</tr>
<tr>
<td>101</td>
<td>Troubleshooting for recording S3 buckets</td>
</tr>
<tr>
<td>102</td>
<td>Permissions for the Amazon S3 Bucket</td>
</tr>
<tr>
<td>103</td>
<td>Required Permissions for the Amazon S3 Bucket When Using IAM Roles</td>
</tr>
<tr>
<td>103</td>
<td>Required Permissions for the Amazon S3 Bucket When Using Service-Linked Roles</td>
</tr>
<tr>
<td>103</td>
<td>Granting AWS Config access to the Amazon S3 Bucket</td>
</tr>
<tr>
<td>104</td>
<td>Permissions for the Amazon SNS Topic</td>
</tr>
<tr>
<td>105</td>
<td>Required Permissions for the Amazon SNS Topic When Using IAM Roles</td>
</tr>
<tr>
<td>105</td>
<td>Required Permissions for the Amazon SNS Topic When Using Service-Linked Roles</td>
</tr>
<tr>
<td>106</td>
<td>Troubleshooting for the Amazon SNS Topic</td>
</tr>
<tr>
<td>107</td>
<td>AWS Config Rules</td>
</tr>
<tr>
<td>107</td>
<td>Viewing Configuration Compliance</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>174</td>
</tr>
<tr>
<td>Getting Started with Custom Rules</td>
<td>174</td>
</tr>
<tr>
<td>Developing a Custom Rule</td>
<td>176</td>
</tr>
<tr>
<td>Example Functions and Events</td>
<td>180</td>
</tr>
<tr>
<td>Managing your AWS Config Rules</td>
<td>188</td>
</tr>
<tr>
<td>Add, View, Update and Delete Rules (Console)</td>
<td>188</td>
</tr>
<tr>
<td>View, Update, and Delete Rules (AWS CLI)</td>
<td>190</td>
</tr>
<tr>
<td>View, Update, and Delete Rules (API)</td>
<td>191</td>
</tr>
<tr>
<td>Evaluating Your Resources</td>
<td>192</td>
</tr>
<tr>
<td>Evaluating your Resources (Console)</td>
<td>192</td>
</tr>
<tr>
<td>Evaluating your Resources (CLI)</td>
<td>192</td>
</tr>
<tr>
<td>Evaluating your Resources (API)</td>
<td>193</td>
</tr>
<tr>
<td>Deleting Evaluation Results</td>
<td>193</td>
</tr>
<tr>
<td>Deleting Evaluating Results (Console)</td>
<td>193</td>
</tr>
<tr>
<td>Deleting Evaluating Results (CLI)</td>
<td>193</td>
</tr>
<tr>
<td>Deleting Evaluating Results (API)</td>
<td>193</td>
</tr>
<tr>
<td>Enabling AWS Config Rules Across all Accounts in Your Organization</td>
<td>194</td>
</tr>
<tr>
<td>Remediating Resources and Rules</td>
<td>194</td>
</tr>
<tr>
<td>Prerequisite</td>
<td>195</td>
</tr>
<tr>
<td>Setting Up Manual Remediation (Console)</td>
<td>195</td>
</tr>
<tr>
<td>Setting Up Auto Remediation (Console)</td>
<td>195</td>
</tr>
<tr>
<td>Setting Up and Applying a Remediation Action Remediation Using Rules and Resources (Console)</td>
<td>196</td>
</tr>
<tr>
<td>Delete Remediation Action (Console)</td>
<td>197</td>
</tr>
<tr>
<td>Managing Remediation (API)</td>
<td>198</td>
</tr>
<tr>
<td>Tagging Your Resources</td>
<td>198</td>
</tr>
<tr>
<td>Restrictions Related to Tagging</td>
<td>199</td>
</tr>
<tr>
<td>Managing Tags with AWS Config API Actions</td>
<td>199</td>
</tr>
<tr>
<td>Conformance Packs</td>
<td>200</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>200</td>
</tr>
<tr>
<td>Step 1: Start AWS Config Recording</td>
<td>200</td>
</tr>
<tr>
<td>Step 2: Create a Service-Linked Role</td>
<td>200</td>
</tr>
<tr>
<td>Step 3: Provide Resource-Level Permissions</td>
<td>200</td>
</tr>
<tr>
<td>Prerequisites for Using a Conformance Pack With Remediation</td>
<td>203</td>
</tr>
<tr>
<td>Prerequisites for Using a Conformance Pack With One or More AWS Config Rules</td>
<td>203</td>
</tr>
<tr>
<td>Prerequisites for Organization Conformance Packs</td>
<td>204</td>
</tr>
<tr>
<td>Region Support</td>
<td>204</td>
</tr>
<tr>
<td>Conformance Pack Sample Templates</td>
<td>205</td>
</tr>
<tr>
<td>Operational Best Practices For Amazon S3</td>
<td>205</td>
</tr>
<tr>
<td>Operational Best Practices For Amazon DynamoDB</td>
<td>207</td>
</tr>
<tr>
<td>Operational Best Practices For PCI-DSS</td>
<td>207</td>
</tr>
<tr>
<td>Operational Best Practices For AWS Identity And Access Management</td>
<td>211</td>
</tr>
<tr>
<td>Templates With Remediation</td>
<td>214</td>
</tr>
<tr>
<td>Custom Conformance Pack</td>
<td>219</td>
</tr>
<tr>
<td>Deploying a Conformance Pack Using the AWS Config Console</td>
<td>220</td>
</tr>
<tr>
<td>Deploy a Conformance Pack Using Sample Templates</td>
<td>220</td>
</tr>
<tr>
<td>Edit a Conformance Pack</td>
<td>221</td>
</tr>
<tr>
<td>Delete a Conformance Pack</td>
<td>221</td>
</tr>
<tr>
<td>Deploying a Conformance Pack (AWS CLI)</td>
<td>221</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Multi-Account Multi-Region Data Aggregation</td>
<td>228</td>
</tr>
<tr>
<td>Region Support</td>
<td>228</td>
</tr>
<tr>
<td>Learn More</td>
<td>230</td>
</tr>
<tr>
<td>Viewing Configuration and Compliance Data in the Aggregated View</td>
<td>230</td>
</tr>
<tr>
<td>Use the Aggregated View</td>
<td>230</td>
</tr>
<tr>
<td>Learn More</td>
<td>230</td>
</tr>
<tr>
<td>Setting Up an Aggregator (Console)</td>
<td>231</td>
</tr>
<tr>
<td>Add an Aggregator</td>
<td>232</td>
</tr>
<tr>
<td>Edit an Aggregator</td>
<td>233</td>
</tr>
<tr>
<td>Delete an Aggregator</td>
<td>233</td>
</tr>
<tr>
<td>Learn More</td>
<td>231</td>
</tr>
<tr>
<td>Setting Up an Aggregator (AWS CLI)</td>
<td>233</td>
</tr>
<tr>
<td>Add an Aggregator Using Individual Accounts</td>
<td>234</td>
</tr>
<tr>
<td>Add an Aggregator Using AWS Organizations</td>
<td>235</td>
</tr>
<tr>
<td>View an Aggregator</td>
<td>235</td>
</tr>
<tr>
<td>Edit an Aggregator</td>
<td>236</td>
</tr>
<tr>
<td>Delete an Aggregator</td>
<td>237</td>
</tr>
<tr>
<td>Learn More</td>
<td>231</td>
</tr>
<tr>
<td>Authorizing Aggregator Accounts (Console)</td>
<td>238</td>
</tr>
<tr>
<td>Add Authorization for Aggregator Accounts and Regions</td>
<td>238</td>
</tr>
<tr>
<td>Authorize a Pending Request for an Aggregator Account</td>
<td>239</td>
</tr>
<tr>
<td>Delete Authorization for an Existing Aggregator Account</td>
<td>239</td>
</tr>
<tr>
<td>Learn More</td>
<td>231</td>
</tr>
<tr>
<td>Authorizing Aggregator Accounts (AWS CLI)</td>
<td>240</td>
</tr>
<tr>
<td>Add Authorization for Aggregator Accounts and Regions</td>
<td>240</td>
</tr>
<tr>
<td>Delete an Authorization Account</td>
<td>241</td>
</tr>
<tr>
<td>Learn More</td>
<td>231</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>241</td>
</tr>
<tr>
<td>Learn More</td>
<td>231</td>
</tr>
<tr>
<td>Monitoring</td>
<td>243</td>
</tr>
<tr>
<td>Using Amazon SQS</td>
<td>243</td>
</tr>
<tr>
<td>Permissions for Amazon SQS</td>
<td>243</td>
</tr>
<tr>
<td>Using Amazon CloudWatch Events</td>
<td>244</td>
</tr>
<tr>
<td>Amazon CloudWatch Events format for AWS Config</td>
<td>245</td>
</tr>
<tr>
<td>Creating Amazon CloudWatch Events Rule for AWS Config</td>
<td>245</td>
</tr>
<tr>
<td>Service-Linked AWS Config Rules</td>
<td>247</td>
</tr>
<tr>
<td>Using Service-Linked Roles</td>
<td>248</td>
</tr>
<tr>
<td>Service-Linked Role Permissions for AWS Config</td>
<td>248</td>
</tr>
<tr>
<td>Creating a Service-Linked Role for AWS Config</td>
<td>248</td>
</tr>
<tr>
<td>Editing a Service-Linked Role for AWS Config</td>
<td>248</td>
</tr>
<tr>
<td>Deleting a Service-Linked Role for AWS Config</td>
<td>249</td>
</tr>
<tr>
<td>Using AWS Config with Interface VPC Endpoints</td>
<td>250</td>
</tr>
<tr>
<td>Availability</td>
<td>250</td>
</tr>
<tr>
<td>Create a VPC Endpoint for AWS Config</td>
<td>250</td>
</tr>
<tr>
<td>Logging AWS Config API Calls with AWS CloudTrail</td>
<td>251</td>
</tr>
<tr>
<td>AWS Config Information in CloudTrail</td>
<td>251</td>
</tr>
<tr>
<td>Understanding AWS Config Log File Entries</td>
<td>252</td>
</tr>
<tr>
<td>Example Log Files</td>
<td>252</td>
</tr>
</tbody>
</table>
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 a complete list of AWS resources supported by AWS Config, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

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.

AWS Config Concepts

AWS Config provides a detailed view of the resources associated with your AWS account, including how they are configured, how they are related to one another, and how the configurations and their relationships have changed over time. Let’s take a closer look at the concepts of AWS Config.

Contents

- AWS Config (p. 3)
  - AWS Resources (p. 3)
  - Configuration History (p. 3)
  - Configuration Items (p. 3)
  - Configuration Recorder (p. 3)
  - Configuration Snapshot (p. 4)
  - Configuration Stream (p. 4)
  - Resource Relationship (p. 4)
- AWS Config Managed and Custom Rules (p. 4)
  - AWS Config Custom Rules (p. 4)
- Multi-Account Multi-Region Data Aggregation (p. 5)
  - Source Account (p. 5)
  - Source Region (p. 5)
  - Aggregator (p. 5)
  - Aggregator Account (p. 5)
  - Authorization (p. 5)
- Managing AWS Config (p. 5)
  - AWS Config Console (p. 5)
Understanding the basic components of AWS Config will help you track resource inventory and changes and evaluate configurations of your AWS resources.

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 AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

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 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. AWS Config creates a configuration item whenever it detects a change to a resource type that it is recording. For example, if AWS Config is recording Amazon S3 buckets, AWS Config creates a configuration item whenever a bucket is created, updated, or deleted.

For more information, see Components of a Configuration Item (p. 33).

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. For more information, see Managing the Configuration Recorder (p. 60).

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. 62).

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.
**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.

**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 more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](p. 9).

**AWS Config Managed and Custom 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. If a resource violates a rule, AWS Config flags the resource and the rule as noncompliant, and AWS Config notifies you through Amazon SNS.

**AWS Config Custom Rules**

With AWS Config 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.

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 triggers are defined as part of the rule, and they can include 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** – AWS Config runs evaluations for the rule at a frequency that you choose (for example, every 24 hours).

For more information, see [Evaluating Resources with Rules](p. 107).
Multi-Account Multi-Region Data Aggregation

Multi-account multi-region data aggregation in AWS Config allows you to aggregate AWS Config configuration and compliance data from multiple accounts and regions into a single account. Multi-account multi-region data aggregation is useful for central IT administrators to monitor compliance for multiple AWS accounts in the enterprise.

Source Account

A source account is the AWS account from which you want to aggregate AWS Config resource configuration and compliance data. A source account can be an individual account or an organization in AWS Organizations. You can provide source accounts individually or you can retrieve them through AWS Organizations.

Source Region

A source region is the AWS region from which you want to aggregate AWS Config configuration and compliance data.

Aggregator

An aggregator is a new resource type in AWS Config that collects AWS Config configuration and compliance data from multiple source accounts and regions. Create an aggregator in the region where you want to see the aggregated AWS Config configuration and compliance data.

Aggregator Account

An aggregator account is an account where you create an aggregator.

Authorization

As a source account owner, authorization refers to the permissions you grant to an aggregator account and region to collect your AWS Config configuration and compliance data. Authorization is not required if you are aggregating source accounts that are part of AWS Organizations.

For more information, see topics in Multi-Account Multi-Region Data Aggregation (p. 228) section.

Managing AWS Config

AWS Config Console

You can manage the service using the AWS Config console. The console provides a user interface for performing many AWS Config tasks such as:

- Specifying the types of AWS resources for recording.
- Configuring resources to record, including:
  - Selecting an Amazon S3 bucket.
  - Selecting an Amazon SNS topic.
  - Creating AWS Config role.
- Creating managed rules and custom rules that represent desired configuration settings for specific AWS resources or for an entire AWS account.
- Creating and managing configuration aggregators to aggregate data across multiple accounts and regions.
- Viewing a snapshot of current configurations of the supported resources.
• Viewing relationships between AWS resources.

For more information about the AWS Management Console, see AWS Management Console.

**AWS Config CLI**

The AWS Command Line Interface is a unified tool that you can use to interact with AWS Config from the command line. For more information, see the AWS Command Line Interface User Guide. For a complete list of AWS Config CLI commands, see Available Commands.

**AWS Config APIs**

In addition to the console and the CLI, you can also use the AWS Config RESTful APIs to program AWS Config directly. For more information, see the AWS Config API Reference.

**AWS SDKs**

As an alternative to using the AWS Config API, you can use one of the AWS SDKs. Each SDK consists of libraries and sample code for various programming languages and platforms. The SDKs provide a convenient way to create programmatic access to AWS Config. For example, you can use the SDKs to sign requests cryptographically, manage errors, and retry requests automatically. For more information, see the Tools for Amazon Web Services page.

**Control Access to AWS Config**

AWS Identity and Access Management is a web service that enables Amazon Web Services (AWS) customers to manage users and user permissions. Use IAM to create individual users for anyone who needs access to AWS Config. Create an IAM user for yourself, give that IAM user administrative privileges, and use that IAM user for all of your work. By creating individual IAM users for people accessing your account, you can give each IAM user a unique set of security credentials. You can also grant different permissions to each IAM user. If necessary, you can change or revoke an IAM user's permissions at any time. For more information, see Controlling Permissions for AWS Config (p. 90).

**Partner Solutions**

AWS partners with third-party specialists in logging and analysis to provide solutions that use AWS Config output. For more information, visit the AWS Config detail page at AWS Config.

**How AWS Config Works**

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 Amazon 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 Amazon 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. The configuration history file and configuration snapshot are in JSON format.

Note
AWS Config only delivers the configuration history files and configuration snapshots to the specified S3 bucket; AWS Config doesn't modify the lifecycle policies for objects in the S3
bucket. You can use lifecycle policies to specify whether you want to delete or archive objects to Amazon S3 Glacier. For more information, see Managing Lifecycle Configuration in the Amazon Simple Storage Service Console User Guide. You can also see the Archiving Amazon S3 Data to S3 Glacier blog post.

**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. 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.

- **ConfigRulesEvaluationStarted**

  AWS Config started evaluating your rule against the specified resources.

- **ConfigurationSnapshotDeliveryStarted**

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

- **ConfigurationSnapshotDeliveryCompleted**

  AWS Config successfully delivered the configuration snapshot to your Amazon S3 bucket.

- **ConfigurationSnapshotDeliveryFailed**

  AWS Config failed to deliver the configuration snapshot to your Amazon S3 bucket.

- **ConfigurationHistoryDeliveryCompleted**

  AWS Config successfully delivered the configuration history to your Amazon 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 notifications are delivered within minutes of a change and are collectively known as the configuration stream.

- **OversizedConfigurationItemChangeNotification**

  This message type is delivered when a configuration item change notification exceeded the maximum size allowed by Amazon SNS. The message includes a summary of the configuration item. You can view the complete notification in the specified Amazon S3 bucket location.
AWS Config Developer Guide
AWS Config Supported AWS Resource Types and Resource Relationships

OversizedConfigurationItemChangeDeliveryFailed

AWS Config failed to deliver the oversized configuration item change notification to your Amazon S3 bucket.

For example notifications, see Notifications that AWS Config Sends to an Amazon SNS topic (p. 76).

For more information about Amazon SNS, see the Amazon Simple Notification Service Developer Guide.

AWS Config Supported AWS Resource Types and Resource Relationships

AWS Config supports the following AWS resources types and resource relationships.

Amazon API Gateway

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>API Gateway</td>
<td>AWS::ApiGateway::Stage</td>
<td>contained in</td>
<td>ApiGateway Rest Api</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with</td>
<td>WAFRegional WebACL</td>
</tr>
<tr>
<td></td>
<td>AWS::ApiGatewayV2::Stage</td>
<td>contained in</td>
<td>ApiGatewayV2 Api</td>
</tr>
<tr>
<td></td>
<td>AWS::ApiGateway::RestApi</td>
<td>contains</td>
<td>ApiGateway Stage</td>
</tr>
<tr>
<td></td>
<td>AWS::ApiGatewayV2::RestApi</td>
<td>contains</td>
<td>ApiGatewayV2 Stage</td>
</tr>
</tbody>
</table>

To learn more about how AWS Config integrates with Amazon API Gateway, see Monitoring API Gateway API Configuration with AWS Config.

Amazon CloudFront

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon CloudFront *</td>
<td>AWS::CloudFront::Distribution</td>
<td>is associated with</td>
<td>AWS WAF WebACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ACM Certificate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>S3 Bucket</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IAM Server Certificate</td>
</tr>
<tr>
<td></td>
<td>AWS::CloudFront::StreamingDistribution</td>
<td>is associated with</td>
<td>AWS WAF WebACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ACM Certificate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>S3 Bucket</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IAM Server Certificate</td>
</tr>
</tbody>
</table>

*AWS Config support for Amazon CloudFront is available only in the US East (N. Virginia) region.
# Amazon CloudWatch

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon CloudWatch</td>
<td>AWS::CloudWatch::Alarm</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

# Amazon DynamoDB

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon DynamoDB</td>
<td>AWS::DynamoDB::Table</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

# Amazon Elastic Block Store

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Elastic Block Store</td>
<td>AWS::EC2::Volume</td>
<td>is attached to</td>
<td>EC2 instance</td>
</tr>
</tbody>
</table>

# Amazon Elastic Compute Cloud

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Elastic Compute Cloud</td>
<td>AWS::EC2::Host^</td>
<td>contains</td>
<td>EC2 instance</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::EIP</td>
<td>is attached to</td>
<td>EC2 instance</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Network interface</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::Instance</td>
<td>contains</td>
<td>EC2 network interface</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with</td>
<td>EC2 security group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is attached to</td>
<td>Amazon EBS volume</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>EC2 Elastic IP (EIP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is contained in</td>
<td>EC2 Dedicated IP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Route table</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Subnet</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Virtual private cloud (VPC)</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::NetworkInterface</td>
<td>is associated with</td>
<td>EC2 security group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is attached to</td>
<td>EC2 Elastic IP (EIP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>EC2 instance</td>
</tr>
<tr>
<td>AWS Service</td>
<td>Resource Type Value</td>
<td>Relationship</td>
<td>Related Resource</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------</td>
<td>--------------</td>
<td>------------------</td>
</tr>
<tr>
<td>AWS::EC2::SecurityGroup</td>
<td>is associated with</td>
<td>EC2 instance</td>
<td></td>
</tr>
<tr>
<td>AWS::EC2::NatGateway</td>
<td>is contained in</td>
<td>Subnet</td>
<td></td>
</tr>
<tr>
<td>AWS::EC2::EgressOnlyInternetGateway</td>
<td>is attached to</td>
<td>Virtual private cloud (VPC)</td>
<td></td>
</tr>
<tr>
<td>AWS::EC2::FlowLog</td>
<td>NA</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>AWS::EC2::VPCEndpoint</td>
<td>is contained in</td>
<td>Virtual private cloud (VPC)</td>
<td></td>
</tr>
<tr>
<td>AWS::EC2::VPCEndpointService</td>
<td>is associated with</td>
<td>ElasticLoadBalancingV2 LoadBalancer</td>
<td></td>
</tr>
<tr>
<td>AWS::EC2::VPCPeeringConnection</td>
<td>is associated with</td>
<td>Virtual private cloud (VPC)</td>
<td></td>
</tr>
</tbody>
</table>

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 with AWS Config in the Amazon EC2 User Guide for Linux Instances.

Amazon Elasticsearch Service

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Elasticsearch Service</td>
<td>AWS::Elasticsearch::Domain</td>
<td>is associated with</td>
<td>KMS Key</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>EC2 security group</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>EC2 subnet</td>
</tr>
<tr>
<td>AWS Service</td>
<td>Resource Type Value</td>
<td>Relationship</td>
<td>Related Resource</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------</td>
<td>--------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>Amazon Quantum Ledger Database (QLDB)</td>
<td></td>
<td></td>
<td>Virtual private cloud (VPC)</td>
</tr>
<tr>
<td>Amazon Redshift</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amazon Relational Database Service</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Amazon Quantum Ledger Database (QLDB)

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon QLDB</td>
<td>AWS::QLDB::Ledger</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

### Amazon Redshift

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Redshift</td>
<td>AWS::Redshift::Cluster</td>
<td>is associated with</td>
<td>Cluster parameter group</td>
</tr>
<tr>
<td></td>
<td>AWS::Redshift::ClusterSecurityGroup</td>
<td>is associated with</td>
<td>Cluster security group</td>
</tr>
<tr>
<td></td>
<td>AWS::Redshift::ClusterSubnetGroup</td>
<td>is associated with</td>
<td>Cluster subnet group</td>
</tr>
<tr>
<td></td>
<td>AWS::Redshift::ClusterSnapshot</td>
<td>is associated with</td>
<td>Security group</td>
</tr>
<tr>
<td></td>
<td>AWS::Redshift::EventSubscription</td>
<td>is associated with</td>
<td>Virtual private cloud (VPC)</td>
</tr>
</tbody>
</table>

### Amazon Relational Database Service

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Relational Database Service</td>
<td>AWS::RDS::DBInstance</td>
<td>is associated with</td>
<td>EC2 security group</td>
</tr>
<tr>
<td></td>
<td>AWS::RDS::DBInstance</td>
<td>is associated with</td>
<td>RDS DB security group</td>
</tr>
<tr>
<td></td>
<td>AWS::RDS::DBInstance</td>
<td>is associated with</td>
<td>RDS DB subnet group</td>
</tr>
</tbody>
</table>
### Amazon S3 Bucket Attributes

AWS Config also records the following attributes for the Amazon S3 bucket resource type.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccelerateConfiguration</td>
<td>Transfer acceleration for data over long distances between your client and a bucket.</td>
</tr>
<tr>
<td>BucketAcl</td>
<td>Access control list used to manage access to buckets and objects.</td>
</tr>
<tr>
<td>BucketPolicy</td>
<td>Policy that defines the permissions to the bucket.</td>
</tr>
<tr>
<td>CrossOriginConfiguration</td>
<td>Allow cross-origin requests to the bucket.</td>
</tr>
<tr>
<td>LifecycleConfiguration</td>
<td>Rules that define the lifecycle for objects in your bucket.</td>
</tr>
<tr>
<td>LoggingConfiguration</td>
<td>Logging used to track requests for access to the bucket.</td>
</tr>
<tr>
<td>NotificationConfiguration</td>
<td>Event notifications used to send alerts or trigger workflows for specified bucket events.</td>
</tr>
<tr>
<td>ReplicationConfiguration</td>
<td>Automatic, asynchronous copying of objects across buckets in different AWS Regions.</td>
</tr>
<tr>
<td>RequestPaymentConfiguration</td>
<td>Requester pays is enabled.</td>
</tr>
<tr>
<td>TaggingConfiguration</td>
<td>Tags added to the bucket to categorize. You can also use tagging to track billing.</td>
</tr>
<tr>
<td>WebsiteConfiguration</td>
<td>Static website hosting is enabled for the bucket.</td>
</tr>
</tbody>
</table>
### Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VersioningConfiguration</td>
<td>Versioning is enabled for objects in the bucket.</td>
</tr>
</tbody>
</table>

For more information about the attributes, see `Bucket Configuration Options` in the *Amazon Simple Storage Service Developer Guide*.

## Amazon Simple Queue Service

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Simple Queue Service</td>
<td>AWS::SQS::Queue</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

## Amazon Simple Notification Service

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Simple Notification Service</td>
<td>AWS::SNS::Topic</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

## Amazon Simple Storage Service

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Simple Storage Service</td>
<td>AWS::S3::Bucket</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>AWS::S3::AccountPublicAccessBlock</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

*If you configured AWS Config to record your S3 buckets, and are not receiving configuration change notifications, verify your S3 bucket policies have the required permissions. For more information, see `Troubleshooting for recording S3 buckets (p. 101)`.

## Amazon Virtual Private Cloud

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Virtual Private Cloud</td>
<td>AWS::EC2::CustomerGateway</td>
<td>is attached to</td>
<td>VPN connection</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::InternetGateway</td>
<td>is attached to</td>
<td>Virtual private cloud (VPC)</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::NetworkAcl</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::RouteTable</td>
<td>contains</td>
<td>EC2 instance</td>
</tr>
</tbody>
</table>

EC2 network interface
### AWS Auto Scaling

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Scaling</td>
<td>AWS::AutoScaling::AutoScalingGroup</td>
<td>contains</td>
<td>Amazon EC2 instance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with</td>
<td>Classic Load Balancer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Auto Scaling launch</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>configuration</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Subnet</td>
</tr>
</tbody>
</table>
### AWS Config Developer Guide

#### AWS Certificate Manager

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS::AutoScaling::LaunchConfiguration</td>
<td>is associated with</td>
<td>Amazon EC2 security group</td>
<td></td>
</tr>
<tr>
<td>AWS::AutoScaling::ScalingPolicy</td>
<td>is associated with</td>
<td>Auto Scaling group</td>
<td></td>
</tr>
<tr>
<td>AWS::AutoScaling::ScheduledAction</td>
<td>is associated with</td>
<td>Alarm</td>
<td></td>
</tr>
</tbody>
</table>

#### AWS Certificate Manager

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Certificate Manager</td>
<td>AWS::ACM::Certificate</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

#### AWS CloudFormation

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CloudFormation</td>
<td>AWS::CloudFormation:contains</td>
<td>Supported AWS resource types</td>
<td></td>
</tr>
</tbody>
</table>

AWS Config records configuration changes to AWS CloudFormation stacks and supported resource types in the stacks. AWS Config does not record configuration changes for resource types in the stack that are not yet supported. Unsupported resource types appear in the supplementary configuration section of the configuration item for the stack.

#### AWS CloudTrail

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CloudTrail</td>
<td>AWS::CloudTrail::Trail</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

#### AWS CodeBuild

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CloudBuild</td>
<td>AWS::CodeBuild::Project</td>
<td>is associated with</td>
<td>S3 bucket</td>
</tr>
</tbody>
</table>

AWS Config integrates with AWS CodeBuild Sample.
AWS CodePipeline

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CodePipeline</td>
<td>AWS::CodePipeline::Pipeline</td>
<td>is attached to S3 bucket</td>
<td></td>
</tr>
<tr>
<td></td>
<td>is associated with IAM role</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with Code project</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with Lambda function</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with Cloudformation stack</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with ElasticBeanstalk application</td>
<td></td>
</tr>
</tbody>
</table>

AWS Config records configuration changes to CodePipeline pipelines and supported resource types in the pipelines. AWS Config does not record configuration changes for resource types in the pipelines that are not yet supported. Unsupported resource types such as CodeCommit repository, CodeDeploy application, ECS cluster, and ECS service appear in the supplementary configuration section of the configuration item for the stack.

AWS Elastic Beanstalk

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Elastic Beanstalk</td>
<td>AWS::ElasticBeanstalk::Application</td>
<td>contains Elastic Beanstalk Application Version</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with Elastic Beanstalk Environment</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with IAM role</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AWS::ElasticBeanstalk::ApplicationVersion</td>
<td>contains Elastic Beanstalk Application</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with Elastic Beanstalk Environment</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with S3 bucket</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AWS::ElasticBeanstalk::Application</td>
<td>contains Elastic Beanstalk Application</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is associated with IAM role</td>
<td></td>
</tr>
<tr>
<td></td>
<td>contains</td>
<td>CloudFormation Stack</td>
<td></td>
</tr>
</tbody>
</table>
AWS Identity and Access Management

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Identity and Access Management</td>
<td>AWS::IAM::User*</td>
<td>is attached to</td>
<td>IAM group</td>
</tr>
<tr>
<td></td>
<td>AWS::IAM::Group*</td>
<td>contains</td>
<td>IAM user</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is attached to</td>
<td>IAM customer managed policy</td>
</tr>
<tr>
<td></td>
<td>AWS::IAM::Role*</td>
<td>is attached to</td>
<td>IAM customer managed policy</td>
</tr>
<tr>
<td></td>
<td>AWS::IAM::Policy</td>
<td>is attached to</td>
<td>IAM user</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IAM group</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IAM role</td>
</tr>
</tbody>
</table>

*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. 62).

AWS Config includes inline policies with the configuration details that it records.

AWS Key Management Service

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Key Management Service</td>
<td>AWS::KMS::Key</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

AWS Lambda Function

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Lambda Function</td>
<td>AWS::Lambda::Function</td>
<td>is associated with IAM role</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>contains EC2 security group</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>contains EC2 subnet</td>
<td></td>
</tr>
</tbody>
</table>
## AWS Service Catalog

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Service Catalog</td>
<td>AWS::ServiceCatalog::Portfolio</td>
<td>is contained in</td>
<td>Portfolio</td>
</tr>
<tr>
<td>AWS Service Catalog</td>
<td>AWS::ServiceCatalog::CloudFormationProduct</td>
<td>is associated with</td>
<td>CloudFormationProvisionedProduct</td>
</tr>
<tr>
<td>AWS Service Catalog</td>
<td>AWS::ServiceCatalog::CloudFormationProvisionedProduct</td>
<td>is associated with</td>
<td>CloudFormationProduct</td>
</tr>
<tr>
<td>AWS Service Catalog</td>
<td>AWS::ServiceCatalog::CloudFormationProvisionedProduct</td>
<td>is associated with</td>
<td>CloudFormationStack</td>
</tr>
<tr>
<td>AWS Service Catalog</td>
<td>AWS::ServiceCatalog::Portfolio</td>
<td>contains</td>
<td>CloudFormationProduct</td>
</tr>
</tbody>
</table>

## AWS Shield

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Shield*</td>
<td>AWS::Shield::Protection</td>
<td>is associated with</td>
<td>Amazon CloudFront distribution</td>
</tr>
<tr>
<td>AWS Shield*</td>
<td>AWS::ShieldRegional::Protection</td>
<td>is associated with</td>
<td>EC2 EIP</td>
</tr>
<tr>
<td>AWS Shield*</td>
<td>AWS::ShieldRegional::Protection</td>
<td>is associated with</td>
<td>ElasticLoadBalancing Balancer</td>
</tr>
<tr>
<td>AWS Shield*</td>
<td>AWS::ShieldRegional::Protection</td>
<td>is associated with</td>
<td>ElasticLoadBalancingV2 LoadBalancer</td>
</tr>
</tbody>
</table>

*AWS Config support for AWS::Shield::Protection is available only in the US East (N. Virginia) region. The AWS::ShieldRegional::Protection is available in all regions where AWS Shield is supported.

## AWS Systems Manager

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Systems Manager</td>
<td>AWS::SSM::ManagedInstanceInventory</td>
<td>is associated with</td>
<td>EC2 instance</td>
</tr>
<tr>
<td>AWS Systems Manager</td>
<td>AWS::SSM::PatchCompliance</td>
<td>is associated with</td>
<td>Managed Instance Inventory</td>
</tr>
<tr>
<td>AWS Systems Manager</td>
<td>AWS::SSM::AssociationCompliance</td>
<td>is associated with</td>
<td>Managed Instance Inventory</td>
</tr>
</tbody>
</table>

*To learn more about managed instance inventory, see [Recording Software Configuration for Managed Instances](p. 65).
AWS WAF

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS WAF</td>
<td>AWS::WAF::RateBasedRule</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>AWS WAF</td>
<td>AWS::WAF::Rule</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>AWS WAF</td>
<td>AWS::WAF::WebACL</td>
<td>is associated with</td>
<td>WAF rule</td>
</tr>
<tr>
<td>AWS WAF</td>
<td>AWS::WAF::RuleGroup</td>
<td>is associated with</td>
<td>WAF rule</td>
</tr>
<tr>
<td>AWS WAFRegional</td>
<td>AWS::WAFRegional::RateBasedRule</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>AWS WAFRegional</td>
<td>AWS::WAFRegional::Rule</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>AWS WAFRegional</td>
<td>AWS::WAFRegional::WebACL</td>
<td>is associated with</td>
<td>ElasticLoadBalancingV2 LoadBalancer</td>
</tr>
<tr>
<td>AWS WAFRegional</td>
<td>AWS::WAFRegional::RuleGroup</td>
<td>is associated with</td>
<td>WAFRegional rule</td>
</tr>
</tbody>
</table>

*The AWS WAF resource type values are available only in the US East (N. Virginia) Region. The AWS::WAFRegional::RateBasedRule, AWS::WAFRegional::Rule, AWS::WAFRegional::WebACL, and AWS::WAFRegional::RuleGroup are available in all regions where AWS WAF is supported.*

AWS X-Ray

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS X-Ray</td>
<td>AWS::XRay::EncryptionConfig</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

Elastic Load Balancing

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Resource Type Value</th>
<th>Relationship</th>
<th>Related Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elastic Load Balancing</td>
<td>Application Load Balancer</td>
<td>is associated with</td>
<td>EC2 security group</td>
</tr>
<tr>
<td>Elastic Load Balancing</td>
<td>AWS::ElasticLoadBalancing::LoadBalancer</td>
<td>is attached to</td>
<td>Subnet</td>
</tr>
<tr>
<td>Elastic Load Balancing</td>
<td>AWS::ElasticLoadBalancing::LoadBalancer</td>
<td>is contained in</td>
<td>Virtual private cloud (VPC)</td>
</tr>
<tr>
<td>Classic Load Balancer</td>
<td>is associated with</td>
<td></td>
<td>EC2 security group</td>
</tr>
<tr>
<td>AWS Service</td>
<td>Resource Type Value</td>
<td>Relationship</td>
<td>Related Resource</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>Network Load Balancer</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>AWS::ElasticLoadBalancingV2::LoadBalancer</td>
<td>is attached to Subnet is contained in Virtual private cloud (VPC)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Getting Started with AWS Config

When you sign up for AWS, your account has access to all AWS services. You pay only for the services that you use.

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

To sign up for an AWS account

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

After you sign up for an AWS account, you can get started with AWS Config with the AWS Management Console, AWS CLI, or the AWS SDKs. Use the console for a quick and streamlined process.

When you set up AWS Config, you can complete the following:

- Specify the resource types that you want AWS Config to record.
- 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.
- Specify the rules that you want AWS Config to use to evaluate compliance information for the recorded resource types.

For more information about using the AWS CLI, see Setting Up AWS Config with the AWS CLI (p. 24).

For more information about using the AWS SDKs, see AWS Software Development Kits for AWS Config (p. 258).

Topics
- Setting Up AWS Config with the Console (p. 22)
- Setting Up AWS Config with the AWS CLI (p. 24)
- Setting Up AWS Config Rules with the Console (p. 29)
- Viewing the AWS Config Dashboard (p. 31)

Setting Up AWS Config with the Console

You can use the AWS Management Console to get started with AWS Config to do the following:

- Specify the resource types you want AWS Config to record.
- Set up Amazon SNS to notify you of configuration changes.
- Specify an Amazon S3 bucket to receive configuration information.
- Add AWS Config managed rules to evaluate the resource types.

If you are using AWS Config for the first time or configuring AWS Config for a new region, you can choose managed rules to evaluate resource configurations. For regions that support AWS Config and AWS Config Rules, see AWS Config Regions and Endpoints in the Amazon Web Services General Reference.
To set up AWS Config with the 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 are opening the AWS Config console or you are setting up AWS Config in a new region, the AWS Config console page looks like the following:

3. Choose Get Started Now.

4. On the Settings page, for Resource types to record, specify the AWS resource types 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 resource type, AWS Config 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 global resource type, AWS Config automatically starts recording resources of that type.
     - **Specific types** – AWS Config records configuration changes for only the AWS resource types that you specify.

   For more information about these options, see Selecting Which Resources AWS Config Records (p. 62).

5. For 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
can't change the bucket name 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**, choose your preferred bucket.
- **Choose a bucket from another account** – For **Bucket Name**, type the bucket name.

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. 102).

6. For **Amazon SNS Topic**, choose whether AWS Config streams information by selecting the Stream configuration changes and notifications to an Amazon SNS topic. AWS Config sends notifications such as configuration history delivery, configuration snapshot delivery, and compliance.

7. 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. 104).

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

8. For **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**, choose an IAM role in your account. AWS Config will attach the required policies. For more information, see Permissions for the IAM Role Assigned to AWS Config (p. 99).

  **Note**
  Check the box if you want to use the IAM role as it. AWS Config will not attach policies to the role.

9. If you are setting up AWS Config in a region that supports rules, choose **Next**. See Setting Up AWS Config Rules with the Console (p. 29).

Otherwise, choose **Save**. AWS Config displays the **Resource inventory** page.

For information about looking up the existing resources in your account and understanding the configurations of your resources, see View, and Manage Your AWS Resources (p. 40).

You can also use Amazon Simple Queue Service to monitor AWS resources programmatically. For more information, see Monitoring AWS Resource Changes with Amazon SQS (p. 243).

### Setting Up AWS Config with the AWS CLI

You can use the AWS Command Line Interface to control and automate the services from AWS.

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

- **AWS Command Line Interface User Guide**
Prerequisites

Follow this procedure to create an Amazon S3 bucket, an Amazon SNS topic, and an IAM role with attached policies. You can then use the AWS CLI to specify the bucket, topic, and role for AWS Config.

Contents

- Creating an Amazon S3 Bucket (p. 25)
- Creating an Amazon SNS Topic (p. 25)
- Creating an IAM Role (p. 26)

Creating 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 Creating an Amazon SNS Topic (p. 25).

To create an Amazon S3 bucket with the AWS CLI, use the create-bucket command.

To create an Amazon S3 bucket with the console

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.

   **Note**
   You can also use an Amazon S3 bucket from a different account, but you may 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. 102), and then go to Creating an Amazon SNS Topic (p. 25).

Creating 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 Creating an IAM Role (p. 26).
To create an Amazon SNS topic with the AWS CLI, use the `create-topic` command.

**To create an Amazon SNS topic with the console**

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 for the next task.

   For more information, see ARN Format in the AWS General Reference.

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, choose Email.
5. For Endpoint, type an email address that you can use to receive the notification and 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.

   **Note**

   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. 104) and then go to Creating an IAM Role (p. 26).

**Creating 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 with the AWS CLI, use the `create-role` command. You can then attach a policy to the role with the `attach-role-policy` command.

**To create an IAM role with the console**

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 Policies** 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 IAM Role Policy for Amazon S3 Bucket (p. 100) 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 Policies** 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 IAM Role Policy for Amazon SNS Topic (p. 101) 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**.

**Turning on AWS Config**

You can use the AWS CLI to turn on AWS Config with the **subscribe** command and a few parameters.

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 can have one configuration recorder and one delivery channel per region in your account.
To turn on AWS Config, use the `subscribe` command with the following parameters:

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.

Your command should look like the following example:

```
```

After you run the `subscribe` command, AWS Config records all supported resources that it finds in the region. If you don't want AWS Config to record supported resources, specify the types of resources to record by updating the configuration recorder to use a recording group. For more information, see Selecting Resources (AWS CLI) (p. 63).

**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.

**Contents**

- Verify that the Delivery Channel Is Created (p. 28)
- Verify that the Configuration Recorder Is Created (p. 29)
- Verify that AWS Config has started recording (p. 29)

**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": [
    {
        "name": "my-delivery-channel",
        "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 Creating an IAM Role (p. 26).

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

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 View, and Manage Your AWS Resources (p. 40).

Setting Up AWS Config Rules with the Console

The `Rules` page provides initial AWS managed rules that you can add to your account. After set up, AWS Config evaluates your AWS resources against the rules that you choose. You can update the rules and create additional managed rules after set up.
To see the complete list of AWS managed rules, see List of AWS Config Managed Rules (p. 112).

For example, you can choose the `cloudtrail-enabled` rule, which evaluates whether your account has a CloudTrail trail. If your account doesn't have a trail, AWS Config flags the resource type and the rule as noncompliant.

On the Rules page, you can do the following:

A. Type in the search field to filter results by rule name, description, or label. For example, type `EC2` to return rules that evaluate EC2 resource types or type `periodic` to return rules that have a periodic trigger. Type "new" to search for newly added rules. For more information about trigger types, see Specifying Triggers for AWS Config Rules (p. 110).

B. Choose Select all to add all rules or Clear all to remove all rules.

C. Choose the arrow icon to see the next page of rules.

D. Recently added rules are marked as New.

E. See the labels to identify the service that the rule evaluates and if the rule has a periodic trigger.

To set up AWS Config rules

1. On the Rules page, choose the rules that you want. You can customize these rules and add other rules to your account after set up.
2. Choose **Next**.

3. On the **Review** page, verify your setup details, and then choose **Confirm**.

The **Rules** page shows your rules and their current compliance results in the table. The result for each rule is **Evaluating**... until AWS Config finishes evaluating your resources against the rule. You can update the results with the refresh button. When AWS Config finishes evaluations, you can see the rules and resource types that are compliant or noncompliant. For more information, see Viewing Configuration Compliance (p. 107).

**Note**

AWS Config evaluates only the resource types that it is recording. For example, if you add the **cloudtrail-enabled** rule but don't record the CloudTrail trail resource type, AWS Config can't evaluate whether the trails in your account are compliant or noncompliant. For more information, see Selecting Which Resources AWS Config Records (p. 62).

You can view, edit, and delete your existing rules. You can also create additional AWS managed rules or create your own. For more information, see Managing your AWS Config Rules (p. 188).

---

### Viewing the AWS Config Dashboard

Use the **Dashboard** to see an overview of your resources, rules, and their compliance state. This page helps you quickly identify the top resources in your account, and if you have any rules or resources that are noncompliant.

After setup, AWS Config starts recording the specified resources and then evaluates them against your rules. It may take a few minutes for AWS Config to display your resources, rules, and their compliance states on the **Dashboard**.

**To use the AWS Config Dashboard**

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

2. Choose **Dashboard**.

3. Use the **Dashboard** to see an overview of your resources, rules, and their compliance state.

![Dashboard Image](image)

On the **Dashboard**, you can do the following:

A. View the total number of resources that AWS Config is recording.
B. View the resource types that AWS Config is recording, in descending order (the number of resources). Choose a resource type to go to the Resources inventory page.
C. Choose View all resources to go to the Resources inventory page.
D. View the number of noncompliant rules.
E. View the number of noncompliant resources.
F. View the top noncompliant rules, in descending order (the number of resources).
G. Choose View all noncompliant rules to go to the Rules page.

The Dashboard shows the resources and rules specific to your region and account. It does not show resources or rules from other regions or other AWS accounts.

Note
The Evaluate your AWS resource configuration using Config rules message can appear on the Dashboard for the following reasons:

• You haven’t set up AWS Config Rules for your account. You can choose Add rule to go to the Rules page.
• AWS Config is still evaluating your resources against your rules. You can refresh the page to see the latest evaluation results.
• AWS Config evaluated your resources against your rules and did not find any resources in scope. You can specify the resources for AWS Config to record in the Settings page. For more information, see Selecting Which Resources AWS Config Records (p. 62).
Viewing AWS Resource Configurations and Managing AWS Config

Use AWS Config for the following:

- View all the resources that AWS Config is recording in your account.
- Customize the types of resources that AWS Config records.
- View configuration changes over a specific period of time for a resource in AWS Config console and AWS CLI.
- View AWS resource configuration history
- View AWS resource compliance history
- View all the notifications that AWS Config sends to an Amazon SNS topic.
- Modify settings for your IAM role
- Modify or delete your delivery channels

Topics
- Components of a Configuration Item (p. 33)
- Record Configurations for Third-Party Resources (p. 34)
- Viewing AWS Resource Configurations and History (p. 40)
- Managing AWS Config (p. 56)
- Notifications that AWS Config Sends to an Amazon SNS topic (p. 76)

Components of a Configuration Item

A configuration item consists of the following components.

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
<th>Contains</th>
</tr>
</thead>
</table>
| Metadata  | Information about this configuration item | • Version 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 |
| Attributes¹ | Resource attributes | • Resource ID  
• List of key-value tags² for this resource  
• Resource type; see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9) |
Record Configurations for Third-Party Resources

Record configurations for third-party resources or custom resource types such as on premise servers, SAAS monitoring tools, and version control systems (like GitHub). You can publish the configuration data of third-party resources into AWS Config and view and monitor the resource inventory and configuration history using AWS Config console and APIs. Now, you can use AWS Config to manage all your resources and evaluate resource configuration for compliance against best practices using AWS Config rules. You can also create AWS Config rules or conformance packs to evaluate these third-party resources against best practices, internal policies, and regulatory policies.

**Note**
This feature is only available in the redesigned AWS Config console.

**Prerequisite:** The third-party resources or custom resource type must be registered using AWS CloudFormation.

**Topics**
Step 1: Setup Your Development Environment

Install and configure AWS CloudFormation AWS CLI. The AWS CLI allows you to model and register your custom resources. For more information, see Setting Up Your Environment for Developing Resource Providers.

Step 2: Model Your Resource

Create a resource provider schema that conforms to and validates the configuration of the resource type.

1. Use the `cfn init` command to create your resource provider project and generate the files it requires.

   ```bash
   $ cfn init
   Initializing new project
   ```

2. The `cfn init` command launches a wizard that walks you through setting up the project, including specifying the resource name. For this walkthrough, specify `MyCustomNamespace::Testing::WordPress`.

   ```
   Enter resource type identifier (Organization::Service::Resource):
   MyCustomNamespace::Testing::WordPress
   ```

3. Enter a package name for your resource.

   ```
   Enter a package name (empty for default 'com.custom.testing.wordpress'):
   com.custom.testing.wordpress
   Initialized a new project in /workplace/user/custom-testing-wordpress
   ```

4. Open the `custom-testing-wordpress.json` file that contains the schema for your resource. Copy and paste the following schema into `custom-testing-wordpress.json`.

   ```json
   {
     "typeName": "MyCustomNamespace::Testing::WordPress",
     "description": "An example resource that creates a website based on WordPress 5.2.2.",
     "sourceUrl": "The URL of the source code for this resource, if public.",
     "properties": {
       "Name": {
         "description": "A name associated with the website.",
         "type": "string",
         "pattern": "^[a-zA-Z0-9]{1,219}\Z",
         "minLength": 1, "maxLength": 219
         }
       },
     "SubnetId": {
       "description": "A subnet in which to host the website.",
       "pattern": "^[\(subnet-\[a-f0-9\]{13}\)|\(subnet-\[a-f0-9\]{8}\)\Z",
       "type": "string"
     }
   }
   ```
Step 3: Register Your Resource

AWS Config does not require resource provider handlers to perform configuration tracking for your resource. Run the following command to register your resource.

```
# cfn submit
```

For more information, see Registering Resource Providers for Use in AWS CloudFormation Templates.

Step 4: Publish Resource Configuration

Determine the configuration for MyCustomNamespace::Testing::WordPress.

```
{
    "Name": "MyWordPressSite",
    "SubnetId": "subnet-abcd0123",
    "InstanceId": "i-01234567",
    "PublicIp": "my-wordpress-site.com"
}
```

Determine the schema version id from AWS CloudFormation DescribeType.

In the AWS Config see if this resource configuration is accepted. To evaluate compliance you can write AWS Config rules using this resource.

Optional: To automate recording of configuration, implement a periodic or change-based configuration collectors.
Record and Delete a Configuration State for Third-Party Resources Using AWS CLI

The AWS CLI is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and use scripts to automate them.

