
AWS CloudTrail

User Guide

Version 1.0



AWS CloudTrail: User Guide

Copyright © 2016 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 CloudTrail?	1
How AWS CloudTrail Works	1
Additional Information	2
CloudTrail Workflow	2
CloudTrail Concepts	3
What Are Trails?	4
How Do You Manage CloudTrail?	4
How Do You Control Access to CloudTrail?	4
How Do You Perform Monitoring with CloudTrail?	5
How Does CloudTrail Behave Regionally and Globally?	5
About global service events	6
How Does CloudTrail Relate to Other AWS Monitoring Services?	7
Partner Solutions	7
CloudTrail Supported Services	7
Analytics	8
Application Services	9
Compute	9
Database	10
Developer Tools	11
Enterprise Applications	11
Internet of Things	12
Game Development	12
Management Tools	12
Mobile Services	13
Networking	14
Security and Identity	14
Storage and Content Delivery	15
Support	16
CloudTrail Topics by AWS Service	16
CloudTrail Supported Regions	19
CloudTrail Log File Examples	20
CloudTrail Log File Name Format	20
Log File Examples	21
Getting Started with CloudTrail	27
Creating and Updating Your Trail	27
Tips for managing trails	28
Creating and Updating a Trail with the CloudTrail Console	28
Creating and Updating a Trail with the AWS CLI	34
CloudTrail Trail Naming Requirements	42
Amazon S3 Bucket Naming Requirements	42
Amazon S3 Bucket Policy for CloudTrail	42
AWS KMS Alias Naming Requirements	45
Viewing Events with CloudTrail API Activity History	45
Viewing CloudTrail Events in the CloudTrail Console	46
Viewing CloudTrail Events with the AWS CLI	48
Regions Supported by CloudTrail API Activity History	53
Services Supported by CloudTrail API Activity History	54
Resource Types Supported by CloudTrail API Activity History	78
Controlling User Permissions for CloudTrail	81
Granting Permissions for CloudTrail Administration	82
Granting Custom Permissions for CloudTrail Users	83
Getting and Viewing Your CloudTrail Log Files	88
Finding Your CloudTrail Log Files	88
Reading Your CloudTrail Log Files	89
Configuring Amazon SNS Notifications for CloudTrail	90

Configuring CloudTrail to Send Notifications	91
CloudTrail Permissions for SNS Notifications	92
Working with CloudTrail Log Files	93
Create Multiple Trails	93
Receiving CloudTrail Log Files from Multiple Regions	94
Monitoring CloudTrail Log Files with Amazon CloudWatch Logs	95
Sending CloudTrail Events to CloudWatch Logs	96
Using an AWS CloudFormation Template to Create CloudWatch Alarms	99
Creating CloudWatch Alarms for CloudTrail Events: Examples	108
Creating CloudWatch Alarms for CloudTrail Events: Additional Examples	131
Configuring Notifications for CloudWatch Logs Alarms	137
Stopping CloudTrail from Sending Events to CloudWatch Logs	137
CloudWatch Log Group and Log Stream Naming for CloudTrail	138
Role Policy Document for CloudTrail to Use CloudWatch Logs for Monitoring	138
Receiving CloudTrail Log Files from Multiple Accounts	139
Setting Bucket Policy for Multiple Accounts	139
Turning on CloudTrail in Additional Accounts	140
Sharing CloudTrail Log Files Between AWS Accounts	142
Scenario 1: Granting Access to the Account that Generated the Log Files	143
Scenario 2: Granting Access to All Logs	144
Creating a Role	145
Creating an Access Policy to Grant Access to Accounts You Own	147
Creating an Access Policy to Grant Access to a Third Party	148
Assuming a Role	150
Stop Sharing CloudTrail Log Files Between AWS Accounts	152
Encrypting CloudTrail Log Files with AWS KMS–Managed Keys (SSE-KMS)	152
Enabling log file encryption	153
Granting Permissions to Create a CMK	154
AWS KMS Key Policy for CloudTrail	154
Updating a Trail to Use Your CMK	161
Enabling and disabling CloudTrail log file encryption with the AWS CLI	162
Validating CloudTrail Log File Integrity	163
Why Use It?	163
How It Works	164
Enabling Log File Integrity Validation for CloudTrail	164
Validating CloudTrail Log File Integrity with the AWS CLI	165
CloudTrail Digest File Structure	170
Custom Implementations of CloudTrail Log File Integrity Validation	175
Using the CloudTrail Processing Library	184
Minimum requirements	184
Processing CloudTrail Logs with the CloudTrail Processing Library	184
Advanced Topics	188
Additional Resources	191
CloudTrail Event Reference	192
CloudTrail Record Contents	193
sharedEventID Example	195
CloudTrail userIdentity Element	196
Examples	196
Fields	197
Values for AWS STS APIs with SAML and Web Identity Federation	199
Non-API Events Captured by CloudTrail	200
AWS Console Sign-in Events	200
CloudTrail Document History	203

What is AWS CloudTrail?

You can use AWS CloudTrail to get a history of AWS API calls and related events for your account. This includes calls made by using the AWS Management Console, AWS SDKs, command line tools, and higher-level AWS services.

You can identify which users and accounts called AWS for services that support CloudTrail, the source IP address the calls were made from, and when the calls occurred. You can integrate CloudTrail into applications using the API, automate trail creation for your organization, check the status of your trails, and control how administrators turn CloudTrail logging on and off.

Topics

- [How AWS CloudTrail Works \(p. 1\)](#)
- [CloudTrail Workflow \(p. 2\)](#)
- [CloudTrail Concepts \(p. 3\)](#)
- [CloudTrail Supported Services \(p. 7\)](#)
- [CloudTrail Supported Regions \(p. 19\)](#)
- [CloudTrail Log File Examples \(p. 20\)](#)

How AWS CloudTrail Works

AWS CloudTrail captures AWS API calls and related events made by or on behalf of an AWS account and delivers log files to an Amazon S3 bucket that you specify. Optionally, you can configure CloudTrail to deliver events to a log group to be monitored by CloudWatch Logs. You can also choose to receive Amazon SNS notifications each time a log file is delivered to your bucket. You can create a trail with the CloudTrail console, the AWS CLI, or the CloudTrail API. A trail is a configuration that enables logging of the AWS API activity and related events in your account.

You can create two types of trails:

- **A trail that applies to all regions** - When you create a trail that applies to all regions, CloudTrail creates the same trail in each region, records the log files in each region, and delivers the log files to the single S3 bucket (and optionally to the CloudWatch Logs log group) that you specify. This is the default option when you create a trail using the CloudTrail console. If you choose to receive Amazon SNS notifications for log file deliveries, one SNS topic will suffice for all regions. If you choose to have CloudTrail send events from a trail that applies to all regions to a CloudWatch Logs log group, events from all regions will be sent to the single log group.

- **A trail that applies to one region** - You specify a bucket that receives events only from that region. The bucket can be in any region that you specify. If you create additional individual trails that apply to specific regions, you can have those trails deliver event logs to a single S3 bucket.

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

CloudTrail typically delivers log files within 15 minutes of an API call. In addition, the service publishes new log files multiple times an hour, usually about every five minutes. These log files contain API calls from all of the account's services that support CloudTrail. For a list of AWS services that support CloudTrail, see [CloudTrail Supported Services \(p. 7\)](#).

Note

AWS CloudTrail records API calls made on an AWS account directly by the user or on behalf of the user by an AWS service. Examples of services that make API calls on behalf of users include, but are not limited to, AWS CloudFormation, Elastic Beanstalk, AWS OpsWorks, and Auto Scaling. For example, an AWS CloudFormation `CreateStack` call can result in additional API calls to Amazon EC2, Amazon RDS, Amazon EBS or other services as required by the AWS CloudFormation template. This behavior is normal and expected. CloudTrail logs all of these API calls and provides a history of the calls made by the users directly or made by an AWS service as a result of the calls made by the user. You can identify the latter type of API call by examining the **invokedBy** field in the CloudTrail event.

Additional Information

To get started with CloudTrail, see [Getting Started with CloudTrail \(p. 27\)](#).

For information on sending CloudTrail events to a CloudWatch Logs log group, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 95\)](#).

For information on SNS notifications, see [Configuring Amazon SNS Notifications for CloudTrail \(p. 90\)](#). You can also choose to have CloudTrail receive SNS notifications that you configured for CloudWatch alarms.

For information on aggregating log files from AWS regions into a single Amazon S3 bucket, see [Receiving CloudTrail Log Files from Multiple Regions \(p. 94\)](#).

You can share CloudTrail log files between AWS accounts. For more information, see [Sharing CloudTrail Log Files Between AWS Accounts \(p. 142\)](#).

You can aggregate log files from multiple accounts into a single bucket. For more information, see [Receiving CloudTrail Log Files from Multiple Accounts \(p. 139\)](#).

Note

When configuring a trail, you can choose an Amazon S3 bucket that belongs to a different account. However, if you choose to have CloudTrail deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

For CloudTrail pricing, see the [AWS CloudTrail Pricing](#) page. For associated Amazon S3 and Amazon SNS usage charges, see [Amazon S3 Pricing](#) and [Amazon SNS Pricing](#).

CloudTrail Workflow

Here are the steps you take to use CloudTrail, which are described in detail in this guide:

1. Using the console, the AWS CLI, or the CloudTrail API, you create a trail, which consists of the information that CloudTrail uses to deliver log files to your Amazon S3 bucket or CloudWatch Logs log group. For information, see [Creating and Updating Your Trail \(p. 27\)](#). By default, a trail created in the console is applied across regions so that activity from all regions in an AWS partition are logged to the Amazon S3 bucket that you specify.
2. (Optional) You create an Amazon SNS topic to which you subscribe for notifications that a new log file has arrived in your bucket. Amazon SNS can notify you in multiple ways, including programmatically using Amazon Simple Queue Service. For information, see [Configuring Amazon SNS Notifications for CloudTrail \(p. 90\)](#).
3. You use the Amazon S3 API or console to retrieve the log files. For information, see [Getting and Viewing Your CloudTrail Log Files \(p. 88\)](#).
4. You use the CloudTrail API, AWS CLI, or console to update your trail.
5. (Optional) You can use AWS Identity and Access Management to control which AWS users can create, configure, or delete trails, start and stop logging, and access the buckets that contain log information. For information, see [Controlling User Permissions for CloudTrail \(p. 81\)](#).
6. (Optional) You can use CloudWatch Logs to monitor events from API activity captured by CloudTrail. If so, you also create an IAM role with permissions for CloudTrail to send events to a CloudWatch Logs log group for monitoring. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 95\)](#).
7. (Optional) You can enable log file encryption, which provides an extra layer of security for your log files. For more information, see [Encrypting CloudTrail Log Files with AWS KMS–Managed Keys \(SSE-KMS\) \(p. 152\)](#).
8. (Optional) You can enable log file integrity validation, which enables verification that log files have remained unchanged since CloudTrail delivered them. For more information, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).
9. (Optional) You can analyze your CloudTrail output by using one of the partner solutions that integrate with CloudTrail. These solutions offer a broad set of capabilities, such as change tracking, troubleshooting, and security analysis. For more information, see the [Amazon CloudTrail](#) page.

CloudTrail Concepts

This section summarizes basic concepts related to CloudTrail.

Contents

- [What Are Trails? \(p. 4\)](#)
- [How Do You Manage CloudTrail? \(p. 4\)](#)
 - [CloudTrail Console \(p. 4\)](#)
 - [CloudTrail CLI \(p. 4\)](#)
 - [CloudTrail APIs \(p. 4\)](#)
 - [AWS SDKs \(p. 4\)](#)
- [How Do you Control Access to CloudTrail? \(p. 4\)](#)
- [How Do You Perform Monitoring with CloudTrail? \(p. 5\)](#)
 - [CloudWatch Logs and CloudTrail \(p. 5\)](#)
- [How Does CloudTrail Behave Regionally and Globally? \(p. 5\)](#)
 - [What are the advantages of applying a trail to all regions? \(p. 5\)](#)
 - [What happens when you apply a trail to all regions? \(p. 5\)](#)
 - [Multiple trails per region \(p. 6\)](#)
 - [AWS Security Token Service \(AWS STS\) and CloudTrail \(p. 6\)](#)
- [About global service events \(p. 6\)](#)
- [How Does CloudTrail Relate to Other AWS Monitoring Services? \(p. 7\)](#)

- [Partner Solutions \(p. 7\)](#)

What Are Trails?

A trail is a configuration that enables logging of the AWS API activity and related events in your account. CloudTrail delivers the logs to an Amazon S3 bucket that you specify, and optionally to a CloudWatch Logs log group. You can also specify an Amazon SNS topic that receives notifications of log file deliveries. For a trail that applies to all regions, the trail configuration in each region is identical.

How Do You Manage CloudTrail?

CloudTrail Console

You can manage the CloudTrail service by using the AWS CloudTrail console web application. The console provides a user interface for performing many CloudTrail tasks such as turning on or editing CloudTrail, selecting an Amazon S3 bucket, setting a prefix, including or preventing API calls from global services such as IAM and AWS STS, and receiving Amazon SNS notifications for log file deliveries. For more information about the AWS management console in general, see [AWS Management Console](#).

CloudTrail CLI

The AWS Command Line Interface is a unified tool that enables you to act easily with CloudTrail from the command line. For more information, see the [CLI User Guide](#). For a complete list of the available CloudTrail CLI commands, see [Available Commands](#).

CloudTrail APIs

In addition to the console and the CLI, you can also use the CloudTrail RESTful APIs to program CloudTrail directly. For more information see the [AWS CloudTrail API Reference](#).

AWS SDKs

As an alternative to using the CloudTrail 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 CloudTrail. For example, the SDKs take care of cryptographically signing requests, managing errors, and retrying requests automatically. For more information, see the [Tools For AWS](#) page.

How Do you Control Access to CloudTrail?

AWS Identity and Access Management is a web service that enables Amazon Web Services (AWS) customers to manage users and user permissions. Without IAM, organizations with multiple users and systems must either create multiple AWS accounts, each with its own billing and subscriptions to AWS products, or employees must all share the security credentials of a single AWS account. Also, without IAM, you have no control over the tasks a particular user or system can do and what AWS resources they might use.

Use IAM to create individual users for anyone who needs access to AWS CloudTrail. Create an IAM user for yourself as well, give that IAM user administrative privileges, and use that IAM user for all 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 any time. For more information, see [Controlling User Permissions for CloudTrail \(p. 81\)](#).

How Do You Perform Monitoring with CloudTrail?

CloudWatch Logs and CloudTrail

Amazon CloudWatch is a web service that collects and tracks metrics to monitor in real time your Amazon Web Services (AWS) resources and the applications that you run on AWS. Amazon CloudWatch Logs is a feature of CloudWatch that you can use specifically to monitor log data. Integration with CloudWatch Logs enables CloudTrail to send events containing API activity in your AWS account to a CloudWatch Logs log group. CloudTrail events that are sent to CloudWatch Logs can trigger alarms according to the metric filters you define. You can optionally configure CloudWatch alarms to send notifications or make changes to the resources that you are monitoring based on log stream events that your metric filters extract. Using CloudWatch Logs, you can also track CloudTrail events alongside events from the operating system, applications, or other AWS services that are sent to CloudWatch Logs. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 95\)](#).

How Does CloudTrail Behave Regionally and Globally?

A trail can be applied to all regions or a single region. As a best practice, create a trail that applies to all regions in the [AWS partition](#) in which you are working. This is the default setting when you create a trail in the CloudTrail console.

Note

'Turning on a trail' means that you create a trail and start logging. In the CloudTrail console, logging is turned on automatically when you create a trail.

What are the advantages of applying a trail to all regions?

A trail that applies to all regions has the following advantages:

- The configuration settings for the trail apply consistently across all regions.
- You receive log files from all regions in a single S3 bucket and optionally in a CloudWatch Logs log group.
- You manage trail configuration for all regions from one location.
- You immediately receive events from a new region. When a new region launches, CloudTrail automatically creates a trail for you in the new region with the same settings as your original trail.
- You can create trails in regions that you don't use often to monitor for unusual activity.

What happens when you apply a trail to all regions?

When you apply a trail to all regions, CloudTrail uses the trail that you create in a particular region to create trails with identical configuration in all other regions in your account.

This has the following effects:

- CloudTrail delivers log files for API activity from all regions to the single Amazon S3 bucket that you specify, and optionally to a CloudWatch Logs log group.
- If you configured an Amazon SNS topic for the trail, SNS notifications about log file deliveries in all regions are sent to that single SNS topic.
- Global service events will be delivered from a single region to your specified S3 bucket and to your CloudWatch Logs log group if you configured one for the trail. For information about global service events, see [About global service events \(p. 6\)](#).

- If you enabled log file integrity validation, log file integrity validation is enabled in all regions for the trail. For information about log file integrity validation, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).

Multiple trails per region

If you have different but related user groups such as developers, security personnel, and IT auditors, you can create multiple trails per region. This allows each group to receive its own copy of the log files.

CloudTrail supports five trails per region. A trail that applies to all regions counts as one trail in every region.

The following example is a region with 5 trails:

- You create two trails in the (US West (N. California) Region) that apply only to this region.
- You create two more trails in US West (N. California) Region that apply to all regions.
- You create a trail in the (Asia Pacific (Sydney) Region) that applies to all regions. This trail also exists as a trail in the US West (N. California) Region.

You can see a list of your trails in all regions on the Trails page of the CloudTrail console. For more information, see [Updating a Trail \(p. 32\)](#).

AWS Security Token Service (AWS STS) and CloudTrail

AWS STS is a service that has a global endpoint and that also supports region-specific endpoints. An endpoint is a URL that is the entry point for web service requests. For example, `https://cloudtrail.us-west-2.amazonaws.com` is the US West (Oregon) regional entry point for the AWS CloudTrail service. Regional endpoints help reduce latency in your applications.

When you use an AWS STS region-specific endpoint, the trail in that region delivers only the AWS STS events that occur in that region. For example, if you are using the endpoint `sts.us-west-2.amazonaws.com`, the trail in us-west-2 delivers only the AWS STS events that originate from us-west-2. For more information about AWS STS regional endpoints, see [Activating AWS STS in a New Region](#).

For a complete list of AWS web service regional endpoints, see [Regions and Endpoints](#). For details about events from the global AWS STS endpoint, see [About global service events \(p. 6\)](#).

About global service events

For most services, events are sent to the region where the action happened. For global services such as IAM, AWS STS, and Amazon CloudFront, events are delivered to any trail that has the **Include global services** option enabled. AWS OpsWorks and Amazon Route 53 actions are logged in the US East (N. Virginia) Region.

To avoid receiving duplicate global service events, remember the following:

- Global service events are always delivered to trails that have the **Apply trail to all regions** option enabled. Events are delivered from a single region to the bucket for the trail. This setting cannot be changed.
- If you have a single region trail, you should enable the **Include global services** option.
- If you have multiple single region trails, you should enable the **Include global services** option in only one of the trails.

Example:

1. You have a trail with the **Apply trail to all regions** option enabled.
2. You have multiple single region trails.
3. You do not need to enable the **Include global services** option for the single region trails. Global service events are delivered for the first trail.

Note

When you create a new trail with the AWS CLI, AWS SDKs, or CloudTrail API, you can include or exclude global service events for single region trails. You can also use the **Additional Configuration** section in the CloudTrail console to update an existing trail to include or exclude global service events.

How Does CloudTrail Relate to Other AWS Monitoring Services?

CloudTrail adds another dimension to the monitoring capabilities already offered by AWS; it does not change or replace logging features you might already be using such as those for Amazon S3 or Amazon CloudFront subscriptions. Amazon CloudWatch focuses on performance monitoring and system health; CloudTrail focuses on API activity. While CloudTrail does not report on system performance or health, you can use CloudTrail in conjunction with CloudWatch Logs alarms to notify you about activity that you might be interested in, as mentioned above.

Partner Solutions

AWS partners with third-party specialists in logging and analysis to provide solutions that leverage CloudTrail output. For more information, visit the CloudTrail detail page at [AWS CloudTrail](#).