To install the AWS CLI on your local machine, see Installing the AWS CLI in the AWS CLI User Guide.

If necessary, type `aws configure` to configure the AWS CLI.

**Topics**
- Record a Configuration Item (p. 37)
- Read the Configuration Item using AWS Config APIs (p. 37)
- Delete the Third-Party Resource (p. 38)

**Record a Configuration Item**

Record a configuration item for a third-party resource or a custom resource type using the following procedure:

Ensure you register the resource type `MyCustomNamespace::Testing::WordPress` with its matching schema.

1. Open a command prompt or a terminal window.
2. Type the following command:

```bash
aws configservice put-resource-config --resource-type MyCustomNamespace::Testing::WordPress --resource-id resource-001 --schema-version-id 00000001 --configuration '{
  "Id": "resource-001",
  "Name": "My example custom resource.",
  "PublicAccess": false
}'
```

**Read the Configuration Item using AWS Config APIs**

1. Open a command prompt or a terminal window.
2. Type the following command:

```bash
aws configservice list-discovered-resources --resource-type MyCustomNamespace::Testing::WordPress
```
3. Press Enter.

You should see output similar to the following:

```json
{
  "resourceIdentifiers": [ {
    "resourceType": "MyCustomNamespace::Testing::WordPress",
    "resourceId": "resource-001"  
  }
```
4. Type the following command:

```bash
aws configservice batch-get-resource-config --resource-keys '[]
}']
```

5. Press Enter.

You should see output similar to the following:

```json
{
  "unprocessedResourceKeys": [],
  "baseConfigurationItems": [
    {
      "configurationItemCaptureTime": 1569605832.673,
      "resourceType": "MyCustomNamespace::Testing::WordPress",
      "resourceId": "resource-001",
      "configurationStateId": "1569605832673",
      "awsRegion": "us-west-2",
      "version": "1.3",
      "supplementaryConfiguration": {},
      "configuration": "{Id":"resource-001","Name":"My example custom resource.","PublicAccess":false},
      "configurationItemStatus": "ResourceDiscovered",
      "accountId": "AccountId"
    }
  ]
}
```

### Delete the Third-Party Resource

You can record the configuration state for a third-party resource or custom resource type that you want to delete.

- Type the following command:

```bash
aws configservice delete-resource-config --resource-type MyCustomNamespace::Testing::WordPress --resource-id resource-002
```

If successful, the command executes with no additional output.

### Managing a Configuration State for Third-Party Resources Type Using APIs

You can manage a configuration state for third-party resources or custom resource type using `PutResourceConfig` and `DeleteResourceConfig` APIs. For more information, see the API Reference.

- `PutResourceConfig`
- `DeleteResourceConfig`
# Region Support

This feature is supported in the following regions:

<table>
<thead>
<tr>
<th>Region Name</th>
<th>Region</th>
<th>Endpoint</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>ap-south-1</td>
<td>config.ap-south-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>ap-northeast-2</td>
<td>config.ap-northeast-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>ap-southeast-1</td>
<td>config.ap-southeast-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>ap-southeast-2</td>
<td>config.ap-southeast-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>ap-northeast-1</td>
<td>config.ap-northeast-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>ca-central-1</td>
<td>config.ca-central-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>eu-central-1</td>
<td>config.eu-central-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>eu-west-1</td>
<td>config.eu-west-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (London)</td>
<td>eu-west-2</td>
<td>config.eu-west-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Paris)</td>
<td>eu-west-3</td>
<td>config.eu-west-3.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Stockholm)</td>
<td>eu-north-1</td>
<td>config.eu-north-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>South America (São Paulo)</td>
<td>sa-east-1</td>
<td>config.sa-east-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US East (N. Virginia)</td>
<td>us-east-1</td>
<td>config.us-east-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US East (Ohio)</td>
<td>us-east-2</td>
<td>config.us-east-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US West (N. California)</td>
<td>us-west-1</td>
<td>config.us-west-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>us-west-2</td>
<td>config.us-west-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
</tbody>
</table>
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 also view compliance state changes for resources as evaluated by AWS Config Rules displayed in a timeline.

Topics
- Looking Up Resources That Are Discovered by AWS Config (p. 40)
- Viewing Configuration Details (p. 41)
- Viewing Compliance History for Resources as Evaluated by AWS Config Rules (p. 47)
- Delivering Configuration Snapshot to an Amazon S3 Bucket (p. 51)

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 resources that AWS Config is not currently recording. AWS Config discovers supported resource types only. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

Looking Up Resources (AWS Config Console)

You can use resource types or tag information to look up resources in the AWS Config console.

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 look up:
   - Choose Resources and then choose one or more resource types in the list. This list includes resource types that AWS Config supports. To narrow results, type a resource ID or, if applicable, a resource name in the next box. You can also choose Include deleted resources.
   - Choose Tag and type a tag key that is applied to your resources, such as CostCenter. To narrow results, type a tag value in the next box.
3. After you specify the search options, choose Look up.
4. AWS Config lists the resources that match your search options. You can see the following information about the resources:
   - Resource identifier – The resource identifier might be a resource ID or a resource name, if applicable. Choose the resource identifier link to view the resource details page.
   - Resource type – The type of the resource is listed.
   - Compliance – The status of the resource that AWS Config evaluated against your rule.

Looking Up Resources (AWS CLI)

You can use the AWS CLI to list resources that AWS Config has discovered.
To look up resources (AWS CLI)

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

  **Example**

  ```
  $ 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 Viewing Configuration History (p. 43).

Looking up Resources (AWS Config API)

You specify a resource type, and AWS Config returns a list of resource identifiers for resources of that type. For more information, see `ResouceIdentifier` in the AWS Config API Reference.

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

You can view the configuration, relationships, and number of changes made to a resource in the AWS Config console. You can view the configuration history for a resource using AWS CLI.

Viewing Configuration Details (Console)

When you look up resources on the Resource inventory page, click the resource name or ID in the resource identifier column to view 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 that the recording was made.
### 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. Click to return the timeline to the current time.
4. Click to view a configuration item by specifying a date (and, if needed, time) and then choose **Apply**.
5. Click to navigate to the **Changes** section. The number that follows **Changes** is the number of configuration changes that occurred for the resource between the selected time period and the previous block.
6. Click to navigate to the **CloudTrail events** section. The number that follows **Events** is the number of API events that occurred for the resource between the selected time period and the previous block. You can see the API events that AWS CloudTrail logged for the last 90 days. CloudTrail events that occurred prior to the last 90 days can’t be viewed in the timeline.

For more information, see Viewing Events with CloudTrail API Activity History in the *AWS CloudTrail User Guide*.

### Note

CloudTrail events may not be available for the following reasons:

- Verify that you have sufficient read permissions for CloudTrail. For more information, see Read-only access (p. 92).
- There is a service issue and CloudTrail events can’t be displayed at this time. Try refreshing the page.
- You don’t have a CloudTrail trail in this region or your trail is not enabled for logging. For more information, see Creating a Trail for the First Time in the *AWS CloudTrail User Guide*. 
Timeline Navigation Tips for the Selected Resource

The following are tips for using the timeline to view information about the selected resource.

• Use the arrows at either end of the timeline to view the timeline blocks for configuration items that were recorded in other time periods.

• Choose Configuration Details to view the description of the selected resource.

• Choose Relationships to see a list of supported resources in the account that are related to the selected resource. If the Relationships section doesn’t expand, the selected resource was not related to another resource that was in your account during the selected time period.

For more information, see Resource Relationship (p. 4).

• If changes are indicated for the selected time period, choose 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.

• Choose CloudTrail events to view information about API calls that involve the resource, such as the event time, the user name, and the event name. For example, if AWS Config is recording IAM resource types, and an IAM role is updated, you can view the event to see the UpdateRole in the Event name column.

• In the View event column, you can also choose the CloudTrail link to view more information about the event in the CloudTrail console. You must create a trail and enable logging for CloudTrail to view the events in the AWS Config timeline.

• Choose View Details to view the configuration information in text or JSON 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. 33).

• Choose Manage resources to go to the console for the selected resource. If you make a change to the resource, go back to the AWS Config console and choose Now to see the changes. It can take up to 10 minutes to refresh the details page for the resource.

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 on these details pages is limited and ongoing configuration changes are not shown.

Viewing Configuration Details (AWS 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. You can use the AWS CLI to view history of configuration items for each resource.

Viewing Configuration History

Type the get-resource-config-history command and specify the resource type and the resource ID, for example:

```bash
$ aws configservice get-resource-config-history --resource-type AWS::EC2::SecurityGroup --resource-id sg-6fbb3807
```

```json
{
  "configurationItems": [
    {
      "configurationItemCaptureTime": 1414708529.9219999,
      "relationships": [
        {
          "resourceType": "AWS::EC2::Instance",
```
For detailed explanation of the response fields, see Components of a Configuration Item (p. 33) and AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

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. The 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 change.

```json
{
  "fileVersion": "1.0",
  "requestId": "asudf8ow-4e34-4f32-afeb-0ace5bf3trye",
  "configurationItems": [
    {
      "snapshotVersion": "1.0",
      "resourceId": "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"],
      "availabilityZone": "us-west-2b",
      "resourceType": "AWS::EC2::Volume",
      "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",
        "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",
      "relatedEvents": [
        "06c12a39-eb35-11de-ae07-adb69edbb1e4", "c376e30d-71a2-4694-89b7-a5a04ad92281"
      ]
    }
  ]
}
```
"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"
],
"availabilityZone": "us-west-2b",
"resourceType": "AWS::EC2::Volume",
"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",
    "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"
],
"availabilityZone": "us-west-2b",
"resourceType": "AWS::EC2::Volume",
"tags": {},
"relationships": [
AWS Config supports storing compliance state changes of resources as evaluated by AWS Config Rules. The resource compliance history is presented in the form of a timeline. The timeline captures changes as ConfigurationItems over a period of time for a specific resource. The Compliance timeline is available in the AWS Config console adjacent to the Configuration timeline.

You can opt in or opt out to record all resource types in AWS Config. If you have opted to record all resource types, AWS Config will automatically begin to record the resource compliance history as evaluated by AWS Config Rules. You can select all the resources or specific types of resources for which you want AWS Config to record configuration changes. By default, AWS Config records the configuration changes for all supported resources.

Topics
- Recording Resource Types in the AWS Console (p. 48)
- Viewing Compliance Timeline Using Resources (p. 48)
- Viewing Compliance Timeline Using Rules (p. 50)
- Querying Resource Compliance History (p. 50)
Recording Resource Types in the AWS Console

On the Settings page, for Resource types to record, specify the AWS resource types 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 resource type, AWS Config 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 global resource type, AWS Config automatically starts recording resources of that type.
- **Specific types** – AWS Config records configuration changes for only the AWS resource types that you specify. Choose resource type `Config:ResourceCompliance` to record compliance history.

Viewing Compliance Timeline Using Resources

Access the compliance timeline by selecting a specific resource from the Resource inventory page.

1. Select the Resources from the left navigation.
2. On the Resource inventory page, select all the existing resources from the drop-down and if appropriate, select include deleted resources.
3. Click Lookup.

   The table displays the resource identifier for the resource type and the resource compliance status for that resource. The resource identifier might be a resource ID or a resource name, if applicable.
4. Select the resource from the resource identifier column.
5. Select the Compliance timeline from the Resource actions drop-down.

   The compliance timeline is displayed.
Note
Alternatively, on the Resource inventory page, you can directly click the resource name. The Resource details page is displayed. To access the compliance timeline from the resource details page, click the **Compliance timeline** button.
Viewing Compliance History

Access the compliance timeline by selecting a specific rule from the Rule page.

1. Select the **Rules** from the left navigation.
2. On the Rule page, click the name of the rule evaluating your relevant resources. If no rules are displayed on the screen, add rules using the **Add rule** button.
3. On the Rule details page, select the resources from the Resources evaluated table.
4. Select **Compliance timeline** from the Resource actions drop-down. The compliance timeline is displayed.

Querying Resource Compliance History

Query the resource compliance history using `get-resource-config-history` using the resource type `AWS::Config::ResourceCompliance`.

```
aws configservice get-resource-config-history --resource-type AWS::Config::ResourceCompliance --resource-id AWS::S3::Bucket/configrules-bucket
```

You should see output similar to the following:

```
{
"configurationItems": [
{
"configurationItemCaptureTime": 1539799966.921,
"relationships": [
{
"resourceType": "AWS::S3::Bucket",
"resourceId": "configrules-bucket",
"relationshipName": "Is associated with "
}
]
"tags": {}
}
```
Delivering Configuration Snapshot to an Amazon S3 Bucket

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.

Topics
- Delivering Configuration Snapshot (p. 51)
- Example Configuration Snapshot from AWS Config (p. 51)
- Verifying Delivery Status (p. 55)
- Viewing Configuration Snapshot in Amazon S3 bucket (p. 56)

Delivering Configuration Snapshot

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.

Type 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"
}
```

Example Configuration Snapshot from 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",
  "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"],
  "availabilityZone": "us-west-2b",
  "resourceType": "AWS::EC2::Volume",
  "tags": {},
  "relationships": [  
  // omitted
  ]
  },
"configuration": {  
  "volumeId": "vol-ce676ccc",
  "size": 1,
  "snapshotId": "",
  "availabilityZone": "us-west-2b",
  "state": "in-use",
  "attachments": [  
  // omitted
  ],
  "tags": [  
  // omitted
  ],
  "volumeType": "standard"
  }
},
{  
  "configurationItemVersion": "1.0",
  "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",
  "attachments": [  
  // omitted
  ]
  },
"tags": [  
  // omitted
  ],
  "configurationStateId": "3e660fdf-4e34-4f32-afeb-0ace5bf3d63a",
  "configurationItemStatus": "OK",
}
"relatedEvents": [
    "06c12a39-eb35-11de-ae07-adb69edbb1e4",
    "c376e30d-71a2-4694-89b7-a5a04ad92381"
],
"availabilityZone": "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.amazonaws.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": "in-use"}}]
"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-47b4cf5c",
    "vpcId": "vpc-41b4cf2a",
    "description": "",
    "ownerId": "12345678910",
    "status": "in-use",
    "privateIpAddress": "172.31.21.63",
    "privateDnsName": "ip-172-31-21-63.us-west-2.compute.internal",
    "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.compute.amazonaws.com",
      "ipOwnerId": "amazon"
    },
    "privateIpAddresses": [
AWS Conﬁg Developer Guide
Delivering Conﬁguration Snapshot
{
west-2.compute.internal",

west-2.compute.amazonaws.com",

]

}

]

}

}

}

"privateIpAddress": "172.31.21.63",
"privateDnsName": "ip-172-31-21-63.us"primary": true,
"association": {
"publicIp": "54.218.4.189",
"publicDnsName": "ec2-54-218-4-189.us}

"ipOwnerId": "amazon"

}
],
"ebsOptimized": false

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

Verifying Delivery Status
Type 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.

55


Viewing Configuration Snapshot in 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.

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
- Managing the Delivery Channel (p. 56)
- Updating the IAM Role Assigned to AWS Config (p. 59)
- Managing the Configuration Recorder (p. 60)
- Selecting Which Resources AWS Config Records (p. 62)
- Recording Software Configuration for Managed Instances (p. 65)
- Querying the Current Configuration State of AWS Resources (p. 66)
- Deleting AWS Config Data (p. 73)

Managing the Delivery Channel

As AWS Config continually records the changes that occur to your AWS resources, it sends notifications and updated configuration states through the delivery channel. You can manage the delivery channel to control where AWS Config sends configuration updates.

You can have only one delivery channel per region per AWS account, and the delivery channel is required to use AWS Config.

Updating the Delivery Channel

When you update the delivery channel, you can set the following options:

- The Amazon S3 bucket to which AWS Config sends configuration snapshots and configuration history files.
- How often AWS Config delivers configuration snapshots to your Amazon S3 bucket.
- The Amazon SNS topic to which AWS Config sends notifications about configuration changes.

To update the delivery channel (console)

- You can use the AWS Config console to set the Amazon S3 bucket and the Amazon SNS topic for your delivery channel. For steps to manage these settings, see Setting Up AWS Config with the Console (p. 22).
The console does not provide options to rename the delivery channel, set the frequency for configuration snapshots, or delete the delivery channel. To do these tasks, you must use the AWS CLI, the AWS Config API, or one of the AWS SDKs.

To update the delivery channel (AWS CLI)

1. Use the `put-delivery-channel` command:

   ```
   $ aws configservice put-delivery-channel --delivery-channel file://deliveryChannel.json
   ```

   The `deliveryChannel.json` file specifies the delivery channel attributes:

   ```json
   {
       "name": "default",
       "s3BucketName": "config-bucket-123456789012",
       "configSnapshotDeliveryProperties": {
           "deliveryFrequency": "Twelve_Hours"
       }
   }
   ```

   This example sets the following attributes:

   - **name** – The name of the delivery channel. By default, AWS Config assigns the name `default` to a new delivery channel.

     You cannot update the delivery channel name with the `put-delivery-channel` command. For the steps to change the name, see Renaming the Delivery Channel (p. 58).

   - **s3BucketName** – The name of the Amazon S3 bucket to which AWS Config delivers configuration snapshots and configuration history files.

     If you specify a bucket that belongs to another AWS account, that bucket must have policies that grant access permissions to AWS Config. For more information, see Permissions for the Amazon S3 Bucket (p. 102).

   - **snsTopicARN** – The Amazon Resource Name (ARN) of the Amazon SNS topic to which AWS Config sends notifications about configuration changes.

     If you choose a topic from another account, that topic must have policies that grant access permissions to AWS Config. For more information, see Permissions for the Amazon SNS Topic (p. 104).

   - **configSnapshotDeliveryProperties** – Contains the `deliveryFrequency` attribute, which sets how often AWS Config delivers configuration snapshots.

2. (Optional) You can use the `describe-delivery-channels` command to verify that the delivery channel settings are updated:

   ```
   $ aws configservice describe-delivery-channels
   ```

   ```json
   {
     "DeliveryChannels": [
       {
         "configSnapshotDeliveryProperties": {
           "deliveryFrequency": "Twelve_Hours"
         },
         "name": "default",
         "s3BucketName": "config-bucket-123456789012"
       }
     ]
   }
   ```
Renaming the Delivery Channel

To change the delivery channel name, you must delete it and create a new delivery channel with the desired name. Before you can delete the delivery channel, you must temporarily stop the configuration recorder.

The AWS Config console does not provide the option to delete the delivery channel, so you must use the AWS CLI, the AWS Config API, or one of the AWS SDKs.

To rename the delivery channel (AWS CLI)

1. Use the `stop-configuration-recorder` command to stop the configuration recorder:

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

2. Use the `describe-delivery-channels` command, and take note of your delivery channel's attributes:

   ```
   $ aws configservice describe-delivery-channels
   {
   "DeliveryChannels": [
   {
   "configSnapshotDeliveryProperties": {
   "deliveryFrequency": "Twelve_Hours"
   },
   "name": "default",
   "s3BucketName": "config-bucket-123456789012"
   }
   ]
   }
   ```

3. Use the `delete-delivery-channel` command to delete the delivery channel:

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

4. Use the `put-delivery-channel` command to create a delivery channel with the desired name:

   ```
   $ aws configservice put-delivery-channel --delivery-channel file://deliveryChannel.json
   ```

   The deliveryChannel.json file specifies the delivery channel attributes:

   ```
   {
   "name": "myCustomDeliveryChannelName",
   "s3BucketName": "config-bucket-123456789012",
   "configSnapshotDeliveryProperties": {
   "deliveryFrequency": "Twelve_Hours"
   }
   }
   ```

5. Use the `start-configuration-recorder` command to resume recording:
Updating the IAM Role Assigned to AWS Config

You can update the IAM role assumed by AWS Config any time. Before you update the IAM role, ensure that you have created a new role to replace the old one. You must attach policies to the new role 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 Creating an IAM Role (p. 26).

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

Updating the IAM Role

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

**To update the IAM role in a region where rules are supported (console)**

If you are using AWS Config in a region that supports AWS Config rules, complete the following steps. For the list of supported regions, see AWS Config Regions and Endpoints in the Amazon Web Services General Reference.

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Settings in the navigation pane.
3. In the AWS Config role, section, choose the IAM role:
   - 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, choose an IAM role in your account. AWS Config will attach the required policies. For more information, see Permissions for the IAM Role Assigned to AWS Config (p. 99).

   **Note**
   Check the box if you want to use the IAM role as it. AWS Config will not attach policies to the role.
4. Choose Save.

**To update the IAM role in a region where rules are not supported (console)**

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 settings icon (⚙️).
3. Choose Continue.
4. In the AWS Config is requesting permissions to read your resources' configuration page, choose View Details.
5. In the Role Summary section, choose the IAM role:
Managing the Configuration Recorder

AWS Config uses the configuration recorder to detect changes in your resource configurations and capture these changes as configuration items. You must create a configuration recorder before AWS Config can track your resource configurations.

If you set up AWS Config by using the console or the AWS CLI, AWS Config automatically creates and then starts the configuration recorder for you. For more information, see Getting Started With AWS Config (p. 22).

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. 62).

You are charged service usage fees when AWS Config starts recording configurations. For pricing information, see AWS Config Pricing. To control costs, you can stop recording by stopping the configuration recorder. After you stop recording, you can continue to access the configuration information that was already recorded. You will not be charged AWS Config usage fees until you resume recording.

When you start the configuration recorder, AWS Config takes an inventory of all AWS resources in your account.

Managing the Configuration Recorder (Console)

You can use the AWS Config console to stop or start the configuration recorder.

To stop or start the configuration recorder

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Settings in the navigation pane.
3. Stop or start the configuration recorder:
   - If you want to stop recording, under Recording is on, choose Turn off. When prompted, choose Continue.
   - If you want to start recording, under Recording is off, choose Turn on. When prompted, choose Continue.
Managing the Configuration Recorder (AWS CLI)

You can use the AWS CLI to stop or start the configuration recorder. You can also rename or delete the configuration recorder using the AWS CLI, the AWS Config API, or one of the AWS SDKs. The following steps help you use the AWS CLI.

To stop the configuration recorder

- Use the `stop-configuration-recorder` command:

  ```sh
  $ aws configservice stop-configuration-recorder --configuration-recorder-name configRecorderName
  ```

To start the configuration recorder

- Use the `start-configuration-recorder` command:

  ```sh
  $ aws configservice start-configuration-recorder --configuration-recorder-name configRecorderName
  ```

To rename the configuration recorder

To change the configuration recorder name, you must delete it and create a new configuration recorder with the desired name.

1. Use the `describe-configuration-recorders` command to look up the name of your current configuration recorder:

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

   ```json
   {  
     "ConfigurationRecorders": [  
       {  
         "roleARN": "arn:aws:iam::012345678912:role/myConfigRole",  
         "name": "default"  
       }  
     ]  
   }
   ```

2. Use the `delete-configuration-recorder` command to delete your current configuration recorder:

   ```sh
   $ aws configservice delete-configuration-recorder --configuration-recorder-name default
   ```

3. Use the `put-configuration-recorder` command to create a configuration recorder with the desired name:

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

4. Use the `start-configuration-recorder` command to resume recording:

   ```sh
   $ aws configservice start-configuration-recorder --configuration-recorder-name configRecorderName
   ```
To delete the configuration recorder

- Use the `delete-configuration-recorder` command:

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

Selecting Which Resources AWS Config Records

AWS Config continuously detects when any resource of a supported type is created, changed, or deleted. AWS Config records these events as configuration items. You can customize AWS Config to record 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 AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

Recording 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 region and can be used in all regions. The global resource types that AWS Config supports are IAM users, groups, roles, and customer managed policies.

**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.

Recording Specific Resource Types

If you don’t want AWS Config to record the changes for all supported resources, you can customize it to record changes for only specific types. AWS Config records configuration changes for the types of resources that you specify, including the creation and deletion of such resources.

If a resource is not recorded, AWS Config captures only the creation and deletion of that resource, and no other details, at no cost to you. When a nonrecorded resource is created or deleted, AWS Config sends a notification, and it displays the event on the resource details page. The details page for a nonrecorded resource provides null values for most configuration details, and it does not provide information about relationships and configuration changes.

The relationship information that AWS Config provides for recorded resources is not limited because of missing data for nonrecorded resources. If a recorded resource is related to a nonrecorded resource, that relationship is provided in the details page of the recorded resource.

You can stop AWS Config from recording a type of resource 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.

AWS Config rules can be used to evaluate compliance for only those resources that AWS Config records.

Selecting Resources (Console)

You can use the AWS Config console to select the types of resources that AWS Config records.
Selecting Which Resources are Recorded

**To select resources**

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

2. Open the **Settings** page:
   - If you are using AWS Config in a region that supports AWS Config rules, choose **Settings** in the navigation pane. For the list of supported regions, see [AWS Config Regions and Endpoints](https://docs.aws.amazon.com/config/latest/developerguide/configuration-recorder-endpoints.html) in the Amazon Web Services General Reference.
   - Otherwise, choose the settings icon (⚙️) on the **Resource inventory** 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 resource types** – 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. Save your changes:
   - If you are using AWS Config in a region that supports AWS Config rules, choose **Save**.
   - Otherwise, choose **Continue**. In the **AWS Config is requesting permissions to read your resources' configuration** page, choose **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:

   ```bash
   $ 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:

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

   The following is an example response:

   ```json
   {
   "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:

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

   The `recordingGroup.json` file specifies which types of resources AWS Config will record:

   ```json
   {
   "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:

   ```bash
   $ aws configservice describe-configuration-recorders
   ```
Recording Software Configuration for Managed Instances

You can use AWS Config to record software inventory changes on Amazon EC2 instances and on-premises servers. This enables you to see the historical changes to software configuration. For example, when a new Windows update is installed on a managed Windows instance, AWS Config records the changes and then sends the changes to your delivery channels, so that you are notified about the change. With AWS Config, you can see the history of when Windows updates were installed for the managed instance and how they changed over time.

You must complete the following steps to record software configuration changes:

- Turn on recording for the managed instance inventory resource type in AWS Config.
- Configure EC2 and on-premises servers as managed instances in AWS Systems Manager. A managed instance is a machine that has been configured for use with Systems Manager.
- Initiate collection of software inventory from your managed instances using the Systems Manager Inventory capability.

You can also use AWS Config rules to monitor software configuration changes and be notified whether the changes are compliant or noncompliant against your rules. For example, if you create a rule that checks whether your managed instances have a specified application, and an instance doesn’t have that application installed, AWS Config flags that instance as noncompliant against your rule. For a list of AWS Config managed rules, see List of AWS Config Managed Rules (p. 112).

To enable recording of software configuration changes in AWS Config:

1. Turn on recording for all supported resource types or selectively record the managed instance inventory resource type in AWS Config. For more information, see Selecting Which Resources AWS Config Records (p. 62).

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"
      }
   ]
}
```
2. Launch an Amazon EC2 instance with an instance profile for Systems Manager that includes the `AmazonSSMManagedInstanceCore` managed policy. This AWS managed policy enables an instance to use Systems Manager service core functionality.

For information about other policies you can add to the instance profile for Systems Manager, see Create an IAM Instance Profile for Systems Manager in the *AWS Systems Manager User Guide*.

**Important**

SSM Agent is Amazon software that must be installed on a managed instance in order to communicate with the Systems Manager in the cloud. If your EC2 instance was created from an AMI for one of the following operating systems, the agent is preinstalled:

- Windows Server 2003-2012 R2 AMIs published in November 2016 or later
- Windows Server 2016 and 2019
- Amazon Linux
- Amazon Linux 2
- Ubuntu Server 16.04
- Ubuntu Server 18.04

On EC2 instances that were not created from an AMI with the agent preinstalled, you must install the agent manually. For information, see the following topics in the *AWS Systems Manager User Guide*:

- Installing and Configuring SSM Agent on Windows Instances
- Installing and Configuring SSM Agent on Amazon EC2 Linux Instances

3. Initiate inventory collection as described in Configuring Inventory Collection in the *AWS Systems Manager User Guide*. The procedures are the same for Linux and Windows instances.

AWS Config can record configuration changes for the following inventory types:

- **Applications** – A list of applications for managed instances, such as antivirus software.
- **AWS components** – A list of AWS components for managed instances, such as the AWS CLI and SDKs.
- **Instance information** – Instance information such as OS name and version, domain, and firewall status.
- **Network configuration** – Configuration information such as IP address, gateway, and subnet mask.
- **Windows Updates** – A list of Windows updates for managed instances (Windows instances only).

**Note**

AWS Config doesn’t support recording the custom inventory type at this time.

Inventory collection is one of many Systems Manager capabilities, which are grouped in the categories *Operations Management*, *Actions & Change*, *Instances & Nodes*, and *Shared Resources*. For more information, see What is Systems Manager? and Systems Manager Capabilities in the *AWS Systems Manager User Guide*.

**Querying the Current Configuration State of AWS Resources**

You can use AWS Config to query the current configuration state of AWS resources based on configuration properties for a single account and Region or across multiple accounts and Regions. You
can perform ad hoc, property-based queries against current AWS resource state metadata across all
resources that AWS Config supports. The **advanced query** feature provides a single query endpoint and
a powerful query language to get current resource state metadata without performing service-specific
describe API calls. You can use configuration aggregators to run the same queries from a central account
across multiple accounts and AWS Regions.

AWS Config uses a subset of structured query language (SQL) SELECT syntax to perform property-based
queries and aggregations on the current configuration item (CI) data. The queries range in complexity
from simple matches against tag and/or resource identifiers, to more complex queries, such as viewing
all Amazon S3 buckets that have versioning disabled. This allows you to query exactly the current
resource state you need without performing AWS service-specific API calls.

You can use advanced query for:

- Inventory management; for example, to retrieve a list of Amazon EC2 instances of a particular size.
- Security and operational intelligence; for example, to retrieve a list of resources that have a specific
  configuration property enabled or disabled.
- Cost optimization; for example, to identify a list of Amazon EBS volumes that are not attached to any
  EC2 instance.

**Features**

The query language supports querying AWS resources based on CI properties of all AWS resource types
supported by AWS Config, including configuration data, tags, and relationships. It is a subset of SQL
SELECT command with limitations, as mentioned in the following section. It supports aggregation
functions such as **AVG**, **COUNT**, **MAX**, **MIN**, and **SUM**.

**Limitations**

As a subset of SQL SELECT, the query syntax has following limitations:

- No support for **ALL**, **AS**, **DISTINCT**, **FROM**, **HAVING**, **JOIN**, and **UNION** keywords in a query.
- When querying against multiple properties within an array of objects, matches are computed
  against all array elements (that is, arrays of objects are flattened for indexing). For example, if the
database had a document of the form `{ "a": [ { "b": 1, "c": 2, ... }, { "b": 3, "c":
4, ... } ] }`, a query like `SELECT a WHERE a.b=1 AND a.c=4` would match this document,
even though no single array element had the specified values of `{ "b": 1, "c": 4 }`.
- The **SELECT** all columns shorthand (that is **SELECT ***) selects only the top-level, scalar properties
  of a CI. The scalar properties returned are `accountId`, `awsRegion`, `arn`, `availabilityZone`,
  `configurationItemCaptureTime`, `resourceCreationTime`, `resourceId`, `resourceName`,
  `resourceType`, and `version`.
- Wildcard limitations:
  - Wildcards are supported only for property values and not for property keys (for example, `...WHERE
    someKey LIKE 'someValue%'` is supported but `...WHERE 'someKey%' LIKE 'someValue%
    ' is not supported).
  - Support for only suffix wildcards (for example, `...LIKE 'AWS::EC2:: Instance'` is supported but
    `...LIKE '% EC2 Instance'` is not supported).
  - Wildcard matches must be at least three characters long (for example, `...LIKE 'ab%'` is not
    allowed but `...LIKE 'abc%'` is allowed).
- Aggregation limitations:
  - **GROUP BY** clauses in aggregations may contain only a single property.
  - Aggregate functions can accept only a single argument or property.
  - Aggregate functions cannot take other functions as arguments.
• No support for pagination for aggregate queries; a maximum of 500 results is returned.
• No support for `HAVING` clauses in aggregations.

Query Using the SQL Query Editor (Console)

Prerequisites

You must have permissions for `config:SelectResourceConfig` and `config:SelectAggregateResourceConfig` APIs. For more information, see `SelectResourceConfig API` and `SelectAggregateResourceConfig API`.

You must have permissions for the `AWSConfigUserAccess` IAM managed policy. For more information, see `Granting Permissions for AWS Config Administration (p. 90)`.

1. Sign in to the AWS Management Console and open the AWS Config console at `https://console.aws.amazon.com/config/`.
2. Choose `Advanced queries` from the left navigation to query your resource configurations for a single account and Region or for multiple accounts and Regions. On the `Advanced queries` page, you can use sample queries or create your own query.
   • To use a sample query as a starting point, choose an appropriate query from the list of queries. The query is displayed in the SQL query editor. If required, you can edit this query.
   OR
   • To create your own query, choose `New query`.

   **Important**
   An updated list of properties and their data types is available in GitHub.

3. On the `Query editor` page, for `New query`, create your own query for this account and Region. You can also select an appropriate aggregator to create a query for multiple accounts and Regions.

   **Note**
   To run a query on an aggregator, create an aggregator. For more information, see `Setting Up an Aggregator Using the Console (p. 231)`. If you already have an aggregator set up, in the query scope, choose the aggregator to run an advanced query on that aggregator. When you select an aggregator, consider adding the AWS account ID and AWS Region in the query statement to view that information in the results.

4. Choose `Run`. The query results are displayed in the table below the query editor.
5. Choose `Export as` to export the query results in CSV or JSON format.

   **Note**
   The query results are paginated. When you choose export, up to 500 results are exported. You can also use the APIs to retrieve all the results. The results are paginated and you can retrieve 100 results at a time.

Query Using the SQL Query Editor AWS CLI

To install the AWS Command Line Interface (AWS CLI) on your local computer, see `Installing the AWS CLI in the AWS CLI User Guide`.

To query your resource configuration data using the query editor (AWS CLI) for a single account and Region

1. Open a command prompt or a terminal window.
2. Type the following command to query your resource configuration data.
aws configservice select-resource-config --expression "SELECT resourceId WHERE resourceType='AWS::EC2::Instance'"

Depending on your query, the output looks like the following.

```json
{
  "QueryInfo": {
    "SelectFields": [
      {
        "Name": "resourceId"
      }
    ],
  },
  "Results": [
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}"
  ]
}
```

To query your resource configuration data using the query editor (AWS CLI) for multiple accounts and Regions

1. Open a command prompt or a terminal window.
2. Type the following command to query your resource configuration data.

```bash
aws configservice select-aggregate-resource-config --expression "SELECT resourceId WHERE resourceType='AWS::EC2::Instance'
```

Depending on your query, the output looks like the following.

```json
{
  "QueryInfo": {
    "SelectFields": [
      {
        "Name": "resourceId"
      }
    ],
  },
  "Results": [
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}",
    "{"resourceId": "ResourceId"}"
  ]
}
```

**Note**
While using the AWS::IAM::User resource type in an advanced query, use `awsRegion = 'global'`. 
Example Queries

Query to list all EC2 instances with AMI ID ami-12345

```
SELECT
    resourceId,
    resourceType,
    configuration.instanceType,
    configuration.placement.tenancy,
    configuration.imageId,
    availabilityZone
WHERE
    resourceType = 'AWS::EC2::Instance'
    AND configuration.imageId = 'ami-12345'
```

Results

```json
{
  "QueryInfo": {
    "SelectFields": [
      {
        "Name": "resourceId"
      },
      {
        "Name": "resourceType"
      },
      {
        "Name": "configuration.instanceType"
      },
      {
        "Name": "configuration.placement.tenancy"
      },
      {
        "Name": "configuration.imageId"
      },
      {
        "Name": "availabilityZone"
      }
    ],
    "Results": [
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2c",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2a",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2c",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2c",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t1.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2a",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2c",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2c",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t1.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2a",
          "resourceType": "AWS::EC2::Instance"},
        {
          "resourceId": "resourceid",
          "configuration": {
            "imageId": "ami-12345",
            "instanceType": "t2.micro",
            "placement": {
              "tenancy": "default"},
            "availabilityZone": "us-west-2c",
          "resourceType": "AWS::EC2::Instance"},
        }
    ]
}
```
Query for count of resources grouped by their AWS Config rules compliance status

```sql
SELECT
    configuration.complianceType,
    COUNT(*)
WHERE
    resourceType = 'AWS::Config::ResourceCompliance'
GROUP BY
    configuration.complianceType
```

Result

```
{
    "QueryInfo": {
        "SelectFields": [
            {
                "Name": "configuration.complianceType"
            },
            {
                "Name": "COUNT(*)"
            }
        ],
        "Results": [
            {
                "COUNT(*)": 163,
                "configuration": {
                    "complianceType": "NON_COMPLIANT"
                }
            },
            {
                "COUNT(*)": 2,
                "configuration": {
                    "complianceType": "COMPLIANT"
                }
            }
        ]
    }
}
```

Query to get counts of AWS resources grouped by account ID

```
aws configservice select-aggregate-resource-config --expression "SELECT COUNT(*), accountId group by accountId" --configuration-aggregator-name my-aggregator
```

Result

```
{
    "Results": [
        {
            "COUNT(*)": 2407,
            "accountId": "accountId"
        },
        {
            "COUNT(*)": 726,
            "accountId": "accountId"
        }
    ],
    "QueryInfo": {
        "SelectFields": [
            {
                "Name": "COUNT(*)"
            },
            {
                "Name": "accountId"
            }
        ]
    }
}
```

Query to list all EC2 volumes that are not in use

```
SELECT
    resourceId,
    accountId,
```
awsRegion,
resourceType,
configuration.volumeType,
configuration.size,
resourceCreationTime,
tags,
configuration.encrypted,
configuration.availabilityZone,
configuration.state.value
WHERE
resourceType = 'AWS::EC2::Volume'
AND
configuration.state.value <> 'in-use'

Result

```json
{
"Results": [
   {
      "accountId": "accountId",
      "resourceId": "vol-0174de9c962f6581c",
      "awsRegion": "us-west-2",
      "configuration": {
         "volumeType": "gp2",
         "encrypted": false,
         "size": 100.0,
         "state": {
            "value": "available"
         },
         "availabilityZone": "us-west-2a",
         "resourceCreationTime": "2020-02-21T07:39:43.771Z",
         "tags": [],
         "resourceType": "AWS::EC2::Volume"
      }
   },
   {
      "accountId": "accountId",
      "resourceId": "vol-0cbeb652a74af2f8f",
      "awsRegion": "us-east-1",
      "configuration": {
         "volumeType": "gp2",
         "encrypted": false,
         "size": 100.0,
         "state": {
            "value": "available"
         },
         "availabilityZone": "us-east-1a",
         "resourceCreationTime": "2020-02-21T07:39:43.771Z",
         "tags": [],
         "resourceType": "AWS::EC2::Volume"
      }
   },
   {
      "accountId": "accountId",
      "resourceId": "vol-0a49952d528ec0ba2",
      "awsRegion": "ap-south-1",
      "configuration": {
         "volumeType": "gp2",
         "encrypted": false,
         "size": 100.0,
         "state": {
            "value": "available"
         },
         "availabilityZone": "ap-south-1a",
         "resourceCreationTime": "2020-02-21T07:39:31.800Z",
         "tags": [],
         "resourceType": "AWS::EC2::Volume"
      }
   }
],
"QueryInfo": {
   "SelectFields": [
      {
         "Name": "resourceId"
      },
      {
         "Name": "accountId"
      },
      {
         "Name": "awsRegion"
      },
      {
         "Name": "resourceType"
      },
      {
         "Name": "configuration.volumeType"
      },
      {
         "Name": "configuration.size"
      },
      {
         "Name": "resourceCreationTime"
      },
      {
         "Name": "tags"
      },
      {
         "Name": "configuration.encrypted"
      }
   ]
}
"Name": "configuration.availabilityZone"
},
{
   "Name": "configuration.state.value"
}
}
]
}

Query Components

The SQL SELECT query components are as follows.

**Synopsis**

```sql
SELECT property [, ...] 
[ WHERE condition ] 
[ GROUP BY property ] 
[ ORDER BY property [ ASC | DESC ] [, property [ ASC | DESC ] ...] ]
```

**Parameters**

[ **WHERE condition** ]

Filters results according to the condition you specify.

[ **GROUP BY property** ]

Aggregates the result set into groups of rows with matching values for the given property.

The GROUP BY clause is applicable to aggregations. AWS Config supports grouping by only one property.

[ **ORDER BY property [ ASC | DESC ] [, property [ ASC | DESC ] ...]** ]

Sorts a result set by one or more output properties.

When the clause contains multiple properties, the result set is sorted according to the first property, then according to the second property for rows that have matching values for the first property, and so on.

**Examples**

```sql
SELECT resourceId WHERE resourceType='AWS::EC2::Instance'
```

```sql
SELECT configuration.complianceType, COUNT(*) WHERE resourceType = 'AWS::Config::ResourceCompliance' GROUP BY configuration.complianceType
```

**Deleting AWS Config Data**

AWS Config allows you to delete your data by specifying a retention period for your ConfigurationItems. When you specify a retention period, AWS Config retains your ConfigurationItems for that specified period. You can choose a period between a minimum of 30 days and a maximum of 7 years (2557 days). AWS Config deletes data older than your specified retention period. If you do not specify a retention period, AWS Config continues to store ConfigurationItems for the default period of 7 years (2557 days). When recording is switched on, the current state
of the resource is when a ConfigurationItem is recorded and until the next change (a new ConfigurationItem) is recorded.

To understand the behavior of retention period, let's take a look at the timeline.

- When recording is switched on, the current state of a resource always exists and can't be deleted irrespective of the date the ConfigurationItem is recorded.
- When AWS Config records new ConfigurationItems, the previous ConfigurationItems are deleted depending on the specified retention period.

In the following timeline, AWS Config records ConfigurationItems at the following dates. For the purpose of this timeline, today is represented as May 24, 2018.

The following table explains which ConfigurationItems are displayed on the AWS Config timeline based on selected retention period.

<table>
<thead>
<tr>
<th>Retention Period</th>
<th>Configuration Items displayed on timeline</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 days</td>
<td>December 12, 2017</td>
<td>The current state of the resource started from December 12, 2017 when the ConfigurationItem was recorded and is valid until today (May</td>
</tr>
</tbody>
</table>
Deleting Data

<table>
<thead>
<tr>
<th>Retention Period</th>
<th>Configuration Items displayed on timeline</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>365 days</td>
<td>December 12, 2017; November 12, 2017, and March 10, 2017</td>
<td>The retention period shows the current state December 12, 2017 and previous ConfigurationItems November 12, 2017 and March 10, 2017. The ConfigurationItem for March 10, 2017 is displayed on the timeline because that configuration state represented the current state 365 days ago.</td>
</tr>
</tbody>
</table>

After you specify a retention period, AWS Config APIs no longer return ConfigurationItems that represent a state older than the specified retention period.

Note

- AWS Config cannot record your ConfigurationItems if recording is switched off.
- AWS Config cannot record your ConfigurationItems if your IAM role is broken.

Setting Data Retention Period in AWS Management Console

In the AWS Management Console, if you do not select a data retention period, the default period is 7 years or 2557 days.

To set a custom data retention period for configuration items select the checkbox. You can select 1 year, 3 years, 5 years, or a custom period. For a custom period, enter the number of days between 30 and 2557 days.
Notifications that AWS Config Sends to an Amazon SNS topic

You can configure AWS Config to stream configuration changes and notifications to an Amazon SNS topic. For example, when a resource is updated, you can get a notification sent to your email, so that you can view the changes. You can also be notified when AWS Config evaluates your custom or managed rules against your resources.

AWS Config sends notifications for the following events:

- Configuration item change for a resource.
- Configuration history for a resource was delivered for your account.
- Configuration snapshot for recorded resources was started and delivered for your account.
- Compliance state of your resources and whether they are compliant with your rules.
- Evaluation started for a rule against your resources.
- AWS Config failed to deliver the notification to your account.

Topics

- Example Configuration Item Change Notifications (p. 76)
- Example Configuration History Delivery Notification (p. 84)
- Example Configuration Snapshot Delivery Started Notification (p. 84)
- Example Configuration Snapshot Delivery Notification (p. 85)
- Example Compliance Change Notification (p. 86)
- Example Rules Evaluation Started Notification (p. 87)
- Example Oversized Configuration Item Change Notification (p. 87)
- Example Delivery Failed Notification (p. 88)

Example Configuration Item Change Notifications

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. AWS Config also sends notifications that show whether your resources are compliant against your rules. If you choose to have 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 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 an ID of `i-344c463d`. The notification contains the configuration item change for the resource.

```
"Type": "Notification",
"MessageId": "8b945cb0-db34-5b72-b032-1724878af488",
"Message": {
  "MessageVersion": "1.0",
  "NotificationCreateTime": "2014-03-18T10:11:00Z",
  "messageType": "ConfigurationItemChangeNotification",
  "configurationItems": [  
    
```

76
"configurationItemVersion": "1.0",
"configurationItemCaptureTime": "2014-03-07T23:47:08.918Z",
"resourceId": "vol-ce676ccc",
"accountId": "123456789012",
"configurationStateId": "3e660fdf-4e34-4f32-af8b-0ace5bf363a",
"configurationItemStatus": "OK",
"relatedEvents": [
  "06c12a39-eb35-11de-ae07-adb6f9edbb164",
  "c376e30d-71a2-4694-89b7-a50a4a9f2281"
],
"availabilityZone": "us-west-2b",
"resourceType": "AWS::EC2::VOLUME",
"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",
  "attachments": [
    {
      "volumeId": "vol-ce676ccc",
      "instanceId": "i-344c463d",
      "device": "/dev/sdf",
      "state": "attached",
      "attachTime": "FriMar0723:46:28UTC2014",
      "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"
    }
  }
}
Configuration Items for Resources with Relationships

If a resource is related to other resources, a change to that resource can result in multiple configuration items. The following example shows how AWS Config creates configuration items for resources with relationships.

1. You have an Amazon EC2 instance with an ID of i-007d374c8912e3e90, and the instance is associated with an Amazon EC2 security group, sg-c8b141b4.
2. You update your EC2 instance to change the security group to another security group, sg-3f1fef43.
3. Because the EC2 instance is related to another resource, AWS Config creates multiple configuration items like the following examples:

This notification contains the configuration item change for the EC2 instance when the security group is replaced.

```json
{
    "Type": "Notification",
    "MessageId": "faeba85e-ef46-570a-b01c-f8b0faae8d5d",
    "Subject": "[AWS Config:us-east-2] AWS::EC2::Instance i-007d374c8912e3e90 Updated in Account 123456789012",
    "Message": {
        "configurationItemDiff": {
            "changedProperties": {
                "Configuration.NetworkInterfaces.0": {
                    "previousValue": {
                        "networkInterfaceId": "eni-fde9493f",
                        "subnetId": "subnet-2372be7b",
                        "vpcId": "vpc-14400670",
                        "description": "",
                        "ownerId": "123456789012",
                        "status": "in-use",
                        "macAddress": "0e:36:a2:2d:c5:e0",
                        "privateIpAddress": "172.31.16.84",
                        "privateDnsName": "ip-172-31-16-84.ec2.internal",
                        "sourceDestCheck": true,
                        "groups": [{
                            "groupName": "example-security-group-1",
                            "groupId": "sg-c8b141b4"
                        }],
                        "attachment": {
                            "attachmentId": "eni-attach-85bd89d9",
                            "deviceIndex": 0,
                            "status": "attached",
                            "attachTime": "2017-01-09T19:36:02.000Z",
                            "deleteOnTermination": true
                        }
                    }
                }
            }
        }
    }
}
```
Example Configuration Item Change Notifications

```
{
  "updatedValue": null,
  "changeType": "DELETE"
}
```

```
"Relationships.0": {
  "previousValue": {
    "resourceId": "sg-c8b141b4",
    "resourceName": null,
    "resourceType": "AWS::EC2::SecurityGroup",
    "name": "Is associated with SecurityGroup"
  },
  "updatedValue": null,
  "changeType": "DELETE"
}
```

```
"Configuration.NetworkInterfaces.1": {
  "previousValue": null,
  "updatedValue": {
    "networkInterfaceId": "eni-fde9493f",
    "subnetId": "subnet-2372be7b",
    "vpcId": "vpc-14400670",
    "description": "",
    "ownerId": "123456789012",
    "status": "in-use",
    "macAddress": "0e:36:a2:2d:c5:e0",
    "privateIpAddress": "172.31.16.84",
    "privateDnsName": "ip-172-31-16-84.ec2.internal",
    "sourceDestCheck": true,
    "groups": [{
      "groupName": "example-security-group-2",
      "groupId": "sg-3f1fef43"
    }],
    "attachment": {
      "attachmentId": "eni-attach-85bd89d9",
      "deviceIndex": 0,
      "status": "attached",
      "attachTime": "2017-01-09T19:36:02.000Z",
      "deleteOnTermination": true
    },
    "association": {
      "publicIp": "54.175.43.43",
      "publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
      "ipOwnerId": "amazon"
    },
    "privateIpAddresses": [{
      "privateIpAddress": "172.31.16.84",
      "privateDnsName": "ip-172-31-16-84.ec2.internal",
      "primary": true,
      "association": {
        "publicIp": "54.175.43.43",
        "publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
        "ipOwnerId": "amazon"
      }
    }
  }
}
```

```
}]}"association": {
  "publicIp": "54.175.43.43",
  "publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
  "ipOwnerId": "amazon"
},
"privateIpAddresses": [{
  "privateIpAddress": "172.31.16.84",
  "privateDnsName": "ip-172-31-16-84.ec2.internal",
  "primary": true,
  "association": {
    "publicIp": "54.175.43.43",
```
"publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
  "ipOwnerId": "amazon"
}
)}
,"changeType": "CREATE"
},
"Relationships.1": {
  "previousValue": null,
  "updatedValue": {
    "resourceId": "sg-3f1fef43",
    "resourceName": null,
    "resourceType": "AWS::EC2::SecurityGroup",
    "name": "Is associated with SecurityGroup"
  },
  "changeType": "CREATE"
},
"Configuration.SecurityGroups.1": {
  "previousValue": null,
  "updatedValue": {
    "groupName": "example-security-group-2",
    "groupId": "sg-3f1fef43"
  },
  "changeType": "CREATE"
},
"Configuration.SecurityGroups.0": {
  "previousValue": {
    "groupName": "example-security-group-1",
    "groupId": "sg-c8b141b4"
  },
  "updatedValue": null,
  "changeType": "DELETE"
}
},
"changeType": "UPDATE"
},
"configurationItem": {
  "relatedEvents": ["e61e1419-7cb0-477f-8dde-bbfe27467a96"],
  "relationships": [ {
    "resourceId": "eni-fde9493f",
    "resourceName": null,
    "resourceType": "AWS::EC2::NetworkInterface",
    "name": "Contains NetworkInterface"
  },
  {
    "resourceId": "sg-3f1fef43",
    "resourceName": null,
    "resourceType": "AWS::EC2::SecurityGroup",
    "name": "Is associated with SecurityGroup"
  },
  {
    "resourceId": "subnet-2372be7b",
    "resourceName": null,
    "resourceType": "AWS::EC2::Subnet",
    "name": "Is contained in Subnet"
  },
  {
    "resourceId": "vol-0a2d63a256bce35c5",
    "resourceName": null,
    "resourceType": "AWS::EC2::Volume",
    "name": "Is attached to Volume"
  },
  {
    "resourceId": "vpc-144000670",
    "changeType": "DELETE"}]}
"resourceName": null,
"resourceType": "AWS::EC2::VPC",
"name": "Is contained in Vpc"
}
]
"configuration": {
"instanceId": "i-007d374c0912e3e90",
"imageId": "ami-9be6f38c",
"state": {
  "code": 16,
  "name": "running"
},
"privateDnsName": "ip-172-31-16-84.ec2.internal",
"publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
"stateTransitionReason": "",
"keyName": "ec2-micro",
"amiLaunchIndex": 0,
"productCodes": [],
"instanceType": "t2.micro",
"launchTime": "2017-01-09T20:13:28.000Z",
"placement": {
  "availabilityZone": "us-east-2c",
  "groupName": "",
  "tenancy": "default",
  "hostId": null,
  "affinity": null
},
"kernelId": null,
"ramdiskId": null,
"platform": null,
"monitoring": {"state": "disabled"},
"subnetId": "subnet-2372be7b",
"vpcId": "vpc-14400670",
"privateIpAddress": "172.31.16.84",
"publicIpAddress": "54.175.43.43",
"stateReason": null,
"architecture": "x86_64",
"rootDeviceType": "ebs",
"rootDeviceName": "/dev/xvda",
"blockDeviceMappings": [{
  "deviceName": "/dev/xvda",
  "ebs": {
    "volumeId": "vol-0a2d63a256bce35c5",
    "status": "attached",
    "attachTime": "2017-01-09T19:36:03.000Z",
    "deleteOnTermination": true
  }
}],
"virtualizationType": "hvm",
"instanceLifecycle": null,
"spotInstanceRequestId": null,
"clientToken": "bIYqA1483990561516",
"tags": [{
  "key": "Name",
  "value": "value"
}],
"securityGroups": [{
  "groupName": "example-security-group-2",
  "groupId": "sg-3f1f6e43"
}],
"sourceDestCheck": true,
"hypervisor": "xen",
"networkInterfaces": [{
  "networkInterfaceId": "eni-fde9493f",
  "subnetId": "subnet-2372be7b",
  "vpcId": "vpc-14400670",
"description": "",
"ownerId": "123456789012",
"status": "in-use",
"macAddress": "0e:36:a2:2d:c5:e0",
"privateIpAddress": "172.31.16.84",
"privateDnsName": "ip-172-31-16-84.ec2.internal",
"sourceDestCheck": true,
"groups": [{
  "groupName": "example-security-group-2",
  "groupId": "sg-3f1f43f3"
}],
"attachment": {
  "attachmentId": "eni-attach-85bd89d9",
  "deviceIndex": 0,
  "status": "attached",
  "attachTime": "2017-01-09T19:36:02.000Z",
  "deleteOnTermination": true
},
"association": {
  "publicIp": "54.175.43.43",
  "publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
  "ipOwnerId": "amazon"
},
"privateIpAddresses": [{
  "privateIpAddress": "172.31.16.84",
  "privateDnsName": "ip-172-31-16-84.ec2.internal",
  "primary": true,
  "association": {
    "publicIp": "54.175.43.43",
    "publicDnsName": "ec2-54-175-43-43.compute-1.amazonaws.com",
    "ipOwnerId": "amazon"
  }
}],
"iamInstanceProfile": null,
"ebsOptimized": false,
"srivmNetSupport": null,
"enaSupport": true
},
"supplementaryConfiguration": {},
"tags": {
  "Name": "value"
},
"configurationItemVersion": "1.2",
"configurationItemCaptureTime": "2017-01-09T22:50:14.328Z",
"configurationStateId": 1484002214328,
"awsAccountId": "123456789012",
"configurationStateMd5Hash": "8d0f41750f5965e0071ae9be063ba306",
"arn": "arn:aws:ec2:us-east-2:123456789012:instance/i-007d374c8912e3e90",
"awsRegion": "us-east-2",
"availabilityZone": "us-east-2c",
"configurationStateMd5Hash": "8d0f41750f5965e0071ae9be063ba306",
"resourceCreationTime": "2017-01-09T20:13:28.000Z"
},
"notificationCreationTime": "2017-01-09T22:50:15.928Z",
"messageType": "ConfigurationItemChangeNotification",
"recordVersion": "1.2"
},
"Timestamp": "2017-01-09T22:50:16.358Z",
"SignatureVersion": "1",
"Signature": "lpJTEYOSr8fufiBaaRw1EbfWkFWoD7I67SmLeEzkfAMjWqvpAplULH1LC +I0sS/0IA4P1Yc8GSK/OCOC/OXfBntw4CAtbMugTQvb3452ZYWcpcK0kPMie6vN5IvDuZ/6D28ECE +gVTNT009xtNIH8amvLmsglUSxU37xayg5yyxC5yTrXXe9+1kdzRd4QzS7ob8KrRrZW6ipxPNL6d5d/ VvPfyhcb57Vnm40/2/eonVbxbjWNHg7jQXqsa1Xhuc9eP2gEsmC4sl32gbqdeDU1Y4dFGukuzFYoHuWt5DPh +GlUq3KeiDAQshxAEZLmO1CQ71j/bELDJDtM9AcX6lDZ79W=="
This notification contains the configuration item change for the EC2 security group, sg-3f1fef43, which is associated with the instance.

```json
{
   "Type": "Notification",
   "MessageId": "564d873e-711e-51a3-b48c-d7d064f65bf4",
   "Subject": "[AWS Config:us-east-2] AWS::EC2::SecurityGroup sg-3f1fef43 Created in Account 123456789012",
   "Message": {
      "configurationItemDiff": {
         "changedProperties": {},
         "changeType": "CREATE"
      },
      "configurationItem": {
         "relatedEvents": "e61e1419-7cb0-477f-8dde-bbfe27467a96",
         "relationships": [{
            "resourceId": "vpc-14400670",
            "resourceName": null,
            "resourceType": "AWS::EC2::VPC",
            "name": "Is contained in Vpc"
         }],
         "configuration": {
            "ownerId": "123456789012",
            "groupName": "example-security-group-2",
            "groupId": "sg-3f1fef43",
            "description": "This is an example security group.",
            "ipPermissions": [],
            "ipPermissionsEgress": [{
               "ipProtocol": "-1",
               "fromPort": null,
               "toPort": null,
               "userIdGroupPairs": [],
               "ipRanges": ["0.0.0.0/0"],
               "prefixListIds": []
            }],
            "vpcId": "vpc-14400670",
            "tags": []
         },
         "supplementaryConfiguration": {},
         "tags": {},
         "configurationItemVersion": "1.2",
         "configurationItemCaptureTime": "2017-01-09T22:50:15.156Z",
         "awsAccountId": "123456789012",
         "configurationItemStatus": "ResourceDiscovered",
         "resourceType": "AWS::EC2::SecurityGroup",
         "resourceId": "sg-3f1fef43",
         "resourceName": null,
         "awsRegion": "us-east-2",
         "availabilityZone": "Not Applicable",
         "configurationStateMd5Hash": "7399608745296f67f7fe1c9ca56d5205",
         "resourceCreationTime": null
      },
      "notificationCreationTime": "2017-01-09T22:50:16.021Z",
      "messageType": "ConfigurationItemChangeNotification"
   }
}
```
Example Configuration History Delivery Notification

The configuration history is a collection of the configuration items for a resource type over a time period. The following is an example notification that AWS Config sends when the configuration history for a CloudTrail trail resource is delivered for your account.

```json
{
  "Type": "Notification",
  "MessageId": "ce49bf2c-d03a-51b0-8b6a-ef480a8b39fe",
  "Subject": "[AWS Config:us-east-2] Configuration History Delivery Completed for Account 123456789012",
  "Message": {
    "s3ObjectKey": "AWSLogs/123456789012/Config/us-east-2/2016/9/27/ConfigHistory/123456789012_Config_us-east-2_ConfigHistory_AWS::CloudTrail::Trail_20160927T195818Z_20160927T195818Z_1.json.gz",
    "s3Bucket": "config-bucket-123456789012-ohio",
    "notificationCreationTime": "2016-09-27T20:37:05.217Z",
    "messageType": "ConfigurationHistoryDeliveryCompleted",
    "recordVersion": "1.1"
  }
}
```

Example Configuration Snapshot Delivery Started Notification

The following is an example notification that AWS Config sends when AWS Config starts delivering the configuration snapshot for your account.

```json
{
  "Type": "Notification",
  "MessageId": "a32d0487-94b1-53f6-b4e6-5407c9c00be6",
  "Subject": "[AWS Config:us-east-2] Configuration Snapshot Delivery Started for Account 123456789012",
  "Message": {
    "s3ObjectKey": "AWSLogs/123456789012/Config/us-east-2/2016/9/27/ConfigHistory/123456789012_Config_us-east-2_ConfigHistory_AWS::CloudTrail::Trail_20160927T195818Z_20160927T195818Z_1.json.gz",
    "s3Bucket": "config-bucket-123456789012-ohio",
    "notificationCreationTime": "2016-09-27T20:37:05.315Z",
    "messageType": "ConfigurationSnapshotDeliveryCompleted",
    "recordVersion": "1.1"
  }
}
```
Example Configuration Snapshot Delivery Notification

The configuration snapshot is a collection of configuration items for all recorded resources and their configurations in your account. The following is an example notification that AWS Config sends when the configuration snapshot is delivered for your account.

```json

"Message": {
    "configSnapshotId": "108e0794-84ca-47e9-a719-76a199ddd11a",
    "notificationCreationTime": "2016-10-18T17:26:09.572Z",
    "messageType": "ConfigurationSnapshotDeliveryStarted",
    "recordVersion": "1.1"
},
"Timestamp": "2016-10-18T17:26:09.840Z",
"SignatureVersion": "1",
"Signature": "BBA0DeKeFteTyYH5HANpOLmW/jun0MBegbRq/kym9y7jNlkoF/V13bpgL1GnVmqDQpqB60EO80xzaqy5FXF5W5rZxKlEnS9xugFzALPuX/
ol5J4evWAlBKNq1lxvAQgu9ghfD7Dsd3aCwe84scqGfoj1rBv7plZgMx+u35R/
C54cbfcdudP5sdWdo8s+cTpzVmtA30y5x0yfOqXk1ABAO/JEnjdwQs0z/d4SYxmh/H9wcvwXB9XECe1HhR70Y
+wNqixfx4OS1AsSrZvnmJE+m9m9phFq64YrARDrv6tMaeN66CVFQ+81ceAXIg2E1m7z7l24Pa==",
"SigningCertURL": "https://sns.us-east-2.amazonaws.com/SimpleNotificationService-b95095be82e6a46b3aafc7f4149a.htm",

}```
Example Compliance Change Notification

When AWS Config evaluates your resources against a custom or managed rule, AWS Config sends a notification that shows whether the resources are compliant against the rule.

The following is an example notification where the CloudTrail trail resource is compliant against the cloudtrail-enabled managed rule.

```
{
  "Type": "Notification",
  "MessageId": "11fd05d5-47e1-5523-bc01-55b988b9478",
  "Subject": "[AWS Config:us-east-2] AWS::Account 123456789012 is COMPLIANT with cloudtrail-enabled in Account...",
  "Message": {
    "awsAccountId": "123456789012",
    "configRuleName": "cloudtrail-enabled",
    "resourceType": "AWS::Account",
    "resourceId": "123456789012",
    "awsRegion": "us-east-2",
    "newEvaluationResult": {
      "evaluationResultIdentifier": {
        "evaluationResultQualifier": {
          "configRuleName": "cloudtrail-enabled",
          "resourceType": "AWS::Account",
          "resourceId": "123456789012"
        },
        "orderingTimestamp": "2016-09-27T19:48:40.619Z"
      },
      "complianceType": "COMPLIANT",
      "resultRecordedTime": "2016-09-27T19:48:41.405Z",
      "configRuleInvokedTime": "2016-09-27T19:48:40.914Z",
      "annotation": null,
      "resultToken": null
    },
    "oldEvaluationResult": {
      "evaluationResultIdentifier": {
        "evaluationResultQualifier": {
          "configRuleName": "cloudtrail-enabled",
          "resourceType": "AWS::Account",
          "resourceId": "123456789012"
        },
        "orderingTimestamp": "2016-09-27T16:30:49.531Z"
      },
      "complianceType": "NON_COMPLIANT",
      "resultRecordedTime": "2016-09-27T16:30:50.717Z",
      "configRuleInvokedTime": "2016-09-27T16:30:50.105Z",
      "annotation": null,
      "resultToken": null
    },
    "notificationCreationTime": "2016-09-27T19:48:42.620Z",
    "messageType": "ComplianceChangeNotification",
    "recordVersion": "1.0"
  },
  "Timestamp": "2016-09-27T19:48:42.749Z",
  "SignatureVersion": "1",
  "Signature": "XZ9FfLb2yrWw9yj0y6BkNtIP5g7Cry6JtCEyULhmG9gp0Zl3seq41udhtAgCZoiMrizAE1
+6gctHRcV1hNemzp/8yMmTf06aYXtc0FJDaEvd8k68VCS9aqRlBiY1LNo7IIi4Pqd5rE48X2YBQSCzQyERGKutfZzBIPyAmb1Q/y4/6e28rDyi1s45FDSlg0GEb4LKLNR6eDi4FbRtW9IHA7Nz8obqs1dHbgWYnp3c80mVL17ohP4hiwxdwAgXrbaN32eKYr15gdHozx8+BIZ12tkcUtYSB31mgR1U07Yhn3L3c6rZxQ=="
}
```
Example Rules Evaluation Started Notification

AWS Config sends a notification when it starts to evaluate your custom or managed rule against your resources. The following is an example notification when AWS Config starts to evaluate the `iam-password-policy` managed rule.

```
{
  "Type": "Notification",
  "MessageId": "358c8e65-e27a-594e-82d0-de1fe77393d7",
  "Subject": "[AWS Config:us-east-2] Config Rules Evaluation Started for Account 123456789012",
  "Message": {
    "awsAccountId": "123456789012",
    "awsRegion": "us-east-2",
    "configRuleNames": ["iam-password-policy"],
    "messageType": "ConfigRulesEvaluationStarted",
    "recordVersion": "1.0"
  },
  "SignatureVersion": "1",
  "Signature": "DE431D+24zzFRboyFY2bPTSznJWe8L6TjDC+ItY1lFkE9jACSBl13sQ1uSjYzEhEbN7Cs+wBoHnJ/DxOspyCxt4gqrgCd+H21636BrvQRDhjWw7Lq168IozEliRvRWBm38zDTvHqkmnXq8dHRSk/KMswMeVTBkuW0X8iVmrj+KpnuF57vE6eXeFhjBeJ0DKvQ+nA+i3onSuT7hvXQDBpDMcSUlRJaMQJ6tCmU5G76qg/ql4941b4Vj4ubGdGW5ySx1uSsTrlWXQKXaxWtsCQFdhOkKgmVlCfMrZnP8Pjnu+wspYQELkEtwBChvDvzLi15MrAsQ==",
  "SigningCertURL": "https://sns.us-east-2.amazonaws.com/SimpleNotificationService-b95095bebed8e86fa046b3aaefc7f4149a.pem",
}
```

Example Oversized Configuration Item Change Notification

When AWS Config detects a configuration change for a resource, it sends a configuration item notification. If the notification exceeds the maximum size allowed by Amazon Simple Notification Service (Amazon SNS), the notification includes a brief summary of the configuration item. You can view the complete notification in the Amazon S3 bucket location specified in the `s3BucketLocation` field.

The following example notification shows a configuration item for an Amazon EC2 instance. The notification includes a summary of the changes and the location of the notification in the Amazon S3 bucket.

```
View the Timeline for this Resource in AWS Config Management Console:
```
The full configuration item change notification for this resource exceeded the maximum size allowed by Amazon Simple Notification Service (SNS). A summary of the configuration item is provided here. You can view the complete notification in the specified Amazon S3 bucket location.

New State Record Summary:
-------------------------
{
  "configurationItemSummary": {
    "changeType": "UPDATE",
    "configurationItemVersion": "1.2",
    "configurationItemCaptureTime": "2016-10-06T16:46:16.261Z",
    "configurationStateId": 0,
    "awsAccountId": "123456789012",
    "configurationItemStatus": "OK",
    "resourceType": "AWS::EC2::Instance",
    "resourceId": "resourceId_14b76876-7969-4097-ab8e-a31942b02e80",
    "resourceName": null,
    "ARN": "arn:aws:ec2:us-west-2:123456789012:instance/resourceId_14b76876-7969-4097-ab8e-a31942b02e80",
    "awsRegion": "us-west-2",
    "availabilityZone": null,
    "configurationStateMd5Hash": "8f1ee69b287895a0f0b5753eca68e96",
    "resourceCreationTime": "2016-10-06T16:46:10.489Z"
  },
  "s3DeliverySummary": {
    "s3BucketLocation": "my-bucket/AWSLogs/123456789012/Config/us-west-2/2016/10/6/OversizedChangeNotification/AWS::EC2::Instance/resourceId_14b76876-7969-4097-ab8e-a31942b02e80/123456789012_Config_us-west-2_ChangeNotification_AWS:EC2::Instance_resourceId_14b76876-7969-4097-ab8e-a31942b02e80_20161006T164616Z_0.json.gz",
    "errorCode": null,
    "errorMessage": null
  },
  "notificationCreationTime": "2016-10-06T16:46:16.261Z",
  "messageType": "OversizedConfigurationItemChangeNotification",
  "recordVersion": "1.0"
}

Example Delivery Failed Notification

AWS Config sends a delivery failed notification if AWS Config can't deliver the configuration snapshot or an oversized configuration item change notification to your Amazon S3 bucket. Verify that you specified a valid Amazon S3 bucket.

View the Timeline for this Resource in AWS Config Management Console:

The full configuration item change notification for this resource exceeded the maximum size allowed by Amazon Simple Notification Service (SNS). A summary of the configuration item is provided here. You can view the complete notification in the specified Amazon S3 bucket location.

New State Record Summary:
-------------------------
{
  "configurationItemSummary": {
    "changeType": "UPDATE",
    "configurationItemVersion": "1.2",
    "configurationItemCaptureTime": "2016-10-06T16:46:13.749Z",
    "configurationStateId": 0,
    "awsAccountId": "123456789012",
    "configurationItemStatus": "OK",
    "resourceType": "AWS::EC2::Instance",
    "resourceId": "resourceId_14b76876-7969-4097-ab8e-a31942b02e80",
    "awsRegion": "us-west-2",
    "availabilityZone": null,
    "configurationStateMd5Hash": "8f1ee69b287895a0f0b5753eca68e96",
    "resourceCreationTime": "2016-10-06T16:46:10.489Z"
  }
}
"awsAccountId": "123456789012",
"configurationItemStatus": "OK",
"resourceType": "AWS::EC2::Instance",
"resourceId": "test_resourceId_014b953d-75e3-40ce-96b9-c7240b975457",
"resourceName": null,
test_resourceId_014b953d-75e3-40ce-96b9-c7240b975457",
"awsRegion": "us-west-2",
"availabilityZone": null,
"configurationStateMd5Hash": "6de64b95eacd30e7b63d4bba7cd80814",
"resourceCreationTime": "2016-10-06T16:46:10.489Z"
},
"s3DeliverySummary": {
  "s3BucketLocation": null,
  "errorCode": "NoSuchBucket",
  "errorMessage": "Failed to deliver notification to bucket: bucket-example for
  account 123456789012 in region us-west-2."
},
"notificationCreationTime": "2016-10-06T16:46:10.489Z",
"messageType": "OversizedConfigurationItemChangeDeliveryFailed",
"recordVersion": "1.0"}
Controlling Permissions for AWS Config

AWS Config integrates with AWS Identity and Access Management (IAM), which allows you to create permission policies to attach to your IAM role, Amazon S3 buckets and Amazon Simple Notification Service (Amazon SNS) topics. You can use AWS Identity and Access Management to create AWS Config permission policies to attach to the IAM roles. A policy is a set of statements that grants AWS Config permissions.

**Important**
We consider it a best practice not to use root account credentials to perform everyday work in AWS. Instead, we recommend that you create an IAM administrators group with appropriate permissions, create IAM users for the people in your organization who need to perform administrative tasks (including for yourself), and add those users to the administrative group. For more information, see IAM Best Practices in the IAM User Guide guide.

The first two topics control user permissions for AWS Config followed by topics that provide accurate configuration information about permissions needed for AWS Config. The topics provide examples of recommended IAM policies to use with the AWS Config console and the AWS Command Line Interface.

**Topics**
- Granting Permissions for AWS Config Administration (p. 90)
- Granting Custom Permissions for AWS Config Users (p. 92)
- Supported Resource-Level Permissions for AWS Config Rules APIs Actions (p. 98)
- Permissions for the IAM Role Assigned to AWS Config (p. 99)
- Permissions for the Amazon S3 Bucket (p. 102)
- Permissions for the Amazon SNS Topic (p. 104)

Granting Permissions for AWS Config Administration

To allow users to administer AWS Config, you must grant explicit permissions to IAM users to perform the actions associated with AWS Config tasks. For most scenarios, you can do this using an AWS managed policy that contains predefined permissions.

**Note**
The permissions you grant to users to perform AWS Config administration tasks are not the same as the permissions that AWS Config itself requires in order to deliver log files to Amazon S3 buckets or send notifications to Amazon SNS topics.

Users who set up and manage AWS Config must have full-access permissions. With full-access permissions, users can provide Amazon S3 and Amazon SNS endpoints that AWS Config delivers data to, create a role for AWS Config, and turn on and turn off recording.

Users who use AWS Config but don't need to set up AWS Config should have read-only permissions. With read-only permissions, users can look up the configurations of resources or search for resources by tags.

A typical approach is to create an IAM group that has the appropriate permissions and then add individual IAM users to that group. For example, you might create an IAM group for users who should...
have full access to AWS Config actions, and a separate group for users who should be able to view the configurations but not create or change a role.

Contents
- Creating an IAM Group and Users for AWS Config Access (p. 91)
- Granting Full-Access Permission for AWS Config Access (p. 91)
- Additional Resources (p. 92)

Creating an IAM Group and Users for AWS Config Access

2. From the dashboard, choose Groups in the navigation pane, and then choose Create New Group.
3. Type a name, and then choose Next Step.
4. On the Attach Policy page, find and choose AWSConfigUserAccess. This policy provides user access to use AWS Config, including searching by tags on resources, and reading all tags. This does not provide permission to configure AWS Config which requires administrative privileges.
   **Note**
   You can also create a custom policy that grants permissions to individual actions. For more information, see Granting Custom Permissions for AWS Config Users (p. 92).
5. Choose Next Step.
6. Review the information for the group you are about to create.
   **Note**
   You can edit the group name, but you will need to choose the policy again.
7. Choose Create Group. The group that you created appears in the list of groups.
8. Choose the group name that you created, choose Group Actions, and then choose Add Users to Group.
9. On the Add Users to Group page, choose the existing IAM users, and then choose Add Users. If you don't already have IAM users, choose Create New Users, enter user names, and then choose Create.
10. If you created new users, choose Users in the navigation pane and complete the following for each user:
   a. Choose the user.
   b. If the user will use the console to manage AWS Config, in the Security Credentials tab, choose Manage Password, and then create a password for the user.
   c. If the user will use the AWS CLI or API to manage AWS Config, and if you didn't already create access keys, in the Security Credentials tab, choose Manage Access Keys and then create access keys. Store the keys in a secure location.
   d. Give each user his or her credentials (access keys or password).

Granting Full-Access Permission for AWS Config Access

2. In the navigation pane, choose Policies, and then choose Create Policy.
3. For Create Your Own Policy, choose Select.
4. Type a policy name and description. For example: AWSConfigFullAccess.
5. For Policy Document, type or paste the full-access policy into the editor. You can use the Full access (p. 93).
6. Choose Validate Policy and ensure that no errors display in a red box at the top of the screen. Correct any errors that are reported.
7. Choose Create Policy to save your new policy.
8. In the list of policies, select the policy that you created. You can use the Filter menu and the Search box to find the policy.
9. Choose Policy Actions, and then choose Attach.
10. Select the users, groups, or roles, and then choose Attach Policy. You can use the Filter menu and the Search box to filter the list.
11. Choose Apply Policy.

Note
Instead of creating a managed policy, you can also create an inline policy from the IAM console and attach it to an IAM user, group, or role. For more information, see Working with Inline Policies in the IAM User Guide.

Additional Resources
To learn more about creating IAM users, groups, policies, and permissions, see Creating an Admins Group Using the Console and Permissions and Policies in the IAM User Guide.

Granting Custom Permissions for AWS Config Users

AWS Config policies grant permissions to users who work with AWS Config. If you need to grant different permissions to users, you can attach a AWS Config policy to an IAM group or to a user. You can edit the policy to include or exclude specific permissions. You can also create your own custom policy. Policies are JSON documents that define the actions a user is allowed to perform and the resources that the user is allowed to perform those actions on.

Contents
- Read-only access (p. 92)
- Full access (p. 93)
- Controlling User Permissions for Actions on Multi-Account Multi-Region Data Aggregation (p. 95)
- Additional Information (p. 92)

Read-only access

The following example shows a AWS managed policy, AWSConfigUserAccess that grants read-only access to AWS Config. The policy also grants permission to read objects in Amazon S3 buckets, but not create or delete them.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "aws:config:ListScope",
      "Effect": "Allow",
      "Action": "s3:GetObject"
    }
  ]
}
```
In the policy statements, the **Effect** element specifies whether the actions are allowed or denied. The **Action** element lists the specific actions that the user is allowed to perform. The **Resource** element lists the AWS resources the user is allowed to perform those actions on. For policies that control access to AWS Config actions, the **Resource** element is always set to ", a wildcard that means "all resources."

The values in the **Action** element correspond to the APIs that the services support. The actions are preceded by **config:** to indicate that they refer to AWS Config actions. You can use the `*` wildcard character in the **Action** element, such as in the following examples:

- "Action": ["config:*ConfigurationRecorder"]

  This allows all AWS Config actions that end with "ConfigurationRecorder" (StartConfigurationRecorder, StopConfigurationRecorder).

- "Action": ["config:*"]

  This allows all AWS Config actions, but not actions for other AWS services.

- "Action": ["*"]

  This allows all AWS actions. This permission is suitable for a user who acts as an AWS administrator for your account.

The read-only policy doesn't grant user permission for the actions such as StartConfigurationRecorder, StopConfigurationRecorder, and DeleteConfigurationRecorder. Users with this policy are not allowed to start configuration recorder, stop configuration recorder, or delete configuration recorder. For the list of AWS Config actions, see the [AWS Config API Reference](#).

## Full access

The following example shows a policy that grants full access to AWS Config. It grants users the permission to perform all AWS Config actions. It also lets users manage files in Amazon S3 buckets and manage Amazon SNS topics in the account that the user is associated with.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sns:AddPermission",
                  "sns:CreateTopic",
                  "sns:DeleteTopic",
                  "sns:GetTopicAttributes",
```
"sns:ListPlatformApplications",
"sns:ListTopics",
"sns:SetTopicAttributes"
],
"Resource": "*
},
{
"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",
  "iam:DeletePolicyVersion",
  "iam:CreateServiceLinkedRole"
],
"Resource": "*
},
{
"Effect": "Allow",
"Action": [
  "cloudtrail:DescribeTrails",
  "cloudtrail:GetTrailStatus",
  "cloudtrail:LookupEvents"
],
"Resource": "*
},
{
"Effect": "Allow",
"Action": [
  "config:*",
  "tag:Get**
],
"Resource": "*
},
{
"Effect": "Allow",
"Action": [
  "ssm:DescribeDocument",
  "ssm:GetDocument",
  "ssm:DescribeAutomationExecutions",
  "ssm:GetAutomationExecution"
]
Controlling User Permissions for Actions on Multi-Account Multi-Region Data Aggregation

You can use resource-level permissions to control a user's ability to perform specific actions on multi-account multi-region data aggregation. AWS Config multi-account multi-region data aggregation API's support resource level permissions. With resource level permission can restrict to access/modify the resource data to specific users.

For example, you want to restrict access to resource data to specific users. You can create two aggregators AccessibleAggregator and InAccessibleAggregator. Then attach an IAM policy that allows access to the AccessibleAggregator.

In the first policy, you allow the aggregator actions such as DescribeConfigurationAggregators and DeleteConfigurationAggregator actions for the config ARN that you specify. In the following example, the config ARN is arn:aws:config:ap-northeast-1:AccountID:config-aggregator/config-aggregator-mocpsqhs.

```json
{"Version": "2012-10-17",
"Statement": [
{
"Sid": "ConfigReadOnly",
"Effect": "Allow",
"Action": [
"config:PutConfigurationAggregator",
"config:DescribePendingAggregationRequests",
"config:DeletePendingAggregationRequest",
"config:GetAggregateConfigRuleComplianceSummary",
"config:DescribeAggregateComplianceByConfigRules",
"config:GetAggregateComplianceDetailsByConfigRule",
"config:DescribeConfigurationAggregators",
"config:DescribeConfigurationAggregatorSourcesStatus",
"config:DeleteConfigurationAggregator"
],
}
]
}
```

In the second policy, you deny the aggregator actions for the config ARN that you specify. In the following example, the config ARN is arn:aws:config:ap-northeast-1:AccountID:config-aggregator/config-aggregator-pokxzldx.

```json
{"Version": "2012-10-17",
"Statement": [
{
"Sid": "ConfigReadOnly",
"Effect": "Deny",
"Action": [
"config:PutConfigurationAggregator",
"config:DescribePendingAggregationRequests",
"config:DeletePendingAggregationRequest",
"config:GetAggregateConfigRuleComplianceSummary",
"config:DescribeAggregateComplianceByConfigRules",
"config:GetAggregateComplianceDetailsByConfigRule",
"config:DescribeConfigurationAggregators",
"config:DescribeConfigurationAggregatorSourcesStatus",
"config:DeleteConfigurationAggregator"
],
"Resource": "arn:aws:config:ap-northeast-1:AccountID:config-aggregator/config-aggregator-pokxzldx"
}
]
}
```
Controlling User Permissions for Actions on Multi-Account Multi-Region Data Aggregation

{
    "Sid": "ConfigReadOnly",
    "Effect": "Deny",
    "Action": [
        "config:PutConfigurationAggregator",
        "config:DescribePendingAggregationRequests",
        "config:DeletePendingAggregationRequest",
        "config:GetAggregateConfigRuleComplianceSummary",
        "config:DescribeAggregateComplianceByConfigRules",
        "config:GetAggregateComplianceDetailsByConfigRule",
        "config:DescribeConfigurationAggregators",
        "config:DescribeConfigurationAggregatorSourcesStatus",
        "config:DeleteConfigurationAggregator"
    ],
    "Resource": "arn:aws:config:ap-northeast-1:AccountID:config-aggregator/config-aggregator-pokxzldx"
}

If a user of the developer group tries to describe or delete configuration aggregators on the config that you specified in the second policy, that user gets an access denied exception.

The following AWS CLI examples show that the user creates two aggregators, AccessibleAggregator and InAccessibleAggregator.

aws configservice describe-configuration-aggregators

The command complete successfully:

{
    "ConfigurationAggregators": [
        {
            "CreationTime": 1517942461.442,
            "ConfigurationAggregatorName": "AccessibleAggregator",
            "AccountAggregationSources": [
                {
                    "AllAwsRegions": true,
                    "AccountIds": [
                        "AccountID1",
                        "AccountID2",
                        "AccountID3"
                    ]
                }
            ],
            "LastUpdatedTime": 1517942461.455
        }
    ]
}

{
    "ConfigurationAggregators": [
        {
            "CreationTime": 1517942461.442,
            "ConfigurationAggregatorName": "InAccessibleAggregator",
            "AccountAggregationSources": [
                {
                    "AllAwsRegions": true,
                    "AccountIds": [
                        "AccountID1",
                        "AccountID2",
                        "AccountID3"
                    ]
                }
            ],
            "LastUpdatedTime": 1517942461.455
        }
    ]
}
Controlling User Permissions for Actions on Multi-Account Multi-Region Data Aggregation

Note
For account-aggregation-sources enter a comma-separated list of AWS account IDs for which you want to aggregate data. Wrap the account IDs in square brackets, and be sure to escape quotation marks (for example, "["AccountID1", "AccountID2", "AccountID3"]").

The user then creates an IAM policy that denies access to InAccessibleAggregator.

```json
{
   "Version": "2012-10-17",
   "Statement": [
   {
   "Sid": "ConfigReadOnly",
   "Effect": "Deny",
   "Action": [
   "config:PutConfigurationAggregator",
   "config:DescribePendingAggregationRequests",
   "config:DeletePendingAggregationRequest",
   "config:GetAggregateConfigRuleComplianceSummary",
   "config:DescribeAggregateComplianceByConfigRules",
   "config:GetAggregateComplianceDetailsByConfigRule",
   "config:DescribeConfigurationAggregators",
   "config:GetAggregateConfigurationAggregatorSourcesStatus",
   "config:DeleteConfigurationAggregator"
   ],
   "Resource": "arn:aws:config:ap-northeast-1:AccountID:config-aggregator/config-aggregator-pokxzldx"
   }
   ]
}
```

Next, the user confirms that IAM policy works for restricting access to specific aggregator and rules.

```
aws configservice get-aggregate-compliance-details-by-config-rule --configuration-aggregator-name InAccessibleAggregator --config-rule-name rule name --account-id AccountID --aws-region AwsRegion
```

The command returns an access denied exception:

```
```

With resource-level permissions, you can grant or deny access to perform specific actions on multi-account multi-region data aggregation.
## Additional Information

To learn more about creating IAM users, groups, policies, and permissions, see [Creating Your First IAM User and Administrators Group](#) and Access Management in the [IAM User Guide](#).

### Supported Resource-Level Permissions for AWS Config Rules APIs Actions

Resource-level permissions refers to the ability to specify which resources users are allowed to perform actions on. AWS Config supports resource-level permissions for certain AWS Config Rules API actions. This means that for certain AWS Config Rules actions, you can control when users are allowed to use those actions based on conditions that have to be fulfilled, or specific resources that users are allowed to use.

The following table describes the AWS Config Rules API actions that currently support resource-level permissions, as well as the supported resources (and their ARNs) for each action. When specifying an ARN, you can use the * wildcard in your paths; for example, when you cannot or do not want to specify exact resource IDs.

**Important**
If an AWS Config Rules API action is not listed in this table, then it does not support resource-level permissions. If an AWS Config Rules action does not support resource-level permissions, you can grant users permissions to use the action, but you have to specify a * for the resource element of your policy statement.

<table>
<thead>
<tr>
<th>API Action</th>
<th>Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteConfigRule</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>DeleteEvaluationResults</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>DescribeComplianceByConfigRule</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>DescribeConfigRuleEvaluationStatus</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>DescribeConfigRules</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>GetComplianceDetailsByConfigRule</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>PutConfigRule</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
<tr>
<td>StartConfigRulesEvaluation</td>
<td>Config Rule</td>
</tr>
<tr>
<td></td>
<td>arn:aws:config:region:accountID:config-rule/config-rule-ID</td>
</tr>
</tbody>
</table>
For example, you want to allow read access and deny write access to specific rules to specific users.

In the first policy, you allow the AWS Config Rules read actions such as DescribeConfigRules and DescribeConfigRuleEvaluationStatus on the specified rules.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "config:DescribeConfigRules",
            "config:StartConfigRulesEvaluation",
            "config:DescribeComplianceByConfigRule",
            "config:DescribeConfigRuleEvaluationStatus",
            "config:GetComplianceDetailsByConfigRule"
         ],
         "Resource": [
            "arn:aws:config:region:accountID:config-rule/config-rule-ID"
         ]
      }
   ]
}
```

In the second policy, you deny the AWS Config Rules write actions on the specific rule.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Deny",
         "Action": [
            "config:PutConfigRule",
            "config:DeleteConfigRule",
            "config:DeleteEvaluationResults"
         ],
      }
   ]
}
```

With resource-level permissions, you can allow read access and deny write access to perform specific actions on AWS Config Rules API actions.

Permissions for the IAM Role Assigned to AWS Config

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 write to your S3 bucket, publish to your SNS topic, and to
make Describe or List API requests to get configuration details for your AWS resources. For more information on IAM roles, see IAM Roles in the IAM User Guide.

When you use the AWS Config console to create or update an IAM role, AWS Config automatically attaches the required permissions for you. For more information, see Setting Up AWS Config with the Console (p. 22).

Contents
- Creating IAM Role Policies (p. 100)
  - Adding an IAM Trust Policy to your Role (p. 100)
  - IAM Role Policy for Amazon S3 Bucket (p. 100)
  - IAM Role Policy for Amazon SNS Topic (p. 101)
  - IAM Role Policy for Getting Configuration Details (p. 101)
- Troubleshooting for recording S3 buckets (p. 101)

Creating IAM Role Policies

When you use the AWS Config console to create an IAM role, AWS Config automatically attaches the required permissions to the role for you.

If you are using the AWS CLI to set up AWS Config or you are updating an existing IAM role, you must manually update the policy to allow AWS Config to access your S3 bucket, publish to your SNS topic, and get configuration details about your resources.

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 in the IAM User Guide.

The following is an example trust policy for AWS Config roles:

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

IAM Role Policy for Amazon S3 Bucket

The following example policy grants AWS Config permissions to access your Amazon S3 bucket:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:*"]
    }
  ]
}
```
IAM Role Policy for Amazon SNS Topic

The following example policy grants AWS Config permissions to access your SNS topic:

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

If your SNS topic is encrypted for additional setup instructions, see Configuring AWS KMS Permissions in the Amazon Simple Notification Service Developer Guide.

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.

Use the AWS managed policy `AWSConfigRole` and attach it to the IAM role that you assign to AWS Config. AWS updates this policy each time AWS Config adds support for an AWS resource type, which means AWS Config will continue to have the required permissions to get configuration details as long as the role has this managed policy attached.

If you create or update a role with the console, AWS Config attaches the `AWSConfigRole` for you.

If you use the AWS CLI, use the `attach-role-policy` command and specify the Amazon Resource Name (ARN) for `AWSConfigRole`:

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

Troubleshooting for recording S3 buckets

If you configured AWS Config to record S3 buckets for your account, AWS Config records and delivers notifications when an S3 bucket is created, updated, or deleted.
If you configured AWS Config to record S3 buckets, and are not receiving configuration change notifications:

- Verify that the IAM role assigned to AWS Config has the `AWSConfigRole` managed policy.
- If you have S3 bucket policies attached to your buckets, verify that they allow AWS Config permission to record changes to your buckets.

If you have a custom policy for your S3 bucket, you can add the following policy to your existing bucket policy. Your policy input must be alpha-numeric. This policy grants AWS Config permission to record the S3 bucket.

```json
{
    "Sid": "AWSConfig_ReadConfiguration_Access",
    "Effect": "Allow",
    "Principal": {"AWS": "arn:aws:iam::myAccountID:role/config-role"},
    "Action": [
        "s3:GetAccelerateConfiguration",
        "s3:GetBucketAcl",
        "s3:GetBucketCors",
        "s3:GetBucketLocation",
        "s3:GetBucketLogging",
        "s3:GetBucketNotification",
        "s3:GetBucketPolicy",
        "s3:GetBucketRequestPayment",
        "s3:GetBucketTagging",
        "s3:GetBucketWebsite",
        "s3:GetLifecycleConfiguration",
        "s3:GetReplicationConfiguration"
    ],
    "Resource": "arn:aws:s3:::myBucketName"
}
```

## 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.

### Contents
- Required Permissions for the Amazon S3 Bucket When Using IAM Roles (p. 103)
- Required Permissions for the Amazon S3 Bucket When Using Service-Linked Roles (p. 103)
- Granting AWS Config access to the Amazon S3 Bucket (p. 103)
Required Permissions for the Amazon S3 Bucket
When Using IAM Roles

When AWS Config sends configuration information (history files and snapshots) to Amazon S3 bucket in your account, it assumes the IAM role that you assigned when you set up AWS Config. When AWS Config sends configuration information to an Amazon 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. You must attach an access policy, mentioned in step 6 below to the Amazon S3 bucket in another account to grant AWS Config access to the Amazon S3 bucket.

Before AWS Config can deliver logs to your Amazon S3 bucket AWS Config checks whether the bucket exists and in which AWS region the bucket is located. AWS Config attempts to call Amazon S3 HeadBucket API to check whether the bucket exists and to get the bucket region. If permissions are not provided to locate the bucket when the location check is performed, you see AccessDenied error in AWS CloudTrail logs. However, the log delivery to your Amazon S3 bucket succeeds if you do not provide bucket location permissions.

Required Permissions for the Amazon S3 Bucket
When Using Service-Linked Roles

If you set up AWS Config using a service-linked role, you need to attach an access policy, mentioned in step 6 below to the Amazon S3 bucket in your own account or another account to grant AWS Config access to the Amazon S3 bucket.

Granting AWS Config access to the Amazon S3 Bucket

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

```json
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "AWSConfigBucketPermissionsCheck",
 "Effect": "Allow",
 "Principal": {
 "Service": ["config.amazonaws.com"
 ]
 }]
}
```
Permissions for the Amazon SNS Topic

Use the information in this topic only if you want to configure AWS Config to deliver Amazon SNS topics owned by your account or by a different account. AWS Config must have permissions to send notifications to an Amazon SNS topic.

Contents
- Required Permissions for the Amazon SNS Topic When Using IAM Roles (p. 105)
- Required Permissions for the Amazon SNS Topic When Using Service-Linked Roles (p. 105)
- Troubleshooting for the Amazon SNS Topic (p. 106)
Required Permissions for the Amazon SNS Topic When Using IAM Roles

You can attach a permission policy to the Amazon SNS topic owned by a different account. If you want to use an Amazon SNS topic from another account, make sure to attach the following policy to an existing Amazon SNS topic.

```
{
   "Id": "Policy1415489375392",
   "Statement": [
      {
         "Sid": "AWSConfigSNSPolicy20150201",
         "Action": [ "SNS:Publish" ],
         "Effect": "Allow",
         "Principal": { "AWS": [ "account-id1", "account-id2", "account-id3" ] }
      }
   ]
}
```

For the `Resource` key, `account-id` is the account number of the topic owner. For `account-id1`, `account-id2`, and `account-id3`, use the AWS accounts that will send data to an Amazon SNS topic. You must substitute appropriate values for `region` and `myTopic`.

Required Permissions for the Amazon SNS Topic When Using Service-Linked Roles

If you set up AWS Config using a service-linked role, you need to attach a permission policy to the Amazon SNS topic. If you want to use an Amazon SNS topic from your own account, make sure to attach the following policy to an existing Amazon SNS topic.