CloudTrail Supported Services

CloudTrail supports the following services:

Service Categories

- [Analytics](#) (p. 8)
- [Application Services](#) (p. 9)
- [Compute](#) (p. 9)
- [Database](#) (p. 10)
- [Developer Tools](#) (p. 11)
- [Enterprise Applications](#) (p. 11)
- [Internet of Things](#) (p. 12)
- [Game Development](#) (p. 12)
- [Management Tools](#) (p. 12)
- [Mobile Services](#) (p. 13)
- [Networking](#) (p. 14)
- [Security and Identity](#) (p. 14)
- [Storage and Content Delivery](#) (p. 15)
- [Support](#) (p. 16)
- [CloudTrail Topics by AWS Service](#) (p. 16)

Analytics

- **Amazon EMR** (Supported beginning 04/04/2014)

Amazon EMR (Amazon EMR) is a web service that makes it easy to process large amounts of data efficiently. Amazon EMR uses Hadoop processing combined with several services from AWS to perform such tasks as web indexing, data mining, log file analysis, machine learning, scientific simulation, and data warehousing. For more information, see the [Amazon EMR Developer Guide](#). For more information about the Amazon EMR calls logged by CloudTrail, see [Logging Amazon EMR API Calls in AWS CloudTrail](#).

- **AWS Data Pipeline** (Supported beginning 12/02/2014)

AWS Data Pipeline is a web service that you can use to automate the movement and transformation of data through data-driven workflows. For more information, see the [AWS Data Pipeline Developer Guide](#). For more information about the AWS Data Pipeline calls logged by CloudTrail, see [Logging AWS Data Pipeline API Calls by using AWS CloudTrail](#).

- **Amazon Kinesis Firehose** (Supported beginning 03/17/2016)

Amazon Kinesis Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon S3 and Amazon Redshift. Firehose is part of the Amazon Kinesis streaming data family, along with Amazon Kinesis Streams. For more information, see the [Amazon Kinesis Firehose Developer Guide](#). For more information about the Firehose calls logged by CloudTrail, see [Logging Amazon Kinesis Firehose API Calls with AWS CloudTrail](#).

- **Amazon Kinesis Streams** (Supported beginning 04/25/2014)

Amazon Kinesis Streams is a managed service that scales elastically for real-time processing of streaming big data. The service takes in large streams of data records that can then be consumed in real time by multiple data-processing applications that can be run on Amazon EC2 instances. For more information, see the [Amazon Kinesis Streams Developer Guide](#). For more information about the Streams calls logged by CloudTrail, see [Logging Amazon Kinesis Streams API Calls by Using AWS CloudTrail](#).

- **Amazon Redshift** (Supported beginning 06/10/2014)

Amazon Redshift is a fast, fully managed, petabyte-scale data warehouse service that makes it simple and cost-effective to efficiently analyze all your data by using your existing business intelligence tools. It is optimized for datasets that range from a few hundred gigabytes to a petabyte or more. An Amazon Redshift data warehouse is a collection of computing resources called nodes which are organized into groups called clusters. Each cluster runs an Amazon Redshift engine and contains one or more databases. For more information, see the [Amazon Redshift Database Developer Guide](#). For a complete list of the Amazon Redshift actions logged by CloudTrail, see the [Amazon Redshift API Reference](#).

- **Amazon Elasticsearch Service** (Supported beginning 10/01/2015)

Amazon Elasticsearch Service is a managed service that makes it easy to deploy, operate, and scale Amazon ES in the AWS cloud. For more information, see the [Amazon Elasticsearch Service Developer Guide](#). For information on the Amazon ES API calls logged by CloudTrail, see [Auditing Amazon ES Domains with CloudTrail](#).

- **Amazon Machine Learning** (Supported beginning 12/10/2015)

Amazon Machine Learning makes it easy for developers to build smart applications, including applications for fraud detection, demand forecasting, targeted marketing, and click prediction. The powerful algorithms of Amazon Machine Learning create machine learning (ML) models by finding patterns in your existing data. For more information, see the [Amazon Machine Learning Developer Guide](#). For information on the Amazon ML API calls logged by CloudTrail, see [Logging Amazon ML API Calls By Using AWS CloudTrail](#).

Application Services

- **Amazon API Gateway** (Supported beginning 07/09/2015)

Amazon API Gateway helps developers deliver robust, reliable, secure and scalable access to backend APIs for mobile apps, web apps, and server apps. For more information, see the [API Gateway Developer Guide](#). For more information about the Amazon API Gateway calls logged by CloudTrail, see [Logging Amazon API Gateway API Calls By Using AWS CloudTrail](#).

- **Amazon CloudSearch** (Supported beginning 10/16/2014)

Amazon CloudSearch is a fully-managed service in the cloud that makes it easy to set up, manage, and scale a search solution for your website. Amazon CloudSearch enables you to search large collections of data such as web pages, document files, forum posts, or product information. For more information about Amazon CloudSearch, see the [Amazon CloudSearch Developer Guide](#). For more information about the Amazon CloudSearch calls logged by CloudTrail, see [Logging Amazon CloudSearch Configuration Service Calls Using AWS CloudTrail](#).

- **Amazon Elastic Transcoder** (Supported beginning 10/27/2014)

Amazon Elastic Transcoder lets you convert media files that you have stored in Amazon S3 into media files in the formats required by consumer playback devices. For more information about Elastic Transcoder, see [Amazon Elastic Transcoder Developer Guide](#). For more information about the Elastic Transcoder calls logged by CloudTrail, see [Logging Elastic Transcoder API Calls Using CloudTrail](#).

- **Amazon Simple Email Service** (Supported beginning 05/07/2015)

Amazon Simple Email Service is an outbound-only email-sending service that provides an easy, cost-effective way for you to send email. For more information about Amazon Simple Email Service, see the [Amazon Simple Email Service Developer Guide](#). For more information about the Amazon SES calls logged by CloudTrail, see [Logging Amazon SES API Calls By Using AWS CloudTrail](#).

- **Amazon Simple Queue Service** (Supported beginning 07/16/2014)

Amazon Simple Queue Service (Amazon SQS) offers reliable and scalable hosted queues for storing messages as they travel between computers. By using Amazon SQS, you can move data between distributed components of your applications that perform different tasks without losing messages or requiring each component to be always available. For more information, see the [Amazon Simple Queue Service Developer Guide](#). For more information about the Amazon SQS calls logged by CloudTrail, see [Logging Amazon SQS API Calls By Using AWS CloudTrail](#).

- **Amazon Simple Workflow Service** (Supported beginning 05/13/2014)

The Amazon Simple Workflow Service (Amazon SWF) makes it easy to build applications that coordinate work across distributed components. In Amazon SWF, a task represents a logical unit of work that is performed by a component of your application. Coordinating tasks across the application involves managing intertask dependencies, scheduling, and concurrency in accordance with the logical flow of the application. Amazon SWF gives you full control over implementing tasks and coordinating them without worrying about underlying complexities such as tracking their progress and maintaining their state. For more information, see the [Amazon SWF Developer Guide](#). For more information about the Amazon SWF calls logged by CloudTrail, see [Logging Amazon Simple Workflow Service API Calls with AWS CloudTrail](#).

Compute

- **Amazon Elastic Compute Cloud (EC2)** (Supported beginning 11/13/2013)

Amazon Elastic Compute Cloud (Amazon EC2) provides resizable computing capacity in the AWS cloud. You can launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 can also scale up or down quickly to handle changes in requirements

or spikes in popularity, thereby reducing your need to forecast server traffic. For more information about Amazon EC2, see the [Amazon EC2 User Guide](#). For more information about the Amazon EC2 calls logged by CloudTrail, see [Logging Amazon EC2 API Calls Using AWS CloudTrail](#). [Amazon EC2 Simple Systems Manager \(SSM\)](#) is a feature of [EC2Config](#) that enables you to manage the configuration of running Windows instances. For information about the Amazon EC2 Simple Systems Manager API calls logged by CloudTrail, see [Logging SSM API Calls Using AWS CloudTrail](#).

- **Amazon EC2 Container Service** (Supported beginning 04/09/2015)

Amazon EC2 Container Service (Amazon ECS) is a highly scalable, fast, container management service that makes it easy to run, stop, and manage Docker containers on a cluster of Amazon EC2 instances. For more information about Amazon EC2 Container Service, see the [Amazon EC2 Container Service Developer Guide](#). For more information about the Amazon ECS calls logged by CloudTrail, see [Logging Amazon ECS API Calls By Using AWS CloudTrail](#).

- **AWS Elastic Beanstalk** (Supported beginning 03/31/2014)

You can use Elastic Beanstalk to quickly deploy and manage applications in the AWS cloud without worrying about the infrastructure that runs those applications. For more information, see the [Elastic Beanstalk Developer's Guide](#). For more information about the Elastic Beanstalk calls logged by using CloudTrail, see [Using AWS Elastic Beanstalk with AWS CloudTrail](#).

- **AWS Lambda** (Supported beginning 04/09/2015)

AWS Lambda is a zero-administration compute platform that runs your code in the AWS cloud, providing the high availability, security, performance, and scalability of AWS infrastructure. For more information about Lambda, see the [AWS Lambda Developer Guide](#). For more information about the Lambda calls logged by CloudTrail, see [Logging AWS Lambda API Calls By Using AWS CloudTrail](#).

- **Auto Scaling** (Supported beginning 07/16/2014)

Auto Scaling is a web service that enables you to automatically launch or terminate Amazon Elastic Compute Cloud (Amazon EC2) instances based on user-defined policies, health status checks, and schedules. For more information, see the [Auto Scaling Developer Guide](#). For a list of the Auto Scaling calls logged by CloudTrail, see [Logging Auto Scaling API Calls By Using CloudTrail](#).

- **Elastic Load Balancing** (Supported beginning 04/04/2014)

You can use Elastic Load Balancing to automatically distribute your incoming application traffic across multiple Amazon EC2 instances. Elastic Load Balancing automatically scales request handling capacity in response to incoming traffic. For more information about Elastic Load Balancing, see the [Elastic Load Balancing User Guide](#). For more information about the Elastic Load Balancing calls logged by CloudTrail, see [Logging Elastic Load Balancing API Calls Using AWS CloudTrail](#).

- **Amazon EC2 Container Registry** (Supported beginning 12/21/2015)

Amazon EC2 Container Registry (Amazon ECR) is a secure and scalable managed AWS Docker registry service. Amazon ECR supports private Docker repositories with resource-level permissions. You can use the Docker CLI to author, push, pull, and manage images. For more information about Amazon ECR, see the [Amazon EC2 Container Registry User Guide](#). For more information about the Amazon ECR calls logged by CloudTrail, see [Logging Amazon ECR API Calls By Using AWS CloudTrail](#).

Database

- **AWS Database Migration Service** (Supported beginning 02/04/2016)

AWS Database Migration Service (AWS DMS) can migrate your data to and from most widely used commercial and open-source databases such as Oracle, PostgreSQL, Microsoft SQL Server, Amazon Aurora, MariaDB, and MySQL. For more information, see the [AWS Database Migration Service User Guide](#). For more information about the AWS DMS calls logged by CloudTrail, see [Logging AWS Database Migration Service API Calls Using AWS CloudTrail](#).

- **Amazon DynamoDB** (Supported beginning 05/28/2015)

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. For more information, see the [Amazon DynamoDB Developer Guide](#). For more information about the DynamoDB calls logged by CloudTrail, see [Logging DynamoDB API Calls By Using AWS CloudTrail](#).

- **Amazon ElastiCache** (Supported beginning 09/15/2014)

Amazon ElastiCache is a web service that makes it easy to set up, manage, and scale distributed in-memory cache environments in the cloud. It provides a high performance, resizable, and cost-effective in-memory cache, while removing the complexity associated with deploying and managing a distributed cache environment. For more information, see the [Amazon ElastiCache User Guide](#). For more information about the ElastiCache calls logged by CloudTrail, see [Logging Amazon ElastiCache API Calls Using AWS CloudTrail](#).

- **Amazon Relational Database Service** (Supported beginning 11/13/2013)

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks. For more information, see the [Amazon RDS User Guide](#). For more information about the Amazon RDS calls logged by CloudTrail, see [Logging Amazon RDS API Calls Using CloudTrail](#).

Developer Tools

- **AWS CodeDeploy** (Supported beginning 12/16/2014)

AWS CodeDeploy is a deployment service that enables developers to automate the deployment of applications to Amazon Elastic Compute Cloud (Amazon EC2) instances, and to update the applications as required. For more information, see the [AWS CodeDeploy User Guide](#). For more information about the AWS CodeDeploy calls logged by CloudTrail, see [Logging CodeDeploy API Calls By Using AWS CloudTrail](#).

- **AWS CodePipeline** (Supported beginning 07/09/2015)

AWS CodePipeline is a continuous delivery and automation service hosted by Amazon Web Services that enables you to model, configure, and automate the steps required to release your software. For more information, see the [AWS CodePipeline User Guide](#). For more information about the AWS CodePipeline calls logged by CloudTrail, see [Logging AWS CodePipeline API Calls By Using AWS CloudTrail](#).

Enterprise Applications

- **Amazon WorkSpaces** (Supported beginning 04/09/2015)

Amazon WorkSpaces offers an easy way to provide a cloud-based desktop experience to your end-users. A choice of bundles offer a range of different amounts of CPU, memory, storage, and a choice of applications. Users can connect from a PC, Mac desktop computer, iPad, Kindle, or Android tablet. For more information about Amazon WorkSpaces, see [Amazon WorkSpaces Administration Guide](#). For more information about the Amazon WorkSpaces actions logged by CloudTrail, see [Logging Amazon WorkSpaces API Calls by Using CloudTrail](#).

- **Amazon WorkDocs** (Supported beginning 08/27/2014)

Amazon WorkDocs is a fully managed enterprise storage and sharing service. Your files are stored in the cloud safely and securely. Amazon WorkDocs also includes a synchronization application that keeps selected folders on your local computer in sync with your files in the cloud. Your files are visible to only you and to your designated contributors and viewers. For more information about Amazon

WorkDocs, see [Amazon WorkDocs Administration Guide](#). For more information about the Amazon WorkDocs actions logged by CloudTrail, see [Logging Amazon WorkDocs API Calls By Using CloudTrail](#).

Internet of Things

- **AWS IoT** (Supported beginning 04/11/2016)

AWS IoT provides secure, bi-directional communication between Internet-connected things (such as sensors, actuators, embedded devices, or smart appliances) and the AWS cloud. This enables you to collect telemetry data from multiple devices and store and analyze the data. For more information about AWS IoT, see the [AWS IoT Developer Guide](#). For more information about the list of AWS IoT calls logged by CloudTrail, see [Logging AWS IoT API calls with AWS CloudTrail](#).

Game Development

- **Amazon GameLift** (Supported beginning 01/27/2016)

Amazon GameLift is a fully managed service for deploying, operating, and scaling session-based multiplayer game servers in the cloud. You can deploy your first game server in the cloud in just minutes, eliminating up to thousands of hours in upfront software development. For more information about GameLift, see the [Amazon GameLift Developer Guide](#). For more information about the list of GameLift calls logged by CloudTrail, see [Logging Amazon GameLift API Calls with AWS CloudTrail](#).

Management Tools

- **AWS CloudFormation** (Supported beginning 04/02/2014)

AWS CloudFormation enables you to create and provision AWS infrastructure deployments predictably and repeatedly. It helps you leverage AWS products such as Amazon EC2, Amazon EBS, Amazon SNS, Elastic Load Balancing, and Auto Scaling to build highly reliable, highly scalable, cost-effective applications without worrying about creating and configuring the underlying AWS infrastructure. For more information, see the [AWS CloudFormation User Guide](#). For more information about the AWS CloudFormation calls logged by CloudTrail, see [Logging AWS CloudFormation API Calls in AWS CloudTrail](#).

- **AWS CloudTrail** (Supported beginning 11/13/2013)

Like any supported service, when logging is turned on, CloudTrail logs actions to an Amazon S3 bucket that you specify. For a complete list of the actions that can be logged, see the [AWS CloudTrail API Reference](#).

- **Amazon CloudWatch** (Supported beginning 04/30/2014)

Amazon CloudWatch monitors your AWS resources and the applications you run on AWS in real time. You can use CloudWatch to collect and track metrics which are the variables you want to measure for your resources and applications. CloudWatch alarms send notifications or automatically make changes to the resources you are monitoring based on rules that you define. For more information, see the [Amazon CloudWatch Developer Guide](#). For more information about the list of CloudWatch calls logged by CloudTrail, see [Logging Amazon CloudWatch API Calls in AWS CloudTrail](#).

- **Amazon CloudWatch Events** (Supported beginning 01/16/2016)

Amazon CloudWatch Events delivers a timely stream of system events that describe changes in AWS resources to AWS Lambda functions, streams in Amazon Kinesis Streams, Amazon SNS topics, or built-in targets. Using simple rules that you can set up quickly, you can match events and route them to one or more target functions or streams. For more information, see the [Amazon CloudWatch Developer](#)

[Guide](#). For more information about the list of CloudWatch Events calls logged by CloudTrail, see [Logging CloudWatch API Calls in AWS CloudTrail](#).

- **Amazon CloudWatch Logs** (Supported beginning 03/10/2016)

Amazon CloudWatch Logs monitors, stores, and accesses your log files from Amazon EC2 instances, AWS CloudTrail, and other sources. You can then retrieve the associated log data from CloudWatch Logs. For more information, see the [Amazon CloudWatch Developer Guide](#). For more information about the list of CloudWatch Logs calls logged by CloudTrail, see [Logging Amazon CloudWatch API Calls in AWS CloudTrail](#).

- **AWS Config** (Supported beginning 02/10/2015)

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. For more information, see the [AWS Config Developer Guide](#). For information about the AWS Config calls logged by CloudTrail, see [Logging AWS Config API Calls By Using AWS CloudTrail](#).

- **AWS OpsWorks** (Supported beginning 06/04/2014)

AWS OpsWorks provides a simple and flexible way to create and manage stacks and applications. It supports a standard set of components—including application servers, database servers, load balancers, and more—that you can use to assemble your stack. These components all come with a standard configuration and are ready to run. For more information, see the [AWS OpsWorks User Guide](#). For more information about the list of AWS OpsWorks calls logged by CloudTrail, see [Logging AWS OpsWorks API Calls By Using AWS CloudTrail](#).

- **AWS Service Catalog** (Supported beginning 07/06/2016)

AWS Service Catalog allows organizations to create and manage catalogs of IT services that are approved for use on AWS. These IT services can include everything from virtual machine images, servers, software, and databases to complete multi-tier application architectures. For more information, see the [AWS Service Catalog Developer Guide](#). For more information about the list of AWS Service Catalog calls logged by CloudTrail, see [Logging AWS Service Catalog API Calls with AWS CloudTrail](#).

Mobile Services

- **Amazon Cognito** (Supported beginning 02/18/2016)

Amazon Cognito is a service that you can use to create unique identities for your users, authenticate these identities with identity providers, and save mobile user data in the AWS cloud. For more information, see the [Amazon Cognito Developer Guide](#). For information about the Amazon Cognito calls logged by CloudTrail, see [Logging Amazon Cognito API Calls with AWS CloudTrail](#).

- **AWS Device Farm** (Supported beginning 07/13/2015)

AWS Device Farm is an app testing service that enables you to test your Android and Fire OS apps on real, physical phones and tablets that are hosted by Amazon Web Services (AWS). For more information about AWS Device Farm, see the [Device Farm Developer Guide](#). For information about the AWS Device Farm calls logged by CloudTrail, see [Logging AWS Device Farm API Calls By Using AWS CloudTrail](#).

- **Amazon Simple Notification Service** (Supported beginning 10/09/2014)

Amazon Simple Notification Service (Amazon SNS) is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients. For more information about Amazon SNS, see the [Amazon Simple Notification Service Developer Guide](#). For more information about the Amazon SNS calls logged by CloudTrail, see [Logging Amazon Simple Notification Service API Calls By Using AWS CloudTrail](#).

Networking

- **AWS Direct Connect** (Supported beginning 03/08/2014)

You can use AWS Direct Connect to establish a direct connection from your premises to AWS. This may reduce your network costs and increase bandwidth throughput. For more information about AWS Direct Connect, see the [AWS Direct Connect User Guide](#). For more information about the AWS Direct Connect calls logged by CloudTrail, see [Logging AWS Direct Connect API Calls in AWS CloudTrail](#).

- **Amazon Route 53** (Supported beginning 02/11/2015)

Amazon Route 53 is a Domain Name System (DNS) and domain name registration web service. To use Amazon Route 53 with CloudTrail, you must choose US East (N. Virginia) as the region when you create the trail. For more information about Amazon Route 53, see the [Amazon Route 53 Developer Guide](#). For information about the Amazon Route 53 calls logged by CloudTrail, see [Using AWS CloudTrail to Capture Requests Sent to the Amazon Route 53 API](#).

- **Amazon Virtual Private Cloud** (Supported beginning 11/13/2013)

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you would operate in your own data center with the added benefit of using the scalable AWS infrastructure. For more information, see the [Amazon Virtual Private Cloud User Guide](#). The Amazon VPC API is a subset of the Amazon EC2 API. For more information about the Amazon EC2 calls logged by CloudTrail (including those for Amazon VPC), see [Logging Amazon EC2 API Calls Using AWS CloudTrail](#).

Security and Identity

- **AWS Certificate Manager** (Supported beginning 03/25/2016)

AWS Certificate Manager (ACM) handles the complexity of provisioning, deploying, and managing certificates provided by ACM (ACM Certificates) for your AWS-based websites and applications. For more information, see the [AWS Certificate Manager User Guide](#). For more information about the ACM calls logged by CloudTrail, see [Using AWS CloudTrail](#).

- **AWS CloudHSM** (Supported beginning 01/08/2015)

AWS CloudHSM provides secure cryptographic key storage to customers by making hardware security modules (HSMs) available in the AWS cloud. For more information, see the [AWS CloudHSM Getting Started Guide](#). For a complete list of AWS CloudHSM calls logged by CloudTrail, see [Logging AWS CloudHSM API Calls By Using AWS CloudTrail](#).

- **AWS Directory Service** (Supported beginning 05/14/2015)

The AWS Directory Service is a managed service that makes it easy to connect to your existing on-premises Microsoft Active Directory and deploy and manage Windows workloads in the AWS cloud. For more information about AWS Directory Service, see the [AWS Directory Service Administration Guide](#). For information about the AWS Directory Service calls logged by CloudTrail, see [Logging AWS Directory Service API Calls by Using CloudTrail](#).

- **AWS Identity and Access Management** (Supported beginning 11/13/2013)

AWS Identity and Access Management (IAM) is a web service that enables AWS customers to manage users and user permissions. By using IAM, you can centrally manage users, security credentials such as access keys, and permissions that control which AWS resources users can access. For more information, see the [IAM User Guide](#). For more information about the IAM calls logged by CloudTrail, see [Logging IAM Events with AWS CloudTrail](#).

- **Amazon Inspector** (Supported beginning 04/20/2016)

Amazon Inspector enables you to analyze the behavior of your AWS resources and helps you to identify potential security issues. With Amazon Inspector, you can define a collection of AWS resources that you want to include in an assessment target. For more information, see the [Amazon Inspector User Guide](#). For more information about the Amazon Inspector calls logged by CloudTrail, see [Logging Amazon Inspector API calls with AWS CloudTrail](#).

- **AWS Key Management Service** (Supported beginning 11/12/2014)

The AWS Key Management Service is a managed service that makes it easy for you to create and control the encryption keys used to encrypt your data. AWS KMS is integrated with other AWS services including Amazon EBS, Amazon S3, and Amazon Redshift. For more information, see the [AWS Key Management Service Developer Guide](#). For more information about the list of AWS KMS calls logged by CloudTrail, see [Logging AWS KMS API Calls](#).

- **AWS Security Token Service** (Supported beginning 11/13/2013)

You can use the AWS Security Token Service (STS) to grant a trusted user temporary, limited access to your AWS resources. For more information, see the [AWS Security Token Service User Guide](#). For information about both the IAM and AWS STS calls logged by CloudTrail, see [Logging IAM Events with AWS CloudTrail](#). For a complete list of AWS STS calls, see the [AWS Security Token Service API Reference](#).

- **AWS WAF** (Supported beginning 04/28/2016)

AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to Amazon CloudFront and lets you control access to your content. For more information, see the [AWS WAF Developer Guide](#). For more information about the list of AWS WAF calls logged by CloudTrail, see [Logging AWS WAF API Calls with AWS CloudTrail](#).

Storage and Content Delivery

- **Amazon CloudFront** (Supported beginning 05/28/2014)

Amazon CloudFront speeds up distribution of your static and dynamic web content to end users. CloudFront delivers your content through a worldwide network of edge locations. When an end user requests content that you're serving with CloudFront, the user is routed to the edge location that provides the lowest latency, so that content is delivered with the best possible performance. For more information, see the [Amazon CloudFront Developer Guide](#). For more information about the CloudFront calls logged by CloudTrail, see [Using AWS CloudTrail to Capture Requests Sent to the CloudFront API](#).

- **Amazon EBS** (Supported beginning 08/09/2013)

Amazon EBS allows you to create persistent storage volumes and attach them to Amazon EC2 instances. Once attached, you can create a file system on top of these volumes, run a database, or use them in any other way you would use a block device. Amazon EBS volumes are placed in a specific Availability Zone, where they are automatically replicated to protect you from the failure of a single component. For more information about Amazon EBS, see the [Amazon EC2 User Guide](#). For more information about the Amazon EBS calls logged by CloudTrail, see [Logging API Calls Using AWS CloudTrail](#).

- **Amazon Glacier** (Supported beginning 12/11/2014)

Amazon Glacier is a storage service optimized for data archiving and backup of infrequently used data. The service is durable, extremely low-cost, and includes security features. For more information, see the [Amazon Glacier Developer Guide](#). For more information about the Amazon Glacier calls logged by CloudTrail, see [Logging Amazon Glacier API Calls By Using AWS CloudTrail](#).

- **Amazon S3 bucket level events** (Supported beginning 09/01/2015)

You can use Amazon Simple Storage Service (Amazon S3) to store and retrieve any amount of data at any time, from anywhere on the web. For more information, see the [Amazon Simple Storage Service Developer Guide](#). With this release, CloudTrail logs Amazon S3 bucket-related events such as the creation and deletion of buckets, changes to bucket policy, and changes to replication status. You can

also use CloudTrail logs together with Amazon S3 server access logs. For detailed information, see [Logging Amazon S3 API Calls By Using AWS CloudTrail](#).

- **AWS Storage Gateway** (Supported beginning 12/16/2014)

AWS Storage Gateway is a service that connects an on-premises software appliance with cloud-based storage to provide seamless and secure integration between your on-premises IT environment and the AWS storage infrastructure in the cloud. For more information about AWS Storage Gateway, see the [AWS Storage Gateway User Guide](#). For information about the AWS Storage Gateway Volume Gateway calls logged by CloudTrail, see [Logging Volume Gateway API Calls by Using AWS CloudTrail](#).

Support

- **AWS Support** (Supported beginning 04/21/2016)

AWS Support offers a range of plans that provide access to tools and expertise that support the success and operational health of your AWS solutions. All support plans provide 24x7 access to customer service, AWS documentation, whitepapers, and support forums. For more information about AWS Support, see the [AWS Support User Guide](#). For more information about the list of AWS Support API calls logged by CloudTrail, see [Logging AWS Support API Calls with AWS CloudTrail](#).

CloudTrail Topics by AWS Service

See the CloudTrail topics for specific AWS services.

AWS Service	CloudTrail Topics
Amazon API Gateway	Logging Amazon API Gateway API Calls By Using AWS CloudTrail
Auto Scaling	Logging Auto Scaling API Calls By Using CloudTrail
AWS Certificate Manager	Using AWS CloudTrail
AWS CloudFormation	Logging AWS CloudFormation API Calls in AWS CloudTrail
Amazon CloudFront	Using AWS CloudTrail to Capture Requests Sent to the CloudFront API
AWS CloudHSM	Logging AWS CloudHSM API Calls By Using AWS CloudTrail
Amazon CloudSearch	Logging Amazon CloudSearch Configuration Service Calls Using AWS CloudTrail
Amazon CloudWatch, CloudWatch Events, and CloudWatch Logs	Logging Amazon CloudWatch API Calls in AWS CloudTrail
AWS CodeDeploy	Logging CodeDeploy API Calls By Using AWS CloudTrail
AWS CodePipeline	Logging AWS CodePipeline API Calls By Using AWS CloudTrail
Amazon Cognito	Logging Amazon Cognito API Calls with AWS CloudTrail

AWS CloudTrail User Guide
CloudTrail Topics by AWS Service

AWS Service	CloudTrail Topics
AWS Config	Logging AWS Config API Calls By Using AWS CloudTrail
AWS Data Pipeline	Logging AWS Data Pipeline API Calls by using AWS CloudTrail
AWS Database Migration Service (AWS DMS)	Logging AWS Database Migration Service API Calls Using AWS CloudTrail
AWS Device Farm	Logging AWS Device Farm API Calls By Using AWS CloudTrail
AWS Direct Connect	Logging AWS Direct Connect API Calls in AWS CloudTrail
AWS Directory Service	Logging AWS Directory Service API Calls by Using CloudTrail
Amazon DynamoDB	Logging DynamoDB API Calls By Using AWS CloudTrail
Amazon EC2 Container Registry (Amazon ECR)	Logging Amazon ECR API Calls By Using AWS CloudTrail
Amazon EC2 Container Service (Amazon ECS)	Logging Amazon ECS API Calls By Using AWS CloudTrail
AWS Elastic Beanstalk (Elastic Beanstalk)	Using AWS Elastic Beanstalk with AWS CloudTrail
Amazon Elastic Block Store (Amazon EBS)	Logging API Calls Using AWS CloudTrail
Amazon Elastic Compute Cloud (Amazon EC2)	Logging API Calls Using AWS CloudTrail
Elastic Load Balancing	Logging Elastic Load Balancing API Calls Using AWS CloudTrail
Amazon EMR (Amazon EMR)	Logging Amazon EMR API Calls in AWS CloudTrail
Amazon Elastic Transcoder	Logging Elastic Transcoder API Calls Using CloudTrail
Amazon ElastiCache	Logging Amazon ElastiCache API Calls Using AWS CloudTrail
Amazon Elasticsearch Service	Auditing Amazon ES Domains with CloudTrail
Amazon GameLift	Logging Amazon GameLift API Calls with AWS CloudTrail
Amazon Glacier	Logging Amazon Glacier API Calls By Using AWS CloudTrail
AWS Identity and Access Management (IAM)	Logging IAM Events with AWS CloudTrail
Amazon Inspector	Logging Amazon Inspector API calls with AWS CloudTrail
AWS IoT	Logging AWS IoT API Calls with AWS CloudTrail
AWS Key Management Service (AWS KMS)	Logging AWS KMS API Calls

AWS CloudTrail User Guide
CloudTrail Topics by AWS Service

AWS Service	CloudTrail Topics
Amazon Kinesis Firehose	Logging Amazon Kinesis Firehose API Calls with AWS CloudTrail
Amazon Kinesis Streams	Logging Amazon Kinesis Streams API Calls Using AWS CloudTrail
AWS Lambda	Logging AWS Lambda API Calls By Using AWS CloudTrail AWS Lambda Walkthrough 5: Handling AWS CloudTrail Events Using the AWS CLI (Node.js)
Amazon Machine Learning	Logging Amazon ML API Calls By Using AWS CloudTrail
AWS OpsWorks	Logging AWS OpsWorks API Calls By Using AWS CloudTrail
Amazon Relational Database Service (Amazon RDS)	Logging Amazon RDS API Calls Using CloudTrail
Amazon Route 53	Using AWS CloudTrail to Capture Requests Sent to the Amazon Route 53 API
AWS Security Token Service (AWS STS)	Logging IAM Events with AWS CloudTrail The IAM topic includes information for AWS STS.
AWS Service Catalog	Logging AWS Service Catalog API Calls with AWS CloudTrail
Amazon S3 Bucket Level Events	Logging Amazon S3 API Calls By Using AWS CloudTrail
Amazon Simple Email Service (Amazon SES)	Logging Amazon SES API Calls By Using AWS CloudTrail
Amazon Simple Notification Service (Amazon SNS)	Logging Amazon Simple Notification Service API Calls By Using AWS CloudTrail
Amazon Simple Queue Service (Amazon SQS)	Logging Amazon SQS API Calls By Using AWS CloudTrail
Amazon EC2 Simple Systems Manager (SSM)	Logging SSM API Calls Using AWS CloudTrail
Amazon Simple Workflow Service (Amazon SWF)	Logging Amazon Simple Workflow Service API Calls with AWS CloudTrail
AWS Storage Gateway	Logging Volume Gateway API Calls by Using AWS CloudTrail
AWS Support	Logging AWS Support API Calls with AWS CloudTrail
Amazon Virtual Private Cloud (Amazon VPC)	Logging Amazon EC2 API Calls Using AWS CloudTrail The Amazon VPC API is a subset of the Amazon EC2 API.

AWS Service	CloudTrail Topics
AWS WAF	Logging AWS WAF API Calls with AWS CloudTrail
Amazon WorkDocs	Logging Amazon WorkDocs API Calls By Using CloudTrail
Amazon WorkSpaces	Logging Amazon WorkSpaces API Calls by Using CloudTrail

CloudTrail Supported Regions

Region Name	Region	Endpoint	Protocol	AWS Account ID	Support Date	Supports Amazon CloudWatch Logs
US East (N. Virginia)	us-east-1	cloudtrail.us-east-1.amazonaws.com	HTTPS	086441151436	11/13/2013	Yes
US West (N. California)	us-west-1	cloudtrail.us-west-1.amazonaws.com	HTTPS	388731089494	05/13/2014	Yes
US West (Oregon)	us-west-2	cloudtrail.us-west-2.amazonaws.com	HTTPS	113285607260	11/13/2013	Yes
Asia Pacific (Mumbai)	ap-south-1	cloudtrail.ap-south-1.amazonaws.com	HTTPS	977081816279	06/27/2016	Yes
Asia Pacific (Seoul)	ap-northeast-2	cloudtrail.ap-northeast-2.amazonaws.com	HTTPS	492519147666	01/06/2016	Yes
Asia Pacific (Singapore)	ap-southeast-1	cloudtrail.ap-southeast-1.amazonaws.com	HTTPS	903692715234	06/30/2014	Yes
Asia Pacific (Sydney)	ap-southeast-2	cloudtrail.ap-southeast-2.amazonaws.com	HTTPS	284668455005	05/13/2014	Yes
Asia Pacific (Tokyo)	ap-northeast-1	cloudtrail.ap-northeast-1.amazonaws.com	HTTPS	216624486486	06/30/2014	Yes

Region Name	Region	Endpoint	Protocol	AWS Account ID	Support Date	Supports Amazon CloudWatch Logs
EU (Frankfurt)	eu-central-1	cloudtrail.eu-central-1.amazonaws.com	HTTPS	035351147821	10/23/2014	Yes
EU (Ireland)	eu-west-1	cloudtrail.eu-west-1.amazonaws.com	HTTPS	859597730677	05/13/2014	Yes
South America (São Paulo)	sa-east-1	cloudtrail.sa-east-1.amazonaws.com	HTTPS	814480443879	06/30/2014	Yes

For information about CloudTrail in the AWS GovCloud (US) region, see [AWS CloudTrail in the AWS GovCloud \(US\) User Guide](#).

CloudTrail Log File Examples

CloudTrail monitors events for your account and then delivers those events as log files to your Amazon S3 bucket. See the following to learn more about log files.

Contents

- [CloudTrail Log File Name Format \(p. 20\)](#)
- [Log File Examples \(p. 21\)](#)
 - [Amazon EC2 Log Examples \(p. 21\)](#)
 - [IAM Log Examples \(p. 23\)](#)
 - [Error Code and Message Log Example \(p. 26\)](#)

CloudTrail Log File Name Format

CloudTrail uses the following file name format for the log file objects that it delivers to your Amazon S3 bucket:

```
AccountID_CloudTrail_RegionName_YYYYMMDDTHHmmZ_UniqueString.FileNameFormat
```

- The `YYYY`, `MM`, `DD`, `HH`, and `mm` are the digits of the year, month, day, hour, and minute when the log file was delivered. Hours are in 24-hour format. The `z` indicates that the time is in UTC.

Note

A log file delivered at a specific time can contain records written at any point before that time.

- The 16-character `UniqueString` component of the log file name is there to prevent overwriting of files. It has no meaning, and log processing software should ignore it.
- `FileNameFormat` is the encoding of the file. Currently, this is `json.gz`, which is a JSON text file in compressed gzip format.

Example CloudTrail Log File Name

```
111122223333_CloudTrail_ap-northeast-1_20150801T0210Z_Mu0KsOhtHlar15ZZ.json.gz
```

Log File Examples

A log file contains one or more records. The following examples are snippets of logs that show the records for an action that started the creation of a log file.

Amazon EC2 Log Examples

Amazon Elastic Compute Cloud (Amazon EC2) provides resizable computing capacity in the AWS Cloud. You can launch virtual servers, configure security and networking, and manage storage. Amazon EC2 can also scale up or down quickly to handle changes in requirements or spikes in popularity, thereby reducing your need to forecast server traffic. For more information, see the [Amazon EC2 User Guide](#).

The following example shows that an IAM user named Alice used the AWS CLI to call the Amazon EC2 `startInstances` action by using the `ec2-start-instances` command for instance `i-ebeaf9e2`.

```
{
  "Records": [{
    "eventVersion": "1.0",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn:aws:iam::123456789012:user/Alice",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "accountId": "123456789012",
      "userName": "Alice"
    },
    "eventTime": "2014-03-06T21:22:54Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.176",
    "userAgent": "ec2-api-tools 1.6.12.2",
    "requestParameters": {
      "instancesSet": {
        "items": [{
          "instanceId": "i-ebeaf9e2"
        }]
      }
    },
    "responseElements": {
      "instancesSet": {
        "items": [{
          "instanceId": "i-ebeaf9e2",
          "currentState": {
            "code": 0,
            "name": "pending"
          },
          "previousState": {
            "code": 80,
            "name": "stopped"
          }
        }]
      }
    }
  }]
}
```

```
        }
    },
    ... additional entries ...
]
}
```

The following example shows that an IAM user named Alice used the AWS CLI to call the Amazon EC2 `stopInstances` action by using the `ec2-stop-instances`.

```
{
  "Records": [{
    "eventVersion": "1.0",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn:aws:iam::123456789012:user/Alice",
      "accountId": "123456789012",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "userName": "Alice"
    },
    "eventTime": "2014-03-06T21:01:59Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StopInstances",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.176",
    "userAgent": "ec2-api-tools 1.6.12.2",
    "requestParameters": {
      "instancesSet": {
        "items": [{
          "instanceId": "i-ebeaf9e2"
        }]
      },
      "force": false
    },
    "responseElements": {
      "instancesSet": {
        "items": [{
          "instanceId": "i-ebeaf9e2",
          "currentState": {
            "code": 64,
            "name": "stopping"
          },
          "previousState": {
            "code": 16,
            "name": "running"
          }
        }]
      }
    }
  },
  ... additional entries ...
]
}
```

The following example shows that the Amazon EC2 console back-end called the `CreateKeyPair` action in response to requests initiated by the IAM user Alice. Note that the `responseElements` contain a hash of the key pair and that the key material has been removed by AWS.

```
{
  "Records": [{
    {
      "eventVersion": "1.0",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2014-03-06T15:15:06Z"
          }
        }
      },
      "eventTime": "2014-03-06T17:10:34Z",
      "eventSource": "ec2.amazonaws.com",
      "eventName": "CreateKeyPair",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "72.21.198.64",
      "userAgent": "EC2ConsoleBackend, aws-sdk-java/Linux/x.xx.fleetxen
Java_HotSpot(TM)_64-Bit_Server_VM/xx",
      "requestParameters": {
        "keyName": "mykeypair"
      },
      "responseElements": {
        "keyName": "mykeypair",
        "keyFingerprint":
"30:1d:46:d0:5b:ad:7e:1b:b6:70:62:8b:ff:38:b5:e9:ab:5d:b8:21",
        "keyMaterial": "\u003csensitiveDataRemoved\u003e"
      },
      ... additional entries ...
    }
  ]
}
```

IAM Log Examples

AWS Identity and Access Management (IAM) is a web service that enables AWS customers to manage users and user permissions. With IAM, you can manage users, security credentials such as access keys, and permissions that control which AWS resources users can access. For more information, see the [IAM User Guide](#)

The following example shows that the IAM user Alice used the AWS CLI to call the `CreateUser` action to create a new user named Bob.

```
{
  "Records": [{
```

```
"eventVersion": "1.0",
"userIdentity": {
  "type": "IAMUser",
  "principalId": "EX_PRINCIPAL_ID",
  "arn": "arn:aws:iam::123456789012:user/Alice",
  "accountId": "123456789012",
  "accessKeyId": "EXAMPLE_KEY_ID",
  "userName": "Alice"
},
"eventTime": "2014-03-24T21:11:59Z",
"eventSource": "iam.amazonaws.com",
"eventName": "CreateUser",
"awsRegion": "us-east-1",
"sourceIPAddress": "127.0.0.1",
"userAgent": "aws-cli/1.3.2 Python/2.7.5 Windows/7",
"requestParameters": {
  "userName": "Bob"
},
"responseElements": {
  "user": {
    "createDate": "Mar 24, 2014 9:11:59 PM",
    "userName": "Bob",
    "arn": "arn:aws:iam::123456789012:user/Bob",
    "path": "/",
    "userId": "EXAMPLEUSERID"
  }
}
}]
}
```