```
{
   "Id": "Policy_ID",
   "Statement": [
      {
         "Sid": "AWSConfigSNSPolicy",
         "Effect": "Allow",
         "Principal": { "AWS": "[configRoleArn]" },
         "Action": "SNS:Publish",
      }
   ]
}
```

You must substitute appropriate values for `region`, `account-id`, and `myTopic`. 
Note
AWS Config does not recommend using a service-linked role when using Amazon SNS topic from other accounts.

Troubleshooting for the Amazon SNS Topic

AWS Config must have permissions to send notifications to an Amazon SNS topic. If an Amazon SNS topic cannot receive notifications, verify that the IAM role that AWS Config was assuming must have sns:publish permissions.
Evaluating Resources with Rules

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 called managed rules to help you get started. You can also create your own custom rules. 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 violates a rule, AWS Config flags the resource and the rule as noncompliant.

For example, when an EC2 volume is created, AWS Config can evaluate the volume against a rule that requires volumes to be encrypted. If the volume is not encrypted, AWS Config flags the volume and the rule as noncompliant. AWS Config can also check all of your resources for account-wide requirements. For example, AWS Config 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.

The AWS Config console shows the compliance status of your rules and resources. You can see how your AWS resources comply overall with your desired configurations, and learn which specific resources are noncompliant. 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 assess how well your resource configurations comply with internal practices, industry guidelines, and regulations.

For regions that support AWS Config rules, see AWS Config Regions and Endpoints in the Amazon Web Services General Reference.

You can create up to 150 AWS Config rules per region in your account. For more information, see AWS Config Limits in the Amazon Web Services General Reference.

You can also create custom rules to evaluate additional resources that AWS Config doesn’t yet record. For more information, see Evaluating Additional Resource Types (p. 179).

Topics
- Viewing Configuration Compliance (p. 107)
- Specifying Triggers for AWS Config Rules (p. 110)
- AWS Config Managed Rules (p. 112)
- AWS Config Custom Rules (p. 174)
- Managing your AWS Config Rules (p. 188)
- Evaluating Your Resources (p. 192)
- Deleting Evaluation Results (p. 193)
- Enabling AWS Config Rules Across all Accounts in Your Organization (p. 194)
- Remediation Noncompliant AWS Resources by AWS Config Rules (p. 194)
- Tagging Your AWS Config Resources (p. 198)

Viewing Configuration Compliance

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 (p. 41).

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. 40).

Example 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:

```shell
$ 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:

```shell
$ 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",
```

"ResourceType": "AWS::EC2::Instance",
"ResourceId": "i-nnnnnnnn",
```
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": 1443751426.494,
      "ConfigRuleInvokedTime": 1443751421.208,
      "ComplianceType": "NON_COMPLIANT"
    }
  ]
}
Example 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.

Specifying Triggers for AWS Config Rules

When you add a rule to your account, you can specify when you want AWS Config to run the rule; this is called a trigger. AWS Config evaluates your resource configurations against the rule when the trigger occurs.

Contents

- Trigger types (p. 110)
- Example rules with triggers (p. 111)
- Rule evaluations when the configuration recorder is turned off (p. 111)

Trigger types

There are two types of triggers:

Configuration changes

AWS Config runs evaluations for the rule when certain types of resources are created, changed, or deleted.

You choose which resources trigger the evaluation by defining the rule's scope. The scope can include the following:

- One or more resource types
- A combination of a resource type and a resource ID
- A combination of a tag key and value
- When any recorded resource is created, updated, or deleted

AWS Config runs the evaluation when it detects a change to a resource that matches the rule's scope. You can use the scope to constrain which resources trigger evaluations. Otherwise, evaluations are triggered when any recorded resource changes.
Periodic

AWS Config runs evaluations for the rule at a frequency that you choose (for example, every 24 hours).

If you choose configuration changes and periodic, AWS Config invokes your Lambda function when it detects a configuration change and also at the frequency that you specify.

Example rules with triggers

Example rule with configuration change trigger

1. You add the AWS Config managed rule, S3_BUCKET_LOGGING_ENABLED, to your account to check whether your Amazon S3 buckets have logging enabled.
2. The trigger type for the rule is configuration changes. AWS Config runs the evaluations for the rule when an Amazon S3 bucket is created, changed, or deleted.
3. When a bucket is updated, the configuration change triggers the rule and AWS Config evaluates whether the bucket is compliant against the rule.

Example rule with periodic trigger

1. You add the AWS Config managed rule, IAM_PASSWORD_POLICY, to your account. The rule checks whether the password policy for your IAM users comply with your account policy, such as requiring a minimum length or requiring specific characters.
2. The trigger type for the rule is periodic. AWS Config runs evaluation for the rule at a frequency that you specify, such as every 24 hours.
3. Every 24 hours, the rule is triggered and AWS Config evaluates whether the passwords for your IAM users are compliant against the rule.

Example rule with configuration change and periodic triggers

1. You create a custom rule that evaluates whether CloudTrail trails in your account are turned on and logging for all regions.
2. You want AWS Config to run evaluations for the rule every time a trail is created, updated, or deleted. You also want AWS Config to run the rule every 12 hours.
3. For the trigger type, choose configuration changes and periodic.

Rule evaluations when the configuration recorder is turned off

If you turn off the configuration recorder, AWS Config stops recording changes to your resource configurations. This affects your rule evaluations in the following ways:

- Rules with a periodic trigger continue to run evaluations at the specified frequency.
- Rules with a configuration change trigger do not run evaluations.
- Rules with both trigger types run evaluations only at the specified frequency. The rules do not run evaluations for configuration changes.
- If you run an on-demand evaluation for a rule with a configuration change trigger, the rule evaluates the last known state of the resource, which is the last recorded configuration item.
AWS Config Managed 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 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 required if you want to create 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 triggers are defined as part of the rule, and they can include 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** – AWS Config runs evaluations for the rule at a frequency that you choose (for example, every 24 hours).

The AWS Config console shows which resources comply with the rule and which rules are being followed. For more information, see Viewing Configuration Compliance (p. 107).

Topics

- List of AWS Config Managed Rules (p. 112)
- Working with AWS Config Managed Rules (p. 172)
- Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173)

List of AWS Config Managed Rules

AWS Config currently supports the following managed rules in the compute; management and governance; network and content delivery; security, identity, and compliance; and storage categories. This page also lists the rules that AWS Config does not currently support in the China (Beijing) and China (Ningxia) regions.

Analytics

- elasticache-redis-cluster-automatic-backup-check (p. 138)
- elasticsearch-encrypted-at-rest (p. 137)
- elasticsearch-in-vpc-only (p. 137)

Compute

- approved-amis-by-id (p. 119)
- approved-amis-by-tag (p. 119)
• autoscaling-group-elb-healthcheck-required (p. 120)
• desired-instance-tenancy (p. 128)
• desired-instance-type (p. 128)
• ebs-optimized-instance (p. 130)
• ec2-stopped-instance (p. 131)
• ec2-instance-detailed-monitoring-enabled (p. 132)
• ec2-instance-managed-by-systems-manager (p. 132)
• ec2-instance-no-public-ip (p. 132)
• ec2-instances-in-vpc (p. 132)
• ec2-managedinstance-applications-blacklisted (p. 133)
• ec2-managedinstance-applications-required (p. 133)
• ec2-managedinstance-association-compliance-status-check (p. 134)
• ec2-managedinstance-inventory-blacklisted (p. 134)
• ec2-managedinstance-patch-compliance-status-check (p. 135)
• ec2-managedinstance-platform-check (p. 135)
• ec2-security-group-attached-to-eni (p. 135)
• ec2-volume-inuse-check (p. 136)
• eip-attached (p. 137)
• elb-acm-certificate-required (p. 138)
• elb-custom-security-policy-ssl-check (p. 138)
• elb-logging-enabled (p. 139)
• elb-predefined-security-policy-ssl-check (p. 139)
• encrypted-volumes (p. 141)
• lambda-concurrency-check (p. 152)
• lambda-dlq-check (p. 152)
• lambda-function-settings-check (p. 153)
• lambda-function-public-access-prohibited (p. 153)*
• lambda-inside-vpc (p. 153)
• restricted-common-ports (p. 159)
• restricted-ssh (p. 159)

Cryptography and PKI
• kms-cmk-not-scheduled-for-deletion (p. 151)

Database
• db-instance-backup-enabled (p. 127)
• dynamodb-autoscaling-enabled (p. 129)
• dynamodb-table-encryption-enabled (p. 130)
• dynamodb-throughput-limit-check (p. 130)
• rds-enhanced-monitoring-enabled (p. 155)
• rds-instance-public-access-check (p. 155)
• rds-multi-az-support (p. 155)
• rds-snapshots-public-prohibited (p. 156)
• rds-storage-encrypted (p. 156)
• redshift-cluster-configuration-check (p. 156)
• redshift-cluster-maintenancesettings-check (p. 157)
• redshift-cluster-public-access-check (p. 157)

**Machine Learning**
• sagemaker-endpoint-configuration-kms-key-configured (p. 168)
• sagemaker-notebook-no-direct-internet-access (p. 169)
• sagemaker-notebook-kms-configured (p. 168)

**Management and Governance**
• cloudformation-stack-drift-detection-check (p. 120)
• cloudformation-stack-notification-check (p. 120)
• cloud-trail-cloud-watch-logs-enabled (p. 121)
• cloudtrail-enabled (p. 122)
• cloud-trail-encryption-enabled (p. 122)
• cloud-trail-log-file-validation-enabled (p. 122)
• cloudtrail-s3-dataevents-enabled (p. 123)
• cloudwatch-alarm-action-check (p. 123)
• cloudwatch-alarm-resource-check (p. 124)
• cloudwatch-alarm-settings-check (p. 124)
• cloudwatch-log-group-encrypted (p. 125)
• codebuild-project-envvar-awscred-check (p. 126)
• codebuild-project-source-repo-url-check (p. 126)
• codepipeline-deployment-count-check (p. 126)
• codepipeline-region-fanout-check (p. 127)
• multi-region-cloud-trail-enabled (p. 154)
• required-tags (p. 158)

**Migration and Transfer**
• dms-replication-not-public (p. 129)

**Network and Content Delivery**
• alb-http-to-https-redirection-check (p. 117)
• api-gw-execution-logging-enabled (p. 118)
• api-gw-cache-enabled-and-encrypted (p. 118)
• api-gw-endpoint-type-check (p. 118)
• cloudfront-viewer-policy-https (p. 121)
• internet-gateway-authorized-vpc-only (p. 151)
• service-vpc-endpoint-enabled (p. 169)
• vpc-default-security-group-closed (p. 170)
• vpc-flow-logs-enabled (p. 171)
• vpc-sg-open-only-to-authorized-ports (p. 171)
• vpc-vpn-2-tunnels-up (p. 171)

Security, Identity & Compliance

• access-keys-rotated (p. 117)
• acm-certificate-expiration-check (p. 117)
• cmk-backing-key-rotation-enabled (p. 125)
• fms-security-group-audit-policy-check (p. 141)
• fms-security-group-content-check (p. 142)
• fms-security-group-resource-association-check (p. 143)
• fms-shield-resource-policy-check (p. 144)
• fms-webacl-resource-policy-check (p. 144)
• fms-webacl-rulegroup-association-check (p. 145)
• guardduty-non-archived-findings (p. 146)
• guardduty-enabled-centralized (p. 146)
• iam-group-has-users-check (p. 146)
• iam-password-policy (p. 147)
• iam-policy-blacklisted-check (p. 147)
• iam-policy-in-use (p. 148)
• iam-policy-no-statements-with-admin-access (p. 148)
• iam-role-managed-policy-check (p. 149)
• iam-root-access-key-check (p. 149)
• iam-user-group-membership-check (p. 150)
• iam-user-mfa-enabled (p. 150)
• iam-user-no-policies-check (p. 150)
• iam-user-unused-credentials-check (p. 151)
• mfa-enabled-for-iam-console-access (p. 154)
• root-account-hardware-mfa-enabled (p. 160)
• root-account-mfa-enabled (p. 160)
• shield-advanced-enabled-autorenew (p. 170)
• shield-drt-access (p. 170)

Storage

• ebs-snapshot-public-restorable-check (p. 131)
• efs-encrypted-check (p. 136)
• elb-deletion-protection-enabled (p. 139)
• emr-master-no-public-ip (p. 140)
• emr-kerberos-enabled (p. 140)
• s3-account-level-public-access-blocks (p. 160)
• s3-bucket-blacklisted-actions-prohibited (p. 161)*
• s3-bucket-logging-enabled (p. 163)
• s3-bucket-policy-grantee-check (p. 164)*
• s3-bucket-policy-not-more-permissive (p. 161)*
• s3-bucket-public-read-prohibited (p. 165)*
• s3-bucket-public-write-prohibited (p. 165)*
• s3-bucket-replication-enabled (p. 166)
• s3-bucket-server-side-encryption-enabled (p. 166)*
• s3-bucket-ssl-requests-only (p. 166)*
• s3-bucket-versioning-enabled (p. 168)

*This rule uses automated reasoning tools (ART) to evaluate IAM permissions and resource policies for correctness.

**Rules not Supported in China (Beijing) Region**

AWS Config does not currently support the following rules in the Beijing region:

• acm-certificate-expiration-check
• cmk-backing-key-rotation-enabled
• cloud-trail-encryption-enabled
• cloud-trail-log-file-validation-enabled
• cloudformation-stack-drift-detection-check
• codebuild-project-envvar-awscred-check
• codebuild-project-source-repo-url-check
• codepipeline-deployment-count-check
• codepipeline-region-fanout-check
• elb-acm-certificate-required
• encrypted-volumes
• fms-webacl-resource-policy-check
• fms-webacl-rulegroup-association-check
• guardduty-enabled-centralized
• lambda-function-public-access-prohibited
• rds-storage-encrypted
• root-account-hardware-mfa-enabled
• root-account-mfa-enabled
• s3-bucket-blacklisted-actions-prohibited
• s3-bucket-policy-grantee-check
• s3-bucket-policy-not-more-permissive
• s3-bucket-public-read-prohibited
• s3-bucket-public-write-prohibited
• s3-bucket-server-side-encryption-enabled
• s3-bucket-ssl-requests-only

**Rules not Supported in China (Ningxia) Region**

All the rules that are not available in the Beijing region are also not available in the Ningxia region. In addition to the above mentioned rules, AWS Config does not currently support the following rules in the Ningxia region:

• lambda-function-settings-check
• cloudformation-stack-notification-check
• dynamodb-table-encryption-enable

access-keys-rotated

Checks whether the active access keys are rotated within the number of days specified in maxAccessKeyAge. The rule is NON_COMPLIANT if the access keys have not been rotated for more than maxAccessKeyAge number of days.

Note
Re-evaluating this rule within 4 hours of the first evaluation will have no effect on the results.

Identifier: ACCESS_KEYS_ROTATED
Trigger type: Periodic
Parameters:
maxAccessKeyAge
Maximum number of days within which the access keys must be rotated. The default value is 90 days.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

acm-certificate-expiration-check

Checks whether ACM Certificates in your account are marked for expiration within the specified number of days. Certificates provided by ACM are automatically renewed. ACM does not automatically renew certificates that you import.

Identifier: ACM_CERTIFICATE_EXPIRATION_CHECK
Trigger type: Configuration changes and periodic
Parameters:
daysToExpiration
Specify the number of days before the rule flags the ACM Certificate as NON_COMPLIANT.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

alb-http-to-https-redirection-check

Checks whether HTTP to HTTPS redirection is configured on all HTTP listeners of Application Load Balancers. The rule is NON_COMPLIANT if one or more HTTP listeners of Application Load Balancers do not have HTTP to HTTPS redirection configured.

Identifier: ALB_HTTP_TO_HTTPS_REDIRECT_CHECK
Trigger type: Periodic

Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

api-gw-cache-enabled-and-encrypted

Checks that all methods in Amazon API Gateway stages have caching enabled and encrypted. The rule is NON_COMPLIANT if any method in an API Gateway stage is not configured for caching or the cache is not encrypted.

Identifier: API_GW_CACHE_ENABLED_AND_ENCRYPTED

Trigger type: Configuration changes

Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

api-gw-endpoint-type-check

Checks that Amazon API Gateway APIs are of the type specified in the rule parameter endpointConfigurationType. The rule returns NON_COMPLIANT if the REST API does not match the endpoint type configured in the rule parameter.

Identifier: API_GW_ENDPOINT_TYPE_CHECK

Trigger type: Configuration changes

Parameters:

endpointConfigurationType

(Required) Comma-separated list of allowed endpoint types. Allowed values are REGIONAL, PRIVATE and EDGE.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

api-gw-execution-logging-enabled

Checks that all methods in Amazon API Gateway stage has logging enabled. The rule is NON_COMPLIANT if logging is not enabled. The rule is NON_COMPLIANT if loggingLevel is neither ERROR nor INFO.
Identifier: API_GW_EXECUTION_LOGGING_ENABLED

Trigger type: Configuration changes

AWS Region: All supported AWS regions

Parameters:

loggingLevel

(Optional) Comma-separated list of specific logging levels (for example, ERROR, INFO or ERROR,INFO).

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

approved-amis-by-id

Checks whether running instances are using specified AMIs. Specify a list of approved AMI IDs. Running instances with AMIs that are not on this list are NON_COMPLIANT.

Identifier: APPROVED_AMIS_BY_ID

Trigger type: Configuration changes

Parameters:

amisIds

The AMI IDs (comma-separated list of up to 10).

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

approved-amis-by-tag

Checks whether running instances are using specified AMIs. Specify the tags that identify the AMIs. Running instances with AMIs that don't have at least one of the specified tags are NON_COMPLIANT.

Identifier: APPROVED_AMIS_BY_TAG

Trigger type: Configuration changes

Parameters:

amisByTagKeyAndValue

The AMIs by tag (comma-separated list up to 10; for example, "tag-key:tag-value").

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
autoscaling-group-elb-healthcheck-required

Checks whether your Auto Scaling groups that are associated with a load balancer are using Elastic Load Balancing health checks.

Identifier: AUTOSCALING_GROUP_ELB_HEALTHCHECK_REQUIRED

Trigger type: Configuration changes

Parameters:
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

cloudformation-stack-drift-detection-check

Checks whether an AWS CloudFormation stack's actual configuration differs, or has drifted, from its expected configuration. A stack is considered to have drifted if one or more of its resources differ from their expected configuration. The rule and the stack are COMPLIANT when the stack drift status is IN_SYNC. The rule and the stack are NON_COMPLIANT when the stack drift status is DRIFTED.

Note
If the stacks you created are not visible, choose Re-evaluate and check again.

Identifier: CLOUDFORMATION_STACK_DRIFT_DETECTION_CHECK

Trigger type: Configuration changes and periodic

Parameters:

cloudformationRoleArn

The AWS CloudFormation role ARN with IAM policy permissions to detect drift for AWS CloudFormation stacks.

Note
If the role does not have all of the permissions, the rule fails. The error appears as an annotation at the top of the page. Ensure to attach config.amazonaws.com trusted permissions and ReadOnlyAccess policy permissions. For specific policy permissions, refer to the Detecting Unmanaged Configuration Changes to Stacks and Resources in the AWS CloudFormation User Guide.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

cloudformation-stack-notification-check

Checks whether your CloudFormation stacks are sending event notifications to an SNS topic. Optionally checks whether specified SNS topics are used.

Identifier: CLOUDFORMATION_STACK_NOTIFICATION_CHECK
**Trigger type:** Configuration changes

**Parameters:**

- snsTopic1
  - SNS Topic ARN.
- snsTopic2
  - SNS Topic ARN.
- snsTopic3
  - SNS Topic ARN.
- snsTopic4
  - SNS Topic ARN.
- snsTopic5
  - SNS Topic ARN.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloudfront-viewer-policy-https**

Checks whether your Amazon CloudFront distributions use HTTPS (directly or via a redirection). The rule is NON_COMPLIANT if the value of ViewerProtocolPolicy is set to allow-all for defaultCacheBehavior or for cacheBehaviors. This means that the rule is non compliant when viewers can use HTTP or HTTPS.

**Identifier:** CLOUDFRONT_VIEWER_POLICY_HTTPS

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloud-trail-cloud-watch-logs-enabled**

Checks whether AWS CloudTrail trails are configured to send logs to Amazon CloudWatch Logs. The trail is NON_COMPLIANT if the CloudWatchLogsLogGroupArn property of the trail is empty.

**Identifier:** CLOUD_TRAIL_CLOUD_WATCH_LOGS_ENABLED

**Trigger type:** Periodic

**Parameters:**

None
**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloudtrail-enabled**

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.

**Identifier:** CLOUD_TRAIL_ENABLED

**Trigger type:** Periodic

**Parameters:**

- `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.
- `cloudWatchLogsLogGroupArn`
  The ARN of the Amazon CloudWatch log group for AWS CloudTrail to send data to.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloud-trail-encryption-enabled**

Checks whether AWS CloudTrail is configured to use the server side encryption (SSE) AWS Key Management Service (AWS KMS) customer master key (CMK) encryption. The rule is COMPLIANT if the `KmsKeyId` is defined.

**Identifier:** CLOUD_TRAIL_ENCRYPTION_ENABLED

**Trigger type:** Periodic

**Parameters:**

- None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloud-trail-log-file-validation-enabled**

Checks whether AWS CloudTrail creates a signed digest file with logs. AWS recommends that the file validation must be enabled on all trails. The rule is NON_COMPLIANT if the validation is not enabled.

**Identifier:** CLOUD_TRAIL_LOG_FILE_VALIDATION_ENABLED

**Trigger type:** Periodic
Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

cloudtrail-s3-dataevents-enabled

Checks whether at least one AWS CloudTrail trail is logging Amazon S3 data events for all S3 buckets. The rule is NON_COMPLIANT if trails that log data events for S3 buckets are not configured.

Note

The rule checks for trails in their home region only. The rule does not support Amazon S3 prefixes.

Identifier: CLOUDTRAIL_S3_DATAEVENTS_ENABLED

Trigger type: Periodic

Parameters:

S3BucketNames

(Optional) Comma-separated list of S3 bucket names for which data events logging should be enabled. Default behavior checks for all S3 buckets.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

cloudwatch-alarm-action-check

Checks whether CloudWatch alarms have at least one alarm action, one INSUFFICIENT_DATA action, or one OK action enabled. Optionally, checks whether any of the actions matches one of the specified ARNs.

Identifier: CLOUDWATCH_ALARM_ACTION_CHECK

Trigger type: Configuration changes

Parameters:

alarmActionRequired

Alarms have at least one action.

The default value is true.

insufficientDataActionRequired

Alarms have at least one action when the alarm transitions to the INSUFFICIENT_DATA state from any other state.

The default value is true.

okActionRequired

Alarms have at least one action when the alarm transitions to an OK state from any other state.
The default value is `false`.

**action1**
The action to execute, specified as an ARN.

**action2**
The action to execute, specified as an ARN.

**action3**
The action to execute, specified as an ARN.

**action4**
The action to execute, specified as an ARN.

**action5**
The action to execute, specified as an ARN.

**AWS CloudFormation template**
To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloudwatch-alarm-resource-check**
Checks whether the specified resource type has a CloudWatch alarm for the specified metric. For resource type, you can specify EBS volumes, EC2 instances, RDS clusters, or S3 buckets.

**Identifier:** CLOUDWATCH_ALARM_RESOURCE_CHECK

**Trigger type:** Periodic

**Parameters:**

**resourceType**
AWS resource type. The value can be one of the following:
- AWS::EC2::Volume
- AWS::EC2::Instance
- AWS::S3::Bucket

**metricName**
The name of the metric associated with the alarm (for example, "CPUUtilization" for EC2 instances).

**AWS CloudFormation template**
To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**cloudwatch-alarm-settings-check**
Checks whether CloudWatch alarms with the given metric name have the specified settings.

**Identifier:** CLOUDWATCH_ALARM_SETTINGS_CHECK

**Trigger type:** Configuration changes
Parameters:

metricName
   The name for the metric associated with the alarm.
threshold
   The value against which the specified statistic is compared.
evaluationPeriod
   The number of periods in which data is compared to the specified threshold.
period
   The period, in seconds, during which the specified statistic is applied.
   The default value is 300 seconds.
comparisonOperator
   The operation for comparing the specified statistic and threshold (for example, "GreaterThanThreshold").
statistic
   The statistic for the metric associated with the alarm (for example, "Average" or "Sum").

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

cloudwatch-log-group-encrypted

Checks whether a log group in Amazon CloudWatch Logs is encrypted. The rule is NON_COMPLIANT if CloudWatch Logs has a log group without encryption enabled.

Identifier: CLOUDWATCH_LOG_GROUP_ENCRYPTED

Trigger type: Periodic

Parameters:

KmsKeyId
   (Optional) Amazon Resource Name (ARN) of an AWS Key Management Service (KMS) key that is used to encrypt the CloudWatch Logs log group.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

cmk-backing-key-rotation-enabled

Checks that key rotation is enabled for each customer master key (CMK). The rule is COMPLIANT, if the key rotation is enabled for specific key object. The rule is not applicable to CMKs that have imported key material.

Identifier: CMK_BACKING_KEY_ROTATION_ENABLED
**Trigger type:** Periodic  
**Parameters:**  
None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**codebuild-project-envvar-awscred-check**

Checks whether the project contains environment variables AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY. The rule is NON_COMPLIANT when the project environment variables contains plaintext credentials.

**Identifier:** CODEBUILD_PROJECT_ENVVAR_AWSCREDS_CHECK  
**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**codebuild-project-source-repo-url-check**

Checks whether the GitHub or Bitbucket source repository URL contains either personal access tokens or user name and password. The rule is COMPLIANT with the usage of OAuth to grant authorization for accessing GitHub or Bitbucket repositories.

**Identifier:** CODEBUILD_PROJECT_SOURCE_REPO_URL_CHECK  
**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**codepipeline-deployment-count-check**

Checks whether the first deployment stage of the AWS CodePipeline performs more than one deployment. Optionally, checks if each of the subsequent remaining stages deploy to more than the specified number of deployments (deploymentLimit). The rule is NON_COMPLIANT if the first stage in the AWS CodePipeline deploys to more than one region and the AWS CodePipeline deploys to more than the number specified in the deploymentLimit.
**Identifier:** CODEPIPELINE_DEPLOYMENT_COUNT_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- deploymentLimit

The maximum number of deployments each stage can perform.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**codepipeline-region-fanout-check**

Checks whether each stage in the AWS CodePipeline deploys to more than N times the number of the regions the AWS CodePipeline has deployed in all the previous combined stages, where N is the region fanout number. The first deployment stage can deploy to a maximum of one region and the second deployment stage can deploy to a maximum number specified in the regionFanoutFactor. If you do not provide a regionFanoutFactor, by default the value is three. For example: If 1st deployment stage deploys to one region and 2nd deployment stage deploys to three regions, 3rd deployment stage can deploy to 12 regions, that is, sum of previous stages multiplied by the region fanout (three) number. The rule is NON_COMPLIANT if the deployment is in more than one region in 1st stage or three regions in 2nd stage or 12 regions in 3rd stage.

**Identifier:** CODEPIPELINE_REGION_FANOUT_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- regionFanoutFactor

The number of regions the AWS CodePipeline has deployed to in all previous stages is the acceptable number of regions any stage can deploy to.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**db-instance-backup-enabled**

Checks whether RDS DB instances have backups enabled. Optionally, the rule checks the backup retention period and the backup window.

**Identifier:** DB_INSTANCE_BACKUP_ENABLED

**Trigger type:** Configuration changes

**Parameters:**

- backupRetentionPeriod

  Retention period for backups.
preferredBackupWindow

Time range in which backups are created.

checkReadReplicas

Checks whether RDS DB instances have backups enabled for read replicas.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

desired-instance-tenancy

Checks instances for specified tenancy. Specify AMI IDs to check instances that are launched from those AMIs or specify host IDs to check whether instances are launched on those Dedicated Hosts. Separate multiple ID values with commas.

Identifier: DESIRED_INSTANCE_TENANCY

Trigger type: Configuration changes

Parameters:

tenancy

The desired tenancy of the instances. Valid values are DEDICATED, HOST, and DEFAULT.

imageId

The rule evaluates instances launched only from the AMI with the specified ID. Separate multiple AMI IDs with commas.

hostId

The ID of the Amazon EC2 Dedicated Host on which the instances are meant to be launched. Separate multiple host IDs with commas.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

desired-instance-type

Checks whether your EC2 instances are of the specified instance types.

For a list of supported Amazon EC2 instance types, see Instance Types in the Amazon EC2 User Guide for Linux Instances.

Identifier: DESIRED_INSTANCE_TYPE

Trigger type: Configuration changes

Parameters:

instanceType

Comma-separated list of EC2 instance types (for example, “t2.small, m4.large, i2.xlarge”).
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

dms-replication-not-public

Checks whether AWS Database Migration Service replication instances are public. The rule is NON_COMPLIANT if PubliclyAccessible field is true.

Identifier: DMS_REPLICATION_NOT_PUBLIC

Trigger type: Periodic

AWS Region: All supported AWS regions

Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

dynamodb-autoscaling-enabled

Checks whether Auto Scaling or On-Demand is enabled on your DynamoDB tables and/or global secondary indexes. Optionally you can set the read and write capacity units for the table or global secondary index.

Identifier: DYNAMODB_AUTOSCALING_ENABLED

Trigger type: Periodic

Parameters:

minProvisionedReadCapacity

The minimum number of units that should be provisioned with read capacity in the Auto Scaling group.

minProvisionedWriteCapacity

The minimum number of units that should be provisioned with write capacity in the Auto Scaling group.

maxProvisionedReadCapacity

The maximum number of units that should be provisioned with read capacity in the Auto Scaling group.

maxProvisionedWriteCapacity

The maximum number of units that should be provisioned with write capacity in the Auto Scaling group.

targetReadUtilization

The target utilization percentage for read capacity. Target utilization is expressed in terms of the ratio of consumed capacity to provisioned capacity.
targetWriteUtilization

The target utilization percentage for write capacity. Target utilization is expressed in terms of the ratio of consumed capacity to provisioned capacity.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

dynamodb-table-encryption-enabled

Checks whether the Amazon DynamoDB tables are encrypted and checks their status. The rule is COMPLIANT if the status is enabled or enabling.

Identifier: DYNAMODB_TABLE_ENCRYPTION_ENABLED

Trigger type: Configuration changes

Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

dynamodb-throughput-limit-check

Checks whether provisioned DynamoDB throughput is approaching the maximum limit for your account. By default, the rule checks if provisioned throughput exceeds a threshold of 80% of your account limits.

Identifier: DYNAMODB_THROUGHPUT_LIMIT_CHECK

Trigger type: Periodic

Parameters:

accountRCUThresholdPercentage

Percentage of provisioned read capacity units for your account. When this value is reached, the rule is marked as NON_COMPLIANT.

accountWCUThresholdPercentage

Percentage of provisioned write capacity units for your account. When this value is reached, the rule is marked as NON_COMPLIANT.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

ebs-optimized-instance

Checks whether EBS optimization is enabled for your EC2 instances that can be EBS-optimized.
**Identifier:** EBS_OPTIMIZED_INSTANCE

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ebs-snapshot-public-restorable-check**

Checks whether Amazon Elastic Block Store snapshots are not publicly restorable. The rule is NON_COMPLIANT if one or more snapshots with the `RestorableByUserIds` field is set to `all`. If this field is set to `all`, then Amazon EBS snapshots are public.

**Identifier:** EBS_SNAPSHOT_PUBLIC_RESTORABLE_CHECK

**Trigger type:** Periodic

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-stopped-instance**

Checks whether there are instances stopped for more than the allowed number of days. The instance is NON_COMPLIANT if the state of the ec2 instance has been stopped for longer than the allowed number of days.

**Identifier:** EC2_STOPPED_INSTANCE

**Trigger type:** Periodic

**AWS Region:** All supported AWS regions

**Parameters:**

allowedDays

(Optional) The number of days an ec2 instance can be stopped before it is NON_COMPLIANT. The default number of days is 30.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
**ec2-instance-detailed-monitoring-enabled**

Checks whether detailed monitoring is enabled for EC2 instances.

**Identifier:** EC2_INSTANCE_DETAILED_MONITORING_ENABLED  
**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-instance-managed-by-systems-manager**

Checks whether the Amazon EC2 instances in your account are managed by AWS Systems Manager.

**Identifier:** EC2_INSTANCE_MANAGED_BY_SSM  
**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-instance-no-public-ip**

Checks whether Amazon Elastic Compute Cloud (Amazon EC2) instances have a public IP association. The rule is NON_COMPLIANT if the publicIp field is present in the Amazon EC2 instance configuration item. This rule applies only to IPv4.

**Identifier:** EC2_INSTANCE_NO_PUBLIC_IP  
**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-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.
**Identifier:** INSTANCES_IN_VPC

**Trigger type:** Configuration changes

**Parameters:**

vpcId

The ID of the VPC that contains these instances.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-managedinstance-applications-blacklisted**

Checks that none of the specified applications are installed on the instance. Optionally, specify the application version. Newer versions of the application will not be blacklisted. You can also specify the platform to apply the rule only to instances running that platform.

**Identifier:** EC2_MANAGEDINSTANCE_APPLICATIONS_BLACKLISTED

**Trigger type:** Configuration changes

**Parameters:**

applicationNames

Comma-separated list of application names. Optionally, specify versions appended with "." (for example, "Chrome:0.5.3, FireFox").

**Note** The application names must be an exact match. For example, use `firefox` on Linux or `firefox-compat` on Amazon Linux. In addition, AWS Config does not currently support wildcards for the `applicationNames` parameter (for example, `firefox*`).

platformType

The platform type (for example, "Linux" or "Windows").

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-managedinstance-applications-required**

Checks whether all of the specified applications are installed on the instance. Optionally, specify the minimum acceptable version. You can also specify the platform to apply the rule only to instances running that platform.

**Note** Ensure that SSM agent is running on the EC2 instance and configure SSM agents.

**Identifier:** EC2_MANAGEDINSTANCE_APPLICATIONS_REQUIRED

**Trigger type:** Configuration changes
Parameters:

**applicationNames**

Comma-separated list of application names. Optionally, specify versions appended with ":" (for example, "Chrome:0.5.3, Firefox").

**Note** The application names must be an exact match. For example, use `firefox` on Linux or `firefox-compat` on Amazon Linux. In addition, AWS Config does not currently support wildcards for the `applicationNames` parameter (for example, `firefox*`).

**platformType**

The platform type (for example, "Linux" or "Windows").

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-managedinstance-association-compliance-status-check**

Checks whether the compliance status of the Amazon EC2 Systems Manager association compliance is COMPLIANT or NON_COMPLIANT after the association execution on the instance. The rule is COMPLIANT if the field status is COMPLIANT.

**Identifier:** EC2_MANAGEDINSTANCE_ASSOCIATION_COMPLIANCE_STATUS_CHECK

**Trigger type:** Configuration changes

**Parameters:**

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**ec2-managedinstance-inventory-blacklisted**

Checks whether instances managed by AWS Systems Manager are configured to collect blacklisted inventory types.

**Identifier:** EC2_MANAGEDINSTANCE_INVENTORY_BLACKLISTED

**Trigger type:** Configuration changes

**Parameters:**

**inventoryNames**

Comma-separated list of Systems Manager inventory types (for example, "AWS:Network, AWS:WindowsUpdate").

**platformType**

Platform type (for example, "Linux").
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

ec2-managedinstance-patch-compliance-status-check

Checks whether the compliance status of the Amazon EC2 Systems Manager patch compliance is COMPLIANT or NON_COMPLIANT after the patch installation on the instance. The rule is COMPLIANT if the field status is COMPLIANT.

Identifier: EC2_MANAGEDINSTANCE_PATCH_COMPLIANCE_STATUS_CHECK

Trigger type: Configuration changes

Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

ec2-managedinstance-platform-check

Checks whether EC2 managed instances have the desired configurations.

Identifier: EC2_MANAGEDINSTANCE_PLATFORM_CHECK

Trigger type: Configuration changes

Parameters:

agentVersion

The version of the agent (for example, "2.0.433.0").

platformType

The platform type (for example, "Linux" or "Windows").

platformVersion

The version of the platform (for example, "2016.09").

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

ec2-security-group-attached-to-eni

Checks that security groups are attached to Amazon Elastic Compute Cloud (Amazon EC2) instances or to an elastic network interface. The rule returns NON_COMPLIANT if the security group is not associated with an Amazon EC2 instance or an elastic network interface.
**ec2-volume-inuse-check**

Checks whether EBS volumes are attached to EC2 instances. Optionally checks if EBS volumes are marked for deletion when an instance is terminated.

**Identifier:** EC2_VOLUME_INUSE_CHECK

**Trigger type:** Configuration changes

**Parameters:**
- deleteOnTermination

  EBS volumes are marked for deletion when an instance is terminated.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#) (p. 173).

**efs-encrypted-check**

Checks whether Amazon Elastic File System (Amazon EFS) is configured to encrypt the file data using AWS Key Management Service (AWS KMS). The rule is NON_COMPLIANT if the encrypted key is set to false on `DescribeFileSystems` or if the `KmsKeyId` on `DescribeFileSystems` does not match the `KmsKeyId` parameter.

**Identifier:** EFS_ENCRYPTED_CHECK

**Trigger type:** Periodic

**Parameters:**
- KmsKeyId (optional)

  Amazon Resource Name (ARN) of the AWS KMS key that is used to encrypt the Amazon EFS file system.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#) (p. 173).
**eip-attached**

Checks whether all Elastic IP addresses that are allocated to a VPC are attached to EC2 instances or in-use elastic network interfaces (ENIs).

Results might take up to 6 hours to become available after an evaluation occurs.

**Identifier:** EIP ATTACHED

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elasticsearch-encrypted-at-rest**

Checks whether Amazon Elasticsearch Service (Amazon ES) domains have encryption at rest configuration enabled. The rule is NON_COMPLIANT if the EncryptionAtRestOptions field is not enabled.

**Identifier:** ELASTICSEARCH_ENCRYPTED_AT_REST

**Trigger type:** Periodic

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elasticsearch-in-vpc-only**

Checks whether Amazon Elasticsearch Service (Amazon ES) domains are in Amazon Virtual Private Cloud (Amazon VPC). The rule is NON_COMPLIANT if the Amazon ES domain endpoint is public.

**Identifier:** ELASTICSEARCH_IN_VPC_ONLY

**Trigger type:** Periodic

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
**elasticache-redis-cluster-automatic-backup-check**

Check if the Amazon ElastiCache Redis clusters have automatic backup turned on. The rule is NON_COMPLIANT if the SnapshotRetentionLimit for Redis cluster is less than the SnapshotRetentionPeriod parameter. For example: If the parameter is 15 then the rule is non-compliant if the snapshotRetentionPeriod is between 0-15.

**Identifier:** ELASTICACHE_REDIS_CLUSTER_AUTOMATIC_BACKUP_CHECK

**Trigger type:** Periodic

**AWS Region:** All supported AWS regions

**Parameters:**

- snapshotRetentionPeriod
  
  (Optional) Minimum snapshot retention period in days for Redis cluster. The default is 15 days.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elb-acm-certificate-required**

Checks whether the Classic Load Balancers use SSL certificates provided by AWS Certificate Manager. To use this rule, use an SSL or HTTPS listener with your Classic Load Balancer. This rule is only applicable to Classic Load Balancers. This rule does not check Application Load Balancers and Network Load Balancers.

**Identifier:** ELB_ACM_CERTIFICATE_REQUIRED

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elb-custom-security-policy-ssl-check**

Checks whether your Classic Load Balancer SSL listeners are using a custom policy. The rule is only applicable if there are SSL listeners for the Classic Load Balancer.

**Identifier:** ELB_CUSTOM_SECURITY_POLICY_SSL_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- ssl-protocols-and-ciphers
  
  Comma-separated list of ciphers and protocol.
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elb-deletion-protection-enabled**

Checks whether Elastic Load Balancing has deletion protection enabled. The rule is NON_COMPLIANT if `deletion_protection.enabled` is false.

**Identifier:** ELB_DELETION_PROTECTION_ENABLED

**Trigger type:** Configuration changes

**Parameters:**

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elb-logging-enabled**

Checks whether the Application Load Balancers and the Classic Load Balancers have logging enabled. The rule is NON_COMPLIANT if the `access_logs.s3.enabled` is false or `access_logs.S3.bucket` is not equal to the `s3BucketName` that you provided.

**Identifier:** ELB_LOGGING_ENABLED

**Trigger type:** Configuration changes

**Parameters:**

- **s3BucketNames (optional)**
  
  Comma-separated list of Amazon S3 bucket names for Elastic Load Balancing to deliver the log files.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**elb-predefined-security-policy-ssl-check**

Checks whether your Classic Load Balancer SSL listeners are using a predefined policy. The rule is only applicable if there are SSL listeners for the Classic Load Balancer.

**Identifier:** ELB_PREDEFINED_SECURITY_POLICY_SSL_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- **predefined-policy-name**
  
  Name of the predefined policy.
**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**emr-kerberos-enabled**

Checks that Amazon EMR clusters have Kerberos enabled. The rule is NON_COMPLIANT if a security configuration is not attached to the cluster or the security configuration does not satisfy the specified rule parameters.

**Identifier:** EMR_KERBEROS_ENABLED

**Trigger type:** Periodic

**AWS Regions:** All supported AWS regions

**Parameters:**

- **ticketLifetimeInHours (optional)**
  - Period for which Kerberos ticket issued by cluster's KDC is valid.
- **realm (optional)**
  - Kerberos realm name of the other realm in the trust relationship.
- **domain (optional)**
  - Domain name of the other realm in the trust relationship.
- **adminServer (optional)**
  - Fully qualified domain of the admin server in the other realm of the trust relationship.
- **kdcServer (optional)**
  - Fully qualified domain of the KDC server in the other realm of the trust relationship.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**emr-master-no-public-ip**

Checks whether Amazon Elastic MapReduce (EMR) clusters' master nodes have public IPs. The rule is NON_COMPLIANT if the master node has a public IP.

**Note**

This rule checks clusters that are in RUNNING or WAITING state.

**Identifier:** EMR_MASTER_NO_PUBLIC_IP

**Trigger type:** Periodic

**AWS Region:** All supported AWS regions

**Parameters:**
None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**encrypted-volumes**

Checks whether the EBS volumes that are in an attached state are encrypted. If you specify the ID of a KMS key for encryption using the kmsId parameter, the rule checks if the EBS volumes in an attached state are encrypted with that KMS key.

For more information, see Amazon EBS Encryption in the *Amazon EC2 User Guide for Linux Instances*.

**Identifier:** ENCRYPTED_VOLUMES  
**Trigger type:** Configuration changes  
**Parameters:**

kmsId

ID or ARN of the KMS key that is used to encrypt the volume.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**fms-security-group-audit-policy-check**

Checks whether the security groups associated inScope resources are compliant with the master security groups at each rule level based on allowSecurityGroup and denySecurityGroup flag.

**Note**

Only AWS Firewall Manager can create this rule.

**Identifier:** FMS_SECURITY_GROUP_AUDIT_POLICY_CHECK  
**Trigger type:** Configuration changes  

**AWS Regions:** Only available in US East (N. Virginia), EU (Ireland), US West (N. California), Asia Pacific (Singapore), Asia Pacific (Tokyo), US West (Oregon), Asia Pacific (Sydney), EU (Frankfurt), Asia Pacific (Seoul), US East (Ohio), and EU (London).

**Parameters:**

masterSecurityGroupsIds (mandatory)

Comma-separated list of master security groups IDs. The rule will check if security groups associated inScope resources are compliant with the master security groups at each rule level.

resourceTags (mandatory)

The resource tags associated with the rule (for example, 

```
{ "tagKey1" : ["tagValue1"],
  "tagKey2" : ["tagValue2", "tagValue3"] }
```

).
inScope (mandatory)

If true, the AWS Config rule owner is in Firewall Manager security group audit policy scope.

excludeResourceTags (mandatory)

If true, exclude resources that match resourceTags.

resourceTypes (mandatory)

The resource types such as Amazon EC2 instance or elastic network interface or security group supported by this rule.

fmsRemediationEnabled (mandatory)

If true, AWS Firewall Manager will update NON_COMPLIANT resources according to FMS policy. AWS Config ignores this parameter when you create this rule.

allowSecurityGroup (mandatory)

If true, the rule will check to ensure that all inScope security groups are within the reference security group's inbound/outbound rules.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

fms-security-group-content-check

Checks whether AWS Firewall Manager created security groups content is the same as the master security groups. The rule is NON_COMPLIANT if the content does not match.

Note

Only AWS Firewall Manager can create this rule.

Identifier: FMS_SECURITY_GROUP_CONTENT_CHECK

Trigger type: Configuration changes

AWS Regions: Only available in US East (N. Virginia), EU (Ireland), US West (N. California), Asia Pacific (Singapore), Asia Pacific (Tokyo), US West (Oregon), Asia Pacific (Sydney), EU (Frankfurt), Asia Pacific (Seoul), US East (Ohio), and EU (London).

Parameters:

vpcIds (mandatory)

Comma-separated list of VPC IDs in the account.

securityGroupsIds (mandatory)

Comma-separated list of security groups IDs created by Firewall Manager in every Amazon VPC in an account. They are sorted by VPC IDs.

fmsRemediationEnabled (mandatory)

If true, AWS Firewall Manager will update NON_COMPLIANT resources according to FMS policy. AWS Config ignores this parameter when you create this rule.

revertManualSecurityGroupChangesFlag (mandatory)

If true, AWS Firewall Manager will check the security groups in the securityGroupsIds parameter.
allowSecurityGroup (mandatory)

If true, the rule will check to ensure that all inScope security groups are within the reference security group's inbound/outbound rules.

masterSecurityGroupsIds (optional)

This parameter only applies to AWS Firewall Manager admin account. Comma-separated list of master security groups ID in Firewall Manager admin account.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

fms-security-group-resource-association-check

Checks whether Amazon EC2 or an elastic network interface is associated with AWS Firewall Manager security groups. The rule is NON_COMPLIANT if the resources are not associated with FMS security groups.

Note
Only AWS Firewall Manager can create this rule.

Identifier: FMS_SECURITY_GROUP_RESOURCE_ASSOCIATION_CHECK

Trigger type: Configuration changes

AWS Regions: Only available in US East (N. Virginia), EU (Ireland), US West (N. California), Asia Pacific (Singapore), Asia Pacific (Tokyo), US West (Oregon), Asia Pacific (Sydney), EU (Frankfurt), Asia Pacific (Seoul), US East (Ohio), and EU (London).

Parameters:

vpcIds (mandatory)

Comma-separated list of VPC IDs in the account.

securityGroupsIds (mandatory)

Comma-separated list of security groups IDs created by Firewall Manager in every Amazon VPC in an account. They are sorted by VPC IDs.

resourceTags (mandatory)

The resource tags such as Amazon EC2 instance or elastic network interface associated with the rule (for example, { "tagKey1" : ["tagValue1"], "tagKey2" : ["tagValue2", "tagValue3"] }).

excludeResourceTags (mandatory)

If true, exclude resources that match resourceTags.

resourceTypes (mandatory)

The resource types such as Amazon EC2 instance or elastic network interface or security group supported by this rule.

fmsRemediationEnabled (mandatory)

If true, AWS Firewall Manager will update NON_COMPLIANT resources according to FMS policy. AWS Config ignores this parameter when you create this rule.
exclusiveResourceSecurityGroupManagementFlag (mandatory)

If true, only allows AWS Firewall Manager created security groups associated with resource.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

fms-shield-resource-policy-check

Checks whether an Application Load Balancer, Amazon CloudFront distributions, Elastic Load Balancer or Elastic IP has AWS Shield protection. This rule also checks if they have web ACL associated for Application Load Balancer and Amazon CloudFront distributions.

**Identifier:** FMS_SHIELD_RESOURCE_POLICY_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- **webACLId**
  - The WebACLId of the web ACL.

- **resourceTags**
  - The resource tags associated with the rule (for example, `{ "tagKey1" : ["tagValue1"], "tagKey2" : ["tagValue2", "tagValue3"] }`).

- **excludeResourceTags**
  - If true, exclude the resources that match the resourceTags. If false, include all the resources that match the resourceTags.

- **fmsManagedToken**
  - A token generated by AWS Firewall Manager when creating the rule in your account. AWS Config ignores this parameter when you create this rule.

- **fmsRemediationEnabled**
  - If true, AWS Firewall Manager will update NON_COMPLIANT resources according to FMS policy. AWS Config ignores this parameter when you create this rule.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

fms-webacl-resource-policy-check

Checks whether the web ACL is associated with an Application Load Balancer, API Gateway stage, or Amazon CloudFront distributions. When AWS Firewall Manager creates this rule, the FMS policy owner specifies the WebACLId in the FMS policy and can optionally enable remediation.

**Identifier:** FMS_WEBACL_RESOURCE_POLICY_CHECK

**Trigger type:** Configuration changes

**Parameters:**
webACLId

The WebACLId of the web ACL.

resourceTags

The resource tags (Application Load Balancer and Amazon CloudFront distributions) associated with the rule (for example, `{ "tagKey1" : ["tagValue1"], "tagKey2" : ["tagValue2", "tagValue3"] }`).

excludeResourceTags

If true, exclude the resources that match the resourceTags.

fmsManagedToken

A token generated by AWS Firewall Manager when creating the rule in your account. AWS Config ignores this parameter when you create this rule.

fmsRemediationEnabled

If true, AWS Firewall Manager will update NON_COMPLIANT resources according to FMS policy. AWS Config ignores this parameter when you create this rule.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

fms-webacl-rulegroup-association-check

Checks that the rule groups associate with the web ACL at the correct priority. The correct priority is decided by the rank of the rule groups in the ruleGroups parameter. When AWS Firewall Manager creates this rule, it assigns the highest priority 0 followed by 1, 2, and so on. The FMS policy owner specifies the ruleGroups rank in the FMS policy and can optionally enable remediation.

Identifier: FMS_WEBACL_RULEGROUP_ASSOCIATION_CHECK

Trigger type: Configuration changes

Parameters:

ruleGroups

Comma-separated list of RuleGroupIds and WafOverrideAction pairs (for example, RuleGroupId-1:NONE, RuleGroupId-2:COUNT). For this example, RuleGroupId-1 receives the highest priority 0 and RuleGroupId-2 receives priority 1.

fmsManagedToken

A token generated by AWS Firewall Manager when creating the rule in your account. AWS Config ignores this parameter when you create this rule.

fmsRemediationEnabled

If true, AWS Firewall Manager will update NON_COMPLIANT resources according to FMS policy. AWS Config ignores this parameter when you create this rule.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
**guardduty-enabled-centralized**

Checks whether Amazon GuardDuty is enabled in your AWS account and region. If you provide an AWS account for centralization, the rule evaluates the Amazon GuardDuty results in the centralized account. The rule is COMPLIANT when Amazon GuardDuty is enabled.

**Identifier:** GUARDDUTY_ENABLED_CENTRALIZED

**Trigger type:** Configuration changes

**Parameters:**

CentralMonitoringAccount (optional)

Specify 12-digit AWS Account for centralization of Amazon GuardDuty results.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**guardduty-non-archived-findings**

Checks whether the Amazon GuardDuty has findings that are non archived. The rule is NON_COMPLIANT if Amazon GuardDuty has non archived low/medium/high severity findings older than the specified number in the daysLowSev/daysMediumSev/daysHighSev parameter.

**Identifier:** GUARDDUTY_NON_ARCHIVED_FINDINGS

**Trigger type:** Periodic

**AWS Region:** All supported AWS regions except China (Beijing) and China (Ningxia)

**Parameters:**

daysLowSev

The number of days Amazon GuardDuty low severity findings are allowed to stay non archived. The default is 30 days.

daysMediumSev

The number of days the Amazon GuardDuty medium severity findings are allowed to stay non archived. The default is 7 days.

daysHighSev

The number of days Amazon GuardDuty high severity findings are allowed to stay non archived. The default is 1 day.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**iam-group-has-users-check**

Checks whether IAM groups have at least one IAM user.
Identifier: IAM_GROUP_HAS_USERS_CHECK

Trigger type: Configuration changes

Parameters:
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

iam-password-policy

Checks whether the account password policy for IAM users meets the specified requirements.

Identifier: IAM_PASSWORD_POLICY

Trigger type: Periodic

Parameters:

RequireUppercaseCharacters
  Require at least one uppercase character in password.
RequireLowercaseCharacters
  Require at least one lowercase character in password.
RequireSymbols
  Require at least one symbol in password.
RequireNumbers
  Require at least one number in password.
MinimumPasswordLength
  Password minimum length.
PasswordReusePrevention
  Number of passwords before allowing reuse.
MaxPasswordAge
  Number of days before password expiration.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

iam-policy-blacklisted-check

Checks whether for each IAM resource, a policy ARN in the input parameter is attached to the IAM resource. The rule is NON_COMPLIANT if the policy ARN is attached to the IAM resource. AWS Config marks the resource as COMPLIANT if the IAM resource is part of the exceptionList parameter irrespective of the presence of the policy ARN.
Identifier: IAM_POLICY_BLACKLISTED_CHECK

Trigger type: Configuration changes

Parameters:

policyArns

Comma-separated list of policy ARNs.

exceptionList

Comma-separated list IAM users, groups, or roles that are exempt from this rule. For example, users: [user1;user2], groups:[group1;group2], roles:[role1;role2;role3].

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

iam-policy-in-use

Checks whether the IAM policy ARN is attached to an IAM user, or an IAM group with one or more IAM users, or an IAM role with one or more trusted entity.

Identifier: IAM_POLICY_IN_USE

Trigger type: Periodic

Parameters:

policyARN (mandatory)

An IAM policy Amazon Resource Name (ARN) to be checked

policyUsageType (optional)

Specify the policy to be attached as an IAM user, IAM group, or IAM role. Valid values are IAM_USER, IAM_GROUP, IAM_ROLE, or ANY. Default value is ANY.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

iam-policy-no-statements-with-admin-access

Checks the IAM policies that you create, such as identity-based or resource-based policies, for Allow statements that grant permissions to all actions on all resources. The rule is NON_COMPLIANT if any policy statement includes “Effect”: “Allow” with “Action”: “*” over “Resource”: “*”. For example, the following statement is NON_COMPLIANT:

```
"Statement": [
    {
        "Sid": "VisualEditor",
        "Effect": "Allow",
        "Action": "service:*",
        "Resource": "*"
    }
]
```
This rule checks only the IAM policies that you create. It does not check IAM Managed Policies. When you enable the rule, this rule checks all of the customer managed policies in your account, and all new policies that you create.

**Identifier:** IAM_POLICY_NO_STATEMENTS_WITH_ADMIN_ACCESS

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**iam-role-managed-policy-check**

Checks that AWS Identity and Access Management (IAM) policies in a list of policies are attached to all AWS roles. The rule is NON_COMPLIANT if the IAM managed policy is not attached to the IAM role.

**Identifier:** IAM_ROLE_MANAGED_POLICY_CHECK

**Trigger type:** Configuration changes

**Parameters:**

managedPolicyNames

  Comma-separated list of AWS managed policy ARNs.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**iam-root-access-key-check**

Checks whether the root user access key is available. The rule is COMPLIANT if the user access key does not exist.

**Identifier:** IAM_ROOT_ACCESS_KEY_CHECK

**Trigger type:** Periodic

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
**iam-user-group-membership-check**

Checks whether IAM users are members of at least one IAM group.

**Identifier:** IAM_USER_GROUP_MEMBERSHIP_CHECK

**Trigger type:** Configuration changes

**Parameters:**

*groupName*

Comma-separated list of IAM groups in which IAM users must be members.

**Note**

This rule does not support group names with commas.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**iam-user-mfa-enabled**

Checks whether the AWS Identity and Access Management users have multi-factor authentication (MFA) enabled.

**Identifier:** IAM_USER_MFA_ENABLED

**Trigger type:** Periodic

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**iam-user-no-policies-check**

Checks that none of your IAM users have policies attached. IAM users must inherit permissions from IAM groups or roles.

**Identifier:** IAM_USER_NO_POLICIES_CHECK

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
**iam-user-unused-credentials-check**

Checks whether your AWS Identity and Access Management (IAM) users have passwords or active access keys that have not been used within the specified number of days you provided. Re-evaluating this rule within 4 hours of the first evaluation will have no effect on the results.

**Identifier:** IAM_USER_UNUSED_CREDENTIALS_CHECK

**Trigger type:** Periodic

**Parameters:**

```maxCredentialUsageAge```

Maximum number of days within which a credential must be used. The default value is 90 days.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**internet-gateway-authorized-vpc-only**

Checks that Internet gateways (IGWs) are only attached to an authorized Amazon Virtual Private Cloud (VPCs). The rule is NON_COMPLIANT if IGWs are not attached to an authorized VPC.

**Identifier:** INTERNET_GATEWAY_AUTHORIZED_VPC_ONLY

**Trigger type:** Configuration changes

**AWS Region:** All supported AWS regions

**Parameters:**

```authorizedVpcIds```

Comma-separated list of the authorized VPC IDs with attached IGWs. If parameter is not provided all attached IGWs will be NON_COMPLIANT.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**kms-cmk-not-scheduled-for-deletion**

Checks whether customer master keys (CMKs) are not scheduled for deletion in AWS Key Management Service (KMS). The rule is NON_COMPLIANT if CMKs are scheduled for deletion.

**Identifier:** KMS_CMK_NOT_SCHEDULED_FOR_DELETION

**Trigger type:** Periodic

**AWS Region:** All supported AWS regions

**Parameters:**


kmsKeyIds

Comma-separated list of specific customer managed key IDs not to be scheduled for deletion. If you do not specify any keys, the rule checks all the keys.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**lambda-concurrency-check**

Checks whether the AWS Lambda function is configured with function-level concurrent execution limit. The rule is NON_COMPLIANT if the Lambda function is not configured with function-level concurrent execution limit.

**Identifier:** LAMBDA_CONCURRENCY_CHECK

**Trigger type:** Configuration changes

**Parameters:**

ConcurrentLimitLow

(Optional) Minimum concurrency execution limit

ConcurrentLimitHigh

(Optional) Maximum concurrency execution limit

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**lambda-dlq-check**

Checks whether an AWS Lambda function is configured with a dead-letter queue. The rule is NON_COMPLIANT if the Lambda function is not configured with a dead-letter queue.

**Identifier:** LAMBDA_DLQ_CHECK

**Trigger type:** Configuration changes

**Parameters:**

dlqArns

(Optional) Comma-separated list of Amazon SQS and Amazon SNS ARNs that must be configured as the Lambda function dead-letter queue target.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).
**lambda-function-public-access-prohibited**

Checks whether the AWS Lambda function policy attached to the Lambda resource prohibits public access. If the Lambda function policy allows public access it is NON_COMPLIANT.

**Identifier:** LAMBDA_FUNCTION_PUBLIC_ACCESS_PROHIBITED

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**lambda-function-settings-check**

Checks that the lambda function settings for runtime, role, timeout, and memory size match the expected values.

**Identifier:** LAMBDA_FUNCTION_SETTINGS_CHECK

**Trigger type:** Configuration changes

**Parameters:**

runtime

- Comma-separated list of runtime values.

role

- IAM role.

timeout

- Timeout in seconds.

memorySize

- Memory size in MB.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**lambda-inside-vpc**

Checks whether an AWS Lambda function is in an Amazon Virtual Private Cloud. The rule is NON_COMPLIANT if the Lambda function is not in a VPC.

**Identifier:** LAMBDA_INSIDE_VPC

**Trigger type:** Configuration changes
Parameters:

subnetId

(Optional) Comma-separated list of subnet IDs that Lambda functions must be associated with.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

mfa-enabled-for-iam-console-access

Checks whether AWS Multi-Factor Authentication (MFA) is enabled for all AWS Identity and Access Management (IAM) users that use a console password. The rule is COMPLIANT if MFA is enabled.

Identifier: MFA_ENABLED_FOR_IAM_CONSOLE_ACCESS

Trigger type: Periodic

Parameters:

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

multi-region-cloud-trail-enabled

Checks that there is at least one multi-region AWS CloudTrail. The rule is NON_COMPLIANT if the trails do not match inputs parameters.

Identifier: MULTI_REGION_CLOUD_TRAIL_ENABLED

Trigger type: Periodic

Parameters (optional):

s3BucketName

Name of Amazon S3 bucket for AWS CloudTrail to deliver log files to.

snsTopicArn

Amazon SNS topic ARN for AWS CloudTrail to use for notifications.

cloudWatchLogsLogGroupArn

Amazon CloudWatch log group ARN for AWS CloudTrail to send data to.

includeManagementEvents

Event selector to include management events for the AWS CloudTrail.

readWriteType

Type of events to record. Valid values are ReadOnly, WriteOnly and ALL.
**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#).  

**rds-enhanced-monitoring-enabled**

Checks whether enhanced monitoring is enabled for Amazon Relational Database Service (Amazon RDS) instances.

**Identifier:** RDS_ENHANCED_MONITORING_ENABLED  
**Trigger type:** Configuration changes  
**AWS Region:** All supported AWS regions  
**Parameters:**  
- **monitoringInterval**  
  
  (Optional) An integer value in seconds between points when enhanced monitoring metrics are collected for the database instance. The valid values are 1, 5, 10, 15, 30, and 60.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#).  

**rds-instance-public-access-check**

Check whether the Amazon Relational Database Service instances are not publicly accessible. The rule is NON_COMPLIANT if the `publiclyAccessible` field is true in the instance configuration item.

**Identifier:** RDS_INSTANCE_PUBLIC_ACCESS_CHECK  
**Trigger type:** Configuration changes  
**Parameters:** None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#).  

**rds-multi-az-support**

Checks whether high availability is enabled for your RDS DB instances.

In a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone. For more information, see [High Availability (Multi-AZ)] in the [Amazon RDS User Guide](#).  

**Note**  
This rule does not evaluate Amazon Aurora databases.
**Identifier:** RDS_MULTI_AZ_SUPPORT

**Trigger type:** Configuration changes

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**rds-snapshots-public-prohibited**

Checks if Amazon Relational Database Service (Amazon RDS) snapshots are public. The rule is NON_COMPLIANT if any existing and new Amazon RDS snapshots are public.

**Identifier:** RDS_SNAPSHOTS_PUBLIC_PROHIBITED

**Trigger type:** Configuration changes

**Evaluated resource types:** AWS::RDS::DBSnapshot and AWS::RDS::DBClusterSnapshot

**Parameters:**

None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**rds-storage-encrypted**

Checks whether storage encryption is enabled for your RDS DB instances.

**Identifier:** RDS_STORAGE_ENCRYPTED

**Trigger type:** Configuration changes

**Parameters:**

kmsKeyId

KMS key ID or ARN used to encrypt the storage.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**redshift-cluster-configuration-check**

Checks whether Amazon Redshift clusters have the specified settings.

**Identifier:** REDSHIFT_CLUSTER_CONFIGURATION_CHECK
**Trigger type:** Configuration changes

**Parameters:**

- `clusterDbEncrypted`
  - Database encryption is enabled.
- `nodeTypes`
  - Specify node type.
- `loggingEnabled`
  - Audit logging is enabled.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**redshift-cluster-maintenancesettings-check**

Checks whether Amazon Redshift clusters have the specified maintenance settings.

**Identifier:** REDSHIFT_CLUSTER_MAINTENANCESETTINGS_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- `allowVersionUpgrade`
  - Allow version upgrade is enabled.
- `preferredMaintenanceWindow`
  - Scheduled maintenance window for clusters (for example, Mon:09:30-Mon:10:00).
- `automatedSnapshotRetentionPeriod`
  - Number of days to retain automated snapshots.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**redshift-cluster-public-access-check**

Checks whether Amazon Redshift clusters are not publicly accessible. The rule is NON_COMPLIANT if the `publiclyAccessible` field is true in the cluster configuration item.

**Identifier:** REDSHIFT_CLUSTER_PUBLIC_ACCESS_CHECK

**Trigger type:** Configuration changes

**Parameters:**

None
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

required-tags

Checks whether your resources have the tags that you specify. For example, you can check whether your EC2 instances have the ‘CostCenter’ tag. Separate multiple values with commas.

**Important**
The supported resource types for this rule are as follows:

- ACM::Certificate
- AutoScaling::AutoScalingGroup
- CloudFormation::Stack
- CodeBuild::Project
- DynamoDB::Table
- EC2::CustomerGateway
- EC2::Instance
- EC2::InternetGateway
- EC2::NetworkAcl
- EC2::NetworkInterface
- EC2::RouteTable
- EC2::SecurityGroup
- EC2::Subnet
- EC2::Volume
- EC2::VPC
- EC2::VPNConnection
- EC2::VPNGateway
- ElasticLoadBalancing::LoadBalancer
- ElasticLoadBalancingV2::LoadBalancer
- RDS::DBInstance
- RDS::DBSecurityGroup
- RDS::DBSnapshot
- RDS::DBSubnetGroup
- RDS::EventSubscription
- Redshift::Cluster
- Redshift::ClusterParameterGroup
- Redshift::ClusterSecurityGroup
- Redshift::ClusterSnapshot
- Redshift::ClusterSubnetGroup
- S3::Bucket

**Identifier:** REQUIRED_TAGS

**Trigger type:** Configuration changes

**Parameters:**
tag1Key
   Key of the required tag.
tag1Value
   Optional value of the required tag. Separate multiple values with commas.

AWS CloudFormation template
To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

restricted-common-ports
Checks whether the incoming SSH traffic for the security groups is accessible to the specified ports. The rule is COMPLIANT when the IP addresses of the incoming SSH traffic in the security group are restricted to the specified ports. This rule applies only to IPv4.

Identifier: RESTRICTED_INCOMING_TRAFFIC

Trigger type: Configuration changes

Parameters:
blockedPort1
   Blocked TCP port number.
blockedPort2
   Blocked TCP port number.
blockedPort3
   Blocked TCP port number.
blockedPort4
   Blocked TCP port number.
blockedPort5
   Blocked TCP port number.

AWS CloudFormation template
To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

restricted-ssh
Checks whether the incoming SSH traffic for the security groups is accessible. The rule is COMPLIANT when the IP addresses of the incoming SSH traffic in the security groups are restricted. This rule applies only to IPv4.

Identifier: INCOMING_SSH_DISABLED

Trigger type: Configuration changes

Parameters:
None
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

root-account-hardware-mfa-enabled

Checks whether your AWS account is enabled to use multi-factor authentication (MFA) hardware device to sign in with root credentials. The rule is NON_COMPLIANT if any virtual MFA devices are permitted for signing in with root credentials.

Identifier: ROOT_ACCOUNT_HARDWARE_MFA_ENABLED

Trigger type: Periodic

Parameters:
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

root-account-mfa-enabled

Checks whether users of your AWS account require a multi-factor authentication (MFA) device to sign in with root credentials.

Identifier: ROOT_ACCOUNT_MFA_ENABLED

Trigger type: Periodic

Parameters:
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

s3-account-level-public-access-blocks

Checks whether the required public access block settings are configured from account level. The rule is only NON_COMPLIANT when the fields set below do not match the corresponding fields in the configuration item.

Identifier: S3_ACCOUNT_LEVEL_PUBLIC_ACCESS_BLOCKS

Trigger type: Configuration changes

AWS Region: All supported AWS regions except Middle East (Bahrain)

Parameters:

ignorePublicAcls

(Optional) Either enforced (True) or not (False). The default is True.
blockPublicPolicy

(Optional) Either enforced (True) or not (False). The default is True.

blockPublicAcls

(Optional) Either enforced (True) or not (False). The default is True.

restrictPublicBuckets

(Optional) Either enforced (True) or not (False). The default is True.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

s3-bucket-blacklisted-actions-prohibited

Checks that the Amazon Simple Storage Service bucket policy does not allow blacklisted bucket-level and object-level actions on resources in the bucket for principals from other AWS accounts. For example, the rule checks that the Amazon S3 bucket policy does not allow another AWS account to perform any s3:GetBucket* actions and s3:DeleteObject on any object in the bucket. The rule is NON_COMPLIANT if any blacklisted actions are allowed by the Amazon S3 bucket policy.

Identifier: S3_BUCKET_BLACKLISTED_ACTIONS_PROHIBITED

Trigger type: Configuration changes

Parameters:

blacklistedactionpatterns

Comma-separated list of blacklisted action patterns, for example, s3:GetBucket* and s3:DeleteObject.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

s3-bucket-policy-not-more-permissive

Verifies that your Amazon Simple Storage Service bucket policies do not allow other inter-account permissions than the control Amazon S3 bucket policy that you provide.

Note

If you provide an invalid parameter value, you will see the following error: Value for controlPolicy parameter must be an Amazon S3 bucket policy.

Identifier: S3_BUCKET_POLICY_NOT_MORE_PERMISSIVE

Trigger type: Configuration changes

Parameters:

controlPolicy

Amazon S3 bucket policy that defines an upper bound on the permissions of your S3 buckets. The policy can be a maximum of 1024 characters long.
An example of a control policy is as follows.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": "111122223333"
      },
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "*"
    },
    {
      "Principal": {
        "AWS": "44445556666"
      },
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "*"
    }
  ]
}
```

The first Allow statement specifies that the AWS account ID 111122223333 can retrieve objects (s3:GetObject) on any resource (*). The second Allow statement specifies that the AWS account ID 44445556666 can perform any s3 action (s3:* ) on any resource (*).

Examples of NON_COMPLIANT bucket policies with the above control policy as an input parameter for the rule are as follows.

The following bucket policy is NON_COMPLIANT because the bucket policy allows permissions for the IAM user, Alice, in the AWS account ID 888899998888. These permissions are implicitly denied by the control policy.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": 
        "arn:aws:iam::888899998888:user/Alice"
      },
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::example-bucket/*"
    }
  ]
}
```

The following bucket policy is NON_COMPLIANT because the bucket policy allows the AWS account ID 111122223333 permissions to perform s3:PutBucketPolicy that is implicitly denied by the control policy.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": ["arn:aws:iam::111122223333:"
      },
      "Effect": "Allow",
      "Action": "s3:PutBucketPolicy",
      "Resource": "arn:aws:s3:::example-bucket/*"
    }
  ]
}
```
Examples of **COMPLIANT** bucket policies are as follows.

The following bucket policy is COMPLIANT because the control policy allows principals from the AWS account ID `11112222333` to perform `s3:GetObject` on any object.

```json
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Principal": {
            "AWS": [
                "arn:aws:iam::11112222333:user/Bob"
            ]
        },
        "Effect": "Allow",
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::example-bucket/photos/*"
    }
    ]
}
```

The following bucket policy is COMPLIANT because the control policy allows a principal with the AWS account ID `444455556666` to perform any S3 action.

```json
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Principal": {
            "AWS": [
                "444455556666"
            ]
        },
        "Effect": "Allow",
        "Action": "s3:*Configuration",
        "Resource": "arn:aws:s3:::example-bucket"
    }
    ]
}
```

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](p. 173).

**s3-bucket-logging-enabled**

Checks whether logging is enabled for your S3 buckets.

**Identifier:** `S3_BUCKET_LOGGING_ENABLED`
**Trigger type:** Configuration changes

**Parameters:**

- **targetBucket**
  Target S3 bucket for storing server access logs.

- **targetPrefix**
  Prefix of the target S3 bucket for storing server access logs.

---

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#) (p. 173).

### s3-bucket-policy-grantee-check

Checks that the access granted by the Amazon S3 bucket is restricted by any of the AWS principals, federated users, service principals, IP addresses, or VPCs that you provide. The rule is COMPLIANT if a bucket policy is not present.

For example, if the input parameter to the rule is the list of two principals: 111122223333 and 444455556666 and the bucket policy specifies that only 111122223333 can access the bucket, then the rule is COMPLIANT. With the same input parameters: If the bucket policy specifies that 111122223333 and 444455556666 can access the bucket, it is also compliant. However, if the bucket policy specifies that 999900009999 can access the bucket, the rule is NON-COMPLIANT.

**Identifier:** S3_BUCKET_POLICY_GRANTEE_CHECK

**Trigger type:** Configuration changes

**Parameters:**

- **awsPrincipals**
  Comma-separated list of principals such as IAM User ARNs, IAM Role ARNs and AWS accounts, for example ‘arn:aws:iam::111122223333:user/Alice, arn:aws:iam::444455556666:role/Bob, 123456789012’.

- **servicePrincipals**
  Comma-separated list of service principals, for example ‘cloudtrail.amazonaws.com, lambda.amazonaws.com’.

- **federatedUsers**
  Comma-separated list of identity providers for web identity federation such as Amazon Cognito and SAML identity providers. For example, you can provide as parameter ‘cognito-identity.amazonaws.com, arn:aws:iam::111122223333:saml-provider/my-provider’.

- **ipAddresses**
  Comma-separated list of CIDR formatted IP addresses, for example ‘10.0.0.1, 192.168.1.0/24, 2001:db8::/32’.

- **vpcIds**
  Comma-separated list of Amazon Virtual Private Cloud (Amazon VPC) IDs, for example ‘vpc-1234abc0, vpc-ab1234e0’.
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

s3-bucket-public-read-prohibited

Checks that your Amazon S3 buckets do not allow public read access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).

The rule is compliant when both of the following are true:

- The Block Public Access setting restricts public policies or the bucket policy does not allow public read access.
- The Block Public Access setting restricts public ACLs or the bucket ACL does not allow public read access.

The rule is noncompliant when:

- If the Block Public Access setting does not restrict public policies, AWS Config evaluates whether the policy allows public read access. If the policy allows public read access, the rule is noncompliant.
- If the Block Public Access setting does not restrict public bucket ACLs, AWS Config evaluates whether the bucket ACL allows public read access. If the bucket ACL allows public read access, the rule is noncompliant.

**Identifier:** S3_BUCKET_PUBLIC_READ_PROHIBITED

**Trigger type:** Configuration changes

**Parameters:**

None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

s3-bucket-public-write-prohibited

Checks that your Amazon S3 buckets do not allow public write access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).

The rule is compliant when both of the following are true:

- The Block Public Access setting restricts public policies or the bucket policy does not allow public write access.
- The Block Public Access setting restricts public ACLs or the bucket ACL does not allow public write access.

The rule is noncompliant when:

- If the Block Public Access setting does not restrict public policies, AWS Config evaluates whether the policy allows public write access. If the policy allows public write access, the rule is noncompliant.
If the Block Public Access setting does not restrict public bucket ACLs, AWS Config evaluates whether the bucket ACL allows public write access. If the bucket ACL allows public write access, the rule is noncompliant.

**Identifier:** S3_BUCKET_PUBLIC_WRITE_PROHIBITED

**Trigger type:** Configuration changes

**Parameters:**
None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](p. 173).

**s3-bucket-replication-enabled**

Checks whether S3 buckets have cross-region replication enabled.

**Identifier:** S3_BUCKET_REPLICATION_ENABLED

**Trigger type:** Configuration changes

**Parameters:**
None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](p. 173).

**s3-bucket-server-side-encryption-enabled**

Checks that your Amazon S3 bucket either has Amazon S3 default encryption enabled or that the S3 bucket policy explicitly denies put-object requests without server side encryption.

**Identifier:** S3_BUCKET_SERVER_SIDE_ENCRYPTION_ENABLED

**Trigger type:** Configuration changes

**Parameters:**
None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](p. 173).

**s3-bucket-ssl-requests-only**

Checks whether S3 buckets have policies that require requests to use Secure Socket Layer (SSL).
Identifier: S3_BUCKET_SSL_REQUESTS_ONLY

Trigger type: Configuration changes

Parameters:

None

An example of a bucket policy that is **COMPLIANT** with the SSL AWS Config rule is as follows:

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": ["123412341234"]
            },
            "Action": "s3:Get*",
            "Resource": "arn:aws:s3:::example-bucket/*"
        },
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "*",
            "Resource": "arn:aws:s3:::example-bucket/private/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

An example of a bucket policy that is **NON_COMPLIANT** with the SSL AWS Config rule is as follows:

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": ["123412341234"]
            },
            "Action": "s3:Get*",
            "Resource": "arn:aws:s3:::example-bucket/*"
        },
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "*",
            "Resource": "arn:aws:s3:::example-bucket/private/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

s3-bucket-versioning-enabled

Checks whether versioning is enabled for your S3 buckets. Optionally, the rule checks if MFA delete is enabled for your S3 buckets.

Identifier: S3_BUCKET_VERSIONING_ENABLED

Trigger type: Configuration changes

Parameters:

isMfaDeleteEnabled

MFA delete is enabled for your S3 buckets.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

sagemaker-endpoint-configuration-kms-key-configured

Checks whether AWS Key Management Service (KMS) key is configured for an Amazon SageMaker endpoint configuration. The rule is NON_COMPLIANT if KmsKeyId is not specified for the Amazon SageMaker endpoint configuration.

Identifier: SAGEMAKER_ENDPOINT_CONFIGURATION_KMS_KEY_CONFIGURED

Trigger type: Periodic

AWS Region: All supported AWS regions except China (Beijing), China (Ningxia), and AWS GovCloud (US-Gov-East)

Parameters:

kmsKeyArns

(Optional) Comma-separated list of specific AWS KMS key ARNs allowed for an Amazon SageMaker endpoint configuration.

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

sagemaker-notebook-kms-configured

Check whether an AWS Key Management Service (KMS) key is configured for Amazon SageMaker notebook instance. The rule is not NON_COMPLIANT if kmsKeyId is not specified for the Amazon SageMaker notebook instance.
**Identifier:** SAGEMAKER_NOTEBOOK_INSTANCE_KMS_KEY_CONFIGURED

**Trigger type:** Periodic

**AWS Regions:** All supported AWS regions except AWS GovCloud (US-Gov-East), China (Beijing), and China (Ningxia)

**Parameters:**
- **keyArns** (optional)
  
  Comma-separated list of allowed AWS KMS key IDs allowed for Amazon SageMaker notebook instance.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**sagemaker-notebook-no-direct-internet-access**

Checks whether direct internet access is disabled for an Amazon SageMaker notebook instance. The rule is NON_COMPLIANT if Amazon SageMaker notebook instances are internet-enabled.

**Identifier:** SAGEMAKER_NOTEBOOK_NO_DIRECT_INTERNET_ACCESS

**Trigger type:** Periodic

**AWS Regions:** All supported AWS regions except AWS GovCloud (US-Gov-East), China (Beijing), and China (Ningxia)

**Parameters:**
- None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**service-vpc-endpoint-enabled**

Checks whether Service Endpoint for the service provided in rule parameter is created for each Amazon VPC. The rule returns NON_COMPLIANT if an Amazon VPC doesn't have a VPC endpoint created for the service.

**Identifier:** SERVICE_VPC_ENDPOINT_ENABLED

**Trigger type:** Periodic

**AWS Region:** All supported AWS regions

**Parameters:**
- **serviceName**
  
  (Optional) The short name or suffix for the service. To get a list of available service names or valid suffix list, use DescribeVpcEndpointServices.
AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**shield-advanced-enabled-autorenew**

Checks whether AWS Shield Advanced is enabled in your AWS account and this subscription is set to automatically renew.

*Note*  
The API endpoint of AWS Shield Advanced is only available in us-east-1. This rule should only be scheduled to run in the us-east-1 Region.

**Identifier:** SHIELD_ADVANCED_ENABLED_AUTORENEW  
**Trigger type:** Periodic  
**Parameters:**  
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**shield-drt-access**

Verify that DDoS response team (DRT) can access AWS account. The rule is NON_COMPLIANT if AWS Shield Advanced is enabled but the role for DRT access is not configured.

**Identifier:** SHIELD_DRT_ACCESS  
**Trigger type:** Periodic  
**AWS Regions:** Only available in US East (N. Virginia) region  
**Parameters:**  
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

**vpc-default-security-group-closed**

Checks that the default security group of any Amazon Virtual Private Cloud (VPC) does not allow inbound or outbound traffic. The rule returns NOT_APPLICABLE if the security group is not default. The rule is NON_COMPLIANT if the default security group has one or more inbound or outbound traffic.

**Identifier:** VPC_DEFAULT_SECURITY_GROUP_CLOSED  
**Trigger type:** Configuration changes  
**Parameters:**  
None
None

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#).

**vpc-flow-logs-enabled**

Checks whether Amazon Virtual Private Cloud flow logs are found and enabled for Amazon VPC.

**Identifier:** VPC_FLOW_LOGS_ENABLED

**Trigger type:** Periodic

**Parameters:**

trafficType

The valid trafficType values are ACCEPT, REJECT, or ALL.

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#).

**vpc-sg-open-only-to-authorized-ports**

Checks whether the security group with 0.0.0.0/0 of any Amazon Virtual Private Cloud (Amazon VPC) allows only specific inbound TCP or UDP traffic. The rule and any security group with inbound 0.0.0.0/0 are NON_COMPLIANT if you do not provide any ports in the parameters.

**Identifier:** VPC_SG_OPEN_ONLY_TO_AUTHORIZED_PORTS

**Trigger type:** Configuration changes

**Parameters:**

authorizedTcpPorts (Optional)

Comma-separated list of TCP ports authorized to be open to 0.0.0.0/0. Ranges are defined by a dash; for example, "443,1020-1025".

authorizedUdpPorts (Optional)

Comma-separated list of UDP ports authorized to be open to 0.0.0.0/0. Ranges are defined by a dash; for example, "500,1020-1025".

**AWS CloudFormation template**

To create AWS Config managed rules with AWS CloudFormation templates, see [Creating AWS Config Managed Rules With AWS CloudFormation Templates](#).

**vpc-vpn-2-tunnels-up**

Checks that both AWS Virtual Private Network tunnels provided by AWS Site-to-Site VPN are in UP status. The rule returns NON_COMPLIANT if one or both tunnels are in DOWN status.
Identifier: VPC_VPN_2_TUNNELS_UP
Trigger type: Configuration changes
Parameters:
None

AWS CloudFormation template

To create AWS Config managed rules with AWS CloudFormation templates, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173).

Working with AWS Config Managed Rules

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

Setting Up and Activating 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.
5. On the Rules page, you can do the following:
   a. Type in the search field to filter results by rule name, description, and label. For example, type EC2 to return rules that evaluate EC2 resource types or type periodic to return rules that are triggered periodically.
   b. Choose the arrow icon to see the next page of rules. Recently added rules are marked as New.
6. Choose a rule that you want to create.
7. 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 types for your rule include Configuration changes, specify one of the following options for Scope of changes with which AWS Config invokes your Lambda function:
      - Resources – When a resource that matches the specified resource type, or the type plus identifier, is created, changed, or deleted.
      - Tags – When a resource with the specified tag is created, changed, or deleted.
      - All changes – When a resource recorded by AWS Config is created, changed, or deleted.
   c. If the trigger types for your rule include Periodic, specify the Frequency with which AWS Config invokes your Lambda function.
   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.

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** - AWS Config evaluated your resources against the rule. The rule did not apply to the AWS resources in its scope, the specified resources were deleted, or the evaluation results were deleted. To get evaluation results, update the rule, change its scope, or choose Re-evaluate.

  This message may also appear if the rule didn't report evaluation results.

- **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. To get evaluation results, edit the rule and change its scope, or add resources for AWS Config to record by using 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.

### Activating an AWS Managed Rule (AWS CLI)

Use the `put-config-rule` command.

### Activating an AWS Managed Rule (API)

Use the `PutConfigRule` action.

### Creating AWS Config Managed Rules With AWS CloudFormation Templates

For supported AWS Config managed rules, you can use the AWS CloudFormation templates to create the rule for your account or update an existing AWS CloudFormation stack. A stack is a collection of related resources that you provision and update as a single unit. When you launch a stack with a template, the AWS Config managed rule is created for you. The templates create only the rule, and don't create additional AWS resources.

**Note**

When AWS Config managed rules are updated, the templates are updated for the latest changes. To save a specific version of a template for a rule, download the template, and upload it to your S3 bucket.

For more information about working with AWS CloudFormation templates, see [Getting Started with AWS CloudFormation in the AWS CloudFormation User Guide](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/).  

**To launch an AWS CloudFormation stack for an AWS Config managed rule**

1. Go to the CloudFormation console and create a new stack.
2. For **Specify template**:
   - If you downloaded the template, choose **Upload a template file**, and then **Choose file** to upload the template.
   - If you have an Amazon S3 bucket, choose **Amazon S3 URL**, and enter the template URL `https://s3.amazonaws.com/aws-configservice-us-east-1/cloudformation-templates-for-managed-rules/the rule identifier.template`.
3. Choose **Next**.
4. For **Specify stack details**, type a stack name and enter parameter values for the AWS Config rule. For example, if you are using the `DESIRED_INSTANCE_TYPE` managed rule template, you can specify the instance type such as "m4.large".
5. Choose Next.
6. For Options, you can create tags or configure other advanced options. These are not required.
7. Choose Next.
8. For Review, verify that the template, parameters, and other options are correct.
9. Choose Create. The stack is created in a few minutes. You can view the created rule in the AWS Config console.

You can use the templates to create a single stack for AWS Config managed rules or update an existing stack in your account. If you delete a stack, the managed rules created from that stack are also deleted. For more information, see Working with Stacks in the AWS CloudFormation User Guide.

### AWS Config Custom Rules

You can develop custom rules and add them to AWS Config. You associate each custom rule with an AWS Lambda function, which contains the logic that evaluates whether your AWS resources comply with the rule.

You associate this function with your rule, and the rule invokes the function either in response to configuration changes or periodically. The function then evaluates whether your resources comply with your rule, and sends its evaluation results to AWS Config.

The exercise in Getting Started with Custom Rules for AWS Config (p. 174) 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 how AWS Lambda functions work and how to develop them, see the AWS Lambda Developer Guide.

**Topics**

- Getting Started with Custom Rules for AWS Config (p. 174)
- Developing a Custom Rule for AWS Config (p. 176)
- Example AWS Lambda Functions and Events for AWS Config Rules (p. 180)

### Getting Started with Custom Rules for AWS Config

This procedure guides you through the process of creating a custom 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.

**Topics**

- Creating an AWS Lambda Function for a Custom Config Rule (p. 175)
- Creating a Custom Rule (p. 175)
Creating an AWS Lambda Function for a Custom Config 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 triggers page, choose Next.
6. 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 Create new role from template(s).
   f. For Role name, type a name.
   g. For Policy templates, choose AWS Config Rules permission.
   h. On the Configure function page, choose Next.
   i. On the Review page, verify the details about your function, and choose Create function. The AWS Lambda console displays your function.
7. To verify that your function is set up correctly, test it with the following steps:
   a. Choose Actions, and then choose Configure test event.
   b. In the Input test event window, for Sample event template, choose AWS Config Change Triggered Rule.
   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.

Creating 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, 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** - AWS Config evaluated your resources against the rule. The rule did not apply to the AWS resources in its scope, the specified resources were deleted, or the evaluation results were deleted. To get evaluation results, update the rule, change its scope, or choose **Re-evaluate**.
      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. To get evaluation results, edit the rule and change its scope, or add resources for AWS Config to record by using the **Settings** page.
      Verify that AWS Config is recording EC2 instances.
   - **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** (p. 107).

**Developing a Custom Rule for AWS Config**

Complete the following procedure to create a custom rule. 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:

- **config-rule-change-triggered** – Triggered when your AWS resource configurations change.
- **config-rule-periodic** – Triggered at a frequency that you choose (for example, every 24 hours).

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 the AWS community. 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 triggers page, choose Next.
6. On the Configure function page, type a name and description.
7. For Runtime, choose the programming language in which your function is written.
8. For Code entry type, choose your preferred entry type. If you are using a blueprint, keep Edit code inline.
9. 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 based on the **config-rule-change-triggered** blueprint, or for functions triggered by configuration changes, the event data is the configuration item or an oversized configuration item object for the AWS resource that changed.

• For functions based on the **config-rule-periodic** blueprint, or for functions triggered at a frequency that you choose, the event data is a JSON object that includes information about when the evaluation was triggered.

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

• For example events that AWS Config publishes when it invokes your function, see [Example Events for AWS Config Rules](p. 185).

10. For **Handler**, specify the handler for your function. If you are using a blueprint, keep the default value.
11. For **Role**, choose **Create new role from template(s)**.
12. For **Role name**, type a name.
13. For **Policy templates**, choose **AWS Config Rules permission**.
14. On the **Configure function** page, choose **Next**.
15. 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**

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. For **Trigger type**, choose one or both of the following:
   
   • **Configuration changes** – AWS Config invokes your Lambda function when it detects a configuration change.
   
   • **Periodic** – AWS Config invokes your Lambda function at the frequency that you choose (for example, every 24 hours).
8. If the trigger types for your rule include **Configuration changes**, specify one of the following options for **Scope of changes** with which AWS Config invokes your Lambda function:
   
   • **Resources** – When a resource that matches the specified resource type, or the type plus identifier, is created, changed, or deleted.
   
   • **Tags** – When a resource with the specified tag is created, changed, or deleted.
   
   • **All changes** – When a resource recorded by AWS Config is created, changed, or deleted.
9. If the trigger types for your rule include **Periodic**, specify the **Frequency** with which AWS Config invokes your Lambda function.

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** - AWS Config evaluated your resources against the rule. The rule did not apply to the AWS resources in its scope, the specified resources were deleted, or the evaluation results were deleted. To get evaluation results, update the rule, change its scope, or choose **Re-evaluate**.

This message may also appear if the rule didn’t report evaluation results.

- **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.

**Note**
When you create a custom rule with the AWS Config console, the appropriate permissions are automatically created for you. If you create a custom rule with the AWS CLI, you need to give AWS Config permission to invoke your Lambda function, using the `aws lambda add-permission` command. For more information, see **Using Resource-Based Policies for AWS Lambda (Lambda Function Policies)** in the **AWS Lambda Developer Guide**.

### Evaluating Additional Resource Types

You can create custom rules to run evaluations for resource types not yet recorded by AWS Config. This is useful if you want to evaluate compliance for additional resource types, such as Amazon S3 Glacier vaults or Amazon SNS topics, that AWS Config doesn't currently record. For a list of additional resource types that you can evaluate with custom rules, see **AWS Resource Types Reference**.

**Note**
The list in the AWS CloudFormation User Guide may contain recently added resource types that are not yet available for creating custom rules in AWS Config. AWS Config adds resource types support at regular intervals.

**Example**

1. You want to evaluate Amazon S3 Glacier vaults in your account. Amazon S3 Glacier vault resources are currently not recorded by AWS Config.

2. You create an AWS Lambda function that evaluates whether your Amazon S3 Glacier vaults comply with your account requirements.

3. You create a custom rule named **evaluate-glacier-vaults** and then assign your AWS Lambda function to the rule.

4. AWS Config invokes your Lambda function and then evaluates the Amazon S3 Glacier vaults against your rule.

5. AWS Config returns the evaluations and you can view the compliance results for your rule.
Note
You can view the configuration details in the AWS Config timeline and look up resources in the AWS Config console for resources that AWS Config supports. If you configured AWS Config to record all resource types, newly supported resources will automatically be recorded. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

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) (p. 180)
- Example Events for AWS Config Rules (p. 185)

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 uses data that it receives from the event and that it retrieves from the AWS Config API to evaluate the compliance of 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.

Contents
- Example Function for Evaluations Triggered by Configuration Changes (p. 180)
- Example Function for Periodic Evaluations (p. 183)

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 and OversizedConfigurationItemChangeNotification. These settings enable your rule to be triggered whenever AWS Config generates a configuration item or an oversized configuration item as a result of a resource change.

This example evaluates your resources and checks whether the instances match the resource type, AWS::EC2::Instance. The rule is triggered when AWS Config generates a configuration item or an oversized configuration item notification.

```javascript
'use strict';
```
const aws = require('aws-sdk');
const config = new aws.ConfigService();

// Helper function used to validate input
function checkDefined(reference, referenceName) {
  if (!reference) {
    throw new Error(`Error: ${referenceName} is not defined`);
  }
  return reference;
}

// Check whether the message type is OversizedConfigurationItemChangeNotification,
function isOverSizedChangeNotification(messageType) {
  checkDefined(messageType, 'messageType');
  return messageType === 'OversizedConfigurationItemChangeNotification';
}

// Get the configurationItem for the resource using the getResourceConfigHistory API.
function getConfiguration(resourceType, resourceId, configurationCaptureTime, callback) {
  config.getResourceConfigHistory({ resourceType, resourceId, laterTime: new Date(configurationCaptureTime), limit: 1 }, (err, data) => {
    if (err) {
      callback(err, null);
    } else {
      const configurationItem = data.configurationItems[0];
      callback(null, configurationItem);
    }
  });
}

// Convert the oversized configuration item from the API model to the original invocation model.
function convertApiConfiguration(apiConfiguration) {
  apiConfiguration.awsAccountId = apiConfiguration.accountId;
  apiConfiguration.ARN = apiConfiguration.arn;
  apiConfiguration.configurationStateMd5Hash = apiConfiguration.configurationItemMD5Hash;
  apiConfiguration.configurationItemVersion = apiConfiguration.version;
  apiConfiguration.configuration = JSON.parse(apiConfiguration.configuration);
  if ({}.hasOwnProperty.call(apiConfiguration, 'relationships')) {
    for (let i = 0; i < apiConfiguration.relationships.length; i++) {
      apiConfiguration.relationships[i].name = apiConfiguration.relationships[i].relationshipName;
    }
  }
  return apiConfiguration;
}

// Based on the message type, get the configuration item either from the configurationItem object in the invoking event or with the getResourceConfigHistory API in the getConfiguration function.
function getConfigurationItem(invokingEvent, callback) {
  checkDefined(invokingEvent, 'invokingEvent');
  if (isOverSizedChangeNotification(invokingEvent.messageType)) {
    const configurationItemSummary = checkDefined(invokingEvent.configurationItemSummary, 'configurationItemSummary');
    getConfiguration(configurationItemSummary.resourceType, configurationItemSummary.resourceId, configurationItemSummary.configurationItemCaptureTime, (err, apiConfigurationItem) => {
      if (err) {
        callback(err, null);
      } else {
        const configurationItem = convertApiConfiguration(apiConfigurationItem);
        callback(null, configurationItem);
      }
    });
  } else { // message type is not OversizedConfigurationItemChangeNotification
    getConfiguration(invokingEvent.resourceType, invokingEvent.resourceId, invokingEvent.configCaptureTime, (err, configurationItem) => {
      if (err) {
        callback(err, null);
      } else {
        const configurationItem = convertApiConfiguration(configurationItem);
        callback(null, configurationItem);
      }
    });
  }
}
} else {
    checkDefined(invokingEvent.configurationItem, 'configurationItem');
    callback(null, invokingEvent.configurationItem);
}

// Check whether the resource has been deleted. If the resource was deleted, then the
evaluation returns not applicable.
function isApplicable(configurationItem, event) {
    checkDefined(configurationItem, 'configurationItem');
    checkDefined(event, 'event');
    const status = configurationItem.configurationItemStatus;
    const eventLeftScope = event.eventLeftScope;
    return (status === 'OK' || status === 'ResourceDiscovered') && eventLeftScope ===
false;
}

// In this example, the resource is compliant if it is an instance and its type matches the
type specified as the desired type.
// If the resource is not an instance, then this resource is not applicable.
function evaluateChangeNotificationCompliance(configurationItem, ruleParameters) {
    checkDefined(configurationItem, 'configurationItem');
    checkDefined(configurationItem.configuration, 'configurationItem.configuration');
    checkDefined(ruleParameters, 'ruleParameters');

    if (configurationItem.resourceType !== 'AWS::EC2::Instance') {
        return 'NOT_APPLICABLE';
    } else if (ruleParameters.desiredInstanceType ===
configurationItem.configuration.instanceType) {
        return 'COMPLIANT';
    }
    return 'NON_COMPLIANT';
}

// Receives the event and context from AWS Lambda.
exports.handler = (event, context, callback) => {
    checkDefined(event, 'event');
    const invokingEvent = JSON.parse(event.invokingEvent);
    const ruleParameters = JSON.parse(event.ruleParameters);
    getConfigurationItem(invokingEvent, (err, configurationItem) => {
        if (err) {
            callback(err);
        }
        let compliance = 'NOT_APPLICABLE';
        const putEvaluationsRequest = {};
        if (isApplicable(configurationItem, event)) {
            // Invoke the compliance checking function.
            compliance = evaluateChangeNotificationCompliance(configurationItem,
ruleParameters);
        }
        // Initializes the request that contains the evaluation results.
        putEvaluationsRequest.Evaluations = [
            {
                ComplianceResourceType: configurationItem.resourceType,
                ComplianceResourceId: configurationItem.resourceId,
                ComplianceType: compliance,
                OrderingTimestamp: configurationItem.configurationItemCaptureTime,
            },
        ];
        putEvaluationsRequest.ResultToken = event.resultToken;

        // Sends the evaluation results to AWS Config.
        config.putEvaluations(putEvaluationsRequest, (error, data) => {
            if (error) {
                callback(error, null);
            } else if (data.FailedEvaluations.length > 0) {
                checkDefined(data.FailedEvaluations[0].RuleArn, 'RuleArn');
                callback(null, data.FailedEvaluations[0].ResponseMetadata />
            } else {
                callback(null, null);
            }
        });
    });
}
// Ends the function if evaluation results are not successfully reported to AWS Config.
  callback(JSON.stringify(data), null);
} else {
  callback(null, data);
}
});
});

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 function checks whether the messageType for the event is a configuration item or an oversized configuration item, and then returns the configuration item.

3. The handler calls the isApplicable function to determine whether the resource was deleted.

4. The handler calls the evaluateChangeNotificationCompliance function and passes the configurationItem and ruleParameters objects that AWS Config published in the event.

The function first evaluates whether the resource is an EC2 instance. If the resource is not an EC2 instance, the function returns a compliance value of NOT_APPLICABLE.

The function then evaluates whether the instanceType attribute in the configuration item is equal to the desiredInstanceType parameter value. If the values are equal, the function returns COMPLIANT. If the values are not equal, the function returns NON_COMPLIANT.

5. 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 resource type, and the ID of the resource that was evaluated. The putEvaluationsRequest object also includes the result token from the event, which identifies the rule and the event for AWS Config.

6. The handler sends the evaluation results to AWS Config by passing the object to the putEvaluations method of the config client.

Example Function for Periodic Evaluations

AWS Config will invoke a function like the following example for periodic evaluations. Periodic evaluations occur at the frequency that you specify when you define the rule in AWS Config.

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 ScheduledNotification.

This example checks whether the total number of a specified resource exceeds a specified maximum.

```javascript
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 = {
  COMPLIANT : 'COMPLIANT',
  NON_COMPLIANT : 'NON_COMPLIANT',
  NOT_APPLICABLE : 'NOT_APPLICABLE'
};
```
// Receives the event and context from AWS Lambda.
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),
        noOfResources = 0;

    if (isScheduledNotification(invokingEvent)) {
        countResourceTypes(ruleParameters.applicableResourceType, "", noOfResources,
            function(err, count) {
                if (err === null) {
                    var putEvaluationsRequest;
                    // 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(ruleParameters.maxCount,
                                count),
                            OrderingTimestamp : new Date()
                        } ],
                        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 evaluation results are not
                                successfully reported
                                callback(JSON.stringify(data));
                            }
                            else {
                                callback(null, data);
                            }
                        }
                    });
                } else {
                    callback(err, null);
                }
            });
        } else {
            console.log("Invoked for a notification other than Scheduled Notification... Ignoring.");
        }
    } else {
        console.log("Invoked for a notification other than Scheduled Notification... Ignoring.");
    }

    // Checks whether the invoking event is ScheduledNotification.
    function isScheduledNotification(invokingEvent) {
        return (invokingEvent.messageType === 'ScheduledNotification');
    }

    // Checks whether the compliance conditions for the rule are violated.
    function evaluateCompliance(maxCount, actualCount) {
        if (actualCount > maxCount) {
            return COMPLIANCE_STATES.NON_COMPLIANT;
        } else {
            return COMPLIANCE_STATES.COMPLIANT;
        }
    }

    // Counts the applicable resources that belong to the AWS account.
    function countResourceTypes(applicableResourceType, nextToken, count, callback) {

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. To count the resources of the specified type, the handler calls the `countResourceTypes` function, and it passes the applicableResourceType parameter that it received from the event. The `countResourceTypes` function calls the `listDiscoveredResources` method of the config client, which returns a list of identifiers for the applicable resources. The function uses the length of this list to determine the number of applicable resources, and it returns this count to the handler.

3. 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.

4. Within the `putEvaluationsRequest` object, the handler calls the `evaluateCompliance` function. This function tests whether the number of applicable resources exceeds the maximum assigned to the maxCount parameter, which was provided by the event. 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.

5. The handler sends the evaluation results to AWS Config by passing the object to the `putEvaluations` method of the config client.