The following example shows that the IAM user Alice used the AWS Management Console to call the **AddUserToGroup** action to add Bob to the administrator group.

```
{
  "Records": [
    {
      "eventVersion": "1.0",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2014-03-25T18:45:11Z"
          }
        }
      },
      "eventTime": "2014-03-25T21:08:14Z",
      "eventSource": "iam.amazonaws.com",
      "eventName": "AddUserToGroup",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "127.0.0.1",

```

```
        "userAgent": "AWSConsole",
        "requestParameters": {
            "userName": "Bob",
            "groupName": "admin"
        },
        "responseElements": null
    },
    ...additional entries
]
}
```

The following example shows that the IAM user Alice used the AWS CLI to call the **CreateRole** action to create a new IAM role.

```
{
  "Records": [{
    "eventVersion": "1.0",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn:aws:iam::123456789012:user/Alice",
      "accountId": "123456789012",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "userName": "Alice"
    },
    "eventTime": "2014-03-25T20:17:37Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "CreateRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-cli/1.3.2 Python/2.7.5 Windows/7",
    "requestParameters": {
      "assumeRolePolicyDocument": "{\n  \"Version\": \"2012-10-17\",\n  \"Statement\": [\n    {\n      \"Sid\": \"\",\n      \"Effect\": \"Allow\",\n      \"Principal\": {\n        \"AWS\": \"arn:aws:iam::210987654321:root\"\n      },\n      \"Action\": \"sts:AssumeRole\"\n    }\n  ]\n}",
      "roleName": "TestRole"
    },
    "responseElements": {
      "role": {
        "assumeRolePolicyDocument": "%7B%0A%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20%22Statement%22%3A%20%5B%0A%20%20%20%20%7B%0A%20%20%20%22Sid%22%3A%20%22%22%2C%0A%20%20%20%20%22Effect%22%3A%20%22Allow%22%2C%0A%20%20%20%20%22Principal%22%3A%20%7B%0A%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws%3Aiam%3A%3A803981987763%3Aroot%22%0A%20%20%20%20%20%20%7D%2C%0A%20%20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0A%20%20%20%20%7D%0A%20%20%5D%0A%7D",
        "roleName": "TestRole",
        "roleId": "AROAIUU2EOWSWPGX2UJUO",
        "arn": "arn:aws:iam::123456789012:role/TestRole",
        "createDate": "Mar 25, 2014 8:17:37 PM",
        "path": "/"
      }
    }
  ]
}
```

```
}  
  }  
}
```

Error Code and Message Log Example

The following example shows that the IAM user Alice used the AWS CLI to call the `UpdateTrail` action to update a trail named `myTrail2`, but the trail name was not found. The log shows this error in the `errorCode` and `errorMessage` elements.

```
{  
  "Records":  
  [  
    {  
      "eventVersion": "1.04",  
      "userIdentity": {  
        "type": "IAMUser",  
        "principalId": "EX_PRINCIPAL_ID",  
        "arn": "arn:aws:iam::123456789012:user/Alice",  
        "accountId": "123456789012",  
        "accessKeyId": "EXAMPLE_KEY_ID",  
        "userName": "Alice"  
      },  
      "eventTime": "2016-07-14T19:15:45Z",  
      "eventSource": "cloudtrail.amazonaws.com",  
      "eventName": "UpdateTrail",  
      "awsRegion": "us-west-2",  
      "sourceIPAddress": "205.251.233.182",  
      "userAgent": "aws-cli/1.10.32 Python/2.7.9 Windows/7 botocore/1.4.22",  
  
      "errorCode": "TrailNotFoundException",  
      "errorMessage": "Unknown trail: myTrail2 for the user: 123456789012",  
      "requestParameters": {"name": "myTrail2"},  
      "responseElements": null,  
      "requestID": "5d40662a-49f7-11e6-97e4-d9cb6ff7d6a3",  
      "eventID": "b7d4398e-b2f0-4faa-9c76-e2d316a8d67f",  
      "eventType": "AwsApiCall",  
      "recipientAccountId": "123456789012"  
    }  
  ]  
}
```

Getting Started with CloudTrail

CloudTrail enables you to retrieve a history of API calls and other events for all of the regions in your account. This includes calls and events made by the AWS Management Console and command line tools, by any of the AWS SDKs, or by other AWS services. The following topics discuss how to get started with CloudTrail.

Topics

- [Creating and Updating Your Trail \(p. 27\)](#)
- [Viewing Events with CloudTrail API Activity History \(p. 45\)](#)
- [Controlling User Permissions for CloudTrail \(p. 81\)](#)
- [Getting and Viewing Your CloudTrail Log Files \(p. 88\)](#)
- [Configuring Amazon SNS Notifications for CloudTrail \(p. 90\)](#)

Creating and Updating Your Trail

This section shows you how to create a trail for your AWS account. You can create trails by using the AWS CloudTrail console or the AWS Command Line Interface (AWS CLI). Both methods follow the same steps:

1. Turn on CloudTrail. By default, when you create a trail in one region in the CloudTrail console, the trail will apply to all regions.
2. Create a new Amazon S3 bucket for storing your log files, or specify an existing bucket where you want the log files delivered. By default, log files from all AWS regions in your account will be delivered to the bucket you specify.
3. (Optional) Create a new Amazon SNS topic in order to receive notifications when new log files are delivered. Log file delivery notifications from all regions are sent to the topic that you specify.
4. (Optional) Configure CloudWatch Logs to receive your logs from CloudTrail so that you can monitor for specific kinds of log events.
5. (Optional) Turn on log file encryption. This encrypts your files for added security. For more information, see [Encrypting CloudTrail Log Files with AWS KMS–Managed Keys \(SSE-KMS\) \(p. 152\)](#).
6. (Optional) Turn on log file integrity validation. This enables the delivery of digest files that you can use to validate the integrity of log files after CloudTrail has delivered them. [Validating CloudTrail Log File Integrity \(p. 163\)](#).
7. After a trail is created, you can also configure other features such as tagging. For more information, see [Updating a Trail \(p. 32\)](#).

Tips for managing trails

- You can view all trails from any region in the console.
- To edit a trail in the list, click the trail name. The console will navigate to the region to which the trail belongs so that you can edit it.
- Having at least one trail that applies to all regions will ensure that you receive log files from all the regions in your account.
- If you require that log files from a particular region be stored in an AWS bucket in that same region, you can deselect the option to apply the trail to all regions when you create or update a trail. This might be useful, for example, if you want to keep your CloudWatch Logs alarms separated by region, or if you want to give business units in different regions autonomous control over CloudTrail logging.

Topics

- [Creating and Updating a Trail with the CloudTrail Console \(p. 28\)](#)
- [Creating and Updating a Trail with the AWS CLI \(p. 34\)](#)
- [CloudTrail Trail Naming Requirements \(p. 42\)](#)
- [Amazon S3 Bucket Naming Requirements \(p. 42\)](#)
- [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#)
- [AWS KMS Alias Naming Requirements \(p. 45\)](#)

Creating and Updating a Trail with the CloudTrail Console

Topics

- [Creating a Trail for the First Time \(p. 28\)](#)
- [Adding a Trail \(p. 30\)](#)
- [Updating a Trail \(p. 32\)](#)
- [Deleting a Trail \(p. 33\)](#)
- [Turning off Logging for a Trail \(p. 34\)](#)

Creating a Trail for the First Time

The following steps create a trail which applies to all regions in your account. The trail is based in the region that you choose in the console when you create the trail. In these steps, you also create an Amazon S3 bucket to receive your log files. Log files from all regions will be delivered to your bucket. Optionally, you can configure Amazon SNS to notify you when log files are delivered.

Note

You can use an existing bucket, but we recommend that you create a new one. When you create a new bucket, CloudTrail creates the necessary IAM policies for you and applies them to the bucket.

To create a CloudTrail trail with the AWS Management Console

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
2. Choose **Get Started Now**.

The **Turn on CloudTrail** page appears.

3. Choose the region where you want the trail to be based.
4. In the **Trail name** box, type a name for your trail. For trail naming requirements, see [CloudTrail Trail Naming Requirements \(p. 42\)](#).
5. For **Apply trail to all regions?**, choose **Yes** to receive log files from all regions. **Yes** is the default (and recommended) setting.
6. For **Create a new S3 bucket?**, choose **Yes** to create a new bucket or **No** to use an existing one.

Note

If you chose **No**, choose an existing Amazon S3 bucket. The bucket policy must grant CloudTrail permission to write to it. You can only specify an existing bucket owned by the account under which the trail is created. For information about manually editing the policy for a bucket, see [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#).

7. In the **S3 bucket** field, type a name for the bucket you want to designate for log file storage. The name must be globally unique. For more information about S3 bucket naming rules and conventions, see [Amazon S3 Bucket Naming Requirements \(p. 42\)](#).
8. (Optional) If you want to enter a log file prefix for your bucket, enable log file validation, or configure Amazon SNS notifications, choose **Advanced**.
9. (Advanced option) In the **Log file prefix** field, type a prefix for your Amazon S3 bucket. The prefix is an addition to the URL for an Amazon S3 object that helps create a folder-like organization in your bucket. The location where your log files will be stored is shown under the text field.
10. (Advanced option) For **Enable log file validation**, choose **Yes** to have log digests delivered to your S3 bucket. You can use the digest files to verify that your log files did not change after CloudTrail delivered them. By default, this feature is enabled for new trails. For more information, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).
11. (Advanced option) For **Send SNS notification for every log file delivery**, choose **Yes** if you want to be notified each time a log is delivered to your bucket. CloudTrail stores multiple events in a log file. SNS notifications are sent for every log file, not for every event.
 - For **Create a new SNS topic**, choose **Yes** to create a new topic, or choose **No** to use an existing topic. If you are creating a trail that applies to all regions, SNS notifications for log file deliveries in all regions will be sent to the single SNS topic that you create.

Note

If you chose **No**, choose an existing topic. You can also enter the ARN of a topic from another region or from an account with appropriate permissions. For more information, see [Configuring CloudTrail to Send Notifications \(p. 91\)](#).

12. If you chose **Yes**, in the **SNS topic** field, type a name.

If you create a topic, you must subscribe to the topic to be notified of log file delivery. You can subscribe from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see the [Amazon Simple Notification Service Getting Started Guide](#).

13. Choose **Turn On**.

The new trail will appear on the **Trails** page, which shows your trails from all regions. In about 15 minutes, CloudTrail publishes log files that show the AWS API calls made in your account.

Next Steps

See [Updating a Trail \(p. 32\)](#) if you want to do the following:

- Enable log file encryption.
- Make changes to the Amazon S3 bucket or Amazon SNS notifications for your trail.

- Change the delivery of global service events for a trail that does not receive logs from all regions; for more information, see [About global service events \(p. 6\)](#).
- Add custom tags (key-value pairs) to the trail.
- You can also configure CloudTrail to send log files to CloudWatch Logs. See [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#).

Adding a Trail

You can have any combination of up to five trails in one region. This can be useful when different user groups such as developers and administrators need separate copies of log files. For more information, see [Multiple trails per region \(p. 6\)](#).

Additional trails can incur charges. A trail that applies to all regions counts as one trail in every region. For CloudTrail pricing information, see [AWS CloudTrail Pricing](#).

Follow these steps if you want to create additional trails. If you are new to CloudTrail, see [Creating a Trail for the First Time \(p. 28\)](#).

To add a trail with the AWS Management Console

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
2. Navigate to the **Trails** page of the CloudTrail console for the region where you want to create the trail.
3. Choose **Add new trail**.
4. On the **Create Trail** page, type a name for your trail in the **Trail Name** box. For trail naming requirements, see [CloudTrail Trail Naming Requirements \(p. 42\)](#).
5. For the **Apply trail to all regions?** option, choose **Yes** if you want to receive log files from all regions. **Yes** is the default setting. Choose **No** if you want to receive log files only from the region in which you are creating the trail.
6. For **Create a new S3 bucket?**, choose **Yes** to create a new bucket or **No** to use an existing one.

Note

If you chose **No**, choose an existing Amazon S3 bucket. The bucket policy must grant CloudTrail permission to write to the bucket. You can only specify an existing bucket owned by the account under which the trail is created. For information about manually editing the policy for a bucket, see [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#).

7. If you chose **Yes**, in the **S3 bucket** field, type a name for the bucket you want to designate for log file storage. The name must be globally unique. For more information about S3 bucket naming rules and conventions, see [Amazon S3 Bucket Naming Requirements \(p. 42\)](#).
8. (Optional) If you want to enter a prefix for your bucket, enable log file encryption, enable log file validation, or configure Amazon SNS notifications, choose **Advanced**.
9. (Advanced option) In the **Log file prefix** field, type a prefix for your Amazon S3 bucket. The prefix is an addition to the URL for an Amazon S3 object that helps create a folder-like organization in your bucket. The location where your log files will be stored is shown under the text field.
10. (Advanced option) For **Encrypt log files**, choose **Yes** if you want AWS KMS to encrypt your log files.
 - For **Create a new KMS key**, choose **Yes** to create a new key or **No** to use an existing one.

Note

If you chose **No**, choose an existing KMS key. You can also type the ARN of a key from another account. For more information, see [Updating a Trail to Use Your CMK \(p. 161\)](#). The key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form. For

information about manually editing the key policy, see [AWS KMS Key Policy for CloudTrail \(p. 154\)](#).

11. If you chose **Yes**, in the **KMS key** field, type an alias. CloudTrail encrypts your log files with the key and adds the policy for you.
12. (Advanced option) For **Enable log file validation**, choose **Yes** to have log digests delivered to your S3 bucket. You can use the digest files to verify that your log files did not change after CloudTrail delivered them. By default, this feature is enabled for new trails. For more information, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).
13. (Advanced option) For **Send SNS notification for every log file delivery**, choose **Yes** if you want to be notified each time a log is delivered to your bucket. CloudTrail stores multiple events in a log file. SNS notifications are sent for every log file, not for every event.
 - For **Create a new SNS topic**, choose **Yes** to create a new topic, or choose **No** to use an existing topic. If you are creating a trail that applies to all regions, SNS notifications for log file deliveries in all regions will be sent to the single SNS topic that you create.

Note

If you chose **No**, choose an existing topic. You can also enter the ARN of a topic from another region or from an account with appropriate permissions. For more information, see [Configuring CloudTrail to Send Notifications \(p. 91\)](#).

14. If you chose **Yes**, in the **SNS topic** field, type a name.

If you create a topic, you must subscribe to the topic to be notified of log file delivery. You can subscribe from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see the [Amazon Simple Notification Service Getting Started Guide](#).

15. Choose **Create**.

The new trail will appear on the **Trails** page. In about 15 minutes, CloudTrail starts publishing log files that show the AWS API calls made in your account.

Next Steps

- If you created an SNS topic, you must subscribe to it to get notifications (for example, via email) of your log file deliveries. You can do this from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see the [Amazon Simple Notification Service Getting Started Guide](#).

Note

CloudTrail stores multiple events in a log file. SNS notifications are sent per log file delivery, not per event.

See [Updating a Trail \(p. 32\)](#) if you want to do the following:

- Enable log file encryption.
- Make changes to the Amazon S3 bucket or Amazon SNS notifications for your trail.
- Change the delivery of global service events for a trail that does not receive logs from all regions; for more information, see [About global service events \(p. 6\)](#).
- Add custom tags (key-value pairs) to the trail.
- You can also configure CloudTrail to send log files to CloudWatch Logs. See [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#).

Updating a Trail

To update a trail with the AWS Management Console

1. Navigate to the **Trails** page of the CloudTrail console for the region of the trail that you want to configure.
2. Choose the trail name.
3. To receive log files from all regions, or to receive log files only from the region in which you created the trail, click the pencil icon next to **Apply trail to all regions**.
4. To edit the settings for your S3 bucket, including log file encryption, log file validation, and SNS notifications, click the pencil icon to the right of the **S3** section.
5. For the **Create a new S3 bucket?** option, choose **Yes** to create a new bucket or **No** to use an existing one.

Note

If you chose **No**, choose an existing Amazon S3 bucket. The bucket policy must grant CloudTrail permission to write to the bucket. You can only specify an existing bucket owned by the account under which the trail is created. For information about manually editing the policy for a bucket, see [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#).

6. If you chose **Yes**, in the **S3 bucket** field, type a name for the bucket you want to designate for log file storage. The name must be globally unique. For more information about S3 bucket naming rules and conventions, see [Amazon S3 Bucket Naming Requirements \(p. 42\)](#).

Note

When you designate a different bucket for an existing trail, we recommend that you use the same bucket name prefix that you used previously. Otherwise, you must manually update the bucket policy with the changed prefix.

7. (Optional) If you want to enter a log file prefix for your bucket, enable log file encryption, enable log file validation, or configure SNS notifications, choose **Advanced**.
8. (Advanced option) In the **Log file prefix** field, type a prefix for your Amazon S3 bucket. The prefix is an addition to the URL for an Amazon S3 object that helps create a folder-like organization in your bucket. The location where your log files will be stored is shown under the text field.
9. (Advanced option) For **Encrypt log files**, choose **Yes** if you want AWS KMS to encrypt your log files.
 - For **Create a new KMS key**, choose **Yes** to create a new key or **No** to use an existing one.

Note

If you chose **No**, choose an existing KMS key. You can also type the ARN of a key from another account. For more information, see [Updating a Trail to Use Your CMK \(p. 161\)](#). The key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form. For information about manually editing the key policy, see [AWS KMS Key Policy for CloudTrail \(p. 154\)](#).

10. If you chose **Yes**, in the **KMS key** field, type an alias. CloudTrail encrypts your log files with the key and adds the policy for you.
11. (Advanced option) For **Enable log file validation**, choose **Yes** to have log digests delivered to your S3 bucket. You can use the digest files to verify that your log files did not change after CloudTrail delivered them. For more information, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).
12. (Advanced option) For **Send SNS notification for every log file delivery**, choose **Yes** if you want to be notified each time a log is delivered to your bucket. CloudTrail stores multiple events in a log file. SNS notifications are sent for every log file, not for every event.

- For **Create a new SNS topic**, choose **Yes** to create a new topic, or choose **No** to use an existing topic. If you are updating a trail that applies to all regions, SNS notifications for log file deliveries in all regions will be sent to the single SNS topic that you create.

Note

If you chose **No**, choose an existing topic. You can also enter the ARN of a topic from another region or from an account with appropriate permissions. For more information, see [Configuring CloudTrail to Send Notifications \(p. 91\)](#).

13. If you chose **Yes**, in the **SNS topic** field, type a name.

If you create a topic, you must subscribe to the topic to be notified of log file delivery. You can subscribe from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see the [Amazon Simple Notification Service Getting Started Guide](#).

14. To configure CloudTrail to deliver events to CloudWatch Logs for real-time monitoring, choose **Configure** under the **CloudWatch Logs** section. For more information about these settings, see [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#).
15. Choose **Save** to save your settings for the S3 configuration section.
16. To configure tags (custom key-value pairs) for your trail, click the pencil icon to the right of the **Tags** section. You can add up to 50 key-value pairs per trail. Trail tags must be configured from the region in which the trail was created. You cannot add, view, or edit tags for a trail in one region from the console of another region.
17. (Single Region trails only) To include or exclude global services such as such as IAM or AWS STS, click the pencil icon to the right of **Additional Configuration** to change the **Include global services** option. When enabled, this option records API calls from global services.

Note

The include global services setting is always on for trails that apply to all regions. If your trail applies to all regions, global service events will be delivered from only one region so that you will not receive duplicate copies of global service events. If you have two or more trails that apply to only one region, include global services, and use the same bucket, duplicate global service events will be delivered to your bucket. To avoid this, include global services in the configuration for only one of your trails, or use a trail that applies to all regions instead. For more information, see [About global service events \(p. 6\)](#).

Note

You cannot rename a trail after it has been created. Instead, you can delete the trail and create a new one.

Deleting a Trail

You can delete trails with the CloudTrail console. If you want to delete a trail that receives log files from all regions, you must choose the region where you originally created the trail.