Example Events for AWS Config Rules

When the trigger for a 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

AWS Config publishes an event when it detects a configuration change for a resource that is within a rule's scope. The following example event shows that the rule was triggered by a configuration change for an EC2 instance.

```json
{
    "invokingEvent": "{"configurationItem":{"configurationItemCaptureTime": "2016-02-17T01:36:34.043Z", "awsAccountId": "123456789012", "configurationItemStatus": "COMPLIANT"}
```
Example Event for Evaluations Triggered by Oversized Configuration Changes

Some resource changes generate oversized configuration items. The following example event shows that the rule was triggered by an oversized configuration change for an EC2 instance.

```
{
  "invokingEvent": "{\"configurationItemSummary\": {\"changeType\": \"UPDATE \n\", \"configurationItemVersion\": \"1.2\", \"configurationItemCaptureTime\": \"2016-10-06T16:46:16.261Z\", \"configurationItemStatus\": \"OK\", \"resourceType\": \"AWS::EC2::Instance\", \"resourceId\": \"i-00000000\", \"arn\": \"arn:aws:ec2:us-east-2:123456789012:instance/i-00000000\", \"awsRegion\": \"us-east-2\", \"availabilityZone\": \"us-east-2a\", \"resourceType\": \"AWS::EC2::EIP\", \"name\": \"Is attached to ElasticIp\"}, \"configuration\": {\"foo\": \"bar\"}, \"messageType\": \"ConfigurationItemChangeNotification\"},
  "ruleParameters": "{\"myParameterKey\": \"myParameterValue\"},
  "resultToken": \"myResultToken\",
  "eventLeftScope": false,
  "executionRoleArn": \"arn:aws:iam::123456789012:role/config-role\",
  "configRuleArn": \"arn:aws:config:us-east-2:123456789012:config-rule/config-rule-0123456\",
  "configRuleName": \"change-triggered-config-rule\",
  "configRuleId": \"config-rule-0123456\",
  "accountId": \"123456789012\",
  "version": \"1.0\"}
```

Example Event for Evaluations Triggered by Periodic Frequency

AWS Config publishes an event when it evaluates your resources at a frequency that you specify (such as every 24 hours). The following example event shows that the rule was triggered by a periodic frequency.

```
{
  "invokingEvent": "{\"awsAccountId\": \"123456789012\", \"notificationCreationTime\": \"2016-07-13T21:50:00.373Z\", \"messageType\": \"ScheduledNotification\", \"recordVersion\": \"1.0\"},
  "ruleParameters": "{\"myParameterKey\": \"myParameterValue\"},
  "resultToken": \"myResultToken\",
  "eventLeftScope": false,
  "executionRoleArn": \"arn:aws:iam::123456789012:role/config-role\",
  "configRuleArn": \"arn:aws:config:us-east-2:123456789012:config-rule/config-rule-0123456\",
  "configRuleName": \"periodic-config-rule\",
  "configRuleId": \"config-rule-6543210\",
  "accountId": \"123456789012\",
  "version": \"1.0\"
```
Event Attributes

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

invokingEvent

The event that triggers the evaluation for a rule. If the event is published in response to a resource configuration change, the value for this attribute is a string that contains a JSON configurationItem or a configurationItemSummary (for oversized configuration items). 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 Viewing Configuration History (p. 43).

If the event is published for a periodic evaluation, the value is a string that contains a JSON object. The object includes information about the evaluation that was triggered.

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:

```javascript
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:

```javascript
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.
Managing your AWS Config Rules

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

Contents
- Add, View, Update and Delete Rules (Console) (p. 188)
- View, Update, and Delete Rules (AWS CLI) (p. 190)
- View, Update, and Delete Rules (API) (p. 191)

Add, View, Update and Delete Rules (Console)

On the Rules page, you can view the rules for the region in your account. You can also see the evaluation status for each rule.

To view your 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, 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 Rules. The Rules page shows your rules and the compliance status for each.

A. Choose Add rule to get started with creating a rule.
B. Choose a rule name to see its settings.
C. See the compliance status of the rule when it evaluates your resources.
D. Choose the Edit rule icon (祝愿) to edit the rule.
E. Choose the refresh (🔄) icon to reload compliance results.

To update a rule

1. Choose the Edit rule icon (📝) for the rule that you want to update.
2. Modify the settings on the Config rule page to change your rule as needed.
3. Choose Save.

To delete a rule

1. Choose the Edit rule icon (📝) for the rule that you want to delete.
2. On the Configure rule page, choose Delete rule.
3. When prompted, choose Delete.

To add a rule

If you choose Add rule, you can see the available AWS managed rules on the Add rule page. You can also create your own custom rule.

1. If you want to create your own rule, choose Add custom rule and follow the procedure in Developing a Custom Rule for AWS Config (p. 176).
2. To add a managed rule, choose a rule on the page and follow the procedure in Working with AWS Config Managed Rules (p. 172).

On the Add rule page, you can do the following:
A. Choose **Add custom rule** to create your own rule.

B. Type in the search field to filter results by rule name, description, or label. For example, type **EC2** to return rules that evaluate EC2 resource types or type **periodic** to return rules with periodic triggers. Type **“new”** to search for newly added rules. For more information about trigger types, see **Specifying Triggers for AWS Config Rules** (p. 110).

C. Choose the arrow icon to see the next page of rules.

D. Recently added rules are marked as **New**.

E. See the labels to identify the resource type that the rule evaluates and if the rule has a periodic trigger.

### View, Update, and Delete Rules (AWS CLI)

**To view your rules**

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

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

  AWS Config returns the details for all of your rules.

**To update a rule**

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:

  ```bash
  $ aws configservice put-config-rule --generate-cli-skeleton > putConfigRule.json
  ```

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

   - Include at least one of the following parameters to identify the rule: `ConfigRuleName`, `ConfigRuleArn`, or `ConfigRuleId`.
   - If you are updating a custom rule, you must include the `Source` object and its parameters.

3. Fill in the values for the parameters that remain. To reference the details of your rule, use the `describe-config-rules` command.

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

   ```json
   {
   "ConfigRule": {
   "ConfigRuleName": "ConfigRuleName",
   "Scope": {
   "ComplianceResourceTypes": [ "AWS::EC2::Instance",
   "AWS::EC2::Volume",
   "AWS::EC2::VPC"

   },
   "Source": {
   "Owner": "CUSTOM_LAMBDA",
   "SourceDetails": [ ...
   ```
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
```

To delete a rule

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

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

View, Update, and Delete Rules (API)

To view your rules

Use the `DescribeConfigRules` action.
To update or add a rule

Use the `PutConfigRule` action.

To delete a rule

Use the `DeleteConfigRule` action.

Note

If a rule is creating invalid evaluation results, you might want to delete these results before you fix the rule and run a new evaluation. For more information, see Deleting Evaluation Results (p. 193).

Evaluating Your Resources

When you create custom rules or use managed rules, AWS Config evaluates your resources against those rules. You can run on-demand evaluations for resources against your rules. For example, this is helpful when you create a custom rule and want to verify that AWS Config is correctly evaluating your resources or to identify if there is an issue with the evaluation logic of your AWS Lambda function.

Example

1. You create a custom rule that evaluates whether your IAM users have active access keys.
2. AWS Config evaluates the resources against your custom rule.
3. An IAM user who doesn't have an active access key exists in your account. Your rule doesn't correctly flag this resource as noncompliant.
4. You fix the rule and start the evaluation again.
5. Because you fixed your rule, the rule correctly evaluates your resources, and flags the IAM user resource as noncompliant.

Evaluating your Resources (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.
4. Choose a rule from the list.
5. In the Re-evaluate rule section, choose Re-evaluate.
6. AWS Config starts evaluating the resources against your rule.

Note

You can re-evaluate a rule once per minute. You must wait for AWS Config to complete the evaluation for your rule before you start another evaluation. You can't run an evaluation if at the same time the rule is being updated or if the rule is being deleted.

Evaluating your Resources (CLI)

- Use the `start-config-rules-evaluation` command.
Evaluating your Resources (API)

Use the `StartConfigRulesEvaluation` action.

Deleting Evaluation Results

After AWS Config evaluates your rule, you can see the evaluation results on the Rules page or the Rules details page for the rule. If the evaluation results are incorrect or if you want to evaluate again, you can delete the current evaluation results for the rule. For example, if your rule was incorrectly evaluating your resources or you recently deleted resources from your account, you can delete the evaluation results and then run a new evaluation.

Deleting Evaluating Results (Console)

1. Sign in to the AWS Management Console and open the AWS Config console at [https://console.aws.amazon.com/config/](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.
4. Choose a rule from the list.
5. In the Delete evaluation results section, choose Delete results. AWS Config deletes the evaluation results for this rule.
6. When prompted, choose Delete. Deleted evaluations can't be retrieved.
7. After the evaluation results are deleted, you can manually start a new evaluation.

Deleting Evaluating Results (CLI)

- Use the `delete-evaluation-results` command:

  ```bash
  $ aws configservice delete-evaluation-results --config-rule-name ConfigRuleName
  ```

  AWS Config deletes the evaluation results for the rule.

Deleting Evaluating Results (API)

Use the `DeleteEvaluationResults` action.
Enabling AWS Config Rules Across all Accounts in Your Organization

AWS Config allows you to manage AWS Config rules across all AWS accounts within an organization. You can:

- Centrally create, update, and delete AWS Config rules across all accounts in your organization.
- Deploy a common set of AWS Config rules across all accounts and specify accounts where AWS Config rules should not be created.
- Use the APIs from the master account in AWS Organizations to enforce governance by ensuring that the underlying AWS Config rules are not modifiable by your organization’s member accounts.

Ensure AWS Config recording is on before you use the following APIs to manage AWS Config rules across all AWS accounts within an organization:

- PutOrganizationConfigRule
- DescribeOrganizationConfigRules
- GetOrganizationConfigRuleDetailedStatus
- DescribeOrganizationConfigRuleStatuses
- DeleteOrganizationConfigRule

Remediating Noncompliant AWS Resources by AWS Config Rules

AWS Config allows you to remediate noncompliant resources that are evaluated by AWS Config Rules. AWS Config applies remediation using AWS Systems Manager Automation documents. These documents define the actions to be performed on noncompliant AWS resources evaluated by AWS Config Rules. You can associate SSM documents with using the AWS Management Console or by using APIs.

AWS Config provides a set of managed automation documents with remediation actions. You can also create and associate custom automation documents with AWS Config rules.

To apply remediation on noncompliant resources, you can either choose the remediation action you want to associate from a prepopulated list or create your own custom remediation actions using SSM documents. AWS Config provides a recommended list of remediation action in the AWS Management Console.

In the AWS Management Console, you can either choose to manually or automatically remediate noncompliant resources by associating remediation actions with AWS Config rules. With all remediation actions, you can either choose manual or automatic remediation.

Topics

- Prerequisite (p. 195)
- Setting Up Manual Remediation (Console) (p. 195)
- Setting Up Auto Remediation (Console) (p. 195)
- Setting Up and Applying a Remediation Action Remediation Using Rules and Resources (Console) (p. 196)
Prerequisite

Before you begin to apply remediation on noncompliant resources, you must select a rule and set up remediation (manual or auto) for the rule.

Setting Up Manual Remediation (Console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Rules on the left and then on the Rules page, choose Add Rule to add new rules to the rule list.
   For existing rules, select the noncompliant rule from the rule list and choose Edit.
3. On the Edit name of the rule page, in the Choose remediation action section, choose the appropriate remediation action from the recommended list.
   Depending on the selected remediation action, you see specific parameters or no parameters.
4. (Optional): If you want to pass the resource ID of noncompliant resources to the remediation action, choose Resource ID parameter. If selected, at runtime that parameter is substituted with the ID of the resource to be remediated.
   Each parameter has either a static value or a dynamic value. If you do not choose a specific resource ID parameter from the drop-down list, you can enter values for each key. If you choose a resource ID parameter from the drop-down list, you can enter values for all the other keys except the selected resource ID parameter.
5. Choose Save. The Rules page is displayed.

Setting Up Auto Remediation (Console)

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Rules on the left and then on the Rules page, choose Add Rule to add new rules to the rule list.
   For existing rules, select the noncompliant rule from the rule list and choose Edit.
3. On the Edit name of the rule page, in the Choose remediation action section, choose the appropriate remediation action from the recommended list.
   Depending on the selected remediation action, you see specific parameters or no parameters.
4. Choose Auto remediation to automatically remediate noncompliant resources.
   If a resource is still non-compliant after auto remediation, you can set the rule to try auto remediation again. Enter the desired retries and seconds.
   **Note**
   There are costs associated with running a remediation script multiple times.
5. (Optional): If you want to pass the resource ID of noncompliant resources to the remediation action, choose Resource ID parameter. If selected, at runtime that parameter is substituted with the ID of the resource to be remediated.
Each parameter has either a static value or a dynamic value. If you do not choose a specific resource ID parameter from the drop-down list, you can enter values for each key. If you choose a resource ID parameter from the drop-down list, you can enter values for all the other keys except the selected resource ID parameter.

6. Choose Save. The Rules page is displayed.

Setting Up and Applying a Remediation Action
Remediation Using Rules and Resources (Console)

You can also set up remediation from the Rules and the Resources pages. The following procedures provide more details about each workflow.

Using Rules

You can apply a remediation action to a noncompliant rule from Rules on the left.

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Rules on the left and then on the Rules page, choose Add Rule to add new rules to the rule list.

For existing rules, select the noncompliant rule from the rule list and choose Edit.

3. Choose Manage remediation and on the Manage remediation: name of the rule page, select the appropriate remediation action from the recommended list. The remediation actions are related to AWS Systems Manager automation documents.

Depending on the selected remediation action, you will see either specific parameters or no parameters.

4. If you want to pass the resource ID of noncompliant resources to the remediation action, choose Resource ID parameter. If selected, at runtime that parameter is substituted with the ID of the resource to be remediated.

Each parameter has either a static value or a dynamic value. If you do not choose a specific resource ID parameter from the drop-down list, you can enter values for each key. If you choose a resource ID parameter from the drop-down list, you can enter values for all the other keys except the selected resource ID parameter.

5. Choose Save.
6. In the Choose resources in scope section, choose all the noncompliant resources. The resources in scope include those resources where this rule is applied and their compliance status.

For more information about a resource, choose Resource actions and either choose View details, Configuration timeline, or Compliance timeline.

7. Choose Remediate.

If the resources are remediated, the resource compliance status is compliant. To view the compliant resources, select Compliant from the compliance status list.

If the resources are not remediated, the action status column displays Action execution failed (details). Choose (details) to view the main action steps invoked during the execution of the remediation action and the status of each action step.

Note
For troubleshooting failed remediation actions, you can run the AWS Command Line Interface (AWS CLI) command describe-remediation-execution-status to get
detailed view of a Remediation Execution for a set of resources. The details include state, timestamps for remediation execution steps, and any error messages for the failed steps.

Using Resources

You can also apply a remediation action to a noncompliant rule from Resources on the left.

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Resources on the left and then on the Resource inventory page, Resources is selected by default.
3. Select one or more resources from the resource type and choose Look up.
4. Choose the name of the appropriate noncompliant resource from the resource table.
5. On the Resource details: name of the resource page, choose one or more noncompliant rules from the rules in scope.
6. Choose Rule actions and select Manage remediation.
7. On the Manage remediation: name of the rule page, select the appropriate remediation action.
   Depending on the selected remediation action, you will see either specific parameters or no parameters.
8. If you want to pass the resource ID of noncompliant resources to the remediation action, choose Resource ID parameter. If selected, at runtime that parameter is substituted with the ID of the resource to be remediated.
   Each parameter has either a static value or a dynamic value. If you do not choose a specific resource ID parameter from the drop-down list, you can enter values for each key. If you choose a resource ID parameter from the drop-down list, you can enter values for all the other keys except the selected resource ID parameter.
9. Choose Save.
10. In the Choose resources in scope section, choose all the noncompliant resources. The resources in scope include those resources where this rule is applied and their compliance status.
   For more information about a resource, choose Resource actions and either choose View details, Configuration timeline, or Compliance timeline.
11. Choose Remediate.
   If the resources are remediated, the resource compliance status is compliant. To view the compliant resources, select Compliant from the compliance status list.
   If the resources are not remediated, the action status column displays Action execution failed (details). Choose (details) to view the main action steps invoked during the execution of the remediation action and the status of each action step.

   Note
   For troubleshooting failed remediation actions, you can run the AWS Command Line Interface (AWS CLI) command describe-remediation-execution-status to get detailed view of a Remediation Execution for a set of resources. The details include state, timestamps for remediation execution steps, and any error messages for the failed steps.

Delete Remediation Action (Console)

To delete a rule first you must delete remediation action associated with that rule.
1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Rules on the left and then on the Rules page, select the rule from the rule list and choose Edit.
3. On the rule page, choose Edit again.
4. On the Edit name of the rule page, in the Choose remediation action section, choose Delete remediation action and confirm your delete action.

**Note**
If remediation is in progress, the remediation action is not deleted. If you choose Delete remediation action, you cannot retrieve the remediation action. When you delete a remediation action, AWS Config does not delete a rule.

If the remediation action is deleted, the Resource ID parameter is empty and displays N/A. On the Rules page, the remediation action column displays Not set for the corresponding rule.

### Managing Remediation (API)

#### Manual Remediation

Use the following AWS Config API actions to manage remediation:

- DeleteRemediationConfiguration
- DescribeRemediationConfigurations
- DescribeRemediationExecutionStatus
- PutRemediationConfigurations
- StartRemediationExecution

#### Auto Remediation

Use the following AWS Config API actions to manage auto remediation:

- PutRemediationExceptions
- DescribeRemediationExceptions
- DeleteRemediationExceptions

### Tagging Your AWS Config Resources

A tag is a label that you assign to an AWS resource. Each tag consists of a *key* and an optional *value*, both of which you define. Tags make it easier to manage, search for, and filter resources.

Tags enable you to categorize your AWS resources in different ways, for example, by purpose, owner, or environment. This is useful when you have many resources of the same type—you can quickly identify a specific resource based on the tags you've assigned to it. You can assign one or more tags to your AWS resources. Each tag has an associated value.

We recommend that you devise a set of tag keys that meets your needs for each resource type. Using a consistent set of tag keys makes it easier for you to manage your AWS resources. You can search and filter the resources based on the tags you add.

Tags are interpreted strictly as a string of characters and are not automatically assigned to your resources. You can edit tag keys and values, and you can remove tags from a resource at any time. You
can set the value of a tag to an empty string, but you can't set the value of a tag to null. If you add a tag that has the same key as an existing tag on that resource, the new value overwrites the old value. If you delete a resource, any tags for the resource are also deleted.

You can work with tags using the AWS Command Line Interface (AWS CLI) and the AWS Config API reference.

**Restrictions Related to Tagging**

The following basic restrictions apply to tags.

<table>
<thead>
<tr>
<th>Restriction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of tags per resource</td>
<td>50</td>
</tr>
<tr>
<td>Maximum key length</td>
<td>128 Unicode characters in UTF-8</td>
</tr>
<tr>
<td>Maximum value length</td>
<td>256 Unicode characters in UTF-8</td>
</tr>
<tr>
<td>Prefix restriction</td>
<td>Do not use the <code>aws:</code> prefix in your tag names or values because it is reserved for AWS use. You can't edit or delete tag names or values with this prefix. Tags with this prefix do not count against your tags per resource limit.</td>
</tr>
<tr>
<td>Character restrictions</td>
<td>Tags may only contain Unicode letters, digits, whitespace, or these symbols: _ . : / = + - @</td>
</tr>
</tbody>
</table>

**Managing Tags with AWS Config API Actions**

Tag based access controls are available for three resources `ConfigurationAggregator`, `AggregationAuthorization`, and `ConfigRule`. Use the following to add, update, list, and delete the tags for your resources.

- `ListTagsForResource`
- `TagResource`
- `UntagResource`
Conformance Packs

A conformance pack is a collection of AWS Config rules and remediation actions that can be easily deployed as a single entity in an account and a Region or across an organization in AWS Organizations.

Conformance packs are created by authoring a YAML template that contains the list of AWS Config managed or custom rules and remediation actions. You can deploy the template by using the AWS Config console or the AWS CLI. To quickly get started and to evaluate your AWS environment, use one of the sample conformance pack templates.

Note
Conformance packs are only available in the redesigned AWS Config console.

Topics
- Prerequisites (p. 200)
- Region Support (p. 204)
- Conformance Pack Sample Templates (p. 205)
- Deploying a Conformance Pack Using the AWS Config Console (p. 220)
- Deploying a Conformance Pack Using the AWS Command Line Interface (p. 221)
- Managing Conformance Packs (API) (p. 224)
- Managing Conformance Packs Across all Accounts in Your Organization (p. 225)
- Troubleshooting (p. 226)

Prerequisites

Before you deploy your conformance pack, turn on AWS Config recording, create a service-linked role, and provide resource-level permissions for the Amazon S3 bucket to which AWS Config delivers evaluation results.

Step 1: Start AWS Config Recording

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Choose Settings in the navigation pane.
3. To start recording, under Recording is off, choose Turn on. When prompted, choose Continue.

Step 2: Create a Service-Linked Role

Create a service-linked role using the following command.

```
aws iam create-service-linked-role --aws-service-name config-conforms.amazonaws.com --description "my service linked role for config-conforms"
```

There is now a new role, AWSServiceRoleForConfigConforms, in your account.

Step 3: Provide Resource-Level Permissions

Follow these steps to provide resource-level permissions on an Amazon S3 bucket to enable AWS Config to access that S3 bucket. The access policy allows AWS Config to deliver evaluation results to the Amazon S3 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 evaluation results, and then choose Properties.
4. Choose Permissions.
5. Choose Edit Bucket Policy.
6. Copy the appropriate bucket policy into the Bucket Policy Editor window.

**Same Account Same Bucket Policy:** Use the following bucket policy if your template or delivery input bucket is in the same AWS account in which the conformance pack is created. Delivery bucket is a bucket where you want AWS Config to deliver conformance pack artifacts such as reports.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSConfigConformsBucketPermissionsCheck",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::AccountID:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
        ],
        "Action": "s3:GetBucketAcl",
        "Resource": "arn:aws:s3:::delivery-bucket-name"
      }
    },
    {
      "Sid": "AWSConfigConformsBucketDelivery",
      "Effect": "Allow",
      "Principal": { "AWS": [
          "arn:aws:iam::AccountID:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
        ],
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::delivery-bucket-name/[optional] prefix/AWSLogs/AccidentID/Config/**",
        "Condition": { "StringEquals": { "s3:x-amz-acl": "bucket-owner-full-control" } }
      }
    },
    {
      "Sid": "AWSConfigConformsBucketReadAccess",
      "Effect": "Allow",
      "Principal": { "AWS": [
          "arn:aws:iam::AccountID:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
        ],
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::delivery-bucket-name/[optional] prefix/AWSLogs/AccidentID/Config/**"
      }
    }
  ]
}
```
Cross Account Bucket Policy: Use the following bucket policy for the delivery bucket if it present in a different AWS account than the account in which the conformance pack is created.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSConfigConformsBucketPermissionsCheck",
            "Effect": "Allow",
            "Principal": {
                "AWS": [arn:aws:iam::SourceAccountId:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms",
                "PutConformancePack API caller user principal like arn:aws:iam::SourceAccountId:user/username"
            ],
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::awsconfigconforms/suffix in bucket name"
        },
        {
            "Sid": "AWSConfigConformsBucketDelivery",
            "Effect": "Allow",
            "Principal": {
                "AWS": [arn:aws:iam::SourceAccountId:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
            ],
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsconfigconforms/suffix in bucket name/[optional] prefix/AWSLogs/AccountID/Config/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control"
                }
            }
        },
        {
            "Sid": "AWSConfigConformsBucketReadAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": [arn:aws:iam::SourceAccountId:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
            ],
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::awsconfigconforms/suffix in bucket name/[optional] prefix/AWSLogs/AccountID/Config/*"
        }
    ]
}
```

Organizations Bucket Policy: Use the following bucket policy for the delivery bucket of an organization conformance pack.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetObject",
            "Effect": "Allow",
            "Principal": {
                "AWS": [arn:aws:iam::SourceAccountId:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
            ],
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::awsconfigconforms/suffix in bucket name/[optional] prefix/AWSLogs/AccountID/Config/*"
        }
    ]
}
```
Prerequisites for Using a Conformance Pack With Remediation

Before deploying conformance packs using sample templates with remediation, you must create appropriate resources such as automation assume role and other AWS resources based on your remediation target.

If you have an existing automation role that you are using for remediation using SSM documents, you can directly provide the ARN of that role. If you have any resources you can provide those in the template.

AWS Config does not support AWS CloudFormation intrinsic functions for the automation execution role. You must provide the exact ARN of the role as a string.

For more information about how to pass the exact ARN, see Conformance Pack Sample Templates (p. 205). While using example templates, update your Account ID and master Account ID for organization.

Prerequisites for Using a Conformance Pack With One or More AWS Config Rules

Before deploying a conformance pack with one or more custom AWS Config rules, create appropriate resources such as AWS Lambda function and the corresponding execution role.

If you have an existing custom AWS Config rule, you can directly provide the ARN of AWS Lambda function to create another instance of that custom rule as part of the pack.

If you do not have an existing custom AWS Config rule, you can create a AWS Lambda function and use the ARN of the Lambda function. For more information, see AWS Config Custom Rules (p. 174).
If your AWS Lambda function is present in a different AWS account, you can create AWS Config rules with appropriate cross-account AWS Lambda function authorization. For more information, see How to Centrally Manage AWS Config Rules across Multiple AWS Accounts blog post.

## Prerequisites for Organization Conformance Packs

Ensure the name of the delivery S3 bucket must start with prefix `awsconfigconforms`.

Specify an automation execution role ARN for that remediation in the template if the input template has an autoremediation configuration. Ensure a role with the specified name exists in all the accounts (master and member) of an organization. You must create this role in all accounts before calling `PutOrganizationConformancePack`. You can create this role manually or using the AWS CloudFormation stack-sets to create this role in every account.

If your template uses AWS CloudFormation intrinsic function `Fn::ImportValue` to import a particular variable, then that variable must be defined as an `Export Value` in all the member accounts of that organization.

For custom AWS Config rule, see How to Centrally Manage AWS Config Rules across Multiple AWS Accounts blog to setup proper permissions.

## Region Support

Conformance packs are supported in the following Regions.

<table>
<thead>
<tr>
<th>Region name</th>
<th>Region</th>
<th>Endpoint</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>ap-south-1</td>
<td>config.ap-south-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>ap-northeast-2</td>
<td>config.ap-northeast-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>ap-southeast-1</td>
<td>config.ap-southeast-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>ap-southeast-2</td>
<td>config.ap-southeast-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>ap-northeast-1</td>
<td>config.ap-northeast-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>ca-central-1</td>
<td>config.ca-central-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>eu-central-1</td>
<td>config.eu-central-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>eu-west-1</td>
<td>config.eu-west-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (London)</td>
<td>eu-west-2</td>
<td>config.eu-west-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Paris)</td>
<td>eu-west-3</td>
<td>config.eu-west-3.amazonaws.com</td>
<td>HTTPS</td>
</tr>
</tbody>
</table>
Conformance Pack Sample Templates

In the conformance pack template, you can use one or more AWS Config rules and remediation actions. The AWS Config rules and custom rules can be managed.

Topics
- Operational Best Practices For Amazon S3 (p. 205)
- Operational Best Practices For Amazon DynamoDB (p. 207)
- Operational Best Practices For PCI-DSS (p. 207)
- Operational Best Practices For AWS Identity And Access Management (p. 211)
- Templates With Remediation (p. 214)
- Custom Conformance Pack (p. 219)

Few conformance pack YAML templates that you see in AWS Config console are as follows.

Note
The following templates are intended to be samples for using AWS Config rules and remediation actions within a conformance pack. They do not guarantee thorough compliance check as is. You need to modify the template by adding more AWS Config rules to meet your AWS Organizations requirements and ensuring thorough compliance of your AWS resources.

Operational Best Practices For Amazon S3

Resources:
S3BucketPublicReadProhibited:
  Type: AWS::Config::ConfigRule
  Properties:
    ConfigRuleName: S3BucketPublicReadProhibited
    Description: >-
Checks that your Amazon S3 buckets do not allow public read access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).

Scope:
ComplianceResourceTypes:
- "AWS::S3::Bucket"
Source:
Owner: AWS
SourceIdentifier: S3_BUCKET_PUBLIC_READ_PROHIBITED
MaximumExecutionFrequency: Six_Hours

S3BucketPublicWriteProhibited:
Type: "AWS::Config::ConfigRule"
Properties:
ConfigRuleName: S3BucketPublicWriteProhibited
Description: "Checks that your Amazon S3 buckets do not allow public write access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL)."

Scope:
ComplianceResourceTypes:
- "AWS::S3::Bucket"
Source:
Owner: AWS
SourceIdentifier: S3_BUCKET_PUBLIC_WRITE_PROHIBITED
MaximumExecutionFrequency: Six_Hours

S3BucketReplicationEnabled:
Type: "AWS::Config::ConfigRule"
Properties:
ConfigRuleName: S3BucketReplicationEnabled
Description: "Checks whether the Amazon S3 buckets have cross-region replication enabled."

Scope:
ComplianceResourceTypes:
- "AWS::S3::Bucket"
Source:
Owner: AWS
SourceIdentifier: S3_BUCKET_REPLICATION_ENABLED

S3BucketSSLRequestsOnly:
Type: "AWS::Config::ConfigRule"
Properties:
ConfigRuleName: S3BucketSSLRequestsOnly
Description: "Checks whether S3 buckets have policies that require requests to use Secure Socket Layer (SSL)."

Scope:
ComplianceResourceTypes:
- "AWS::S3::Bucket"
Source:
Owner: AWS
SourceIdentifier: S3_BUCKET_SSL_REQUESTS_ONLY

ServerSideReplicationEnabled:
Type: "AWS::Config::ConfigRule"
Properties:
ConfigRuleName: ServerSideReplicationEnabled
Description: "Checks that your Amazon S3 bucket either has S3 default encryption enabled or that the S3 bucket policy explicitly denies put-object requests without server side encryption."

Scope:
ComplianceResourceTypes:
- "AWS::S3::Bucket"
Source:
Owner: AWS
SourceIdentifier: S3_BUCKET_SERVER_SIDE_ENCRYPTION_ENABLED

S3BucketLoggingEnabled:
Type: "AWS::Config::ConfigRule"
Properties:
ConfigRuleName: S3BucketLoggingEnabled
Description: "Checks whether logging is enabled for your S3 buckets."
Operational Best Practices For Amazon DynamoDB

---

Resources:

DynamoDbAutoscalingEnabled:

Properties:
- ConfigRuleName: DynamoDbAutoscalingEnabled
- Description: "This rule checks whether Auto Scaling is enabled on your DynamoDB tables. Optionally you can set the read and write capacity units for the table."
- MaximumExecutionFrequency: Six_Hours
- Scope:
  - ComplianceResourceTypes: "AWS::DynamoDB::Table"
- Source:
  - Owner: AWS
  - SourceIdentifier: DYNAMODB_AUTOSCALING_ENABLED

DynamoDbTableEncryptionEnabled:

Properties:
- ConfigRuleName: DynamoDbTableEncryptionEnabled
- Description: "Checks whether the Amazon DynamoDB tables are encrypted and checks their status. The rule is compliant if the status is enabled or enabling."
- Scope:
  - ComplianceResourceTypes: "AWS::DynamoDB::Table"
- Source:
  - Owner: AWS
  - SourceIdentifier: DYNAMODB_TABLE_ENCRYPTION_ENABLED

DynamoDbThroughputLimitCheck:

Properties:
- ConfigRuleName: DynamoDbThroughputLimitCheck
- Description: "Checks whether provisioned DynamoDB throughput is approaching the maximum limit for your account."
- MaximumExecutionFrequency: Six_Hours
- Source:
  - Owner: AWS
  - SourceIdentifier: DYNAMODB_THROUGHPUT_LIMIT_CHECK

Operational Best Practices For PCI-DSS

This template accepts parameters.

Parameters:

S3BucketBlacklistedActionsProhibitedParameterBlacklistedActionPattern:
- Description: Comma-separated list of blacklisted action patterns, for example, s3:GetBucket* and s3:DeleteObject.
- Type: String

S3BucketPolicyNotMorePermissiveParameterControlPolicy:
- Description: Amazon S3 bucket policy that defines an upper bound on the permissions of your S3 buckets. The policy can be a maximum of 1024 characters long.
- Type: String

Resources:
DMSReplicationNotPublic:
  Properties:
  ConfigRuleName: DMSReplicationNotPublic
  Description: Checks whether AWS Database Migration Service replication instances
  are public. The rule is NON_COMPLIANT if PubliclyAccessible field is True.
  Source:
  Owner: AWS
  SourceIdentifier: DMS_REPLICATION_NOT_PUBLIC
  Type: AWS::Config::ConfigRule

EBSSnapshotPublicRestorableCheck:
  Properties:
  ConfigRuleName: EBSSnapshotPublicRestorableCheck
  Description: Checks whether Amazon Elastic Block Store (Amazon EBS) snapshots
  are not publicly restorable. The rule is NON_COMPLIANT if one or more snapshots
  with RestorableByUserIds field are set to all, that is, Amazon EBS snapshots
  are public.
  Source:
  Owner: AWS
  SourceIdentifier: EBS_SNAPSHOT_PUBLIC_RESTORABLE_CHECK
  Type: AWS::Config::ConfigRule

EC2InstanceNoPublicIP:
  Properties:
  ConfigRuleName: EC2InstanceNoPublicIP
  Description: Checks whether Amazon Elastic Compute Cloud (Amazon EC2) instances
  have a public IP association. The rule is NON_COMPLIANT if the publicIp field
  is present in the Amazon EC2 instance configuration item. This rule applies
  only to IPv4.
  Source:
  Owner: AWS
  SourceIdentifier: EC2_INSTANCE_NO_PUBLIC_IP
  Type: AWS::Config::ConfigRule

ElasticsearchInVPCOnly:
  Properties:
  ConfigRuleName: ElasticsearchInVPCOnly
  Description: Checks whether Amazon Elasticsearch Service domains are in Amazon
  Virtual Private Cloud (Amazon VPC). The rule is NON_COMPLIANT if Amazon
  Elasticsearch Service domain endpoint is public.
  Source:
  Owner: AWS
  SourceIdentifier: ELASTICSEARCH_IN_VPC_ONLY
  Type: AWS::Config::ConfigRule

IAMRootAccessKeyCheck:
  Properties:
  ConfigRuleName: IAMRootAccessKeyCheck
  Description: Checks whether the root user access key is available. The rule
  is compliant if the user access key does not exist.
  Source:
  Owner: AWS
  SourceIdentifier: IAM_ROOT_ACCESS_KEY_CHECK
  Type: AWS::Config::ConfigRule

IAMUserMFAEnabled:
  Properties:
  ConfigRuleName: IAMUserMFAEnabled
  Description: Checks whether the AWS Identity and Access Management users have
  multi-factor authentication (MFA) enabled.
  Source:
  Owner: AWS
  SourceIdentifier: IAM_USER_MFA_ENABLED
  Type: AWS::Config::ConfigRule

IncomingSSHDisabled:
  Properties:
  ConfigRuleName: IncomingSSHDisabled
  Description: Checks whether security groups that are in use disallow unrestricted
  incoming SSH traffic.
  Source:
  Owner: AWS
SourceIdentifier: INCOMING_SSH_DISABLED
Type: AWS::Config::ConfigRule
InstancesInVPC:
  Properties:
    ConfigRuleName: InstancesInVPC
    Description: Checks whether your EC2 instances belong to a virtual private cloud (VPC).
    Source:
      Owner: AWS
SourceIdentifier: INSTANCES_IN_VPC
Type: AWS::Config::ConfigRule
LambdaFunctionPublicAccessProhibited:
  Properties:
    ConfigRuleName: LambdaFunctionPublicAccessProhibited
    Description: Checks whether the Lambda function policy prohibits public access.
    Source:
      Owner: AWS
SourceIdentifier: LAMBDA_FUNCTION_PUBLIC_ACCESS_PROHIBITED
Type: AWS::Config::ConfigRule
LambdaInsideVPC:
  Properties:
    ConfigRuleName: LambdaInsideVPC
    Description: Checks whether an AWS Lambda function is in an Amazon Virtual Private Cloud. The rule is NON_COMPLIANT if the Lambda function is not in a VPC.
    Source:
      Owner: AWS
SourceIdentifier: LAMBDA_INSIDE_VPC
Type: AWS::Config::ConfigRule
MFAEnabledForIAMConsoleAccess:
  Properties:
    ConfigRuleName: MFAEnabledForIAMConsoleAccess
    Description: Checks whether AWS Multi-Factor Authentication (MFA) is enabled for all AWS Identity and Access Management (IAM) users that use a console password. The rule is compliant if MFA is enabled.
    Source:
      Owner: AWS
SourceIdentifier: MFA_ENABLED_FOR_IAM_CONSOLE_ACCESS
Type: AWS::Config::ConfigRule
RDSInstancePublicAccessCheck:
  Properties:
    ConfigRuleName: RDSInstancePublicAccessCheck
    Description: Checks whether the Amazon Relational Database Service (Amazon RDS) instances are not publicly accessible. The rule is non-compliant if the publiclyAccessible field is true in the instance configuration item.
    Source:
      Owner: AWS
SourceIdentifier: RDS_INSTANCE_PUBLIC_ACCESS_CHECK
Type: AWS::Config::ConfigRule
RDSSnapshotsPublicProhibited:
  Properties:
    ConfigRuleName: RDSSnapshotsPublicProhibited
    Description: Checks if Amazon Relational Database Service (Amazon RDS) snapshots are public. The rule is non-compliant if any existing and new Amazon RDS snapshots are public.
    Source:
      Owner: AWS
SourceIdentifier: RDS_SNAPSHOTS_PUBLIC_PROHIBITED
Type: AWS::Config::ConfigRule
RedshiftClusterPublicAccessCheck:
  Properties:
    ConfigRuleName: RedshiftClusterPublicAccessCheck
    Description: Checks whether Amazon Redshift clusters are not publicly accessible. The rule is NON_COMPLIANT if the publiclyAccessible field is true in the cluster configuration item.
    Source:
Owner: AWS
SourceIdentifier: REDSHIFT_CLUSTERS_PUBLIC_ACCESS_CHECK
Type: AWS::Config::ConfigRule

RestrictedIncomingTraffic:
Properties:
  ConfigRuleName: RestrictedIncomingTraffic
  Description: Checks whether security groups that are in use disallow unrestricted incoming TCP traffic to the specified ports.
  Source:
  Owner: AWS
SourceIdentifier: RESTRICTED_INCOMING_TRAFFIC
Type: AWS::Config::ConfigRule

RootAccountHardwareMFAEnabled:
Properties:
  ConfigRuleName: RootAccountHardwareMFAEnabled
  Description: Checks whether your AWS account is enabled to use multi-factor authentication (MFA) hardware device to sign in with root credentials.
  Source:
  Owner: AWS
SourceIdentifier: ROOT_ACCOUNT_HARDWARE_MFA_ENABLED
Type: AWS::Config::ConfigRule

RootAccountMFAEnabled:
Properties:
  ConfigRuleName: RootAccountMFAEnabled
  Description: Checks whether the root user of your AWS account requires multi-factor authentication for console sign-in.
  Source:
  Owner: AWS
SourceIdentifier: ROOT_ACCOUNT_MFA_ENABLED
Type: AWS::Config::ConfigRule

S3BucketBlacklistedActionsProhibited:
Properties:
  ConfigRuleName: S3BucketBlacklistedActionsProhibited
  Description: Checks that the S3 bucket policy does not allow blacklisted bucket-level and object-level actions for principals from other AWS Accounts. The rule is non-compliant if any blacklisted actions are allowed by the S3 bucket policy.
  InputParameters:
    blacklistedActionPattern:
    Ref: S3BucketBlacklistedActionsProhibitedParameterBlacklistedActionPattern
  Source:
  Owner: AWS
SourceIdentifier: S3_BUCKET_BLACKLISTED_ACTIONS_PROHIBITED
Type: AWS::Config::ConfigRule

S3BucketPolicyGranteeCheck:
Properties:
  ConfigRuleName: S3BucketPolicyGranteeCheck
  Description: Checks that the access granted by the Amazon S3 bucket is restricted to any of the AWS principals, federated users, service principals, IP addresses, or VPCs that you provide. The rule is COMPLIANT if a bucket policy is not present.
  Source:
  Owner: AWS
SourceIdentifier: S3_BUCKET_POLICY_GRANTEE_CHECK
Type: AWS::Config::ConfigRule

S3BucketPolicyNotMorePermissive:
Properties:
  ConfigRuleName: S3BucketPolicyNotMorePermissive
  Description: Verifies that your Amazon S3 bucket policies do not allow other inter-account permissions than the control S3 bucket policy that you provide.
  InputParameters:
    controlPolicy:
    Ref: S3BucketPolicyNotMorePermissiveParameterControlPolicy
  Source:
  Owner: AWS
SourceIdentifier: S3_BUCKET_POLICY_NOT_MORE_PERMISSIVE
Type: AWS::Config::ConfigRule
AWS Config Developer Guide
Operational Best Practices For AWS
Identity And Access Management

S3BucketPublicReadProhibited:
Properties:
  ConfigRuleName: S3BucketPublicReadProhibited
  Description: Checks that your Amazon S3 buckets do not allow public read access.
  The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).
  Source:
    Owner: AWS
    SourceIdentifier: S3_BUCKET_PUBLIC_READ_PROHIBITED
  Type: AWS::Config::ConfigRule
S3BucketPublicWriteProhibited:
Properties:
  ConfigRuleName: S3BucketPublicWriteProhibited
  Description: Checks that your Amazon S3 buckets do not allow public write access.
  The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).
  Source:
    Owner: AWS
    SourceIdentifier: S3_BUCKET_PUBLIC_WRITE_PROHIBITED
  Type: AWS::Config::ConfigRule
S3BucketVersioningEnabled:
Properties:
  ConfigRuleName: S3BucketVersioningEnabled
  Description: Checks whether versioning is enabled for your S3 buckets.
  Source:
    Owner: AWS
    SourceIdentifier: S3_BUCKET_VERSIONING_ENABLED
  Type: AWS::Config::ConfigRule
VPCDefaultSecurityGroupClosed:
Properties:
  ConfigRuleName: VPCDefaultSecurityGroupClosed
  Description: Checks that the default security group of any Amazon Virtual Private Cloud (Amazon VPC) does not allow inbound or outbound traffic. The rule is non-compliant if the default security group has inbound or outbound traffic.
  Source:
    Owner: AWS
    SourceIdentifier: VPC_DEFAULT_SECURITY_GROUP_CLOSED
  Type: AWS::Config::ConfigRule
VPCSGOpenOnlyToAuthorizedPorts:
Properties:
  ConfigRuleName: VPCSGOpenOnlyToAuthorizedPorts
  Description: Checks whether any security groups with inbound 0.0.0.0/0 have TCP or UDP ports accessible. The rule is NON_COMPLIANT when a security group with inbound 0.0.0.0/0 has a port accessible which is not specified in the rule parameters.
  Source:
    Owner: AWS
    SourceIdentifier: VPC_SG_OPEN_ONLY_TOAUTHORIZED_PORTS
  Type: AWS::Config::ConfigRule

Operational Best Practices For AWS Identity And Access Management

This template accepts parameters.

Parameters:
AccessKeysRotatedParameterMaxAccessKeyAge:
  Description: Maximum number of days without rotation. Default 90.
  Type: String
IAMPolicyBlacklistedCheckParameterPolicyArns:
  Description: Comma-separated list of IAM policy ARNs that should not be attached to any IAM entity.
IAMRoleManagedPolicyCheck:

Properties:
- ConfigRuleName: IAMRoleManagedPolicyCheck
- Description: Checks that the AWS Identity and Access Management (IAM) role is managed by AWS managed policies.
- InputParameters:
  - ManagedPolicyArns:
    - Description: Comma-separated list of AWS managed policy ARNs.
- Source:
  - Owner: AWS
  - SourceIdentifier: IAM_ROLE_MANAGED_POLICY_CHECK
- Type: AWS::Config::ConfigRule

IAMUserUnusedCredentialsCheck:

Properties:
- ConfigRuleName: IAMUserUnusedCredentialsCheck
- Description: Checks whether IAM users have unused credentials.
- InputParameters:
  - MaxCredentialUsageAge:
    - Description: Maximum number of days a credential cannot be used. The default value is 90 days.
- Source:
  - Owner: AWS
  - SourceIdentifier: IAM_USER_UNUSED_CREDENTIALS_CHECK
- Type: AWS::Config::ConfigRule

Resources:

AccessKeysRotated:

Properties:
- ConfigRuleName: AccessKeysRotated
- Description: Checks whether the active access keys are rotated within the number of days specified in maxAccessKeyAge. The rule is non-compliant if the access keys have not been rotated for more than maxAccessKeyAge number of days.
- InputParameters:
  - maxAccessKeyAge:
    - Description: Time in days after which an access key is deemed expired. The default value is 90 days.
- Source:
  - Owner: AWS
  - SourceIdentifier: ACCESS_KEYS_ROTATED
- Type: AWS::Config::ConfigRule

IAMGroupHasUsersCheck:

Properties:
- ConfigRuleName: IAMGroupHasUsersCheck
- Description: Checks whether IAM groups have at least one IAM user.
- Source:
  - Owner: AWS
  - SourceIdentifier: IAM_GROUP_HAS_USERS_CHECK
- Type: AWS::Config::ConfigRule

IAMPasswordPolicy:

Properties:
- ConfigRuleName: IAMPasswordPolicy
- Description: Checks whether the account password policy for IAM users meets the specified requirements.
- Source:
  - Owner: AWS
  - SourceIdentifier: IAM_PASSWORD_POLICY
- Type: AWS::Config::ConfigRule

IAMPolicyBlacklistedCheck:

Properties:
- ConfigRuleName: IAMPolicyBlacklistedCheck
- Description: Checks that none of your IAM users, groups, or roles (excluding exceptionList) have the specified policies attached.
- InputParameters:
  - PolicyArns:
    - Description: ARNs of the IAM policies to check.
- Source:
  - Owner: AWS
  - SourceIdentifier: IAM_POLICY_BLACKLISTED_CHECK
- Type: AWS::Config::ConfigRule

IAMPolicyNoStatementsWithAdminAccess:

Properties:
- ConfigRuleName: IAMPolicyNoStatementsWithAdminAccess
- Description: Checks whether the default version of AWS Identity and Access Management (IAM) policies do not have administrator access. If any statement has "Effect": "Allow" with "Action": "*" over "Resource": "*", the rule is non-compliant.
- Source:
  - Owner: AWS
  - SourceIdentifier: IAM_POLICY_NO_STATEMENTS_WITH_ADMIN_ACCESS
- Type: AWS::Config::ConfigRule

IAMRoleManagedPolicyCheck:

Properties:
- ConfigRuleName: IAMRoleManagedPolicyCheck
- Description: Checks that the AWS Identity and Access Management (IAM) role is managed by AWS managed policies.
- InputParameters:
  - ManagedPolicyArns:
    - Description: Comma-separated list of AWS managed policy ARNs.
attached to all AWS managed policies specified in the list of managed policies. The rule is non-compliant if the IAM role is not attached to the AWS managed policy.

InputParameters:
  - managedPolicyArns:
    Ref: IAMRoleManagedPolicyCheckParameterManagedPolicyArns

Source:
  - Owner: AWS
  - SourceIdentifier: IAM_ROLE_MANAGED_POLICY_CHECK

Type: AWS::Config::ConfigRule

IAMRootAccessKeyCheck:

Properties:
  - ConfigRuleName: IAMRootAccessKeyCheck
  - Description: Checks whether the root user access key is available. The rule is compliant if the user access key does not exist.

Source:
  - Owner: AWS
  - SourceIdentifier: IAM_ROOT_ACCESS_KEY_CHECK

Type: AWS::Config::ConfigRule

IAMUserGroupMembershipCheck:

Properties:
  - ConfigRuleName: IAMUserGroupMembershipCheck
  - Description: Checks whether IAM users are members of at least one IAM group.

Source:
  - Owner: AWS
  - SourceIdentifier: IAM_USER_GROUP_MEMBERSHIP_CHECK

Type: AWS::Config::ConfigRule

IAMUserMFAEnabled:

Properties:
  - ConfigRuleName: IAMUserMFAEnabled
  - Description: Checks whether the AWS Identity and Access Management users have multi-factor authentication (MFA) enabled.

Source:
  - Owner: AWS
  - SourceIdentifier: IAM_USER_MFA_ENABLED

Type: AWS::Config::ConfigRule

IAMUserNoPoliciesCheck:

Properties:
  - ConfigRuleName: IAMUserNoPoliciesCheck
  - Description: Checks that none of your IAM users have policies attached. IAM users must inherit permissions from IAM groups or roles.

Source:
  - Owner: AWS
  - SourceIdentifier: IAM_USER_NO_POLICIES_CHECK

Type: AWS::Config::ConfigRule

IAMUserUnusedCredentialsCheck:

Properties:
  - ConfigRuleName: IAMUserUnusedCredentialsCheck
  - Description: Checks whether your AWS Identity and Access Management (IAM) users have passwords or active access keys that have not been used within the specified number of days you provided.

InputParameters:
  - maxCredentialUsageAge:
    Ref: IAMUserUnusedCredentialsCheckParameterMaxCredentialUsageAge

Source:
  - Owner: AWS
  - SourceIdentifier: IAM_USER_UNUSED_CREDENTIALS_CHECK

Type: AWS::Config::ConfigRule

MFAEnabledForIAMConsoleAccess:

Properties:
  - ConfigRuleName: MFAEnabledForIAMConsoleAccess
  - Description: Checks whether AWS Multi-Factor Authentication (MFA) is enabled for all AWS Identity and Access Management (IAM) users that use a console password. The rule is compliant if MFA is enabled.

Source:
  - Owner: AWS
SourceIdentifier: MFA_ENABLED_FOR_IAM_CONSOLE_ACCESS
Type: AWS::Config::ConfigRule
RootAccountHardwareMFAEnabled:
  Properties:
    ConfigRuleName: RootAccountHardwareMFAEnabled
    Description: Checks whether your AWS account is enabled to use a multi-factor authentication (MFA) hardware device to sign in with root credentials.
    Source:
      Owner: AWS
    SourceIdentifier: ROOT_ACCOUNT_HARDWARE_MFA_ENABLED
    Type: AWS::Config::ConfigRule
RootAccountMFAEnabled:
  Properties:
    ConfigRuleName: RootAccountMFAEnabled
    Description: Checks whether the root user of your AWS account requires multi-factor authentication for console sign-in.
    Source:
      Owner: AWS
    SourceIdentifier: ROOT_ACCOUNT_MFA_ENABLED
    Type: AWS::Config::ConfigRule

Templates With Remediation

Operational Best Practices For Amazon DynamoDB with Remediation

---

Parameters:
SnsTopicForPublishNotificationArn:
  Description: The ARN of the SNS topic to which the notification about the auto-remediation status should be published.
  Type: String

Resources:
DynamoDbAutoscalingEnabled:
  Properties:
    ConfigRuleName: DynamoDbAutoscalingEnabled
    Description: "This rule checks whether Auto Scaling is enabled on your DynamoDB tables. Optionally you can set the read and write capacity units for the table."
    MaximumExecutionFrequency: Six_Hours
    Scope:
      ComplianceResourceTypes:
        - "AWS::DynamoDB::Table"
    Source:
      Owner: AWS
    SourceIdentifier: DYNAMODB_AUTOSCALING_ENABLED
Type: "AWS::Config::ConfigRule"
DynamoDbAutoscalingEnabledManualRemediation:
  DependsOn: DynamoDbAutoscalingEnabled
  Type: 'AWS::Config::RemediationConfiguration'
  Properties:
    ConfigRuleName: DynamoDbAutoscalingEnabled
    ResourceTypes:
      - "AWS::DynamoDB::Table"
    TargetId: "AWS-PublishSNSNotification"
    TargetType: "SSM_DOCUMENT"
    TargetVersion: "1"
    Parameters:
      AutomationAssumeRole:
        StaticValue:
          Values:
            - "arn:aws:iam::<Account ID>:role/PublishSnsAutomationExecutionRole"
    Message:
DynamoDbTableEncryptionEnabled:
  Properties:
  - ConfigRuleName: DynamoDbTableEncryptionEnabled
  - Description: "Checks whether the Amazon DynamoDB tables are encrypted and checks their status. The rule is compliant if the status is enabled or enabling."
  Scope:
  - ComplianceResourceTypes:
  - "AWS::DynamoDB::Table"
  Source:
  - Owner: AWS
  - SourceIdentifier: DYNAMODB_TABLE_ENCRYPTION_ENABLED
  Type: "AWS::Config::ConfigRule"
DynamoDbTableEncryptionEnabledAutoRemediation:
  DependsOn: DynamoDbTableEncryptionEnabled
  Type: 'AWS::Config::RemediationConfiguration'
  Properties:
  - ConfigRuleName: DynamoDbTableEncryptionEnabled
  - ResourceType: "AWS::DynamoDB::Table"
  - TargetId: "AWS-PublishSNSNotification"
  - TargetType: "SSM_DOCUMENT"
  - TargetVersion: "1"
  Parameters:
  - AutomationAssumeRole:
  - ExecutionControls:
    - SsmControls:
      - ConcurrentExecutionRatePercentage: 10
      - ErrorPercentage: 10
      - Automatic: True
      - MaximumAutomaticAttempts: 10
      - RetryAttemptSeconds: 600
  TopicArn:
  - StaticValue:
  - Values:
  - "A table with no encryption enabled is found"

DynamoDbThroughputLimitCheck:
  Properties:
  - ConfigRuleName: DynamoDbThroughputLimitCheck
  - Description: "Checks whether provisioned DynamoDB throughput is approaching the maximum limit for your account."
  - MaximumExecutionFrequency: Six_Hours
  Source:
  - Owner: AWS
  - SourceIdentifier: DYNAMODB_THROUGHPUT_LIMIT_CHECK
  Type: "AWS::Config::ConfigRule"

Operational Best Practices For Amazon S3 with Remediation

Parameters:
S3TargetBucketNameForEnableLogging:
  Description: The target s3 bucket where the logging should be enabled.
  Type: String

Resources:
S3BucketPublicReadProhibited:
  Type: AWS::Config::ConfigRule
  Properties:
    ConfigRuleName: S3BucketPublicReadProhibited
    Description: Checks that your Amazon S3 buckets do not allow public read access.
    The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).
    Scope:
    ComplianceResourceTypes: - "AWS::S3::Bucket"
    Source:
      Owner: AWS
      SourceIdentifier: S3_BUCKET_PUBLIC_READ_PROHIBITED
      MaximumExecutionFrequency: Six_Hours
    S3PublicReadRemediation:
      DependsOn: S3BucketPublicReadProhibited
      Type: 'AWS::Config::RemediationConfiguration'
      Properties:
        ConfigRuleName: S3BucketPublicReadProhibited
        ResourceId: "AWS::S3::Bucket"
        TargetId: "AWS-DisableS3BucketPublicReadWrite"
        targetType: "SSM_DOCUMENT"
        TargetVersion: "1"
        Parameters:
          AutomationAssumeRole:
            StaticValue:
              Values:
                - arn:aws:iam::Account ID:role/S3OperationsAutomationsExecutionRole
          S3BucketName:
            ResourceValue:
              Value: "RESOURCE_ID"
          ExecutionControls:
            SsmControls:
              ConcurrentExecutionRatePercentage: 10
              ErrorPercentage: 10
              Automatic: True
              MaximumAutomaticAttempts: 10
              RetryAttemptSeconds: 600

S3BucketPublicWriteProhibited:
  Type: "AWS::Config::ConfigRule"
  Properties:
    ConfigRuleName: S3BucketPublicWriteProhibited
    Description: "Checks that your Amazon S3 buckets do not allow public write access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL)."
    Scope:
    ComplianceResourceTypes: - "AWS::S3::Bucket"
    Source:
      Owner: AWS
      SourceIdentifier: S3_BUCKET_PUBLIC_WRITE_PROHIBITED
      MaximumExecutionFrequency: Six_Hours
    S3PublicWriteRemediation:
      DependsOn: S3BucketPublicWriteProhibited
      Type: 'AWS::Config::RemediationConfiguration'
      Properties:
        ConfigRuleName: S3BucketPublicWriteProhibited
        ResourceId: "AWS::S3::Bucket"
        TargetId: "AWS-DisableS3BucketPublicReadWrite"
        targetType: "SSM_DOCUMENT"
TargetVersion: "1"
Parameters:
  AutomationAssumeRole:
    StaticValue:
      Values:
        - arn:aws:iam::Account ID:role/S3OperationsAutomationsExecutionRole
  S3BucketName:
    ResourceValue:
      Value: "RESOURCE_ID"
  ExecutionControls:
    SsmControls:
      ConcurrentExecutionRatePercentage: 10
      ErrorPercentage: 10
      Automatic: True
      MaximumAutomaticAttempts: 10
      RetryAttemptSeconds: 600
S3BucketReplicationEnabled:
  Type: "AWS::Config::ConfigRule"
  Properties:
    ConfigRuleName: S3BucketReplicationEnabled
    Description: "Checks whether the Amazon S3 buckets have cross-region replication enabled."
    Scope:
      ComplianceResourceTypes:
        - "AWS::S3::Bucket"
    Source:
      Owner: AWS
      SourceIdentifier: S3_BUCKET_REPLICATION_ENABLED
S3BucketSSLRequestsOnly:
  Type: "AWS::Config::ConfigRule"
  Properties:
    ConfigRuleName: S3BucketSSLRequestsOnly
    Description: "Checks whether S3 buckets have policies that require requests to use Secure Socket Layer (SSL)."
    Scope:
      ComplianceResourceTypes:
        - "AWS::S3::Bucket"
    Source:
      Owner: AWS
      SourceIdentifier: S3_BUCKET_SSL_REQUESTS_ONLY
S3BucketServerSideEncryptionEnabled:
  Type: "AWS::Config::ConfigRule"
  Properties:
    ConfigRuleName: S3BucketServerSideEncryptionEnabled
    Description: "Checks that your Amazon S3 bucket either has S3 default encryption enabled or that the S3 bucket policy explicitly denies put-object requests without server side encryption."
    Scope:
      ComplianceResourceTypes:
        - "AWS::S3::Bucket"
    Source:
      Owner: AWS
      SourceIdentifier: S3_BUCKET_SERVER_SIDE_ENCRYPTION_ENABLED
S3BucketServerSideEncryptionEnabledRemediation:
  DependsOn: S3BucketServerSideEncryptionEnabled
  Type: 'AWS::Config::RemediationConfiguration'
  Properties:
    ConfigRuleName: S3BucketServerSideEncryptionEnabled
    ResourceType: "AWS::S3::Bucket"
    TargetId: "AWS-EnableS3BucketEncryption"
    TargetType: "SSM_DOCUMENT"
    TargetVersion: "1"
    Parameters:
      AutomationAssumeRole:
StaticValue:
Values:
- arn:aws:iam::Account ID:role/S3OperationsAutomationsExecutionRole
BucketName:
  ResourceValue:
  Value: "RESOURCE_ID"
SSEAlgorithm:
  StaticValue:
  Values:
  - "AES256"
ExecutionControls:
  SsmControls:
    ConcurrentExecutionRatePercentage: 10
    ErrorPercentage: 10
    Automatic: True
    MaximumAutomaticAttempts: 10
    RetryAttemptSeconds: 600
S3BucketLoggingEnabled:
  Type: "AWS::Config::ConfigRule"
  Properties:
    ConfigRuleName: S3BucketLoggingEnabled
    Description: "Checks whether logging is enabled for your S3 buckets."
    Scope:
      ComplianceResourceTypes:
        - "AWS::S3::Bucket"
    Source:
      Owner: AWS
      SourceIdentifier: S3_BUCKET_LOGGING_ENABLED
S3BucketLoggingEnabledRemediation:
  DependsOn: S3BucketLoggingEnabled
  Type: 'AWS::Config::RemediationConfiguration'
  Properties:
    ConfigRuleName: S3BucketLoggingEnabled
    ResourceType: "AWS::S3::Bucket"
    TargetId: "AWS-ConfigureS3BucketLogging"
    targetType: "SSM_DOCUMENT"
    TargetVersion: "1"
    parameters:
      AutomationAssumeRole:
        StaticValue:
        Values:
        - arn:aws:iam::Account ID:role/S3OperationsAutomationsExecutionRole
      BucketName:
        ResourceValue:
        Value: "RESOURCE_ID"
      TargetBucket:
        StaticValue:
        Values:
        - Ref: S3TargetBucketNameForEnableLogging
      GrantedPermission:
        StaticValue:
        Values:
        - "FULL_CONTROL"
      GranteeType:
        StaticValue:
        Values:
        - "Group"
      GranteeUri:
        StaticValue:
        Values:
        - "http://acs.amazonaws.com/groups/s3/LogDelivery"
    ExecutionControls:
      SsmControls:
        ConcurrentExecutionRatePercentage: 10
        ErrorPercentage: 10
Custom Conformance Pack

Parameters:
- CustomConfigRuleLambdaArn:
  
  Description: The ARN of the custom config rule lambda.

  Type: String

Resources:
- CustomRuleForEC2:
  
  Type: AWS::Config::ConfigRule

  Properties:

  ConfigRuleName: "CustomRuleForEC2"

  Scope:

  ComplianceResourceTypes:

  - "AWS::EC2::Volume"

  Source:

  Owner: "CUSTOM_LAMBDA"

  SourceDetails:

  - EventSource: "aws.config"
    MessageType: "ConfigurationItemChangeNotification"

  - EventSource: "aws.config"
    MessageType: "OversizedConfigurationItemChangeNotification"

  SourceIdentifier:

    Ref: CustomConfigRuleLambdaArn

- ConfigRuleForVolumeTags:
  
  Type: AWS::Config::ConfigRule

  Description: "Test CREATE"

  Properties:

  ConfigRuleName: "ConfigRuleForVolumeTags"

  InputParameters:

    tag1Key: CostCenter

  Scope:

  ComplianceResourceTypes:

  - "AWS::EC2::Volume"

  Source:

    Owner: AWS

    SourceIdentifier: "REQUIRED_TAGS"

- CloudTrailEnabled:
  
  Type: AWS::Config::ConfigRule

  Description: "CloudTrail rule"

  Properties:

  ConfigRuleName: "CloudTrailEnabled"

  InputParameters:

    s3BucketName: testBucketName

  Source:

    Owner: AWS

    SourceIdentifier: "CLOUD_TRAIL_ENABLED"

For more information about template structure, see Template Anatomy in AWS CloudFormation user guide.
Deploying a Conformance Pack Using the AWS Config Console

On the **Conformance packs** page, you can deploy a conformance pack for an account in a Region. You can also edit and delete the deployed conformance pack.

**Deploy a Conformance Pack Using Sample Templates**

You can deploy a conformance pack using AWS Config sample templates.

1. Sign in to the AWS Management Console and open the AWS Config console at [https://console.aws.amazon.com/config/](https://console.aws.amazon.com/config/).
2. Navigate to the **Conformance packs** page and choose **Deploy conformance pack**.
3. On the **Specify template** page, either choose a sample template or use an existing template.
   - If you choose **Use sample template**, select a **Sample template** from the drop-down list of sample templates.
     
     For information about the contents of each template, see Conformance Pack Sample Templates.
   - If you choose **Template is ready**, specify the template source. It is either an Amazon S3 URI or a that you upload.
     
     If your template is more than 50 KB, upload it to the S3 bucket and select that S3 bucket location. For example: s3://bucketname/prefix.
4. Choose **Next**.
5. On the **Specify conformance pack details** page, type the name for your conformance pack.

   The conformance pack name must be a unique name with a maximum of 256 alphanumeric characters. The name can contain hyphens but cannot contain spaces.
6. Type the name of the Amazon S3 bucket. Optionally, type a bucket prefix.

   The S3 bucket name must be a minimum of 3 and maximum of 63 lower case alphanumeric characters. The bucket name can contain dashes and must start and end with alphanumeric characters.
7. Optional: Add a parameter.

   Parameters are defined in your template and help you manage and organize your resources.
8. Choose **Next**.
9. On the **Review and deploy** page, review all of the information.

   You can edit the template details and conformance pack details by choosing **Edit**.
10. Choose **Deploy conformance pack**.

    AWS Config displays the conformance pack on the conformance pack page with the appropriate status.

    If your conformance pack deployment fails, check your permissions, verify that you did the prerequisite steps, and try again. Or you can contact AWS Config support.

To deploy a **conformance pack using sample template with remediations**, see the **Prerequisites for Using a Conformance Pack With Remediation** (p. 203) and then use the preceding procedure.
To deploy a **conformance pack with one or more AWS Config rules**, see the Prerequisites for Using a Conformance Pack With One or More AWS Config Rules (p. 203).

### Edit a Conformance Pack

1. To edit a conformance pack, select the conformance pack from the table.
2. Choose **Actions** and then choose **Edit**.
3. On the **Edit conformance pack** page, you can edit the template details, sample template, conformance pack, and parameters section.
   
   You cannot change the name of the conformance pack.
4. Choose **Save changes**.

   The conformance pack is displayed with the AWS Config rules.

### Delete a Conformance Pack

1. To delete a conformance pack, select the conformance pack from the table.
2. Choose **Actions** and then choose **Delete**.
3. On the delete **conformance pack** dialog box, confirm if you would like to permanently delete this conformance pack.
   
   You cannot revert this action. When you delete a conformance pack, you delete all of the AWS Config rules and remediation actions in that conformance pack.
4. Enter **Delete** and choose **Delete**.

   On the **Conformance packs** page, you can see the deployment status as **Deleting** until the conformance pack is completely deleted.

### Deploying a Conformance Pack Using the AWS Command Line Interface

You can deploy, view, update, view compliance status, and delete an AWS Config conformance pack using the AWS Command Line Interface (AWS CLI).

To install the AWS CLI on your local machine see, [AWS Config Conforms Amazon S3 bucket](#).

If necessary, type `aws configure` to configure the AWS CLI to use an AWS Region where AWS Config conformance packs are available.

**Topics**

- Deploy a Conformance Pack (p. 222)
- View a Conformance Pack (p. 222)
- View Conformance Pack Status (p. 222)
- View Conformance Pack Compliance Status (p. 223)
- Get Compliance Details for a Specific Conformance Pack (p. 223)
- Delete a Conformance Pack (p. 224)
Deploy a Conformance Pack

1. Open a command prompt or a terminal window.
2. Type the following command to deploy a conformance pack named `MyConformancePack1`.
   
   ```bash
   aws configservice put-conformance-pack --conformance-pack-name="MyConformancePack1" --template-s3-uri="s3://AmazonS3bucketname/template name.yaml" --delivery-s3-bucket=AmazonS3bucketname
   ```
   
   OR

   You can also upload a YAML template from your local directory.
   
   ```bash
   aws configservice put-conformance-pack --conformance-pack-name="MyConformancePack1" --template-body=template body --delivery-s3-bucket=AmazonS3bucketname
   ```

3. Press Enter to run the command.

   You should see output similar to the following.

   ```json
   {
   }
   ```

View a Conformance Pack

1. Type the following command.

   ```bash
   aws configservice describe-conformance-packs
   ```
   
   OR

   ```bash
   aws configservice describe-conformance-packs --conformance-pack-name="MyConformancePack1"
   ```

2. You should see output similar to the following.

   ```json
   {
   "conformancePackName": "MyConformancePack1",
   "conformancePackId": "conformance-pack-ID",
   "conformancePackInputParameters": [],
   "lastUpdateRequestedTime": "Thu Jul 18 16:07:05 PDT 2019"
   }
   ```

View Conformance Pack Status

1. Type the following command.
View Conformance Pack Compliance Status

1. Type the following command.

```bash
aws configservice describe-conformance-pack-compliance --conformance-pack-name="MyConformancePack1"
```

2. You should see output similar to the following.

```json
{
   "conformancePackName": "MyConformancePack1",
   "conformancePackRuleComplianceList": [
      {
         "configRuleName": "awsconfigconforms-RuleName1-conformance-pack-ID",
         "complianceType": "NON_COMPLIANT"
      },
      {
         "configRuleName": "awsconfigconforms-RuleName2-conformance-pack-ID",
         "complianceType": "COMPLIANT"
      }
   ]
}
```

Get Compliance Details for a Specific Conformance Pack

1. Type the following command.

```bash
aws configservice get-conformance-pack-compliance-details --conformance-pack-name="MyConformancePack1"
```

2. You should see output similar to the following.

```json
{
   "conformancePackRuleEvaluationResults": [
      {
         "evaluationResultIdentifier": {
            "evaluatedResourceId": "account-1234567890",
            "evaluatedResourceType": "AWS::IAM::User",
            "resourceType": "AWS::IAM::User",
            "resourceArn": "arn:aws:iam:us-west-2:1234567890:iam/user/test",
            "evaluatedResourceName": "test",
            "resourceName": "test",
            "evaluationResult": {
               "evaluationResultId": "12345678901234567890",
               "evaluationResultStatus": "COMPLIANT",
               "evaluationResultStatusReason": "Rule RuleName1 is not triggered on resource test",
               "evaluationResultStatusReasonType": "COMPLIANT",
               "evaluationResultStatusReasonDetails": {
                  "ruleName": "RuleName1",
                  "ruleAttributes": {
                     "Key": "Attribute1",
                     "Value": "Value1"
                  }
               }
            }
         }
      }
   ]
}
```
Delete a Conformance Pack

Type the following command.

```
aws configservice delete-conformance-pack --conformance-pack-name MyConformancePack1
```

If successful, the command runs with no additional output.

Managing Conformance Packs (API)

Use the following AWS Config API actions to manage conformance packs:

- DeleteConformancePack
- DescribeConformancePackCompliance
- DescribeConformancePacks
- DescribeConformancePackStatus
- GetConformancePackComplianceDetails
- GetConformancePackComplianceSummary
- PutConformancePack
Managing Conformance Packs Across all Accounts in Your Organization

Use AWS Config to manage conformance packs across all AWS accounts within an organization. You can do the following:

- Centrally deploy, update, and delete conformance packs across member accounts in an organization in AWS Organizations.
- Deploy a common set of AWS Config rules and remediation actions across all accounts and specify accounts where AWS Config rules and remediation actions should not be created.
- Use the APIs from the master account in AWS Organizations to enforce governance by ensuring that the underlying AWS Config rules and remediation actions are not modifiable by your organization's member accounts.

Permissions for cross account bucket access

Each member account must have permission to access the master account bucket. The member account service-linked role must have permissions to access the Amazon S3 bucket and the master account must allow all the member accounts to use this bucket. We recommend having limited permissions to the Amazon S3 bucket policy. To limit access, you can use PrincipalOrgID and PrincipalArn conditions in the Amazon S3 policy.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": ["s3:GetObject", "s3:PutObject"],
      "Resource": "arn:aws:s3:::awsconfigconforms\customer_bucket_name/\**",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "\customer_org_id"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
        }
      }
    },
    {
      "Sid": "AllowGetBucketAcl",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::awsconfigconforms\customer_bucket_name",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "\customer_org_id"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:role/aws-service-role/config-conforms.amazonaws.com/AWSServiceRoleForConfigConforms"
        }
      }
    }
  ]
}
```
Ensure AWS Config recording is on before you use the following APIs to manage conformance pack rules across all AWS accounts within an organization:

- `DeleteOrganizationConformancePack`
- `DescribeOrganizationConformancePacks`
- `DescribeOrganizationConformancePackStatuses`
- `GetOrganizationConformancePackDetailedStatus`
- `PutOrganizationConformancePack`

**Troubleshooting**

If you get an error indicating that the conformance pack failed while creating, updating, or deleting it, you can check the status of your conformance pack.

```
aws configservice describe-conformance-pack-status --conformance-pack-name=ConformancePackName
```

You should see output similar to the following.

```json
"ConformancePackStatusDetails": [
  {
    "ConformancePackName": "ConformancePackName",
    "ConformancePackId": "ConformancePackId",
    "ConformancePackArn": "ConformancePackArn",
    "ConformancePackState": "CREATE_FAILED",
    "StackArn": "CloudFormation stackArn",
    "ConformancePackStatusReason": "Failure Reason",
    "LastUpdateRequestedTime": 1573865201.619,
    "LastUpdateCompletedTime": 1573864244.653
  }
]
```

Check the `ConformancePackStatusReason` for information about the failure.

**When the stackArn is present in the response**

If the error message is not clear or if the failure is due to an internal error, go to the AWS CloudFormation console and do the following:

1. Search for the `stackArn` from the output.
2. Choose the **Events** tab of the AWS CloudFormation stack and check for failed events.

   The status reason indicates why the conformance pack failed.

**When the stackArn is not present in the response**

If you receive a failure while you create a conformance pack but the stackArn is not present in the status response, the possible reason is that the stack creation failed and AWS CloudFormation rolled back and deleted the stack. Go to the AWS CloudFormation console and search for stacks that are in a *Deleted* state. The failed stack might be available there. The AWS CloudFormation stack contains the
conformance pack name. If you find the failed stack, choose the **Events** tab of the AWS CloudFormation stack and check for failed events.

If none of these steps worked and if the failure reason is an internal service error, then try operation again or contact AWS Config support.
Multi-Account Multi-Region Data Aggregation

An aggregator is an AWS Config resource type that collects AWS Config configuration and compliance data from the following:

- Multiple accounts and multiple regions.
- Single account and multiple regions.
- An organization in AWS Organizations and all the accounts in that organization.

Use an aggregator to view the resource configuration and compliance data recorded in AWS Config.

For more information about concepts, see Multi-Account Multi-Region Data Aggregation (p. 5) section in the Concepts topic.

To collect your AWS Config data from source accounts and regions, start with:

1. Adding an aggregator to aggregate AWS Config configuration and compliance data from multiple accounts and regions.
2. Authorizing aggregator accounts to collect AWS Config configuration and compliance data.
   Authorization is required when your source accounts are individual accounts. Authorization is not required if you are aggregating source accounts that are part of AWS Organizations.
3. Monitoring compliance data for rules and accounts in the aggregated view.

Region Support

Currently, multi-account multi-region data aggregation is supported in the following regions:

<table>
<thead>
<tr>
<th>Region Name</th>
<th>Region</th>
<th>Endpoint</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>ap-south-1</td>
<td>config.ap-south-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
</tbody>
</table>
## Region Support

<table>
<thead>
<tr>
<th>Region Name</th>
<th>Region</th>
<th>Endpoint</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>ap-northeast-2</td>
<td>config.ap-northeast-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>ap-southeast-1</td>
<td>config.ap-southeast-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>ap-southeast-2</td>
<td>config.ap-southeast-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>ap-northeast-1</td>
<td>config.ap-northeast-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>AWS GovCloud (US-East)</td>
<td>us-gov-east-1</td>
<td>config.us-gov-east-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>AWS GovCloud (US-West)</td>
<td>us-gov-west-1</td>
<td>config.us-gov-west-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>ca-central-1</td>
<td>config.ca-central-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Stockholm)</td>
<td>eu-north-1</td>
<td>config.eu-north-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>eu-central-1</td>
<td>config.eu-central-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>eu-west-1</td>
<td>config.eu-west-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (London)</td>
<td>eu-west-2</td>
<td>config.eu-west-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>Europe (Paris)</td>
<td>eu-west-3</td>
<td>config.eu-west-3.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>South America (São Paulo)</td>
<td>sa-east-1</td>
<td>config.sa-east-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US East (N. Virginia)</td>
<td>us-east-1</td>
<td>config.us-east-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US East (Ohio)</td>
<td>us-east-2</td>
<td>config.us-east-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US West (N. California)</td>
<td>us-west-1</td>
<td>config.us-west-1.amazonaws.com</td>
<td>HTTPS</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>us-west-2</td>
<td>config.us-west-2.amazonaws.com</td>
<td>HTTPS</td>
</tr>
</tbody>
</table>
Learn More

- AWS Config Concepts (p. 2)
- Viewing Configuration and Compliance Data in the Aggregated View (p. 230)
- Setting Up an Aggregator Using the Console (p. 231)
- Setting Up an Aggregator Using the AWS Command Line Interface (p. 233)
- Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the Console (p. 238)
- Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the AWS Command Line Interface (p. 240)
- Troubleshooting for Multi-Account Multi-Region Data Aggregation (p. 241)

Viewing Configuration and Compliance Data in the Aggregated View

The Aggregated view displays the configuration data of AWS resources and provides an overview of your rules and their compliance state.

It provides the total resource count of AWS resources. The resource types and source accounts are ranked by the highest number of resources. It also provides a count of compliant and noncompliant rules. The noncompliant rules are ranked by highest number of noncompliant resources and source accounts with highest number of noncompliant rules.

After setup, AWS Config starts aggregating data from the specified source accounts into an aggregator. It might take a few minutes for AWS Config to display the compliance status of rules on this page.

Use the Aggregated View

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. In the navigation pane, choose Aggregated view, and then review your rules and their compliance states; AWS resources and their configuration data.

On the Aggregated view page, you can do the following:
Choose an aggregator from the **Aggregator** list.

Choose the region from the **Region** list. By default, **All regions** is selected.

Choose an account from the **Account** list. By default, **All accounts** is selected.

View the top ten resource types, in the descending order according to the number of resources. Choose view all resources to go to the **Aggregated resources** page. On this page, you can view all the aggregated resources for an account.

View the top five accounts by the number of resources, in the descending order according to the number of resources. Choose the number of resources for an account to go to the **Aggregated Resources** page. On this page, you can view all the aggregated resources for an account.

View the top five noncompliant rules, in descending order according to the number of noncompliant resources. Choose a rule to go to the **Rule details** page.

View the top five accounts by noncompliant rules, in descending order according to the number of noncompliant rules. Choose an account to go to the **Aggregated Rules** page. On this page, you can view all the aggregated rules for an account.

**Note**

Data displayed on the tiles is subject to delays.

The **Data collection from all source accounts and regions is incomplete** message is displayed in the aggregated view for the following reasons:

- AWS Config noncompliant rules and configuration data of AWS resources transfer is in progress.
- AWS Config can't find rules to match the filter. Select the appropriate account or region and try again.

The **Data collection from your organization is incomplete. You can view the below data only for 24 hours.** message is displayed in the aggregated view for the following reasons:

- AWS Config is unable to access your organization details due to invalid IAM role. If the IAM role is invalid for more than 24 hours, AWS Config deletes data for entire organization.
- AWS Config service access is disabled in your organization.

**Learn More**

- AWS Config Concepts (p. 2)
- Viewing Configuration and Compliance Data in the Aggregated View (p. 230)
- Setting Up an Aggregator Using the Console (p. 231)
- Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the Console (p. 238)
- Troubleshooting for Multi-Account Multi-Region Data Aggregation (p. 241)

**Setting Up an Aggregator Using the Console**

On the **Aggregator** page, you can do the following:

- Create an aggregator by specifying the source account IDs or organization and regions from which you want to aggregate data.
- Edit and delete an aggregator.
Add an Aggregator

1. Sign in to the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Navigate to the Aggregators page and choose Add aggregator.
3. Allow data replication, gives permission to AWS Config to replicate data from the source accounts into an aggregator account.

   Choose Allow AWS Config to replicate data from source account(s) into an aggregator account. You must select this checkbox to continue to add an aggregator.
4. For Aggregator name, type the name for your aggregator.

   The aggregator name must be a unique name with a maximum of 64 alphanumeric characters. The name can contain hyphens and underscores.
5. For Select source accounts, either choose Add individual account IDs or Add my organization from which you want to aggregate data.

   • If you choose Add individual account IDs, you can add individual account IDs for an aggregator account.
     1. Choose Add source accounts to add account IDs.
     2. Choose Add AWS account IDs to manually add comma-separated AWS account IDs. If you want to aggregate data from the current account, type the account ID of the account.

     OR

     Choose Upload a file to upload a file (.txt or .csv) of comma-separated AWS account IDs.
     3. Choose Add source accounts to confirm your selection.

   • If you choose Add my organization, you can add all accounts in your organization to an aggregator account.

     Note

     You must be signed in to the master account and all features must be enabled in your organization. This option automatically enables the integration between AWS Config and AWS Organizations.
     1. Choose Choose IAM role to create an IAM role or choose an existing IAM role from your account.

     You must assign an IAM role to allow AWS Config to call read-only APIs for your organization.
     2. Choose Create a role and type the IAM role name to create IAM role.

     OR

     Choose Choose a role from your account to select an existing IAM role.

     Note

     In the IAM console, attach the AWSConfigRoleForOrganizations managed policy to your IAM role. Attaching this policy allows AWS Config to call AWS Organizations DescribeOrganization, ListAWSServiceAccessForOrganization, and
List Accounts APIs. You must edit the control policy document to include `config.amazonaws.com` trusted entity.

3. Choose **Choose IAM role** to confirm your selection.

6. For **Regions**, choose the regions for which you want to aggregate data.
   - Select one region or multiple regions or all the AWS regions.
   - Select **Include future AWS regions** to aggregate data from all future AWS regions where multi-account multi-region data aggregation is enabled.

7. Choose **Save**. AWS Config displays the aggregator.

**Edit an Aggregator**

1. To make changes to the aggregator, choose the aggregator name.
2. Choose **Actions** and then choose **Edit**.
3. Use the sections on the **Edit aggregator** page to change the source accounts, IAM roles, or regions for the aggregator.

   **Note**
   You cannot change source type from individual account(s) to organization and vice versa.

4. Choose **Save**.

**Delete an Aggregator**

1. To delete an aggregator, choose the aggregator name.
2. Choose **Actions** and then choose **Delete**.

   A warning message is displayed. Deleting an aggregator results in the loss of all aggregated data. You cannot recover this data but data in the source account(s) is not impacted.

3. Choose **Delete** to confirm your selection.

**Learn More**

- AWS Config Concepts (p. 2)
- Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the Console (p. 238)
- Viewing Configuration and Compliance Data in the Aggregated View (p. 230)
- Troubleshooting for Multi-Account Multi-Region Data Aggregation (p. 241)

**Setting Up an Aggregator Using the AWS Command Line Interface**

You can create, view, update, and delete AWS Config aggregator data using the AWS Command Line Interface (AWS CLI). To use the AWS Management Console, see Setting Up an Aggregator Using the Console (p. 231).

The AWS CLI is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and use scripts to automate them.
Add an Aggregator Using Individual Accounts

1. Open a command prompt or a terminal window.
2. Type the following command to create an aggregator named MyAggregator.

```bash
aws configservice put-configuration-aggregator --configuration-aggregator-name MyAggregator --account-aggregation-sources "[{"AccountIds": ["AccountID1", "AccountID2", "AccountID3"], "AllAwsRegions": true}]"
```

For `account-aggregation-sources`, type one of the following.

- A comma-separated list of AWS account IDs for which you want to aggregate data. Wrap the account IDs in square brackets, and be sure to escape quotation marks (for example, "[{"AccountIds": ["AccountID1", "AccountID2", "AccountID3"], "AllAwsRegions": true}]").
- You can also upload a JSON file of comma-separated AWS account IDs. Upload the file using the following syntax: `--account-aggregation-sources MyFilePath/MyFile.json`

The JSON file must be in the following format:

```json
[
  {
    "AccountIds": [
      "AccountID1",
      "AccountID2",
      "AccountID3"
    ],
    "AllAwsRegions": true
  }
]
```

3. Press Enter to execute the command.

You should see output similar to the following:

```json
{
  "ConfigurationAggregator": {
    "CreationTime": 1517942461.442,
    "ConfigurationAggregatorName": "MyAggregator",
    "AccountAggregationSources": [
```

---

To install the AWS CLI on your local machine, see Installing the AWS CLI in the AWS CLI User Guide.

If necessary, type `aws configure` to configure the AWS CLI to use an AWS Region where AWS Config aggregators are available.

---

**Topics**

- Add an Aggregator Using Individual Accounts (p. 234)
- Add an Aggregator Using AWS Organizations (p. 235)
- View an Aggregator (p. 235)
- Edit an Aggregator (p. 236)
- Delete an Aggregator (p. 237)
- Learn More (p. 231)
Add an Aggregator Using AWS Organizations

Before you begin this procedure, you must be signed in to the master account and all features must be enabled in your organization.

1. Open a command prompt or a terminal window.
2. Type the following command to create an aggregator named `MyAggregator`.

   ```
   aws configservice put-configuration-aggregator --configuration-aggregator-name MyAggregator --organization-aggregation-source "{"RoleArn": "Complete-Arn", "AllAwsRegions": true}"
   ```

3. Press Enter to execute the command.
   
   You should see output similar to the following:

   ```
   {
     "ConfigurationAggregator": {
       "CreationTime": 1517942461.442,
       "ConfigurationAggregatorName": "MyAggregator",
       "OrganizationAggregationSource": {
         "AllAwsRegions": true,
         "RoleArn": "arn:aws:config:Region:AccountID:config-aggregator/config-aggregator-floopus3"
       },
       "LastUpdatedTime": 1517942461.442
     }
   }
   ```

View an Aggregator

1. Type the following command:

   ```
   aws configservice describe-configuration-aggregators
   ```

2. Depending on your source account you should see output similar to the following:

   For individuals accounts

   ```
   {
     "ConfigurationAggregators": [
   ```
Edit an Aggregator

1. You can use the `put-configuration-aggregator` command to update or edit a configuration aggregator.

   Type the following command to add a new account ID to `MyAggregator`:

   ```
   aws configservice put-configuration-aggregator --configuration-aggregator-name MyAggregator --account-aggregation-sources "[{"AccountIds": [{"AccountID1"}, {"AccountID2"}, {"AccountID3"}], "AllAwsRegions": true}]
   ```

2. Depending on your source account you should see output similar to the following:

   For individuals accounts

   ```
   
   ```

OR

For an organization

```

```
Delete an Aggregator

To delete a configuration aggregator using the AWS CLI

- Type the following command:

```bash
aws configservice delete-configuration-aggregator --configuration-aggregator-name MyAggregator
```

If successful, the command executes with no additional output.

Learn More

- AWS Config Concepts (p. 2)
- Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the AWS Command Line Interface (p. 240)
- Viewing Configuration and Compliance Data in the Aggregated View (p. 230)
- Troubleshooting for Multi-Account Multi-Region Data Aggregation (p. 241)
Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the Console

AWS Config allows you to authorize aggregator accounts to collect AWS Config configuration and compliance data.

This flow is not required if you are aggregating source accounts that are part of AWS Organizations.

On the **Authorizations** page, you can do the following:

- Add Authorization to allow an aggregator account and region to collect AWS Config configuration and compliance data.
- Authorize a pending request from an aggregator account to collect AWS Config configuration and compliance data.
- Delete an authorization for an aggregator account.

**Topics**

- Add Authorization for Aggregator Accounts and Regions (p. 238)
- Authorize a Pending Request for an Aggregator Account (p. 239)
- Delete Authorization for an Existing Aggregator Account (p. 239)
- Learn More (p. 231)

**Add Authorization for Aggregator Accounts and Regions**

You can add authorization to grant permission to aggregator accounts and regions to collect AWS Config configuration and compliance data.

1. Sign into the AWS Management Console and open the AWS Config console at https://console.aws.amazon.com/config/.
2. Navigate to the **Authorizations** page and choose **Add authorization**.
3. For **Aggregator account**, type the 12-digit account ID of an aggregator account.
4. For **Aggregator region**, choose the AWS regions where aggregator account is allowed to collect AWS Config configuration and compliance data.
5. Choose **Add authorization** to confirm your selection.

AWS Config displays an aggregator account, region, and authorization status.

**Note**

You can also add authorization to aggregator accounts and regions programmatically using AWS CloudFormation sample template. For more information, see `AWS::Config::AggregationAuthorization` in the *AWS CloudFormation user guide*. 

---

238
Authorize a Pending Request for an Aggregator Account

If you have a pending authorization request from an existing aggregator account you will see the request status on the Authorizations page. You can authorize a pending request from this page.

1. For the aggregator account you want to authorize, choose Authorize in the Actions column.

A confirmation message is displayed to confirm you grant permission to an aggregator account and region for collecting AWS Config data.

2. Choose Authorize to grant this permission for an aggregator account and region.

The authorization status changes from Requesting for authorization to Authorized.

Delete Authorization for an Existing Aggregator Account

1. For the aggregator account you want to delete authorization, choose Delete in the Actions column.

A warning message is displayed. When you delete this authorization, AWS Config data is not shared with an aggregator account.

Note
After authorization for an aggregator is deleted the data will remain in the aggregator account for up to 24 hours before being deleted.

2. Choose Delete to confirm your selection.

The aggregator account is deleted.
You can authorize aggregator accounts to collect AWS Config data from source accounts and delete aggregator accounts using the AWS Command Line Interface (AWS CLI). To use the AWS Management Console, see Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the Console (p. 238).

The AWS CLI is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and use scripts to automate them.

To install the AWS CLI on your local machine, see Installing the AWS CLI in the AWS CLI User Guide.

If necessary, type `aws configure` to configure the AWS CLI to use an AWS Region where AWS Config aggregators are available.

**Topics**

- Add Authorization for Aggregator Accounts and Regions (p. 240)
- Delete an Authorization Account (p. 241)
- Learn More (p. 231)

**Add Authorization for Aggregator Accounts and Regions**

1. Open a command prompt or a terminal window.
2. Type the following command:

   ```bash
   aws configservice put-aggregation-authorization --authorized-account-id AccountID --authorized-aws-region Region
   ```

3. Press Enter.

   You should see output similar to the following:

   ```json
   {
   "AggregationAuthorization": {
   "AuthorizedAccountId": "AccountID",
   }"```
Delete an Authorization Account

To delete an authorized account using the AWS CLI

- Type the following command:

```
aws configservice delete-aggregation-authorization --authorized-account-id AccountID --authorized-aws-region Region
```

If successful, the command executes with no additional output.

Learn More

- AWS Config Concepts (p. 2)
- Setting Up an Aggregator Using the AWS Command Line Interface (p. 233)
- Viewing Configuration and Compliance Data in the Aggregated View (p. 230)
- Troubleshooting for Multi-Account Multi-Region Data Aggregation (p. 241)

Troubleshooting for Multi-Account Multi-Region Data Aggregation

AWS Config might not aggregate data from source accounts for one of the following reasons:

<table>
<thead>
<tr>
<th>If this happens</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config is not enabled in the source account.</td>
<td>Enable AWS Config in the source account and authorize the aggregator account to collect data.</td>
</tr>
<tr>
<td>Authorization is not granted to an aggregator account.</td>
<td>Sign in to the source account and grant authorization to the aggregator account to collect AWS Config data.</td>
</tr>
<tr>
<td>There might be a temporary issue that is preventing data aggregation.</td>
<td>Data aggregation is subject to delays. Wait for a few minutes.</td>
</tr>
</tbody>
</table>

AWS Config might not aggregate data from an organization for one of the following reasons:

<table>
<thead>
<tr>
<th>If this happens</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config is unable to access your organization details due to invalid IAM role.</td>
<td>Create an IAM role or select a valid IAM role from the IAM role list.</td>
</tr>
<tr>
<td></td>
<td>Note If the IAM role is invalid for more than 24 hours, AWS Config deletes data for entire organization.</td>
</tr>
<tr>
<td>If this happens</td>
<td>Do this</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AWS Config service access is disabled in your organization.</td>
<td>You can enable integration between AWS Config and AWS Organizations through the <code>EnableAWSServiceAccess</code> API. If you choose <em>Add my organization</em> in console, AWS Config automatically enables the integration between AWS Config and AWS Organizations.</td>
</tr>
<tr>
<td>AWS Config is unable to access your organization details because all features is not enabled in your organization.</td>
<td><em>Enable all features</em> in AWS Organizations console.</td>
</tr>
</tbody>
</table>

**Learn More**

- [AWS Config Concepts](#) (p. 2)
- [Setting Up an Aggregator Using the Console](#) (p. 231)
- [Authorizing Aggregator Accounts to Collect AWS Config Configuration and Compliance Data Using the Console](#) (p. 238)
- [Viewing Configuration and Compliance Data in the Aggregated View](#) (p. 230)
Monitoring

You can use other AWS services to monitor AWS Config resources.

- You can use 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.
- You can use Amazon CloudWatch Events to detect and react to changes in the status of AWS Config events.

Topics
- Monitoring AWS Resource Changes with Amazon SQS (p. 243)
- Monitoring AWS Config with Amazon CloudWatch Events (p. 244)

Monitoring AWS Resource Changes with Amazon SQS

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-2:444455556666:queue1.

```json
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement":
   {
```
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-2:111122223333:test-topic to perform any actions on the queue named arn:aws:sqs:us-east-2:111122223333:test-topic-queue.

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

```
"Sid":"Queue1_SendMessage",
"Effect": "Allow",
"Principal": {
  "AWS": ["111122223333","444455556666"]
},
"Action": "sqs:SendMessage",
"Resource": "arn:aws:sqs:us-east-2:444455556666:queue1"
}
```

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.

Monitoring AWS Config with Amazon CloudWatch Events

Amazon CloudWatch Events delivers a near real-time stream of system events that describe changes in AWS resources. Use Amazon CloudWatch Events to detect and react to changes in the status of AWS Config events.

You can create a rule that runs whenever there is a state transition, or when there is a transition to one or more states that are of interest. Then, based on rules you create, Amazon CloudWatch Events invokes one or more target actions when an event matches the values you specify in a rule. Depending on the type of event, you might want to send notifications, capture event information, take corrective action, initiate events, or take other actions.

Before you create event rules for AWS Config, however, you should do the following:

- Familiarize yourself with events, rules, and targets in CloudWatch Events. For more information, see What Is Amazon CloudWatch Events?
For more information about how to get started with CloudWatch Events and set up rules, see Getting Started with CloudWatch Events.

Create the target or targets you will use in your event rules.

**Topics**

- Amazon CloudWatch Events format for AWS Config (p. 245)
- Creating Amazon CloudWatch Events Rule for AWS Config (p. 245)

## Amazon CloudWatch Events format for AWS Config

The CloudWatch event for AWS Config has the following format:

```json
{
  "version":"0",
  "id":"cd4d811e-ab12-322b-8255-872ce65b1bc8",
  "detail-type": "event type",
  "source": "aws.config",
  "account": "111122223333",
  "time": "2018-03-22T00:38:11Z",
  "region": "us-east-1",
  "resources": ["resources"],
  "detail": {"specific message type"}
}
```

## Creating Amazon CloudWatch Events Rule for AWS Config

Use the following steps to create a CloudWatch Events rule that triggers on an event emitted by AWS Config.

2. In the navigation pane, choose Events.
3. Choose Create rule.
4. On the Step 1: Create rule page, for Service Name, choose Config.
5. For Event Type, choose the event type that triggers the rule:
   - Choose All Events to make a rule that applies to all AWS services. If you choose this option, you cannot choose specific message types, rule names, resource types, or resource IDs.
   - Choose AWS API Call via CloudTrail to base rules on API calls made to this service. For more information about creating this type of rule, see Creating a CloudWatch Events Rule That Is Triggered on an AWS API Call Using AWS CloudTrail.
   - Choose Config Configuration Item Change to get notifications when a resource in your account changes.
   - Choose Config Rules Compliance Change to get notifications when a compliance check to your rules fails.
   - Choose Config Rules Re-evaluation Status to get reevaluation status notifications.
   - Choose Config Configuration Snapshot Delivery Status to get configuration snapshot delivery status notifications.
   - Choose Config Configuration History Delivery Status to get configuration history delivery status notifications.
6. Choose **Any message type** to receive notifications of any type. Choose **Specific message type(s)** to receive the following types of notifications:

- If you choose **ConfigurationItemChangeNotification**, you receive messages when AWS Config successfully delivers the configuration snapshot to your Amazon S3 bucket.
- If you choose **ComplianceChangeNotification**, you receive messages when the compliance type of a resource that AWS Config evaluates has changed.
- If you choose **ConfigRulesEvaluationStarted**, you receive messages when AWS Config starts evaluating your rule against the specified resources.
- If you choose **ConfigurationSnapshotDeliveryCompleted**, you receive messages when AWS Config successfully delivers the configuration snapshot to your Amazon S3 bucket.
- If you choose **ConfigurationSnapshotDeliveryFailed**, you receive messages when AWS Config fails to deliver the configuration snapshot to your Amazon S3 bucket.
- If you choose **ConfigurationSnapshotDeliveryStarted**, you receive messages when AWS Config starts delivering the configuration snapshot to your Amazon S3 bucket.
- If you choose **ConfigurationHistoryDeliveryCompleted**, you receive messages when AWS Config successfully delivers the configuration history to your Amazon S3 bucket.

7. If you chose a specific event type from the **Event Type** drop-down list, choose **Any resource type** to make a rule that applies to all AWS Config supported resource types.

Or choose **Specific resource type(s)**, and then type the AWS Config supported resource type (for example, `AWS::EC2::Instance`).

8. If you chose a specific event type from the **Event Type** drop-down list, choose **Any resource ID** to include any AWS Config supported resource ID.

Or choose **Specific resource ID(s)**, and then type the AWS Config supported resource ID (for example, `i-04606de676e635647`).

9. If you chose a specific event type from the **Event Type** drop-down list, choose **Any rule name** to include any AWS Config supported rule.

Or choose **Specific rule name(s)**, and then type the AWS Config supported rule (for example, `required-tags`).

10. Review your rule setup to make sure it meets your event-monitoring requirements.

11. In the **Targets** area, choose Add target*.

12. In the **Select target type** list, choose the type of target you have prepared to use with this rule, and then configure any additional options required by that type.

13. Choose **Configure details**.

14. On the **Configure rule details** page, type a name and description for the rule, and then choose the **State** box to enable the rule as soon as it is created.