To delete a trail with the CloudTrail console

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
2. Navigate to the **Trails** page of the CloudTrail console for the region in which the trail was created.
3. Choose the trail name.
4. At the top of the configuration page, click the trash icon.
5. Choose **Delete** to delete the trail permanently. The trail will be removed from the list of trails for the region. Log files that were already delivered will not be deleted.

Turning off Logging for a Trail

When you create a trail, logging is turned on automatically. You can turn off logging for a trail. Previous logs will still be accessible.

To turn off logging for a trail with the CloudTrail console

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
2. In the navigation pane, choose **Trails**, and then choose the trail that you want to configure.
3. At the top of the configuration page, choose **Logging** to turn off logging for the trail.
4. When the **stop logging** message appears, choose **Continue**. CloudTrail stops logging activity for that trail.
5. To resume logging for that trail, choose **Logging** again.

Creating and Updating a Trail with the AWS CLI

Note

The AWS CLI commands in this topic require that you have the AWS command line tools. For more information, see the [AWS Command Line Interface User Guide](#). For help with CloudTrail commands at the AWS CLI command line, type `aws cloudtrail help`.

Two Options for Creating and Updating Trails

When choosing to create or update a trail with the CLI, you have two sets of options: `create-trail` and `update-trail`, and `create-subscription` and `update-subscription`.

`create-trail` and `update-trail`

`create-trail` and `update-trail` offer the following functionality that `create-subscription` and `update-subscription` do not:

- You can create a trail that receives logs across regions, or update a trail to do so, by using the `--is-multi-region-trail` option.
- You can convert a multi-region trail to single-region trail by using the `--no-is-multi-region-trail` option.
- You can enable or disable log file encryption with the `--kms-key-id` parameter. The parameter specifies an AWS KMS key that you have already created and to which you have attached a policy that allows CloudTrail to encrypt your logs. For more information, see [Enabling and disabling CloudTrail log file encryption with the AWS CLI \(p. 162\)](#).
- You can enable or disable log file validation with the `--enable-log-file-validation` and `--no-enable-log-file-validation` options. For information about log file validation, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).
- You can specify a CloudWatch Logs log group and role so that CloudTrail can deliver events to a CloudWatch Logs log group. For information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 95\)](#).

`create-subscription` and `update-subscription`

`create-subscription` and `update-subscription` offer the following advantages:

- You can have CloudTrail create a new S3 bucket for you. With `create-trail`, you must specify an existing bucket to which you have already applied the bucket policy for CloudTrail separately.

- `create-subscription` starts logging on the trail for you. With `create-trail`, you must issue a `start-logging` command separately.

Using create-trail

Creating a trail that applies to one region

The following command creates a trail that applies only to the current region. In order for the command to succeed, the S3 bucket specified must already exist and have the appropriate CloudTrail permissions applied. For information, see [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#).

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name my-bucket
```

For trail naming requirements, see [CloudTrail Trail Naming Requirements \(p. 42\)](#).

Sample output:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "my-trail",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": false,
  "S3BucketName": "my-bucket"
}
```

Notice that, unlike the trails you create in the CloudTrail console, the trails created by `create-trail` by default apply only to a single region. By default, trails created by `create-trail` do not have log file validation enabled.

Start logging on the trail

After the `create-trail` command completes, you must use the `start-logging` command to start logging on the trail that was just created.

Note

When you create a trail by using the CloudTrail console or the `create-subscription` command, logging on the trail is started for you automatically.

To start logging on a trail, use the following syntax:

```
aws cloudtrail start-logging --name my-trail
```

This command produces no output. However, you can verify that logging has started by using the `get-trail-status` command as follows:

```
aws cloudtrail get-trail-status --name my-trail
```

In the output, the `IsLogging` response element will be `true`, as in the following example:

```
{
  "LatestDeliveryTime": 1441139757.497,
  "LatestDeliveryAttemptTime": "2015-09-01T20:35:57Z",
  "LatestNotificationAttemptSucceeded": "2015-09-01T20:35:57Z",
  "LatestDeliveryAttemptSucceeded": "2015-09-01T20:35:57Z",
  "IsLogging": true,
  "TimeLoggingStarted": "2015-09-01T00:54:02Z",
}
```



```
"StartLoggingTime": 1441068842.76,  
"LatestDigestDeliveryTime": 1441140723.629,  
"LatestNotificationAttemptTime": "2015-09-01T20:35:57Z",  
"TimeLoggingStopped": ""  
}
```

Creating a trail that applies to all regions

To create a trail that applies to all regions, you must use the `--is-multi-region-trail` parameter.

The following example creates a trail that delivers logs from all regions into a pre-existing bucket named *my-bucket*:

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name my-bucket  
--is-multi-region-trail
```

To confirm that your trail exists in all regions, the `IsMultiRegionTrail` element in the output shows true:

```
{  
  "IncludeGlobalServiceEvents": true,  
  "Name": "my-trail",  
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",  
  "LogFileValidationEnabled": false,  
  "IsMultiRegionTrail": true,  
  "S3BucketName": "my-bucket"  
}
```

Tip

Don't forget to use the `start-logging` command at this point to start logging on your trail.

Creating a trail that applies to all regions and that has log file validation enabled

To enable log file validation when using `create-trail`, you must use the `--enable-log-file-validation` parameter.

Note

For information about log file validation, see [Validating CloudTrail Log File Integrity \(p. 163\)](#).

The following example command creates a trail that delivers logs from all regions into the bucket specified. The command uses the `--enable-log-file-validation` parameter.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name my-bucket  
--is-multi-region-trail --enable-log-file-validation
```

To confirm that log file validation has been enabled, the `LogFileValidationEnabled` element in the output shows true:

```
{  
  "IncludeGlobalServiceEvents": true,  
  "Name": "my-trail",  
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",  
  "LogFileValidationEnabled": true,  
  "IsMultiRegionTrail": true,  
}
```



```
}
  "S3BucketName": "my-bucket"
}
```

Using update-trail

You can use the `update-trail` command to change the configuration settings for a trail.

Note

If you use the AWS CLI or one of the AWS SDKs to modify a trail, be sure that the trail's bucket policy is up to date. In order for your bucket to automatically receive events from a new AWS region when the new region launches, the policy must contain the full service name, `cloudtrail.amazonaws.com`, as shown on the [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#) page.

Note

The `update-trail` command must be called from the region in which the trail was created.

Converting a trail that applies to one region to apply to all regions

To change an existing a trail so that it applies to all regions, you must use the `--is-multi-region-trail` parameter, as the following example shows:

```
aws cloudtrail update-trail --name my-trail --is-multi-region-trail
```

To confirm that the trail now applies to all regions, the `IsMultiRegionTrail` element in the output shows `true`:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "my-trail",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
  "S3BucketName": "my-bucket"
}
```

Converting a trail that applies to all regions to apply to a single region

To change an existing a trail so that it applies to only to the region in which it was created, you must use the `--no-is-multi-region-trail` parameter, as the following example shows.

```
aws cloudtrail update-trail --name my-trail --no-is-multi-region-trail
```

To confirm that the trail now applies to a single region, the `IsMultiRegionTrail` element in the output shows `false`:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "my-trail",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": false,
  "S3BucketName": "my-bucket"
}
```

Enabling log file validation

To enable log file validation for a trail, use the `--enable-log-file-validation` parameter, as the following example shows. Digest files will be delivered to the S3 bucket for the trail.

```
aws cloudtrail update-trail --name my-trail --enable-log-file-validation
```

To confirm that log file validation is enabled, the `LogFileValidationEnabled` element in the output shows `true`:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "my-trail",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": true,
  "IsMultiRegionTrail": false,
  "S3BucketName": "my-bucket"
}
```

Disabling log file validation

To disable log file validation for a trail, use the `--no-enable-log-file-validation` parameter, as the following example shows.

```
aws cloudtrail update-trail --name my-trail-name --no-enable-log-file-validation
```

To confirm that log file validation is disabled, the `LogFileValidationEnabled` element in the output shows `false`:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "my-trail",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": false,
  "S3BucketName": "my-bucket"
}
```

Note

For information on using the AWS CLI to validate log files, see [Validating CloudTrail Log File Integrity with the AWS CLI](#) (p. 165).

Using create-subscription

The `create-subscription` command creates a trail. Unlike `create-trail`, it can create a new Amazon S3 bucket for log file delivery and a new Amazon SNS topic for notifications. Unlike `create-trail`, the command also starts logging on the trail that it creates.

The `create-subscription` command includes the following options:

- `--name` specifies the name of the trail. This parameter is required. For trail naming requirements, see [CloudTrail Trail Naming Requirements](#) (p. 42).
- `--s3-use-bucket` specifies an existing Amazon S3 bucket for log file storage.
- `--s3-new-bucket` specifies the name of the new bucket created when the command executes. The name of the bucket must be globally unique. For more information, see [Amazon S3 Bucket Naming Requirements](#) (p. 42).

- `--s3-prefix` specifies a prefix for the log file delivery path (optional). The maximum length is 200 characters.

Note

If you want to use a new log file prefix for an existing bucket, add the prefix to the bucket policy first. For more information, see [When adding, changing, or removing a prefix for an existing bucket](#) (p. 44).

- `--sns-new-topic` specifies the name of the Amazon SNS topic to which you can subscribe for notification of log file delivery to your bucket (optional).

Tip

For a full list of options, type `aws cloudtrail create-subscription help` at the command line.

The following example syntax creates a trail, a new S3 bucket for log file delivery, an S3 bucket prefix, and a new SNS topic.

```
aws cloudtrail create-subscription --name=awscloudtrail-example --s3-new-bucket=awscloudtrail-new-bucket-example --s3-prefix=prefix-example --sns-new-topic=awscloudtrail-example-log-deliverytopic
```

If the command executes successfully, you see output similar to the following:

```
Setting up new S3 bucket awscloudtrail-new-bucket-example...
Setting up new SNS topic awscloudtrail-example-log-deliverytopic...
Creating/updating CloudTrail configuration...
CloudTrail configuration:
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "awscloudtrail-example",
      "S3KeyPrefix": "prefix-example",
      "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/awscloudtrail-example",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "awscloudtrail-new-bucket-example"
      "SnsTopicName": "awscloudtrail-example-log-deliverytopic",
      "HomeRegion": "us-west-2"
    }
  ],
  "ResponseMetadata": {
    "HTTPStatusCode": 200,
    "RequestId": "4c55c744-a0ea-4aea-b3b9-eb63dfe68383"
  }
}
Starting CloudTrail service...
Logs will be delivered to awscloudtrail-new-bucket-example:prefix-example
```

Using update-subscription

You can update your trail by using the command `update-subscription` and setting the options to new values. The following example uses the `--s3-use-bucket` parameter to designate a different pre-existing

Amazon S3 bucket. If you want a trail with a different name, you can delete the trail by using the `delete-trail` command and running `create-subscription` again.

```
aws cloudtrail update-subscription --name=awscloudtrail-example --s3-use-bucket=awscloudtrail-new-bucket-example2--s3-prefix=prefix-example
```

If the command executes successfully, you see that the `S3BucketName` value has changed:

```
CloudTrail configuration:
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "awscloudtrail-example",
      "S3KeyPrefix": "prefix-example",
      "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/awscloudtrail-example",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "awscloudtrail-new-bucket-example2"
      "SnsTopicName": "awscloudtrail-example-log-deliverytopic",
      "HomeRegion": "us-west-2"
    }
  ]
}
```

Tip

If you designate an existing bucket for log file publication that was not created with CloudTrail, see [Amazon S3 Bucket Policy for CloudTrail \(p. 42\)](#) to apply the necessary policy to the bucket.

Managing Trails with AWS CLI Commands

The CloudTrail CLI includes several other commands that help you manage your trails. This section demonstrates how to use these commands.

Retrieving Trail Settings and the Status of a Trail

Use the `describe-trails` command to retrieve trail settings:

```
aws cloudtrail describe-trails
```

If the command succeeds, you see output similar to the following:

```
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "my-trail",
      "S3KeyPrefix": "my-prefix",
      "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/my-trail",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "my-bucket"
      "SnsTopicName": "my-topic",
    }
  ]
}
```

```
    "HomeRegion": "us-west-2"  
  }  
]  
}
```

Use the `get-trail-status` command to retrieve the status of a trail:

```
aws cloudtrail get-trail-status --name awscloudtrail-example
```

If the command succeeds, you see output similar to the following:

```
{  
  "LatestDeliveryTime": 1441139757.497,  
  "LatestDeliveryAttemptTime": "2015-09-01T20:35:57Z",  
  "LatestNotificationAttemptSucceeded": "2015-09-01T20:35:57Z",  
  "LatestDeliveryAttemptSucceeded": "2015-09-01T20:35:57Z",  
  "IsLogging": true,  
  "TimeLoggingStarted": "2015-09-01T00:54:02Z",  
  "StartLoggingTime": 1441068842.76,  
  "LatestDigestDeliveryTime": 1441140723.629,  
  "LatestNotificationAttemptTime": "2015-09-01T20:35:57Z",  
  "TimeLoggingStopped": ""  
}
```

In addition to the fields shown in the preceding JSON code, the status contains the following fields if there are Amazon SNS or Amazon S3 errors:

- `LatestNotificationError`. Contains the error emitted by Amazon SNS if a subscription to a topic fails.
- `LatestDeliveryError`. Contains the error emitted by Amazon S3 if CloudTrail cannot deliver a log file to a bucket.

Stopping and Starting Logging for a Trail

The following commands start and stop CloudTrail logging, respectively.

```
aws cloudtrail start-logging --name awscloudtrail-example
```

```
aws cloudtrail stop-logging --name awscloudtrail-example
```

Caution

Before deleting a bucket, you should use `stop-logging` to end all logging to the bucket. If you don't stop logging, CloudTrail will continue to attempt to deliver log files to a bucket of the same name for a limited period of time.

Deleting a Trail

You can delete a trail using the following command. The command must be executed in the region in which the trail was created.

```
aws cloudtrail delete-trail --name awscloudtrail-example
```

When you delete a trail, you do not delete either the Amazon S3 bucket or the Amazon SNS topic associated with it. If you want to delete these items, do so separately using the AWS Management Console, AWS CLI, or service API.

CloudTrail Trail Naming Requirements

CloudTrail trail names must meet the following requirements:

- Contain only ASCII letters (a-z, A-Z), numbers (0-9), periods (.), underscores (_), or dashes (-).
- Start with a letter or number, and end with a letter or number.
- Be between 3 and 128 characters.
- Have no adjacent periods, underscores or dashes. Names like my-_namespace and my-\-namespace are invalid.
- Not be in IP address format (for example, 192.168.5.4).

Amazon S3 Bucket Naming Requirements

The Amazon S3 bucket that you use to store CloudTrail log files must have a name that conforms with naming requirements for non-US Standard regions. Amazon S3 defines a bucket name as a series of one or more labels, separated by periods, that adhere to the following rules:

- The bucket name can be between 3 and 63 characters long, and can contain only lower-case characters, numbers, periods, and dashes.
- Each label in the bucket name must start with a lowercase letter or number.
- The bucket name cannot contain underscores, end with a dash, have consecutive periods, or use dashes adjacent to periods.
- The bucket name cannot be formatted as an IP address (198.51.100.24).

Caution

Because S3 allows your bucket to be used as a URL that can be accessed publicly, the bucket name that you choose must be globally unique. If some other account has already created a bucket with the name that you chose, you must use another name. For more information, see [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

Amazon S3 Bucket Policy for CloudTrail

By default, all Amazon S3 buckets and objects are private. Only the resource owner, the AWS account that created the bucket, can access the bucket and any objects it contains. The resource owner can, however, choose to grant access permissions to other resources and users by writing an access policy. The following topics discuss the bucket policy necessary to enable CloudTrail to write log files to a bucket from supported regions:

Topics

- [Creating an Amazon S3 Bucket Policy by using the CloudTrail Console or CLI](#) (p. 43)
- [Using an Existing Bucket for CloudTrail Logs](#) (p. 43)
- [Receiving log files from other regions and accounts in your Amazon S3 bucket](#) (p. 44)
- [Troubleshooting Amazon S3 Bucket Policy Errors](#) (p. 44)

Creating an Amazon S3 Bucket Policy by using the CloudTrail Console or CLI

When you turn on CloudTrail in the console or in the AWS CLI and create an Amazon S3 bucket, CloudTrail automatically attaches a policy to the new bucket. The policy enables CloudTrail to write log files to the bucket from any of the regions listed on the CloudTrail [CloudTrail Supported Regions \(p. 19\)](#) page.

CloudTrail automatically fills in the following fields in the policy for you:

- The allowed SIDs.
- The name of the folder where the log files will be stored.
- The service principal name for current and future CloudTrail supported regions.
- The bucket name.
- The optional prefix if you specified one at creation.
- The ID of the owning account.

If you created your bucket in this way, you do not need to edit the policy so that CloudTrail can write to it.

Using an Existing Bucket for CloudTrail Logs

If you specified an existing Amazon S3 bucket as the location for log file delivery when you turned on CloudTrail, you must attach a policy to the bucket that allows CloudTrail to write to the bucket.

This section shows you how to use the Amazon S3 console to edit a policy with the necessary permissions for CloudTrail and apply it to an existing Amazon S3 bucket.

For ease of maintenance, it is recommended that you use a dedicated Amazon S3 bucket for CloudTrail logs.

To add the policy required by CloudTrail to an Amazon S3 bucket

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Select the bucket where you want CloudTrail to deliver your log files, and then click **Properties**.
3. Choose **Permissions**.
4. Choose **Edit Bucket Policy**.
5. Copy the following policy into the **Bucket Policy Editor** window and then substitute the correct names of your bucket, prefix, and account number for the placeholders indicated in italics. If you specified a prefix when you created your trail, be sure to include it here. The prefix is an optional addition to the Amazon S3 object key that helps create a folder-like organization in your bucket.

The following policy enables CloudTrail to write log files to your bucket from any current or future [CloudTrail Supported Regions \(p. 19\)](#).

Caution

If the existing bucket already has one or more policies attached to it, add the statements for CloudTrail access to that policy or policies. We recommend that you evaluate the resulting set of permissions to be sure that they are appropriate for the users who will be accessing the bucket.

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

```
{
  "Sid": "AWSCloudTrailAclCheck20150319",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "s3:GetBucketAcl",
  "Resource": "arn:aws:s3:::myBucketName"
},
{
  "Sid": "AWSCloudTrailWrite20150319",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::myBucketName/[optional prefix]/AWSLogs/myAccountID/*",
  "Condition": {
    "StringEquals": {
      "s3:x-amz-acl": "bucket-owner-full-control"
    }
  }
}
]
```

Receiving log files from other regions and accounts in your Amazon S3 bucket

Typically, when you create a trail, you specify the region that CloudTrail will deliver log files from. However, CloudTrail can also aggregate log files from other regions and accounts into your Amazon S3 bucket as long as those regions have write access to your bucket. For more information, see [Receiving CloudTrail Log Files from Multiple Regions](#) (p. 94).

Troubleshooting Amazon S3 Bucket Policy Errors

When adding, changing, or removing a prefix for an existing bucket

If you try to add, modify, or remove a log file prefix for the existing Amazon S3 bucket of a CloudTrail configuration, you might receive the error **There is a problem with the bucket policy**. To resolve this issue, first use the Amazon S3 console to update the prefix in the bucket's policy, and then use the CloudTrail console to specify the same prefix for the bucket in CloudTrail configuration.

To update the log file prefix for an Amazon S3 bucket

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Select the bucket for which you want to modify the prefix, and then click **Properties**.
3. Click **Permissions**.
4. Click **Edit Bucket Policy**.
5. In the bucket policy, under the `s3:PutObject` action, edit the `Resource` entry to add, modify, or remove the log file *prefix* as required.


```
"Action": "s3:PutObject",  
  "Resource": "arn:aws:s3:::myBucketName/prefix/AWSLogs/myAccountID/*",
```

6. Click **Save**.
7. Open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
8. On the **Configuration** page, click the pencil icon in upper right corner of the **S3** box to edit the settings for your S3 bucket.
9. Click **Advanced**.
10. For **S3 bucket**, choose the bucket whose prefix you are changing.
11. In the **Log file prefix** box, update the prefix to match the prefix that you entered in the S3 bucket policy.
12. Click **Save**.

AWS KMS Alias Naming Requirements

When you create a customer master key (CMK), you can choose an alias to identify it. For example, you might choose the alias "KMS-CloudTrail-us-west-2" to encrypt the logs for a specific trail.