15. Choose **Create rule** to confirm your selection.
Service-Linked AWS Config Rules

A service-linked AWS Config rule is a unique type of managed config rule that supports other AWS services to create AWS Config rules in your account. The service-linked AWS Config rules are predefined to include all the permissions required to call other AWS services on your behalf. These rules are similar to standards that an AWS service recommends in your AWS account for compliance verification.

These service-linked AWS Config rules are owned by AWS service teams. The AWS service team creates these rules in your AWS account. You have read-only access to these rules. You cannot edit or delete these rules if you are subscribed to AWS service that these rules are linked to.

In the AWS Config console, the service-linked AWS Config rules are visible in the Rules page. The edit button is greyed in the console thereby restricting you to edit the rule. You can view details of the rule by choosing the rule. On the rule details page, you can view the name of the service that created the rule. The Edit and Delete results is greyed thereby restricting you to edit and delete results of the rule. To edit or delete the rule, contact the AWS service that created the rule.

While using the AWS Command Line Interface, the PutConfigRule, DeleteConfigRule, and DeleteEvaluationResults APIs return access denied with the following error message:

INSUFFICIENT_SLCR_PERMISSIONS = "An AWS service owns ServiceLinkedConfigRule. You do not have permissions to take action on this rule."
Using Service-Linked Roles for AWS Config

AWS Config uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS Config. Service-linked roles are predefined by AWS Config and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS Config easier because you don’t have to manually add the necessary permissions. AWS Config defines the permissions of its service-linked roles, and unless defined otherwise, only AWS Config can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have Yes in the Service-Linked Role column. Choose a Yes with a link to view the service-linked role documentation for that service.

Service-Linked Role Permissions for AWS Config

AWS Config uses the service-linked role named AWSServiceRoleForConfig – AWS Config uses this service-linked role to call other AWS services on your behalf.

The AWSServiceRoleForConfig service-linked role trusts the config.amazonaws.com service to assume the role.

The permissions policy for the AWSServiceRoleForConfig role contains read-only and write-only permissions on the AWS Config resources and read-only permissions for resources in other services that AWS Config supports. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the IAM User Guide.

To use a service-linked role with AWS Config, you must configure permissions on your Amazon S3 bucket and Amazon SNS topic. For more information, see Required Permissions for the Amazon S3 Bucket When Using Service-Linked Roles (p. 103) and Required Permissions for the Amazon SNS Topic When Using Service-Linked Roles (p. 105).

Creating a Service-Linked Role for AWS Config

In the IAM CLI or the IAM API, create a service-linked role with the config.amazonaws.com service name. For more information, see Creating a Service-Linked Role in the IAM User Guide. If you delete this service-linked role, you can use this same process to create the role again.

Editing a Service-Linked Role for AWS Config

AWS Config does not allow you to edit the AWSServiceRoleForConfig service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might
Deleting a Service-Linked Role for AWS Config

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don’t have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

**Note**
If the AWS Config service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

**To delete AWS Config resources used by the AWSServiceRoleForConfig**

Ensure that you do not have ConfigurationRecorders using the service-linked role. You can use the AWS Config console to stop the configuration recorder. To stop recording, under **Recording is on**, choose **Turn off**.

You can delete the ConfigurationRecorder using AWS Config API. To delete, use the `delete-configuration-recorder` command.

```bash
# aws configservice delete-configuration-recorder --configuration-recorder-name default
```

**To manually delete the service-linked role using IAM**

Use the IAM console, the IAM CLI, or the IAM API to delete the AWSServiceRoleForConfig service-linked role. For more information, see *Deleting a Service-Linked Role* in the *IAM User Guide*. 
Using AWS Config with Interface VPC Endpoints

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and AWS Config. You can use this connection to communicate with AWS Config from your VPC without going through the public internet.

Amazon VPC is an AWS service that you can use to launch AWS resources in a virtual network that you define. With a VPC, you have control over your network settings, such as IP address range, subnets, route tables, and network gateways. Interface VPC endpoints are powered by AWS PrivateLink, an AWS technology that enables private communication between AWS services using an elastic network interface with private IP addresses. To connect your VPC to AWS Config, you define an interface VPC endpoint for AWS Config. This type of endpoint enables you to connect your VPC to AWS services. The endpoint provides reliable, scalable connectivity to AWS Config without requiring an internet gateway, network address translation (NAT) instance, or VPN connection. For more information, see What is Amazon VPC in the Amazon VPC User Guide.

The following steps are for users of Amazon VPC. For more information, see Getting Started in the Amazon VPC User Guide.

Availability

AWS Config currently supports VPC endpoints in the following regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Asia Pacific (Mumbai)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- South America (São Paulo)

Create a VPC Endpoint for AWS Config

To start using AWS Config with your VPC, create an interface VPC endpoint for AWS Config. You do not need to change the settings for AWS Config. AWS Config calls other AWS services using their public endpoints. For more information, see Creating an Interface Endpoint in the Amazon VPC User Guide.
Logging AWS Config API Calls with AWS CloudTrail

AWS Config is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Config. CloudTrail captures all API calls for AWS Config as events. The calls captured include calls from the AWS Config console and code calls to the AWS Config API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Config. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in Event history. Using the information collected by CloudTrail, you can determine the request that was made to AWS Config, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS Config Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Config, that activity is recorded in a CloudTrail event along with other AWS service events in Event history. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS Config, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All AWS Config operations are logged by CloudTrail and are documented in the AWS Config API Reference. For example, calls to the DeliverConfigSnapshot, DeleteDeliveryChannel, and DescribeDeliveryChannels operations generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.
Understanding AWS Config Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

Example Log Files

For examples of the CloudTrail log entries, see the following topics.

Contents
- `DeleteDeliveryChannel` (p. 252)
- `DeliverConfigSnapshot` (p. 253)
- `DescribeConfigurationRecorderStatus` (p. 253)
- `DescribeConfigurationRecorders` (p. 254)
- `DescribeDeliveryChannels` (p. 254)
- `GetResourceConfigHistory` (p. 255)
- `PutConfigurationRecorder` (p. 255)
- `PutDeliveryChannel` (p. 256)
- `StartConfigurationRecorder` (p. 256)
- `StopConfigurationRecorder` (p. 257)

DeleteDeliveryChannel

The following is an example CloudTrail log file for the `DeleteDeliveryChannel` operation.

```json
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAACKCEVSQ6C2EXAMPLE",
        "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,
    "recipientAccountID": "222222222222"
}
```
DeliverConfigSnapshot

The following is an example CloudTrail log file for the DeliverConfigSnapshot operation.

```
{
    "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": "58d50f10-212d-4fa4-842e-97c614da67ce",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

DescribeConfigurationRecorderStatus

The following is an example CloudTrail log file for the DescribeConfigurationRecorderStatus operation.

```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::222222222222:user/JohnDoe",
        "accountId": "222222222222",
        "accessKeyId": "AKIAI44QH8DHBEEXAMPLE",
        "userName": "JohnDoe"
    },
    "eventTime": "2014-12-11T18:35:44Z",
    "eventSource": "config.amazonaws.com",
    "eventName": "DescribeConfigurationRecorderStatus",
    "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": {
        "configurationRecorderName": "example-recorder"
    },
    "responseElements": {
        "configurationRecorderStatus": {
            "configurationRecorderName": "example-recorder",
            "status": "ENABLED",
            "statusMessage": "Configuration Recorder is enabled with delivery enabled.",
            "awsRegion": "us-west-2"
        }
    }
}
```
DescribeConfigurationRecorders

The following is an example CloudTrail log file for the DescribeConfigurationRecorders operation.

{
    "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-6e1ee8a8b5",
    "eventType": "AwsApiCall",
    "recipientAccountId": "222222222222"
}

DescribeDeliveryChannels

Following is an example CloudTrail log file for the DescribeDeliveryChannels operation.

{
    "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": "6566b55c-8164-11e4-ab4f-657c7ab282ab",
    "eventID": "6259a9ad-889e-423b-beeb-6e1ee8a8b5",
    "eventType": "AwsApiCall",
    "recipientAccountId": "222222222222"
}
GetResourceConfigHistory

The following is an example CloudTrail log file for the GetResourceConfigHistory operation.

```
{
  "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-b4c6-3fbb87dd8166",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111111111111"
}
```

PutConfigurationRecorder

The following is an example CloudTrail log file for the PutConfigurationRecorder operation.

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::111111111111:user/methuselah",
    "accountId": "111111111111",
    "accessKeyId": "OAIADASDFKJHGFKJOPQ",
    "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": "PutConfigurationRecorder",
  "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": {
    "configurationRecorderName": "configuration recorder name",
    "roleArn": "arn:aws:iam::111111111111:role/ empresas",
    "status": "ENABLED"
  },
  "responseElements": null,
  "requestID": "d9f3490d-80d0-11e4-9f1c-7739d36a3df2",
  "eventID": "ba9c1766-d28f-40e3-b4c6-3fbb87dd8166",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111111111111"
}
```
PutDeliveryChannel

The following is an example CloudTrail log file for the PutDeliveryChannel operation.

```
{
    "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",
        }
    },
    "responseElements": null,
    "requestID": "268b8d4d-8164-11e4-ab4f-657c7ab282ab",
    "eventID": "b2db05f1-1c73-4e52-b238-db69c04e8dd4",
    "eventType": "AwsApiCall",
    "recipientAccountId": "222222222222"
}
```

StartConfigurationRecorder

The following is an example CloudTrail log file for the StartConfigurationRecorder operation.

```
{
    "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",
        }
    },
    "responseElements": null,
    "requestID": "268b8d4d-8164-11e4-ab4f-657c7ab282ab",
    "eventID": "b2db05f1-1c73-4e52-b238-db69c04e8dd4",
    "eventType": "AwsApiCall",
    "recipientAccountId": "222222222222"
}
```

256
StopConfigurationRecorder

The following is an example CloudTrail log file for the StopConfigurationRecorder operation.

```json
{
    "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"
}
```
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.
- **AWS Config Pricing**
- **Technical FAQ**
- **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.

- **Classes & Workshops** – 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, SDKs, IDE toolkits, and command line tools for developing and managing AWS applications.
- **AWS Whitepapers** – Links to a comprehensive list of technical AWS whitepapers, covering topics such as architecture, security, and economics and authored by AWS Solutions Architects or other technical experts.
- **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.

AWS Software Development Kits for AWS Config

An AWS software development kit (SDK) makes it easier to build applications that access 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. The following table lists the available SDKs and third-party libraries you can use to access AWS Config programmatically.

<table>
<thead>
<tr>
<th>Type of Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDKs</td>
<td>AWS provides the following SDKs:</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for C++ Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS Mobile SDK for iOS Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for Go Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for Java Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for JavaScript Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for .NET Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for PHP Documentation</td>
</tr>
<tr>
<td>Type of Access</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for Python (Boto) Documentation</td>
</tr>
<tr>
<td></td>
<td>• AWS SDK for Ruby Documentation</td>
</tr>
<tr>
<td>Third-party libraries</td>
<td>Developers in the AWS developer community also provide their own libraries, which you can find at the following AWS developer centers:</td>
</tr>
<tr>
<td></td>
<td>• AWS Java Developer Center</td>
</tr>
<tr>
<td></td>
<td>• AWS JavaScript Developer Center</td>
</tr>
<tr>
<td></td>
<td>• AWS PHP Developer Center</td>
</tr>
<tr>
<td></td>
<td>• AWS Python Developer Center</td>
</tr>
<tr>
<td></td>
<td>• AWS Ruby Developer Center</td>
</tr>
<tr>
<td></td>
<td>• AWS Windows and .NET Developer Center</td>
</tr>
</tbody>
</table>
### Document History

The following table describes the documentation release history of AWS Config.

- **API version**: 2014-11-12
- **Latest documentation update**: March 6, 2020

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config supports Amazon SNS resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to your Amazon SNS topic. For more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](p. 9).</td>
<td>March 6, 2020</td>
</tr>
<tr>
<td>Multi-Account Multi-Region Data Aggregation Region support</td>
<td>With this release, multi-account multi-region data aggregation is now supported in Europe (Stockholm) Region. For more information, see [Multi-Account Multi-Region Data Aggregation](p. 228).</td>
<td>March 5, 2020</td>
</tr>
<tr>
<td>Advanced queries Region support</td>
<td>With this release, advanced queries is now supported in Europe (Stockholm) Region.</td>
<td>March 5, 2020</td>
</tr>
<tr>
<td>AWS Config allows you to run advanced queries with configuration aggregators</td>
<td>With this release, AWS Config adds support to run advanced queries based on resource configuration properties with configuration aggregators, enabling you to run the same queries across multiple accounts and Regions. For more information, see [Querying the Current Configuration State of AWS Resources](p. 66). With this release, AWS Config adds <code>SelectAggregateResourceConfig</code> API. For more information, see <code>SelectAggregateResourceConfig</code> in the <a href="#">AWS Config API Reference</a>.</td>
<td>February 28, 2020</td>
</tr>
<tr>
<td>AWS Config supports Amazon SQS resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to your Amazon SQS queue. For more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](p. 9).</td>
<td>February 13, 2020</td>
</tr>
<tr>
<td>AWS CloudFormation support for</td>
<td>With this release, AWS CloudFormation support for the following resources was</td>
<td>February 13, 2020</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Conformance Packs</td>
<td>added: AWS::Config::ConformancePack and OrganizationConformancePack.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- AWS::Config::ConformancePack</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use the AWS::Config::ConformancePack resource to create a Conformance Pack that is a collection of AWS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Config rules that can be easily deployed in an account and a Region and across AWS Organization.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- AWS::Config::OrganizationConformancePack</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use the AWS::Config::OrganizationConformancePack resource to create an Organization Conformance Pack</td>
<td></td>
</tr>
<tr>
<td></td>
<td>that has information about conformance packs that AWS Config creates in the member accounts.</td>
<td></td>
</tr>
<tr>
<td>AWS Config updates managed rules</td>
<td>With this release, AWS Config supports the following managed rules:</td>
<td>December 20, 2019</td>
</tr>
<tr>
<td></td>
<td>- api-gw-execution-logging-enabled (p. 118)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- ec2-stopped-instance (p. 131)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- elasticache-redis-cluster-automatic-backup-check (p. 138)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- emr-master-no-public-ip (p. 140)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- guardduty-non-archived-findings (p. 146)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- rds-enhanced-monitoring-enabled (p. 155)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- s3-account-level-public-access-blocks (p. 160)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- sagemaker-endpoint-configuration-kms-key-configured (p. 168)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- service-vpc-endpoint-enabled (p. 169)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td><strong>Record Configurations for Custom Resource Types</strong></td>
<td>With this release, AWS Config introduces support to record configurations for custom resource types. You can publish the configuration data of third-party resources into AWS Config and view and monitor the resource inventory and configuration history using AWS Config console and APIs. For more information, see Record Configurations for Third-Party Resources (p. 34). For more information about APIs, see the AWS Config API Reference:  - DeleteResourceConfig  - PutResourceConfig</td>
<td>November 20, 2019</td>
</tr>
<tr>
<td><strong>Conformance packs</strong></td>
<td>With this release, AWS Config introduces conformance packs. Conformance packs enable you to package a collection of AWS Config rules and remediation actions that can then be deployed together as a single entity across an entire AWS Organization. For more information, see Conformance Packs (p. 200). For more information about APIs, see the AWS Config API Reference:  - DeleteOrganizationConformancePack  - DeleteOrganizationConformancePack  - DescribeConformancePacks  - DescribeConformancePacks  - DescribeConformancePackStatus  - DescribeOrganizationConformancePacks  - DescribeOrganizationConformancePackStatuses  - GetConformancePackComplianceDetails  - GetConformancePackComplianceSummary  - GetOrganizationConformancePackDetailedStatus,  - PutConformancePack  - PutOrganizationConformancePack</td>
<td>November 19, 2019</td>
</tr>
<tr>
<td><strong>AWS Config supports Amazon Elasticsearch Service and AWS Key Management Service resource types</strong></td>
<td>With this release, you can use AWS Config to record configuration changes to your Amazon Elasticsearch Service domain and AWS Key Management Service key. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>November 11, 2019</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| AWS Config updates managed rules | With this release, AWS Config supports the following managed rules:  
  - dms-replication-not-public (p. 129)  
  - emr-kerberos-enabled (p. 140)  
  - fms-security-group-audit-policy-check (p. 141)  
  - fms-security-group-content-check (p. 142)  
  - fms-security-group-resource-association-check (p. 143)  
  - internet-gateway-authorized-vpc-only (p. 151)  
  - kms-cmk-not-scheduled-for-deletion (p. 151)  
  - sagemaker-notebook-no-direct-internet-access (p. 169)  
  - sagemaker-notebook-kms-configured (p. 168)  
  - shield-drt-access (p. 170) | October 10, 2019 |
| AWS Config supports Amazon RDS resource type | With this release, you can use AWS Config to record configuration changes to your Amazon Relational Database Service (Amazon RDS) DBCluster and DBClusterSnapshot.  
For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | September 17, 2019 |
| AWS Config supports Amazon QLDB resource type | With this release, you can use AWS Config to record configuration changes to Amazon Quantum Ledger Database (QLDB) ledger resource type.  
For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | September 10, 2019 |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config allows you to apply auto remediation on noncompliant resources as evaluated by AWS Config Rules</td>
<td>With this release, AWS Config introduces support to apply auto remediation using AWS Systems Manager automation documents on noncompliant resources as evaluated by AWS Config Rules. For more information, see Remediating Noncompliant AWS Resources by AWS Config Rules (p. 194).</td>
<td>September 5, 2019</td>
</tr>
<tr>
<td>AWS Config updates managed rules</td>
<td>With this release, AWS Config supports the following managed rules:</td>
<td>August 22, 2019</td>
</tr>
<tr>
<td></td>
<td>• alb-http-to-https-redirection-check (p. 117)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• api-gw-cache-enabled-and-encrypted (p. 118)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• api-gw-endpoint-type-check (p. 118)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• cloudtrail-s3-dataevents-enabled (p. 123)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• cloudwatch-log-group-encrypted (p. 125)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ebs-snapshot-public-restorable-check (p. 131)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• elb-deletion-protection-enabled (p. 139)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• lambda-concurrency-check (p. 152)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• lambda-dlq-check (p. 152)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• lambda-inside-vpc (p. 153)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• shield-advanced-enabled-autorenew (p. 170)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• vpc-vpn-2-tunnels-up (p. 171)</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>AWS Config updates managed rules</td>
<td>With this release, AWS Config updates the following managed rules:</td>
<td>July 31, 2019</td>
</tr>
<tr>
<td></td>
<td>• cloudfront-viewer-policy-https (p. 121)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2-instance-no-public-ip (p. 132)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2-security-group-attached-to-eni (p. 135)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• efs-encrypted-check (p. 136)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• elasticsearch-encrypted-at-rest (p. 137)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• elasticsearch-in-vpc-only (p. 137)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• redshift-cluster-public-access-check (p. 157)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• vpc-sg-open-only-to-authorized-ports (p. 171)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>AWS Config supports Amazon EC2 resource types</td>
<td>With this release, you can use AWS Config to record configuration changes to the following Amazon EC2 resources; VPC Endpoint, VPC Endpoint Service, and VPC Peering Connection.</td>
<td>July 12, 2019</td>
</tr>
<tr>
<td></td>
<td>For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td></td>
</tr>
<tr>
<td>AWS Config allows you to manage AWS Config rules across all AWS accounts within an organization</td>
<td>With this release, AWS Config introduces support for managing AWS Config rules across all AWS accounts within an organization. You can centrally create, update, and delete AWS Config rules across all accounts in your organization. For more information, see Enabling AWS Config Rules Across All Accounts in Your Organization (p. 194).</td>
<td>July 9, 2019</td>
</tr>
<tr>
<td></td>
<td>For more information about APIs, see the AWS Config API Reference:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• PutOrganizationConfigRule</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• DescribeOrganizationConfigRules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• GetOrganizationConfigRuleDetailedStatus</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• DescribeOrganizationConfigRuleStatuses</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• DeleteOrganizationConfigRule</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>AWS Config supports Amazon S3 and Amazon EC2 resource types</td>
<td>With this release, you can use AWS Config to record configuration changes to the Amazon S3 AccountPublicAccessBlock resource and the following Amazon EC2 resources; NatGateway, EgressOnlyInternetGateway, and FlowLog. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>May 17, 2019</td>
</tr>
<tr>
<td>AWS Config updates managed rules</td>
<td>With this release, AWS Config updates the following managed rules:</td>
<td>May 7, 2019</td>
</tr>
<tr>
<td></td>
<td>• s3-bucket-public-read-prohibited (p. 165)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• s3-bucket-public-write-prohibited (p. 165)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>AWS Config allows you to delete a remediation action using AWS Console.</td>
<td>With this release, AWS Config introduces support to delete a remediation action using AWS Management Console. For more information, see Remediating Noncompliant AWS Resources by AWS Config Rules (p. 194).</td>
<td>April 24, 2019</td>
</tr>
<tr>
<td>AWS Config supports new managed rules</td>
<td>This release supports a new managed rule: fms-shield-resource-policy-check (p. 144).</td>
<td>April 7, 2019</td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>AWS Config supports Amazon API Gateway resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to the following Amazon API Gateway resources: Api (WebSocket API), RestApi (REST API), Stage (WebSocket API stage), and Stage (REST API stage). For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>March 20, 2019</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>AWS Config allows you to run advanced queries</td>
<td>With this release, AWS Config adds support to run advanced queries based on resource configuration properties. For more information, see [Querying the Current Configuration State of AWS Resources](p. 66). With this release, AWS Config adds <code>SelectResourceConfig</code> API. For more information, see <code>SelectResourceConfig</code> in the AWS Config API Reference:</td>
<td>March 19, 2019</td>
</tr>
<tr>
<td>AWS Config allows you to assign tags your AWS Config resources</td>
<td>With this release, AWS Config introduces support for tag based access control for three AWS Config resources—ConfigRule, ConfigurationAggregator, and AggregationAuthorization. For more information, see [Tagging Your AWS Config Resources](p. 198). With this release, you can add, remove or list tags from your AWS Config resources using the following APIs. For more information, see the AWS Config API Reference:</td>
<td>March 14, 2019</td>
</tr>
<tr>
<td>• ListTagsForResource</td>
<td>• TagResource</td>
<td>• UntagResource</td>
</tr>
<tr>
<td>AWS Config allows you to apply remediation on noncompliant resources as evaluated by AWS Config Rules</td>
<td>With this release, AWS Config introduces support to apply remediation using AWS Systems Manager automation documents on noncompliant resources as evaluated by AWS Config Rules. For more information, see [Remediating Noncompliant AWS Resources by AWS Config Rules](p. 194). With this release, AWS Config adds the following new APIs. For more information, see the AWS Config API Reference:</td>
<td>March 12, 2019</td>
</tr>
<tr>
<td>• DeleteRemediationConfiguration</td>
<td>• DescribeRemediationConfigurations</td>
<td>• DescribeRemediationExecutionStatus</td>
</tr>
<tr>
<td>• PutRemediationConfigurations</td>
<td>• StartRemediationExecution</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| AWS Config supports AWS Config Rules in China (Ningxia) Region | This release only supports 54 AWS Config Rules in the China (Ningxia) Region. For more information, see [List of AWS Config Managed Rules](p. 112). However, AWS Config does not currently support the following rules in the China (Ningxia) Region:  
- acm-certificate-expiration-check  
- cmk-backing-key-rotation-enabled  
- cloudformation-stack-drift-detection-check  
- cloudformation-stack-notification-check  
- cloud-trail-encryption-enabled  
- cloud-trail-log-file-validation-enabled  
- codebuild-project-envvar-awscred-check  
- codebuild-project-source-repo-url-check  
- codepipeline-deployment-count-check  
- codepipeline-region-fanout-check  
- dynamodb-table-encryption-enabled  
- elb-acm-certificate-required  
- encrypted-volumes  
- fms-webacl-resource-policy-check  
- fms-webacl-rulegroup-association-check  
- guardduty-enabled-centralized  
- lambda-function-public-access-prohibited  
- lambda-function-settings-check  
- rds-storage-encrypted  
- root-account-mfa-hardware-mfa-enabled  
- root-account-mfa-enabled  
- s3-bucket-blacklisted-actions-prohibited  
- s3-bucket-policy-grantee-check  
- s3-bucket-policy-not-more-permissive  
- s3-bucket-public-read-prohibited  
- s3-bucket-public-write-prohibited  
- s3-bucket-server-side-encryption-enabled | March 12, 2019 |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config supports new managed rules</td>
<td>This release supports the following new managed rules:</td>
<td>January 21, 2019</td>
</tr>
<tr>
<td></td>
<td>• <code>iam-user-mfa-enabled</code> (p. 150)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• <code>s3-bucket-policy-grantee-check</code> (p. 164)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see <strong>List of AWS Config Managed Rules (p. 112)</strong>.</td>
<td></td>
</tr>
<tr>
<td>AWS Config supports AWS Service Catalog resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to the following AWS Service Catalog resources; CloudFormation product, provisioned product, and portfolio. For more information, see <strong>AWS Config Supported AWS Resource Types and Resource Relationships (p. 9)</strong>.</td>
<td>January 11, 2019</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>AWS Config supports new managed rules</td>
<td>This release supports the following new managed rules:</td>
<td>November 19, 2018</td>
</tr>
<tr>
<td></td>
<td>• cloudformation-stack-drift-detection-check (p. 120)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• codepipeline-deployment-count-check (p. 126)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• codepipeline-region-fanout-check (p. 127)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>AWS Config supports new managed rules</td>
<td>This release supports the following new managed rules:</td>
<td>November 12, 2018</td>
</tr>
<tr>
<td></td>
<td>• access-keys-rotated (p. 117)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• cloud-trail-cloud-watch-logs-enabled (p. 121)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• cloud-trail-encryption-enabled (p. 122)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• cloud-trail-log-file-validation-enabled (p. 122)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• cmk-backing-key-rotation-enabled (p. 125)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• iam-policy-no-statements-with-admin-access (p. 148)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• iam-role-managed-policy-check (p. 149)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• iam-root-access-key-check (p. 149)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• iam-user-unused-credentials-check (p. 151)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• mfa-enabled-for-iam-console-access (p. 154)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• multi-region-cloud-trail-enabled (p. 154)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• root-account-hardware-mfa-enabled (p. 160)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• vpc-flow-logs-enabled (p. 171)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| AWS Config supports new managed rules | This release supports the following new managed rules:  
• dynamodb-table-encryption-enabled (p. 130)  
• elb-logging-enabled (p. 139)  
• rds-instance-public-access-check (p. 155)  
• vpc-default-security-group-closed (p. 170)  
For more information, see List of AWS Config Managed Rules (p. 112). | October 24, 2018 |
<p>| Compliance history support | With this release, AWS Config now supports storing compliance history of resources as evaluated by AWS Config Rules. For more information, see Viewing Compliance History for Resources as Evaluated by AWS Config Rules (p. 47). | October 18, 2018 |
| Multi-account multi-region Data Aggregation Region support | With this release, multi-account multi-region Data Aggregation is now supported in six new Regions. For more information, see Multi-Account Multi-Region Data Aggregation (p. 228). | October 4, 2018 |
| AWS Config supports resource-level permissions for AWS Config Rules APIs actions | With this release, AWS Config supports resource-level permissions for certain AWS Config Rules API actions. For more information about the supported APIs, see Supported Resource-Level Permissions for AWS Config Rules APIs Actions (p. 98). | October 1, 2018 |
| AWS Config supports CodePipeline resource type | With this release, you can use AWS Config to record configuration changes to the AWS CodePipeline resource type. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | September 12, 2018 |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config supports new managed rules</td>
<td>This release supports the following new managed rules:</td>
<td>September 5, 2018</td>
</tr>
<tr>
<td></td>
<td>• ec2-instance-managed-by-systems-manager (p. 132)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2-managedinstance-association-compliance-status-check (p. 134)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2-managedinstance-patch-compliance-status-check (p. 135)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• guardduty-enabled-centralized (p. 146)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• rds-snapshots-public-prohibited (p. 156)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• s3-bucket-blacklisted-actions-prohibited (p. 161)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• s3-bucket-policy-not-more-permissive (p. 161)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>AWS Config supports AWS Systems Manager resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to the AWS Systems Manager patch compliance and association compliance resource types. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>August 9, 2018</td>
</tr>
<tr>
<td>AWS Config allows you to delete your AWS Config data using AWS Management Console</td>
<td>With this release, AWS Config introduces support for retention period using AWS Management Console. In the AWS Management Console, you can select a custom data retention period for your ConfigurationItems. For more information, see Deleting AWS Config Data (p. 73).</td>
<td>August 7, 2018</td>
</tr>
<tr>
<td>AWS Config supports AWS Shield resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to the AWS Shield Protection resource type. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>August 7, 2018</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>AWS Config supports AWS PrivateLink</td>
<td>With this release, AWS Config supports AWS PrivateLink, enabling you to route data between your Amazon Virtual Private Cloud (VPC) and AWS Config entirely within the AWS network. For more information, see Using AWS Config with Interface VPC Endpoints (p. 250).</td>
<td>July 31, 2018</td>
</tr>
<tr>
<td>AWS Config allows you to delete your AWS Config data</td>
<td>With this release, AWS Config introduces support for retention period. AWS Config allows you to delete your data by specifying a retention period for your ConfigurationItems. For more information, see Deleting AWS Config Data (p. 73).</td>
<td>May 25, 2018</td>
</tr>
</tbody>
</table>
| AWS Config supports new managed rules | This release supports the following two new managed rules:  
  - lambda-function-settings-check (p. 153)  
  - s3-bucket-replication-enabled (p. 166)  
  - iam-policy-blacklisted-check (p. 147)  
  For more information, see List of AWS Config Managed Rules (p. 112). | May 10, 2018 |
<p>| AWS Config supports AWS X-Ray resource type | With this release, you can use AWS Config to record configuration changes to the AWS X-Ray EncryptionConfig resource type. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | May 1, 2018 |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config supports AWS Lambda resource type and one new managed rule</td>
<td>With this release, you can use AWS Config to record configuration changes to the AWS Lambda function resource type. For more information, see <a href="#">AWS Config Supported AWS Resource Types and Resource Relationships</a>. This release also supports the <a href="#">lambda-function-public-access-prohibited</a> managed rule. For more information, see <a href="#">AWS Config Managed Rules</a>.</td>
<td>April 25, 2018</td>
</tr>
<tr>
<td>AWS Config supports AWS Elastic Beanstalk resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to the AWS Elastic Beanstalk Application, Application Version, and Environment resources. For more information, see <a href="#">AWS Config Supported AWS Resource Types and Resource Relationships</a>.</td>
<td>April 24, 2018</td>
</tr>
</tbody>
</table>
| AWS Config supports new managed rules                                   | This release supports the following two new managed rules:  
  - [fms-webacl-resource-policy-check](#) (p. 144)  
  - [fms-webacl-rulegroup-association-check](#) (p. 145)  

For more information, see [List of AWS Config Managed Rules](#). | April 4, 2018 |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-account multi-region data aggregation</td>
<td>With this release, AWS Config introduces multi-account multi-region data aggregation. This feature allows you to aggregate AWS Config data from multiple accounts or an organization and multiple regions into an aggregator account. For more information, see Multi-Account Multi-Region Data Aggregation (p. 228).&lt;br&gt;&lt;br&gt;With this release, AWS Config adds the following new APIs. For more information, see the AWS Config API Reference:&lt;br&gt;&lt;br&gt;- PutConfigurationAggregator&lt;br&gt;- DescribePendingAggregationRequests&lt;br&gt;- DeletePendingAggregationRequest&lt;br&gt;- PutAggregationAuthorization&lt;br&gt;- DescribeAggregationAuthorizations&lt;br&gt;- GetAggregateConfigRuleComplianceSummary&lt;br&gt;- DescribeAggregateComplianceByConfigRules&lt;br&gt;- GetAggregateComplianceDetailsByConfigRule&lt;br&gt;- DescribeConfigurationAggregators&lt;br&gt;- DescribeConfigurationAggregatorSourcesStatus&lt;br&gt;- DeleteAggregationAuthorization&lt;br&gt;- DeleteConfigurationAggregator</td>
<td>April 4, 2018</td>
</tr>
<tr>
<td>Monitoring AWS Config with Amazon CloudWatch Events</td>
<td>With this release, use Amazon CloudWatch Events to detect and react to changes in the status of AWS Config events.&lt;br&gt;&lt;br&gt;For more information, see Monitoring AWS Config with Amazon CloudWatch Events (p. 244).</td>
<td>March 29, 2018</td>
</tr>
<tr>
<td>New API operation</td>
<td>With this release, AWS Config adds support for BatchGetResourceConfig API, allowing you to batch-retrieve the current state of one or more of your resources.</td>
<td>March 20, 2018</td>
</tr>
<tr>
<td>AWS Config supports AWS WAF RuleGroup resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to the AWS WAF RuleGroup and AWS WAF RuleGroup Regional resources.&lt;br&gt;&lt;br&gt;For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>February 15, 2018</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| AWS Config supports new managed rules | This release supports the following new managed rules:  
- elb-acm-certificate-required (p. 138)  
- elb-custom-security-policy-ssl-check (p. 138)  
- elb-predefined-security-policy-ssl-check (p. 139)  
- codebuild-project-envvar-awscred-check (p. 126)  
- codebuild-project-source-repo-url-check (p. 126)  
- iam-group-has-users-check (p. 146)  
- s3-bucket-server-side-encryption-enabled (p. 166)  

For more information, see List of AWS Config Managed Rules (p. 112). | January 25, 2018 |
| AWS Config supports Elastic Load Balancing resource type | With this release, you can use AWS Config to record configuration changes to your Elastic Load Balancing classic load balancers.  
For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | November 17, 2017 |
| AWS Config supports the Amazon CloudFront and AWS WAF resource type | With this release, you can use AWS Config to record configuration changes to your CloudFront distribution and streaming distribution.  
With this release, you can use AWS Config to record configuration changes to the following AWS WAF and AWS WAF Regional resources; rate based rule, rule, and Web ACL.  
For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | November 15, 2017 |
| AWS Config supports the AWS CodeBuild resource type | With this release, you can use AWS Config to record configuration changes to your AWS CodeBuild projects.  
For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | October 20, 2017 |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config supports Auto Scaling resources and one new managed rule</td>
<td>With this release, you can use AWS Config to record configuration changes to the following Auto Scaling resources; groups, launch configuration, scheduled action, and scaling policy. For more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](p. 9). This release also supports the following managed rule: • autoscaling-group-elb-healthcheck-required (p. 120) For more information, see [AWS Config Managed Rules](p. 112).</td>
<td>September 18, 2017</td>
</tr>
<tr>
<td>AWS Config supports the AWS CodeBuild resource type</td>
<td>With this release, you can use AWS Config to record configuration changes to your AWS CodeBuild projects. For more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](p. 9).</td>
<td>October 20, 2017</td>
</tr>
<tr>
<td>AWS Config supports Auto Scaling resources and one new managed rule</td>
<td>With this release, you can use AWS Config to record configuration changes to the following Auto Scaling resources; groups, launch configuration, scheduled action, and scaling policy. For more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](p. 9). This release also supports the following managed rule: • autoscaling-group-elb-healthcheck-required (p. 120) For more information, see [AWS Config Managed Rules](p. 112).</td>
<td>September 18, 2017</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>AWS Config supports the DynamoDB table resource type and one new managed rule</td>
<td>With this release, you can use AWS Config to record configuration changes to your DynamoDB tables. For more information, see <a href="#">AWS Config Supported AWS Resource Types and Resource Relationships</a> (p. 9). This release supports the following managed rule:</td>
<td>September 8, 2017</td>
</tr>
<tr>
<td></td>
<td>• dynamodb-autoscaling-enabled (p. 129)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see <a href="#">AWS Config Managed Rules</a> (p. 112).</td>
<td></td>
</tr>
<tr>
<td>AWS Config supports two new managed rules for Amazon S3</td>
<td>This release supports two new managed rules:</td>
<td>August 14, 2017</td>
</tr>
<tr>
<td></td>
<td>• s3-bucket-public-read-prohibited (p. 165)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• s3-bucket-public-write-prohibited (p. 165)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see <a href="#">AWS Config Managed Rules</a> (p. 112).</td>
<td></td>
</tr>
<tr>
<td>New page in the AWS Config console</td>
<td>You can use the <strong>Dashboard</strong> in the AWS Config console to see the following:</td>
<td>July 17, 2017</td>
</tr>
<tr>
<td></td>
<td>• Total number of resources</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Total number of rules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Number of noncompliant resources</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Number of noncompliant rules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see <a href="#">Viewing the AWS Config Dashboard</a> (p. 31).</td>
<td></td>
</tr>
<tr>
<td>New API operation</td>
<td>You can use the <code>GetDiscoveredResourceCounts</code> operation to return the number of resource types, the number of each resource type, and the total number of resources that AWS Config is recording in a Region for your AWS account.</td>
<td>July 17, 2017</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| AWS Config supports the AWS CloudFormation stack resource type and one new managed rule | With this release, you can use AWS Config to record configuration changes to your AWS CloudFormation stacks. For more information, see [AWS Config Supported AWS Resource Types and Resource Relationships](#) (p. 9). This release supports the following managed rule:  
  - `cloudformation-stack-notification-check` (p. 120)  
  For more information, see [AWS Config Managed Rules](#) (p. 112). | July 6, 2017 |
<p>| New and updated content | This release adds support for AWS Config Rules in the Canada (Central) Region and South America (São Paulo) Region. For all regions that support AWS Config and Config Rules, see <a href="#">AWS Regions and Endpoints</a> in the <a href="#">AWS General Reference</a>. | July 5, 2017 |
| New and updated content | AWS Config Rules is available in the AWS GovCloud (US) Region. For more information, see the <a href="#">AWS GovCloud (US) User Guide</a>. For regions that support AWS Config, see <a href="#">AWS Regions and Endpoints</a> in the <a href="#">AWS General Reference</a>. | June 8, 2017 |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config supports the Amazon CloudWatch</td>
<td>With this release, you can use AWS Config to record configuration changes to your Amazon CloudWatch alarms. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). This release supports three new managed rules:</td>
<td></td>
</tr>
</tbody>
</table>
| alarm resource type and three new managed   | • cloudwatch-alarm-action-check (p. 123)  
• cloudwatch-alarm-resource-check (p. 124)  
• cloudwatch-alarm-settings-check (p. 124)  
For more information, see AWS Config Managed Rules (p. 112).                                                   | June 1, 2017 |
| rules                                       |                                                                                                                                                                                                          |              |
| New and updated content                     | This release supports specifying the application version number for the following managed rules:                                                                                                          | June 1, 2017 |
|                                             | • ec2-managedinstance-applications-blacklisted (p. 133)  
• ec2-managedinstance-applications-required (p. 133)  
For more information, see AWS Config Managed Rules (p. 112).                                                                                     |              |
<p>| New and updated content                     | This release adds support for AWS Config Rules in the Asia Pacific (Mumbai) Region. For more information, see AWS Regions and Endpoints in the AWS General Reference.                                        | April 27, 2017 |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>New and updated content</td>
<td>This release supports an updated console experience for adding AWS Config managed rules to your account for the first time. When you set up AWS Config Rules for the first time or in a new Region, you can search for AWS managed rules by name, description, or label. You can choose <strong>Select all</strong> to select all rules or choose <strong>Clear all</strong> to clear all rules. For more information, see Setting Up AWS Config Rules with the Console (p. 29).</td>
<td>April 5, 2017</td>
</tr>
</tbody>
</table>
| AWS Config supports new managed rules | This release supports the following new managed rules:  
- `acm-certificate-expiration-check` (p. 117)  
- `ec2-instance-detailed-monitoring-enabled` (p. 132)  
- `ec2-managedinstance-inventory-blacklisted` (p. 134)  
- `ec2-volume-inuse-check` (p. 136)  
- `iam-user-group-membership-check` (p. 150)  
- `iam-user-no-policies-check` (p. 150)  
- `s3-bucket-ssl-requests-only` (p. 166)  
For more information, see List of AWS Config Managed Rules (p. 112). | February 21, 2017 |
<p>| New and updated content | This release adds support for AWS Config Rules in the Europe (London) Region. For more information, see AWS Regions and Endpoints in the AWS General Reference. | February 21, 2017 |
| New and updated content | This release adds AWS CloudFormation templates for AWS Config managed rules. You can use the templates to create managed rules for your account. For more information, see Creating AWS Config Managed Rules With AWS CloudFormation Templates (p. 173). | February 16, 2017 |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>New and updated content</td>
<td>This release adds support for a new test mode for the <code>PutEvaluations</code> API. Set the <code>TestMode</code> parameter to true in your custom rule to verify whether your AWS Lambda function will deliver evaluation results to AWS Config. No updates occur to your existing evaluations, and evaluation results are not sent to AWS Config. For more information, see <code>PutEvaluations</code> in the <em>AWS Config API Reference</em>.</td>
<td>February 16, 2017</td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds support for AWS Config Rules in the Asia Pacific (Seoul), and US West (N. California) Regions. For more information, see <em>AWS Regions and Endpoints</em> in the <em>AWS General Reference</em>.</td>
<td>December 21, 2016</td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds support for AWS Config in the Europe (London) Region. For more information, see <em>AWS Regions and Endpoints</em> in the <em>AWS General Reference</em>.</td>
<td>December 13, 2016</td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds support for AWS Config in the Canada (Central) Region. For more information, see <em>AWS Regions and Endpoints</em> in the <em>AWS General Reference</em>.</td>
<td>December 8, 2016</td>
</tr>
<tr>
<td>AWS Config supports Amazon Redshift resource types and two new managed rules</td>
<td>With this release, you can use AWS Config to record configuration changes to your Amazon Redshift clusters, cluster parameter groups, cluster security groups, cluster snapshots, cluster subnet groups, and event subscriptions. For more information, see <em>AWS Config Supported AWS Resource Types and Resource Relationships</em> (p. 9). This release supports two new managed rules: - <code>redshift-cluster-configuration-check</code> (p. 156) - <code>redshift-cluster-maintenancesettings-check</code> (p. 157) For more information, see <em>List of AWS Config Managed Rules</em> (p. 112).</td>
<td>December 7, 2016</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Release Date</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds support for a new managed rule:</td>
<td>December 7, 2016</td>
</tr>
<tr>
<td></td>
<td>- dynamodb-throughput-limit-check (p. 130)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see List of AWS Config Managed Rules (p. 112).</td>
<td></td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds support for creating up to 50 rules per Region in an account. For more information, see AWS Config Limits in the AWS General Reference.</td>
<td>December 7, 2016</td>
</tr>
</tbody>
</table>
| AWS Config supports the managed instance inventory resource type for Amazon EC2 Systems Manager and three new managed rules | With this release, you can use AWS Config to record software configuration changes on your managed instances with support for managed instance inventory. For more information, see Recording Software Configuration for Managed Instances (p. 65). This release supports three new managed rules:  
- ec2-managedinstance-inventory-blacklisted (p. 134)  
- ec2-managedinstance-applications-required (p. 133)  
- ec2-managedinstance-platform-check (p. 135)  
For more information, see List of AWS Config Managed Rules (p. 112). | December 1, 2016 |
| AWS Config supports the Amazon S3 bucket resource and two new managed rules | With this release, you can use AWS Config to record configuration changes to your Amazon S3 buckets. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).  
This release supports two new managed rules:  
- s3-bucket-logging-enabled (p. 163)  
- s3-bucket-versioning-enabled (p. 168)  
For more information, see AWS Config Managed Rules (p. 112). | October 18, 2016 |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>New and updated content</td>
<td>This release adds support for AWS Config and AWS Config Rules in the US East (Ohio) Region. For more information, see AWS Regions and Endpoints in the AWS General Reference.</td>
<td>October 17, 2016</td>
</tr>
</tbody>
</table>
| New and updated managed rules                | This update adds support for eight new managed rules:  
  - approved-amis-by-id (p. 119)  
  - approved-amis-by-tag (p. 119)  
  - db-instance-backup-enabled (p. 127)  
  - desired-instance-type (p. 128)  
  - ebs-optimized-instance (p. 130)  
  - iam-password-policy (p. 147)  
  - rds-multi-az-support (p. 155)  
  - rds-storage-encrypted (p. 156)  
  You can specify multiple parameter values for the following rules:  
  - desired-instance-tenancy (p. 128)  
  - required-tags (p. 158)  
  For more information, see List of AWS Config Managed Rules (p. 112).                                                                                                                                                                                                                                                                                             | October 4, 2016  |
<p>| New content for the AWS Config console       | This update adds support for viewing AWS CloudTrail API activity in the AWS Config timeline. If CloudTrail is logging for your account, you can view create, update, and delete API events for configuration changes to your resources. For more information, see Viewing Configuration Details (p. 41).                                                                                                                                                                                                                                                     | September 06, 2016|
| AWS Config supports Elastic Load Balancing   | With this release, you can use AWS Config to record configuration changes to your Elastic Load Balancing application load balancers. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).                                                                                                                                                                                                                                               | August 31, 2016  |
| resource type                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                  |
| New and updated content                      | This release adds support for AWS Config Rules in the Asia Pacific (Singapore), and Asia Pacific (Sydney) Regions. For more information, see AWS Regions and Endpoints in the AWS General Reference.                                                                                                                                                                                                                                                                  | August 18, 2016  |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
</table>
| New and updated content for AWS Config Rules | This update adds support for creating a rule that can be triggered by both configuration changes and at a periodic frequency that you choose. For more information, see Specifying Triggers for AWS Config Rules (p. 110).  
This update also adds support for manually evaluating your resources against your rule and deleting evaluation results. For more information, see Evaluating Your Resources (p. 192).  
This update also adds support for evaluating additional resource types using custom rules. For more information, see Evaluating Additional Resource Types (p. 179). | July 25, 2016 |
| AWS Config supports Amazon RDS and AWS Certificate Manager (ACM) resource types | With this release, you can use AWS Config to record configuration changes to your Amazon Relational Database Service (Amazon RDS) DB instances, DB security groups, DB snapshots, DB subnet groups, and event subscriptions. You can also use AWS Config to record configuration changes to certificates provided by ACM.  
For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9). | July 21, 2016 |
<p>| Updated information about managing the configuration recorder | This update adds steps for renaming and deleting the configuration recorder to Managing the Configuration Recorder (p. 60). | July 07, 2016 |
| 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 with the Console (p. 22) are updated, the example policy in Permissions for the Amazon S3 Bucket (p. 102) is updated, and the example policy in Granting Custom Permissions for AWS Config Users (p. 92) is updated. | March 31, 2016 |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example functions and events for Config rules</td>
<td>This update provides updated example functions in Example AWS Lambda Functions for AWS Config Rules (Node.js) (p. 180), and this update adds example events in Example Events for AWS Config Rules (p. 185).</td>
<td>March 29, 2016</td>
</tr>
<tr>
<td>AWS Config Rules GitHub repository</td>
<td>This update adds information about the AWS Config Rules GitHub repository to Evaluating Resources with Rules (p. 107). This repository provides sample functions for custom rules that are developed and contributed by AWS Config users.</td>
<td>March 1, 2016</td>
</tr>
<tr>
<td>AWS Config Rules</td>
<td>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 Rules (p. 107).</td>
<td>December 18, 2015</td>
</tr>
<tr>
<td>AWS Config supports IAM resource types</td>
<td>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 AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>December 10, 2015</td>
</tr>
<tr>
<td>AWS Config supports EC2 Dedicated host</td>
<td>With this release, you can use AWS Config to record configuration changes to your EC2 Dedicated hosts. For more information, see AWS Config Supported AWS Resource Types and Resource Relationships (p. 9).</td>
<td>November 23, 2015</td>
</tr>
</tbody>
</table>
| 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. 101).  
  - **AWSConfigUserAccess** – Grants read-only access to an AWS Config user. For more information, see Granting Custom Permissions for AWS Config Users (p. 92). | October 19, 2015 |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Release Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Config Rules preview</td>
<td>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 Rules (p. 107).</td>
<td>October 7, 2015</td>
</tr>
<tr>
<td>New and updated content</td>
<td>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. 40).</td>
<td>August 27, 2015</td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds the ability to select which resource types AWS Config records. For more information, see Selecting Which Resources AWS Config Records (p. 62).</td>
<td>June 23, 2015</td>
</tr>
<tr>
<td>New and updated content</td>
<td>This release adds support for the following regions: Asia Pacific (Tokyo), Asia Pacific (Singapore), Europe (Frankfurt), South America (São Paulo), and US West (N. California). For more information, see AWS Regions and Endpoints.</td>
<td>April 6, 2015</td>
</tr>
<tr>
<td>New and updated content</td>
<td>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. 251). This release adds support for the US West (Oregon), Europe (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</td>
<td>February 10, 2015</td>
</tr>
<tr>
<td>New guide</td>
<td>This release introduces AWS Config.</td>
<td>November 12, 2014</td>
</tr>
</tbody>
</table>
AWS Glossary

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