The alias must meet the following requirements:

- Between 1 and 32 characters, inclusive
- Contain alphanumeric characters (A-Z, a-z, 0-9), hyphens (-), forward slashes (/), and underscores (_)
- Cannot begin with **aws**

For more information, see [Creating Keys](#) in the *AWS Key Management Service Developer Guide*.

Viewing Events with CloudTrail API Activity History

To troubleshoot operational and security incidents, you can use the CloudTrail **API Activity History** feature. This feature lets you look up and filter events captured by CloudTrail. You can look up events related to the creation, modification, or deletion of resources in your AWS account on a per-region basis. Events can be looked up by using the AWS CloudTrail console, or programmatically by using the AWS SDKs or AWS Command Line Interface. This section describes how to look up events by using the CloudTrail console and the AWS CLI. For information on using the LookupEvents API to retrieve information from CloudTrail events, see the [AWS CloudTrail API Reference](#).

Note

When you stop logging, CloudTrail stops delivering events to your Amazon S3 bucket. As a result, any events that occurred when CloudTrail logging was disabled will not be captured, and they will not be available for viewing.

You can use the CloudTrail console, AWS CLI, or AWS SDKs to review API activity for a region for the times in which you had CloudTrail turned on in that region during the last seven days. For information on other ways to get and view CloudTrail log files, including those older than seven days, see [Getting and Viewing Your CloudTrail Log Files \(p. 88\)](#).

Viewing CloudTrail Events in the CloudTrail Console

Use the CloudTrail console to review API activity for a region during the last seven days.

To view CloudTrail events

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/home/>.
2. In the navigation pane, choose **API Activity History**.

A list of events appears in the content pane with the latest event first. Scroll down to see more events. Events that have not been logged will not appear.

For a list of supported services and regions, see [Services Supported by CloudTrail API Activity History \(p. 54\)](#) and [Regions Supported by CloudTrail API Activity History \(p. 53\)](#).

Contents

- [Filtering CloudTrail Events \(p. 46\)](#)
- [Viewing Details for an Event \(p. 47\)](#)
 - [Viewing Resources Referenced with AWS Config \(p. 47\)](#)

Filtering CloudTrail Events

You can filter events by the following optional attributes. Currently, you can filter by time range with one other attribute at a time.

- User name
- Event name
- Resource type
- Resource name
- Time range
- Event ID

For a list of supported resource types, see [Resource Types Supported by CloudTrail API Activity History \(p. 78\)](#).

To filter by attribute

1. To filter the results by an attribute, choose **Select attribute**, and then type or choose a value in the **Enter lookup value** box.
2. To remove an attribute filter, click the **X** on the right of the attribute filter box.

To filter by a start and end date and time

1. To narrow the time range for the events that you want to see, choose **Select time range**.
2. To remove a time range filter, click the calendar icon on the right of the **Time range** box, and then choose **Remove**.

If there are no events logged for the attribute or time that you chose, the results list is empty.

Note

You can apply only one attribute filter in addition to the time range. Choosing a second attribute filter replaces the first attribute filter while preserving your specified time range.

Viewing Details for an Event

1. Choose an event in the results list to show its details.
2. If the event referenced more than one resource, the additional resources are listed at the bottom of the details pane.
3. Some referenced resources have links. Click a referenced resource link to open the console for that resource.
4. Choose **View Event** in the details pane to view the event in JSON format.
5. Click the event again to close the details pane.

Viewing Resources Referenced with AWS Config

AWS Config records configuration details, relationships, and changes to your AWS resources.

- On the **Resources Referenced** pane, click the icon in the **Config timeline** column to view the resource in the AWS Config console.
 - If the icon is gray, AWS Config is not turned on, or it's not recording the resource type. Click the icon to go to the AWS Config console to turn on the service or start recording that resource type. For more information, see [Set Up AWS Config Using the Console](#) in the *AWS Config Developer Guide*.
 - If a resource type is not currently supported by AWS Config, **Resource type unsupported by Config** appears in the column. For more information, see [Supported Resources, Configuration Items, and Relationships](#) in the *AWS Config Developer Guide*.

Note

Some resources can't be viewed for the following reasons:

- The resource is owned by another AWS account.
- The resource is owned by another AWS service, such as a managed IAM policy.
- The resource was created and then deleted immediately; AWS Config does not record this resource.
- The resource was recently created or updated.

Example

1. You configure AWS Config to record IAM resources.
2. You create an IAM user, Bob-user. The API activity history page shows the `CreateUser` event and Bob-user as an IAM resource. You can click the AWS Config icon to view this IAM resource in the AWS Config timeline.
3. You update the user name to Bob-admin.
4. The API activity history page shows the `UpdateUser` event and Bob-admin as the updated IAM resource.
5. You can click the icon to view the Bob-admin IAM resource in the timeline. However, you can't click the icon for Bob-user, because the resource name changed. AWS Config is now recording the updated resource.

To grant users read-only permission to view resources in the AWS Config console, see [Granting Permission to View AWS Config Information on the CloudTrail Console \(p. 87\)](#).

For more information about AWS Config, see the [AWS Config Developer Guide](#).

Viewing CloudTrail Events with the AWS CLI

You can look up CloudTrail events for the last seven days using the `aws cloudtrail lookup-events` command. `lookup-events` has the following options:

- `--max-results`
- `--start-time`
- `--lookup-attributes`
- `--next-token`
- `--generate-cli-skeleton`
- `--cli-input-json`

These options are explained in this topic. For a list of services supported for event lookup, see [Services Supported by CloudTrail API Activity History \(p. 54\)](#). For a list of regions supported for event lookup, see [Regions Supported by CloudTrail API Activity History \(p. 53\)](#).

For general information on using the AWS Command Line Interface, see the [AWS Command Line Interface User Guide](#).

Prerequisites

- To run AWS CLI commands, you must first install the AWS CLI. For information, see [Installing the AWS Command Line Interface](#).
- Make sure your AWS CLI version is greater than 1.6.6. To verify the CLI version, run `aws --version` on the command line.
- To set the account, region, and default output format for an AWS CLI session, use the `aws configure` command. For more information, see [Configuring the AWS Command Line Interface](#).

Note

The CloudTrail AWS CLI commands are case-sensitive.

Getting command line help

To see the command line help for `lookup-events`, type the following command:

```
aws cloudtrail lookup-events help
```

Looking up events

To see the ten latest events, type the following command:

```
aws cloudtrail lookup-events
```

A returned event looks similar to the following fictitious example, which has been formatted for readability:

```
{
  "NextToken": "kbOt5LlZe++mErCebpy2TgaMgmDvFlkYG
FcH64JSjIbZFjsuvrSqq66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juy3CIZW8=",
  "Events": [
    {
      "EventId": "0ebbaee4-6e67-431d-8225-ba0d81df5972",
      "Username": "root",
      "EventTime": 1424476529.0,
      "CloudTrailEvent": {
        \"eventVersion\": \"1.02\",
        \"userIdentity\": {
          \"type\": \"Root\",
          \"principalId\": \"111122223333\",
          \"arn\": \"arn:aws:iam:111122223333:root\",
          \"accountId\": \"111122223333\"},
        \"eventTime\": \"2015-02-20T23:55:29Z\",
        \"eventSource\": \"signin.amazonaws.com\",
        \"eventName\": \"ConsoleLogin\",
        \"awsRegion\": \"us-east-1\",
        \"sourceIPAddress\": \"203.0.113.4\",
        \"userAgent\": \"Mozilla/5.0\",
        \"requestParameters\": null,
        \"responseElements\": {\"ConsoleLogin\": \"Success\"},
        \"additionalEventData\": {
          \"MobileVersion\": \"No\",
          \"LoginTo\": \"https://console.aws.amazon.com/con
sole/home\",
          \"MFAUsed\": \"No\"},
        \"eventID\": \"0ebbaee4-6e67-431d-8225-ba0d81df5972\",
        \"eventType\": \"AwsApiCall\",
        \"recipientAccountId\": \"111122223333\"},
      "EventName": "ConsoleLogin",
      "Resources": []
    }
  ]
}
```

For an explanation of the lookup-related fields in the output, see the section [Lookup Output Fields \(p. 53\)](#) later in this document. For an explanation of the fields in the CloudTrail event, see [CloudTrail Record Contents \(p. 193\)](#).

Specifying the number of events to return

To specify the number of events to return, type the following command:

```
aws cloudtrail lookup-events --max-results <integer>
```

The default value for *<integer>* is 10. Possible values are 1 through 50. The following example returns one result.

```
aws cloudtrail lookup-events --max-results 1
```

Looking up events by time range

Events from the past seven days are available for lookup. To specify a time range, type the following command:

```
aws cloudtrail lookup-events --start-time <timestamp> --end-time <timestamp>
```

`--start-time <timestamp>` specifies that only events that occur after or at the specified time are returned. If the specified start time is after the specified end time, an error is returned.

`--end-time <timestamp>` specifies that only events that occur before or at the specified time are returned. If the specified end time is before the specified start time, an error is returned.

The default start time is the earliest date that data is available within the last seven days. The default end time is the time of the event that occurred closest to the current time.

Valid <timestamp> formats

The `--start-time` and `--end-time` attributes take UNIX time values or valid equivalents.

The following are examples of valid formats. Date, month, and year values can be separated by hyphens or forward slashes. Double quotes must be used if spaces are present.

```
1422317782
1422317782.0
01-27-2015
01-27-2015,01:16PM
"01-27-2015, 01:16 PM"
"01/27/2015, 13:16"
2015-01-27
"2015-01-27, 01:16 PM"
```

Looking up events by attribute

To filter by an attribute, type the following command:

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=<attribute>,AttributeValue=<string>
```

Note

Although `--lookup-attributes` is technically a list, currently you can specify only one attribute key/value pair per lookup.

Possible values for `AttributeKey` are the following. These names are case sensitive.

- EventId
- EventName
- Username
- ResourceType
- ResourceName

For a list of the resource types supported for event lookup, see [Resource Types Supported by CloudTrail API Activity History \(p. 78\)](#)

Attribute lookup examples

The following example command returns the event for the specified CloudTrail `EventId`.

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=EventId,AttributeValue=b5cc8c40-12ba-4d08-a8d9-2bceb9a3e002
```

The following example command returns events in which the value of `EventName` is `RunInstances`.

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=EventName,AttributeValue=RunInstances
```

The following example command returns events in which the value of `Username` is `root`.

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=Username,AttributeValue=root
```

The following example command returns events in which the value of `ResourceType` is `AWS::S3::Bucket`.

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=ResourceType,AttributeValue=AWS::S3::Bucket
```

The following example command returns events in which the value of `ResourceName` is `CloudTrail_CloudWatchLogs_Role`.

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=ResourceName,AttributeValue=CloudTrail_CloudWatchLogs_Role
```

Specifying the next page of results

To get the next page of results from a `lookup-events` command, type the following command:

```
aws cloudtrail lookup-events <same parameters as previous command> --next-token=<token>
```

where the value for `<token>` is taken from the first field of the output of the previous command.

When you use `--next-token` in a command, you must use the same parameters as in the previous command. For example, suppose you run the following command:

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=Username,AttributeValue=root
```

To get the next page of results, your next command would look like this:

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=Username,AttributeValue=root --next-token=kbOt5LlZe++mErCebpy2TgaMgmDvF1kYG  
FcH64JSjIbZfjsuvrSqq66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juy3CIZW8=
```

Getting JSON input from a file

The AWS CLI for some AWS services has two parameters, `--generate-cli-skeleton` and `--cli-input-json`, that you can use to generate a JSON template which you can modify and use as input to the `--cli-input-json` parameter. This section describes how to use these parameters with `aws cloudtrail lookup-events`. For more general information, see [Generate CLI Skeleton and CLI Input JSON Parameters](#).

To look up CloudTrail events by getting JSON input from a file

1. Create an input template for use with `lookup-events` by redirecting the `--generate-cli-skeleton` output to a file, as in the following example.

```
aws cloudtrail lookup-events --generate-cli-skeleton > LookupEvents.txt
```

The template file generated (in this case, `LookupEvents.txt`) looks like this:

```
{
  "LookupAttributes": [
    {
      "AttributeKey": "",
      "AttributeValue": ""
    }
  ],
  "StartTime": null,
  "EndTime": null,
  "MaxResults": 0,
  "NextToken": ""
}
```

2. Use a text editor to modify the JSON as needed. The JSON input must contain only values that are specified.

Important

All empty or null values must be removed from the template before you can use it.

The following example specifies a time range and maximum number of results to return.

```
{
  "StartTime": "2015-01-01",
  "EndTime": "2015-01-27",
  "MaxResults": 2
}
```

3. To use the edited file as input, use the syntax `--cli-input-json file://<filename>`, as in the following example:

```
aws cloudtrail lookup-events --cli-input-json file://LookupEvents.txt
```


Note

You can use other arguments on the same command line as `--cli-input-json`.

Lookup Output Fields

NextToken

A string to get the next page of results from a previous `lookup-events` command. To use the token, the parameters must be the same as those in the original command. If no `NextToken` entry appears in the output, there are no more results to return.

Events

A list of lookup events based on the lookup attribute and time range that were specified. The events list is sorted by time, with the latest event listed first. Each entry contains information about the lookup request and includes a string representation of the CloudTrail event that was retrieved.

The following entries describe the fields in each lookup event.

EventId

A string that contains the GUID of the event returned.

Username

A string that contains the user name of the account for the event returned.

EventTime

The date and time, in UNIX time format, of the event.

CloudTrailEvent

A JSON string that contains an object representation of the event returned. For information about each of the elements returned, see [Record Body Contents](#).

EventName

A string that contains the name of the event returned.

Resources

A list of resources referenced by the event that was returned. Each resource entry specifies a resource type and a resource name.

ResourceType

A string that contains the type of a resource referenced by the event. When the resource type cannot be determined, null is returned.

ResourceName

A string that contains the name of the resource referenced by the event.

Regions Supported by CloudTrail API Activity History

You can view CloudTrail events for supported services in the following regions:

Region Name	Region
US East (N. Virginia)	us-east-1
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1

Region Name	Region
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
EU (Ireland)	eu-west-1
EU (Frankfurt)	eu-central-1
South America (São Paulo)	sa-east-1

Services Supported by CloudTrail API Activity History

You can look up API activity in the supported regions for the following services. You can look up API activity for create, modify, and delete API calls.

Note

The API activity history feature supports the following APIs. If you're looking for a specific API call that does not appear in the API activity history, check the log files in your S3 bucket. For a list of services for which CloudTrail delivers log files to S3 buckets, see [CloudTrail Supported Services](#) (p. 7).

To view API calls that are supported by the API activity history feature, choose a service from the following list.

Topics

- [Auto Scaling APIs](#) (p. 55)
- [AWS Certificate Manager APIs](#) (p. 56)
- [AWS CloudFormation APIs](#) (p. 56)
- [Amazon CloudFront APIs](#) (p. 56)
- [AWS CloudHSM APIs](#) (p. 57)
- [Amazon CloudSearch APIs](#) (p. 57)
- [AWS CloudTrail APIs](#) (p. 58)
- [Amazon CloudWatch APIs](#) (p. 58)
- [AWS CodeDeploy APIs](#) (p. 58)
- [Amazon Cognito Identity APIs](#) (p. 59)
- [Amazon Cognito Sync APIs](#) (p. 59)
- [AWS Config APIs](#) (p. 59)
- [AWS Data Pipeline APIs](#) (p. 59)
- [AWS Direct Connect APIs](#) (p. 59)
- [Amazon DynamoDB APIs](#) (p. 60)
- [Amazon EC2 APIs](#) (p. 60)
- [AWS Elastic Beanstalk APIs](#) (p. 64)
- [Elastic Load Balancing APIs](#) (p. 65)
- [Amazon EMR APIs](#) (p. 65)
- [Amazon Elastic Transcoder APIs](#) (p. 66)
- [Amazon ElastiCache APIs](#) (p. 66)
- [Amazon Glacier APIs](#) (p. 67)
- [AWS Identity and Access Management APIs](#) (p. 67)

- [Amazon Inspector APIs \(p. 69\)](#)
- [AWS IoT APIs \(p. 70\)](#)
- [Amazon Kinesis Firehose APIs \(p. 71\)](#)
- [Amazon Kinesis Streams APIs \(p. 71\)](#)
- [AWS OpsWorks APIs \(p. 71\)](#)
- [Amazon Redshift APIs \(p. 73\)](#)
- [Amazon Relational Database Service APIs \(p. 74\)](#)
- [Amazon Route 53 APIs \(p. 75\)](#)
- [Amazon S3 Bucket Level APIs \(p. 75\)](#)
- [Amazon Simple Workflow Service APIs \(p. 76\)](#)
- [AWS Storage Gateway APIs \(p. 76\)](#)
- [AWS Support APIs \(p. 77\)](#)
- [AWS WAF APIs \(p. 77\)](#)
- [Amazon WorkDocs APIs \(p. 78\)](#)

Auto Scaling APIs

AttachInstances
CompleteLifecycleAction
CreateAutoScalingGroup
CreateLaunchConfiguration
CreateOrUpdateTags
DeleteAutoScalingGroup
DeleteLaunchConfiguration
DeleteLifecycleHook
DeleteNotificationConfiguration
DeletePolicy
DeleteScheduledAction
DeleteTags
DetachInstances
DisableMetricsCollection
EnableMetricsCollection
EnterStandby
ExecutePolicy
ExitStandby
PutLifecycleHook
PutNotificationConfiguration

PutScalingPolicy
PutScheduledUpdateGroupAction
RecordLifecycleActionHeartbeat
ResumeProcesses
SetDesiredCapacity
SetInstanceHealth
SuspendProcesses
TerminateInstanceInAutoScalingGroup
UpdateAutoScalingGroup

AWS Certificate Manager APIs

DeleteCertificate
RequestCertificate
ResendValidationEmail

AWS CloudFormation APIs

CancelUpdateStack
CreateStack
DeleteStack
SetStackPolicy
SignalResource
UpdateStack

Amazon CloudFront APIs

CreateCloudFrontOriginAccessIdentity
CreateDistribution
CreateInvalidation
CreateStreamingDistribution
DeleteCloudFrontOriginAccessIdentity
DeleteDistribution
DeleteStreamingDistribution

UpdateCloudFrontOriginAccessIdentity
UpdateDistribution
UpdateStreamingDistribution

AWS CloudHSM APIs

AdminCreateHsm
CreateHapg
CreateHsm
CreateLunaClient
DeleteHapg
DeleteHsm
DeleteLunaClient
ModifyHapg
ModifyHsm
ModifyLunaClient

Amazon CloudSearch APIs

BuildSuggesters
CreateDomain
DefineAnalysisScheme
DefineExpression
DefineIndexField
DefineIndexFields
DefineRankExpression
DefineSuggester
DeleteAnalysisScheme
DeleteDomain
DeleteExpression
DeleteIndexField
DeleteRankExpression
DeleteSuggester
IndexDocuments

UpdateAvailabilityOptions
UpdateDefaultSearchField
UpdateScalingParameters
UpdateServiceAccessPolicies
UpdateStemmingOptions
UpdateStopwordOptions
UpdateSynonymOptions

AWS CloudTrail APIs

CreateTrail
DeleteTrail
StartLogging
StopLogging
UpdateTrail

Amazon CloudWatch APIs

DeleteAlarms
DisableAlarmActions
EnableAlarmActions
PutMetricAlarm
SetAlarmState

AWS CodeDeploy APIs

CreateApplication
CreateDeployment
CreateDeploymentConfig
CreateDeploymentGroup
DeleteApplication
DeleteDeploymentConfig
DeleteDeploymentGroup
RegisterApplicationRevision

StopDeployment
UpdateApplication
UpdateDeploymentGroup

Amazon Cognito Identity APIs

CreateIdentityPool
DeleteIdentityPool
UpdateIdentityPool
SetIdentityPoolRoles

Amazon Cognito Sync APIs

SetCognitoEvents
SetIdentityPoolConfiguration

AWS Config APIs

DeleteDeliveryChannel
PutConfigurationRecorder
PutDeliveryChannel
StartConfigurationRecorder
StopConfigurationRecorder

AWS Data Pipeline APIs

ActivatePipeline
CreatePipeline
DeletePipeline
PutPipelineDefinition
SetStatus

AWS Direct Connect APIs

AllocateConnectionOnInterconnect
AllocatePrivateVirtualInterface

AllocatePublicVirtualInterface
ConfirmConnection
ConfirmPrivateVirtualInterface
ConfirmPublicVirtualInterface
CreateConnection
CreateInterconnect
CreatePrivateVirtualInterface
CreatePublicVirtualInterface
DeleteConnection
DeleteInterconnect
DeleteVirtualInterface

Amazon DynamoDB APIs

CreateTable
DeleteTable
PurchaseReservedCapacityOfferings
UpdateTable

Amazon EC2 APIs

AcceptVpcPeeringConnection
AllocateAddress
AllocateHosts
AssignPrivateIpAddresses
AssociateAddress
AssociateDhcpOptions
AssociateRouteTable
AttachClassicLinkVpc
AttachInternetGateway
AttachNetworkInterface
AttachVolume
AttachVpnGateway
AuthorizeSecurityGroupEgress

AuthorizeSecurityGroupIngress
BundleInstance
CancelBundleTask
CancelConversionTask
CancelExportTask
CancelImportTask
CancelReservedInstancesListing
CancelSpotInstanceRequests
CopyImage
CopySnapshot
CreateCustomerGateway
CreateDhcpOptions
CreateImage
CreateInstanceExportTask
CreateInternetGateway
CreateKeyPair
CreateNatGateway
CreateNetworkAcl
CreateNetworkAclEntry
CreateNetworkInterface
CreatePlacementGroup
CreateReservedInstancesListing
CreateRoute
CreateRouteTable
CreateSecurityGroup
CreateSnapshot
CreateSpotDatafeedSubscription
CreateSubnet
CreateTags
CreateVolume
CreateVpc
CreateVpcEndpoint
CreateVpcPeeringConnection

AWS CloudTrail User Guide
Services Supported by CloudTrail API Activity History

CreateVpnConnection
CreateVpnConnectionRoute
CreateVpnGateway
DeleteCustomerGateway
DeleteDhcpOptions
DeleteInternetGateway
DeleteKeyPair
DeleteNatGateway
DeleteNetworkAcl
DeleteNetworkAclEntry
DeleteNetworkInterface
DeletePlacementGroup
DeleteRoute
DeleteRouteTable
DeleteSecurityGroup
DeleteSnapshot
DeleteSpotDatafeedSubscription
DeleteSubnet
DeleteTags
DeleteVolume
DeleteVpc
DeleteVpcEndpoints
DeleteVpcPeeringConnection
DeleteVpnConnection
DeleteVpnConnectionRoute
DeleteVpnGateway
DeregisterImage
DetachClassicLinkVpc
DetachInternetGateway
DetachNetworkInterface
DetachVolume
DetachVpnGateway
DisableVgwRoutePropagation

DisableVpcClassicLink
DisassociateAddress
DisassociateRouteTable
EnableVgwRoutePropagation
EnableVolumeIO
EnableVpcClassicLink
ImportImage
ImportInstance
ImportKeyPair
ImportSnapshot
ImportVolume
ModifyHosts
ModifyImageAttribute
ModifyInstanceAttribute
ModifyInstancePlacement
ModifyNetworkInterfaceAttribute
ModifyReservedInstances
ModifySnapshotAttribute
ModifySubnetAttribute
ModifyVolumeAttribute
ModifyVpcAttribute
ModifyVpcEndpoint
MonitorInstances
MoveAddressToVpc
PurchaseReservedInstancesOffering
RebootInstances
RegisterImage
RejectVpcPeeringConnection
ReleaseAddress
ReleaseHosts
ReplaceNetworkAclAssociation
ReplaceNetworkAclEntry
ReplaceRoute

ReplaceRouteTableAssociation
RequestSpotInstances
ResetImageAttribute
ResetInstanceAttribute
ResetNetworkInterfaceAttribute
ResetSnapshotAttribute
RestoreAddressToClassic
RevokeSecurityGroupEgress
RevokeSecurityGroupIngress
RunInstances
StartInstances
StopInstances
TerminateInstances
UnassignPrivateIPAddresses
UnmonitorInstances

AWS Elastic Beanstalk APIs

AbortEnvironmentUpdate
CreateApplication
CreateApplicationVersion
CreateConfigurationTemplate
CreateEnvironment
CreateStorageLocation
DeleteApplication
DeleteApplicationVersion
DeleteConfigurationTemplate
DeleteEnvironmentConfiguration
RebuildEnvironment
RestartAppServer
SwapEnvironmentCNAMEs
TerminateEnvironment
UpdateApplication

UpdateApplicationVersion
UpdateConfigurationTemplate
UpdateEnvironment

Elastic Load Balancing APIs

AddTags
ApplySecurityGroupsToLoadBalancer
AttachLoadBalancerToSubnets
ConfigureHealthCheck
CreateAppCookieStickinessPolicy
CreateLBCookieStickinessPolicy
CreateLoadBalancer
CreateLoadBalancerListeners
CreateLoadBalancerPolicy
DeleteLoadBalancer
DeleteLoadBalancerListeners
DeleteLoadBalancerPolicy
DeregisterInstancesFromLoadBalancer
DetachLoadBalancerFromSubnets
DisableAvailabilityZonesForLoadBalancer
EnableAvailabilityZonesForLoadBalancer
ModifyLoadBalancerAttributes
RegisterInstancesWithLoadBalancer
RemoveTags
SetLoadBalancerListenerSSLCertificate
SetLoadBalancerPoliciesForBackendServer
SetLoadBalancerPoliciesOfListener

Amazon EMR APIs

AddInstanceGroups
AddJobFlowSteps
AddTags

ModifyInstanceGroups
RemoveTags
RunJobFlow
SetTerminationProtection
SetVisibleToAllUsers
TerminateJobFlows

Amazon Elastic Transcoder APIs

CancelJob
CreateJob
CreatePipeline
CreatePreset
DeletePipeline
DeletePreset
TestRole
UpdatePipeline
UpdatePipelineNotifications
UpdatePipelineStatus

Amazon ElastiCache APIs

AuthorizeCacheSecurityGroupIngress
CopySnapshot
CreateCacheCluster
CreateCacheParameterGroup
CreateCacheSecurityGroup
CreateCacheSubnetGroup
CreateReplicationGroup
CreateSnapshot
DeleteCacheCluster
DeleteCacheParameterGroup
DeleteCacheSecurityGroup
DeleteCacheSubnetGroup

DeleteReplicationGroup
DeleteSnapshot
ModifyCacheCluster
ModifyCacheParameterGroup
ModifyCacheSubnetGroup
ModifyReplicationGroup
PurchaseReservedCacheNodesOffering
RebootCacheCluster
ResetCacheParameterGroup
RevokeCacheSecurityGroupIngress

Amazon Glacier APIs

CreateVault
DeleteVault
DeleteVaultNotifications
SetDataRetrievalPolicy
SetVaultNotifications

AWS Identity and Access Management APIs

AddClientIDToOpenIDConnectProvider
AddRoleToInstanceProfile
AddUserToGroup
AttachGroupPolicy
AttachRolePolicy
AttachUserPolicy
ChangePassword
CreateAccessKey
CreateAccountAlias
CreateGroup
CreateInstanceProfile
CreateLoginProfile
CreateOpenIDConnectProvider

CreatePolicy
CreatePolicyVersion
CreateRole
CreateSAMLProvider
CreateUser
CreateVirtualMFADevice
DeactivateMFADevice
DeleteAccessKey
DeleteAccountAlias
DeleteAccountPasswordPolicy
DeleteGroup
DeleteGroupPolicy
DeleteInstanceProfile
DeleteLoginProfile
DeleteOpenIDConnectProvider
DeletePolicy
DeletePolicyVersion
DeleteRole
DeleteRolePolicy
DeleteSAMLProvider
DeleteServerCertificate
DeleteSigningCertificate
DeleteUser
DeleteUserPolicy
DeleteVirtualMFADevice
DetachGroupPolicy
DetachRolePolicy
DetachUserPolicy
EnableMFADevice
PutGroupPolicy
PutRolePolicy
PutUserPolicy
RemoveClientIDFromOpenIDConnectProvider

RemoveRoleFromInstanceProfile
RemoveUserFromGroup
ResyncMFADevice
SetDefaultPolicyVersion
UpdateAccessKey
UpdateAccountPasswordPolicy
UpdateAssumeRolePolicy
UpdateGroup
UpdateLoginProfile
UpdateOpenIDConnectProviderThumbprint
UpdateSAMLProvider
UpdateServerCertificate
UpdateSigningCertificate
UpdateUser
UploadServerCertificate
UploadSigningCertificate

Amazon Inspector APIs

AddAttributesToFindings
AttachAssessmentAndRulesPackage
CreateApplication
CreateAssessment
CreateResourceGroup
DeleteApplication
DeleteAssessment
DeleteRun
DetachAssessmentAndRulesPackage
RegisterCrossAccountAccessRole
RemoveAttributesFromFindings
RetireRulesPackage
RunAssessment
SetTagsForResource

StartDataCollection
StopDataCollection
UpdateApplication
UpdateAssessment

AWS IoT APIs

AcceptCertificateTransfer
AttachPrincipalPolicy
AttachThingPrincipal
CancelCertificateTransfer
CreateCertificateFromCSR
CreateKeysAndCertificate
CreatePolicy
CreatePolicyVersion
CreateThing
CreateTopicRule
DeleteCACertificate
DeleteCertificate
DeletePolicy
DeletePolicyVersion
DeleteRegistrationCode
DeleteThing
DeleteTopicRule
DetachPrincipalPolicy
DetachThingPrincipal
DisableTopicRule
EnableTopicRule
RegisterCACertificate
RegisterCertificate
RejectCertificateTransfer
ReplaceTopicRule
SetDefaultPolicyVersion

SetLoggingOptions
TransferCertificate
UpdateCACertificate
UpdateCertificate
UpdateThing

Amazon Kinesis Firehose APIs

CreateDeliveryStream
DeleteDeliveryStream
UpdateDestination

Amazon Kinesis Streams APIs

CreateStream
DeleteStream
MergeShards
SplitShard

AWS OpsWorks APIs

AssignInstance
AssignVolume
AssociateElasticIp
AttachElasticLoadBalancer
CloneStack
CreateApp
CreateDeployment
CreateInstance
CreateLayer
CreateStack
CreateUserProfile
DeleteApp
DeleteInstance

AWS CloudTrail User Guide
Services Supported by CloudTrail API Activity History

DeleteLayer
DeleteStack
DeleteUserProfile
DeregisterElasticIp
DeregisterInstance
DeregisterRdsDbInstance
DeregisterVolume
DetachElasticLoadBalancer
DisassociateElasticIp
RebootInstance
RegisterElasticIp
RegisterInstance
RegisterRdsDbInstance
RegisterVolume
SetLoadBasedAutoScaling
SetPermission
SetTimeBasedAutoScaling
StartInstance
StartStack
StopInstance
StopStack
UnassignInstance
UnassignVolume
UpdateApp
UpdateElasticIp
UpdateInstance
UpdateLayer
UpdateMyUserProfile
UpdateRdsDbInstance
UpdateStack
UpdateUserProfile
UpdateVolume

Amazon Redshift APIs

AuthorizeClusterSecurityGroupIngress
AuthorizeSnapshotAccess
CopyClusterSnapshot
CreateCluster
CreateClusterParameterGroup
CreateClusterSecurityGroup
CreateClusterSnapshot
CreateClusterSubnetGroup
CreateEventSubscription
CreateHsmClientCertificate
CreateHsmConfiguration
CreateTags
DeleteCluster
DeleteClusterParameterGroup
DeleteClusterSecurityGroup
DeleteClusterSnapshot
DeleteClusterSubnetGroup
DeleteEventSubscription
DeleteHsmClientCertificate
DeleteHsmConfiguration
DeleteTags
DisableLogging
DisableSnapshotCopy
EnableLogging
EnableSnapshotCopy
ModifyCluster
ModifyClusterParameterGroup
ModifyClusterSubnetGroup
ModifyEventSubscription
ModifySnapshotCopyRetentionPeriod
PurchaseReservedNodeOffering

RebootCluster
ResetClusterParameterGroup
RestoreFromClusterSnapshot
RevokeClusterSecurityGroupIngress
RevokeSnapshotAccess
RotateEncryptionKey

Amazon Relational Database Service APIs

AddSourceIdentifierToSubscription
AddTagsToResource
ApplyPendingMaintenanceAction
AuthorizeDBSecurityGroupIngress
CopyDBParameterGroup
CopyDBSnapshot
CopyOptionGroup
CreateDBInstance
CreateDBInstanceReadReplica
CreateDBParameterGroup
CreateDBSecurityGroup
CreateDBSnapshot
CreateDBSubnetGroup
CreateEventSubscription
CreateOptionGroup
DeleteDBInstance
DeleteDBParameterGroup
DeleteDBSecurityGroup
DeleteDBSnapshot
DeleteDBSubnetGroup
DeleteEventSubscription
DeleteOptionGroup
ModifyDBInstance
ModifyDBParameterGroup

ModifyDBSubnetGroup
ModifyEventSubscription
ModifyOptionGroup
PromoteReadReplica
PurchaseReservedDBInstancesOffering
RebootDBInstance
RemoveSourceIdentifierFromSubscription
RemoveTagsFromResource
ResetDBParameterGroup
RestoreDBInstanceFromDBSnapshot
RestoreDBInstanceToPointInTime
RevokeDBSecurityGroupIngress

Amazon Route 53 APIs

CreateHostedZone
DeleteHostedZone
ChangeResourceRecordSets
CreateHealthCheck
DeleteHealthCheck
AssociateVPCWithHostedZone
DisassociateVPCFromHostedZone

Amazon S3 Bucket Level APIs

CreateBucket
DeleteBucket
DeleteBucketCors
DeleteBucketLifecycle
DeleteBucketPolicy
DeleteBucketReplication
DeleteBucketTagging
DeleteBucketWebsite
PutBucketAcl

PutBucketCors
PutBucketLifecycle
PutBucketLogging
PutBucketNotification
PutBucketPolicy
PutBucketReplication
PutBucketRequestPayment
PutBucketTagging
PutBucketVersioning
PutBucketWebsite

Amazon Simple Workflow Service APIs

DeprecateActivityType
DeprecateDomain
DeprecateWorkflowType
RegisterActivityType
RegisterDomain
RegisterWorkflowType

AWS Storage Gateway APIs

ActivateGateway
AddCache
AddUploadBuffer
AddWorkingStorage
CancelArchival
CancelRetrieval
CreateCachediSCSIVolume
CreateSnapshot
CreateSnapshotFromVolumeRecoveryPoint
CreateStorediSCSIVolume
CreateTapes
DeleteBandwidthRateLimit

DeleteChapCredentials
DeleteGateway
DeleteSnapshotSchedule
DeleteTape
DeleteTapeArchive
DeleteVolume
DisableGateway
ShutdownGateway
StartGateway
UpdateBandwidthRateLimit
UpdateChapCredentials
UpdateGatewayInformation
UpdateGatewaySoftwareNow
UpdateMaintenanceStartTime
UpdateSnapshotSchedule
UpdateVTLDeviceType

AWS Support APIs

AddAttachmentsToSet
AddCommunicationToCase
CreateCase
ResolveCase

AWS WAF APIs

CreateByteMatchSet
CreateIPSet
CreateRule
CreateSizeConstraintSet
CreateSqlInjectionMatchSet
CreateWebACL
CreateXssMatchSet
DeleteByteMatchSet

AWS CloudTrail User Guide
Resource Types Supported by CloudTrail API Activity
History

DeleteIPSet
DeleteRule
DeleteSizeConstraintSet
DeleteSqlInjectionMatchSet
DeleteWebACL
DeleteXssMatchSet
UpdateByteMatchSet
UpdateIPSet
UpdateRule
UpdateSizeConstraintSet
UpdateSqlInjectionMatchSet
UpdateWebACL
UpdateXssMatchSet

Amazon WorkDocs APIs

ActivateUser
AddUserToGroup
CreateInstance
DeactivateUser
DeleteInstance
DeregisterDirectory
DisableDirectoryMFA
EnableDirectoryMFA
RegisterDirectory
RemoveUserFromGroup
ResendUserNotification
UpdateDirectoryMFA
UpdateInstanceAlias

Resource Types Supported by CloudTrail API Activity History

You can look up events by the following resource types for the services indicated.

AWS CloudTrail User Guide
Resource Types Supported by CloudTrail API Activity
History

Auto Scaling

Resource Type	Full syntax
AutoScalingGroup	AWS::AutoScaling::AutoScalingGroup
LaunchConfiguration	AWS::AutoScaling::LaunchConfiguration
ScalingPolicy	AWS::AutoScaling::ScalingPolicy
ScheduledAction	AWS::AutoScaling::ScheduledAction

AWS CloudTrail

Resource Type	Full syntax
Trail	AWS::CloudTrail::Trail

Amazon EC2

Resource Type	Full syntax
Ami	AWS::EC2::Ami
BundleTask	AWS::EC2::BundleTask
ConversionTask	AWS::EC2::ConversionTask
CustomerGateway	AWS::EC2::CustomerGateway
DHCPOptions	AWS::EC2::DHCPOptions
EIP	AWS::EC2::EIP
EIPAssociation	AWS::EC2::EIPAssociation
ExportTask	AWS::EC2::ExportTask
Instance	AWS::EC2::Instance
InternetGateway	AWS::EC2::InternetGateway
KeyPair	AWS::EC2::KeyPair
NetworkAcl	AWS::EC2::NetworkAcl
NetworkInterface	AWS::EC2::NetworkInterface
NetworkInterfaceAttachment	AWS::EC2::NetworkInterfaceAttachment
PlacementGroup	AWS::EC2::PlacementGroup
ReservedInstance	AWS::EC2::ReservedInstance
ReservedInstancesListing	AWS::EC2::ReservedInstancesListing
ReservedInstancesModification	AWS::EC2::ReservedInstancesModification
RouteTable	AWS::EC2::RouteTable
SecurityGroup	AWS::EC2::SecurityGroup

AWS CloudTrail User Guide
Resource Types Supported by CloudTrail API Activity
History

Resource Type	Full syntax
Snapshot	AWS::EC2::Snapshot
SpotInstanceRequest	AWS::EC2::SpotInstanceRequest
Subnet	AWS::EC2::Subnet
SubnetNetworkAclAssociation	AWS::EC2::SubnetNetworkAclAssociation
SubnetRouteTableAssociation	AWS::EC2::SubnetRouteTableAssociation
Volume	AWS::EC2::Volume
VPC	AWS::EC2::VPC
VPCPeeringConnection	AWS::EC2::VPCPeeringConnection
VPNConnection	AWS::EC2::VPNConnection
VPNGateway	AWS::EC2::VPNGateway

Elastic Load Balancing

Resource Type	Full syntax
LoadBalancer	AWS::ElasticLoadBalancing::LoadBalancer

IAM

Resource Type	Full syntax
AccessKey	AWS::IAM::AccessKey
AccountAlias	AWS::IAM::AccountAlias
Group	AWS::IAM::Group
InstanceProfile	AWS::IAM::InstanceProfile
MfaDevice	AWS::IAM::MfaDevice
OpenIDConnectProvider	AWS::IAM::OpenIDConnectProvider
Policy	AWS::IAM::Policy
Role	AWS::IAM::Role
SamlProvider	AWS::IAM::SamlProvider
ServerCertificate	AWS::IAM::ServerCertificate
SigningCertificate	AWS::IAM::SigningCertificate
User	AWS::IAM::User

Amazon RDS

Resource Type	Full syntax
DBInstance	AWS::RDS::DBInstance
DBOptionGroup	AWS::RDS::DBOptionGroup
DBParameterGroup	AWS::RDS::DBParameterGroup
DBSecurityGroup	AWS::RDS::DBSecurityGroup
DBSnapshot	AWS::RDS::DBSnapshot
DBSubnetGroup	AWS::RDS::DBSubnetGroup
EventSubscription	AWS::RDS::EventSubscription
ReservedDBInstance	AWS::RDS::ReservedDBInstance

Amazon S3

Resource Type	Full syntax
Bucket	AWS::S3::Bucket

Controlling User Permissions for CloudTrail

AWS CloudTrail integrates with AWS Identity and Access Management (IAM), which allows you to control access to CloudTrail and to other AWS resources that CloudTrail requires, including Amazon S3 buckets and Amazon Simple Notification Service (Amazon SNS) topics. You can use AWS Identity and Access Management to control which AWS users can create, configure, or delete AWS CloudTrail trails, start and stop logging, and access the buckets that contain log information.

If you work with CloudTrail as the root user in your account, you can perform all the tasks associated with trails, including creating trails, reading logs, and so on. If other people in your organization need to work with CloudTrail, you can create IAM users for those people and give them individual names and passwords. When you do that, you must also give users permissions to work with CloudTrail and with any other AWS services they need to access, such as Amazon S3. (By default, IAM users have no permissions and cannot perform any actions in AWS.)

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.

Topics

- [Granting Permissions for CloudTrail Administration](#) (p. 82)
- [Granting Custom Permissions for CloudTrail Users](#) (p. 83)

Granting Permissions for CloudTrail Administration

To allow users to administer a CloudTrail trail, you must grant explicit permissions to IAM users to perform the actions associated with CloudTrail 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 CloudTrail administration tasks are not the same as the permissions that CloudTrail itself requires in order to deliver log files to Amazon S3 buckets or send notifications to Amazon SNS topics. For more information about those permissions, see [Getting and Viewing Your CloudTrail Log Files \(p. 88\)](#). CloudTrail also requires a role that it can assume to deliver events to an Amazon CloudWatch Logs log group. For more information, see [Granting Custom Permissions for CloudTrail Users \(p. 83\)](#).

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 CloudTrail actions, and a separate group for users who should be able to view trail information but not create or change trails.

To create an IAM group and users for CloudTrail access

1. Open the IAM console at <https://console.aws.amazon.com/iam>.
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 one of the following policies for CloudTrail:
 - **AWSCloudTrailFullAccess**. This policy gives users in the group full access to CloudTrail actions and permissions to manage the Amazon S3 bucket, the log group for CloudWatch Logs, and an Amazon SNS topic for a trail.
 - **AWSCloudTrailReadOnlyAccess**. This policy lets users in the group can view trails and buckets.

Note

You can also create a custom policy that grants permissions to individual actions. For more information, see [Granting Custom Permissions for CloudTrail Users \(p. 83\)](#).

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 CloudTrail, in the **Security Credentials** tab, choose **Manage Password**, and then create a password for the user.

- c. If the user will use the CLI or API to manage CloudTrail, 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).

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 CloudTrail Users

CloudTrail policies grant permissions to users who work with CloudTrail. If you need to grant different permissions to users, you can attach a CloudTrail 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. 83)
- [Full access](#) (p. 84)
- [Controlling User Permissions for Actions on Specific Trails](#) (p. 85)
- [Granting Permission to View AWS Config Information on the CloudTrail Console](#) (p. 87)
- [Additional Information](#) (p. 88)

Read-only access

The following example shows a policy that grants read-only access to CloudTrail trails. It grants users permission to see trail information, but not to create or update trails. The policy also grants permission to read objects in Amazon S3 buckets, but not create or delete them.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetBucketLocation"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudtrail:DescribeTrails",
        "cloudtrail:GetTrailStatus",
        "cloudtrail:LookupEvents",
        "s3:ListAllMyBuckets",
        "kms:ListAliases"
      ],
    }
  ]
}
```

```
    "Resource": "*"
  }
]
}
```

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 CloudTrail 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 `cloudtrail:` to indicate that they refer to CloudTrail actions. You can use the `*` wildcard character in the `Action` element, such as in the following examples:

- "Action": ["cloudtrail:*Logging"]

This allows all CloudTrail actions that end with "Logging" (`StartLogging`, `StopLogging`).

- "Action": ["cloudtrail:*"]

This allows all CloudTrail 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 `CreateTrail`, `UpdateTrail`, `StartLogging`, and `StopLogging` actions. Users with this policy are not allowed to create trails, update trails, or turn logging on and off. For the list of CloudTrail actions, see the [AWS CloudTrail API Reference](#).

Full access

The following example shows a policy that grants full access to CloudTrail. It grants users the permission to perform all CloudTrail actions. It also lets users manage files in Amazon S3 buckets, manage how CloudWatch Logs monitors CloudTrail log events, and manage Amazon SNS topics in the account that the user is associated with.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:AddPermission",
        "sns:CreateTopic",
        "sns>DeleteTopic",
        "sns:ListTopics",
        "sns:SetTopicAttributes",
        "sns:GetTopicAttributes"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
```



```
        "s3:DeleteBucket",
        "s3:GetObject",
        "s3:ListAllMyBuckets",
        "s3:PutBucketPolicy",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicy"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": "cloudtrail:*",
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "logs:CreateLogGroup"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "iam:PassRole",
        "iam:ListRoles",
        "iam:GetRolePolicy",
        "iam:GetUser"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "kms:ListKeys",
        "kms:ListAliases"
    ],
    "Resource": "*"
}
]
```

Controlling User Permissions for Actions on Specific Trails

You can use resource-level permissions to control a user's ability to perform specific actions on CloudTrail trails.

For example, you don't want users of your company's developer group to start or stop logging on a specific trail, but you want to grant them permission to perform the `DescribeTrails` and `GetTrailStatus` actions on the trail. You want the users of the developer group to perform the `StartLogging` or `StopLogging` actions on trails that they create and manage.

You can create two policy statements and then attach them to the developer user group.

In the first policy, you deny the `StartLogging` and `StopLogging` actions for the trail ARN that you specify. In the following example, the trail ARN is `arn:aws:cloudtrail:us-east-1:111122223333:trail/Default`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1446057698000",
      "Effect": "Deny",
      "Action": [
        "cloudtrail:StartLogging",
        "cloudtrail:StopLogging"
      ],
      "Resource": [
        "arn:aws:cloudtrail:us-east-1:111122223333:trail/Default"
      ]
    }
  ]
}
```

In the second policy, the `DescribeTrails` and `GetTrailStatus` actions are allowed on all CloudTrail resources:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1446072643000",
      "Effect": "Allow",
      "Action": [
        "cloudtrail:DescribeTrails",
        "cloudtrail:GetTrailStatus"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

If a user of the developer group tries to start or stop logging on the trail that you specified in the first policy, that user gets an access denied exception. Users of the developer group can start and stop logging on trails that they create and manage.

The following CLI examples show that the developer group has been configured in an AWS CLI profile named `devgroup`. First, a user of `devgroup` runs the `describe-trails` command.

```
$ aws --profile devgroup cloudtrail describe-trails
```

The command complete successfully:

```
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "Default",
      "TrailARN": "arn:aws:cloudtrail:us-east-1:111122223333:trail/Default",

```

```
        "IsMultiRegionTrail": false,  
        "S3BucketName": "myS3bucket ",  
        "HomeRegion": "us-east-1"  
    }  
]  
}
```

The user then runs the `get-trail-status` command on the trail that you specified in the first policy.

```
$ aws --profile devgroup cloudtrail get-trail-status --name Default
```

The command complete successfully:

```
{  
  "LatestDeliveryTime": 1449517556.256,  
  "LatestDeliveryAttemptTime": "2015-12-07T19:45:56Z",  
  "LatestNotificationAttemptSucceeded": "",  
  "LatestDeliveryAttemptSucceeded": "2015-12-07T19:45:56Z",  
  "IsLogging": true,  
  "TimeLoggingStarted": "2015-12-07T19:36:27Z",  
  "StartLoggingTime": 1449516987.685,  
  "StopLoggingTime": 1449516977.332,  
  "LatestNotificationAttemptTime": "",  
  "TimeLoggingStopped": "2015-12-07T19:36:17Z"  
}
```

Next, a user of `devgroup` runs the `stop-logging` command on the same trail.

```
$ aws --profile devgroup cloudtrail stop-logging --name Default
```

The command returns an access denied exception:

```
A client error (AccessDeniedException) occurred when calling the StopLogging  
operation: Unknown
```

The user runs the `start-logging` command on the same trail.

```
$ aws --profile devgroup cloudtrail start-logging --name Default
```

The command returns an access denied exception:

```
A client error (AccessDeniedException) occurred when calling the StartLogging  
operation: Unknown
```

With resource level permissions, you can grant or deny access to specific trails in your account.

Granting Permission to View AWS Config Information on the CloudTrail Console

You can view event information on the CloudTrail console, including resources that are related to that event. For these resources, you can choose the AWS Config icon to view the timeline for that resource

in the AWS Config console. Attach this policy to your users to grant them read-only AWS Config access. The policy doesn't grant them permission to change settings in AWS Config.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "config:Get*",
      "config:Describe*",
      "config:List*"
    ],
    "Resource": "*"
  }]
}
```

For more information, see [Viewing Resources Referenced with AWS Config](#) (p. 47).

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*.

Getting and Viewing Your CloudTrail Log Files

After you've set up CloudTrail to capture the log files you want, you'll need to be able to find the log files and interpret the information they contain.

CloudTrail delivers your log files to an Amazon S3 bucket that you specify when you create the trail. Typically, log files appear in your bucket within 15 minutes of the recorded AWS API call or other AWS event. Log files are generally published every five minutes.

Topics

- [Finding Your CloudTrail Log Files](#) (p. 88)
- [Reading Your CloudTrail Log Files](#) (p. 89)

Finding Your CloudTrail Log Files

CloudTrail publishes log files to your S3 bucket in a gzip archive. In the S3 bucket, the log file has a formatted name that includes the following elements:

- The bucket name that you specified when you created trail (found on the Trails page of the CloudTrail console)
- The (optional) prefix you specified when you created your trail
- The string "AWSLogs"
- The account number
- The string "CloudTrail"
- A region identifier such as us-west-1
- The year the log file was published in YYYY format
- The month the log file was published in MM format
- The day the log file was published in DD format

- An alphanumeric string that disambiguates the file from others that cover the same time period

The following example shows a complete log file object name:

```
bucket_name/prefix_name/AWSLogs/Account ID/CloudTrail/re  
gion/YYYY/MM/DD/file_name.json.gz
```

To retrieve a log file, you can use the Amazon S3 console, the Amazon S3 command line interface (CLI), or the API.

To find your log files with the Amazon S3 console

1. Open the Amazon S3 console.
2. Choose the bucket you specified.
3. Navigate through the object hierarchy until you find the log file you want.
All log files have a .gz extension.

You will navigate through an object hierarchy that is similar to the following example, but with a different bucket name, account ID, region, and date.

```
All Buckets  
  Bucket_Name  
    AWSLogs  
      123456789012  
        CloudTrail  
          us-west-1  
            2014  
              06  
                20
```

A log file for the preceding object hierarchy will look like the following:

```
123456789012_CloudTrail_us-west-1_20140620T1255ZHdkvFTXOA3Vnhbc.json.gz
```

Note

Although uncommon, you may receive log files that contain one or more duplicate events. Duplicate events will have the same **eventID**. For more information about the **eventID** field, see [CloudTrail Record Contents \(p. 193\)](#).

Reading Your CloudTrail Log Files

This topic describes options for retrieving and viewing your CloudTrail log files.

Retrieving your log files

CloudTrail log files are Amazon S3 objects. You can retrieve them by using the Amazon S3 console, the AWS command line interface (CLI), or the Amazon S3 API. For more information, see [Working with Amazon S3 Objects](#) in the Amazon Simple Storage Service Developer Guide. The Amazon Simple Storage Service Console User Guide covers using the console to retrieve your objects. For example, open the

Amazon S3 console, click on the name of the bucket in which you're interested, and keep clicking through the object hierarchy until you get to the log file you're looking for. All log files have a .gz extension.

Viewing your log files

Log files are written in JSON (JavaScript Object Notation) format. If you have a JSON viewer add-on installed, you can view the files directly in your browser by double-clicking the log file name in the Amazon S3 bucket. This will open a new window or a new tab, depending on the add-on and on the browser, that displays the JSON in a readable format. To find a JSON viewer, search on that phrase in your browser of choice.

For example, if you use Google Chrome in Windows or Linux, a popular extension that you can add is named [JSONView](#) which you can download from the Chrome Web Store. If you use Mozilla Firefox, you can also download the [JSONView](#) add-on. With JSONView, you can double-click the compressed .gz file in your Amazon S3 bucket to open the log file in JSON format. There is no comparable extension for Internet Explorer, but there is a [registry edit](#) you can make to enable Internet Explorer to open JSON files after you download and decompress them.

An alternate approach to viewing your CloudTrail logs on Windows is to download them locally and use a text editor such as [Notepad++](#) along with the [JSON Viewer](#) plug-in. To download a log file, right-click on the file in your Amazon S3 bucket and right-click **Download** in the pop-up window. Click **Save link as...** and follow the prompts to save the file locally. This saves the file, however, in compressed format. You must use a product such as [7-Zip](#) to extract the uncompressed JSON data. After decompressing the file, open it in Notepad++, select all of the text, and navigate to **Plugins**, point to **JSON Viewer**, and then click **Format JSON**.

For more information about the event fields that can appear in a log file entry, see [CloudTrail Event Reference](#) (p. 192).

AWS partners with third-party specialists in logging and analysis to provide solutions that leverage CloudTrail output. For more information, visit the CloudTrail detail page at <http://aws.amazon.com/cloudtrail>.

Note

For log files captured during the last seven days, you can use the CloudTrail console, the AWS CLI or the AWS SDKs. For more information, see [Viewing Events with CloudTrail API Activity History](#) (p. 45).

Configuring Amazon SNS Notifications for CloudTrail

You can be notified when CloudTrail publishes new log files to your Amazon S3 bucket. You manage notifications using Amazon Simple Notification Service (Amazon SNS).

Notifications are optional. If you want notifications, you configure CloudTrail to send update information to an Amazon SNS topic whenever a new log file has been sent. To receive these notifications, you can use Amazon SNS to subscribe to the topic. As a subscriber you can get updates sent to a Amazon Simple Queue Service (Amazon SQS) queue, which enables you to handle these notifications programmatically.

Topics

- [Configuring CloudTrail to Send Notifications](#) (p. 91)
- [CloudTrail Permissions for SNS Notifications](#) (p. 92)

Configuring CloudTrail to Send Notifications

You can configure a trail to use an Amazon SNS topic. You can use the CloudTrail console or the [aws cloudtrail create-subscription](#) CLI command to create the topic. CloudTrail creates the Amazon SNS topic for you and attaches an appropriate policy, so that CloudTrail has permission to publish to that topic.

When you create an SNS topic name, the name must meet the following requirements:

- Between 1 and 256 characters long
- Contain uppercase and lowercase ASCII letters, numbers, underscores, or hyphens

When you configure notifications for a trail that applies to all regions, notifications from all regions are sent to the Amazon SNS topic that you specify. If you have one or more region-specific trails, you must create a separate topic for each region and subscribe to each individually.

To receive notifications, subscribe to the Amazon SNS topic or topics that CloudTrail uses. You do this with the Amazon SNS console or Amazon SNS CLI commands. For more information, see [Subscribe to a Topic](#) in the *Amazon Simple Notification Service Developer Guide*.

Tip

CloudTrail sends a notification when log files are written to the Amazon S3 bucket. An active account can generate a large number of notifications. If you subscribe with email or SMS, you can receive a large volume of messages. We recommend that you subscribe using Amazon Simple Queue Service (Amazon SQS), which lets you handle notifications programmatically. For more information, see [Subscribing a Queue to an Amazon SNS Topic](#) in the *Amazon Simple Queue Service Developer Guide*.

The Amazon SNS notification consists of a JSON object that includes a `Message` field. The `Message` field lists the full path to the log file, as shown in the following example:

```
{
  "s3Bucket": "your_bucket_name", "s3ObjectKey": [ "AWS
Logs/123456789012/CloudTrail/us-east-1/2013/12/13/123456789012_CloudTrail_us-
west-2_20131213T1920Z_LnPgDQnpkSKEsppV.json.gz" ]
}
```

If multiple log files are delivered to your Amazon S3 bucket, a notification may contain multiple logs, as shown in the following example:

```
{
  "s3Bucket": "your_bucket_name",
  "s3ObjectKey": [
    "AWSLogs/123456789012/CloudTrail/us-east-
1/2016/08/11/123456789012_CloudTrail_us-east-1_20160811T2215Z_kpaMYavM
QA9Ahp7L.json.gz",
    "AWSLogs/123456789012/CloudTrail/us-east-
1/2016/08/11/123456789012_CloudTrail_us-east-
1_20160811T2210Z_zqDkyQv3TK8ZdLr0.json.gz",
    "AWSLogs/123456789012/CloudTrail/us-east-
1/2016/08/11/123456789012_CloudTrail_us-east-1_20160811T2205Z_jaMVRa6Jf
dLCJYHP.json.gz"
  ]
}
```

If you choose to receive notifications by email, the body of the email consists of the content of the `Message` field. For a complete description of the JSON structure, see [Sending Amazon SNS Messages to Amazon](#)

[SQS Queues](#) in the *Amazon Simple Notification Service Developer Guide*. Only the `Message` field shows CloudTrail information. The other fields contain information from the Amazon SNS service.

If you create a trail with the CloudTrail API, you can specify an existing Amazon SNS topic that you want CloudTrail to send notifications to with the [CreateTrail](#) or [UpdateTrail](#) operations. You must make sure that the topic exists and that it has permissions that allow CloudTrail to send notifications to it. See [CloudTrail Permissions for SNS Notifications](#) (p. 92).

Additional Resources

For more information about Amazon SNS topics and about subscribing to them, see the [Amazon Simple Notification Service Developer Guide](#).

CloudTrail Permissions for SNS Notifications

CloudTrail must have permissions to send notifications to an Amazon SNS topic. If CloudTrail creates the topic for you automatically (for example, if you use the console to set up a new trail or use the `aws cloudtrail create-subscription` command), these permissions are automatically added to the new topic. However, if you specify an existing topic, you must make sure that the topic has the correct permissions.

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

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AWSCloudTrailSNSPolicy20131101",
    "Effect": "Allow",
    "Principal": {
      "Service": "cloudtrail.amazonaws.com"
    },
    "Action": "SNS:Publish",
    "Resource": "arn:aws:sns:Region:SNSTopicOwnerAccountId:SNSTopicName"
  }]
}
```

In the `Resource` field, *SNSTopicOwnerAccountId* is the account number of the topic owner; in topics that you create, this will be your account number. You must substitute appropriate values for *Region* and *SNSTopicName*.

Additional Resources

To learn more about creating Amazon SNS topics and about subscribing to them, see the [Amazon Simple Notification Service Developer Guide](#).

Working with CloudTrail Log Files

You can perform more advanced tasks with your CloudTrail files.

- Create multiple trails per region.
- Monitor CloudTrail log files in real time by sending them to CloudWatch Logs.
- Share log files between accounts.
- Use the AWS CloudTrail Processing Library to write log processing applications in Java.
- Validate your log files to verify that they have not changed after delivery by CloudTrail.

Topics

- [Create Multiple Trails \(p. 93\)](#)
- [Receiving CloudTrail Log Files from Multiple Regions \(p. 94\)](#)
- [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 95\)](#)
- [Receiving CloudTrail Log Files from Multiple Accounts \(p. 139\)](#)
- [Sharing CloudTrail Log Files Between AWS Accounts \(p. 142\)](#)
- [Encrypting CloudTrail Log Files with AWS KMS–Managed Keys \(SSE-KMS\) \(p. 152\)](#)
- [Validating CloudTrail Log File Integrity \(p. 163\)](#)
- [Using the CloudTrail Processing Library \(p. 184\)](#)

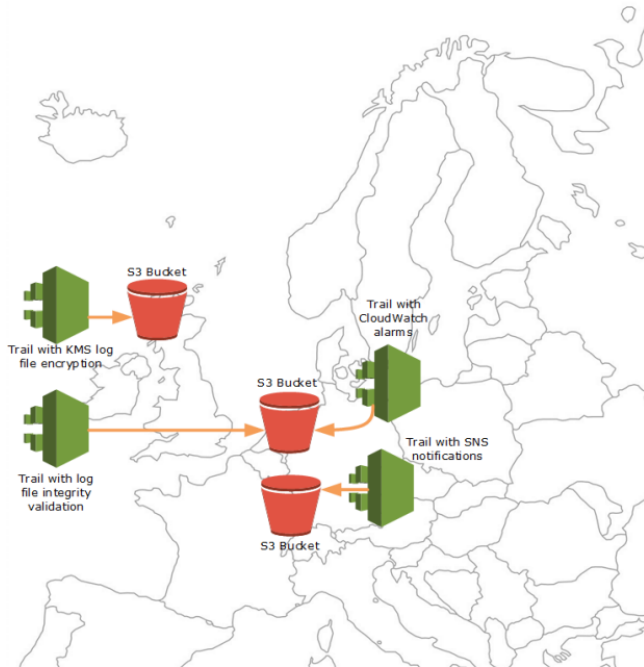
Create Multiple Trails

You can use CloudTrail log files to troubleshoot operational or security issues in your AWS account. You can create trails for different users, who can create and manage their own trails.

For example, you might have the following users:

- A security administrator creates a trail in the EU (Ireland) Region and configures KMS log file encryption. The log files are delivered to a separate S3 bucket.
- A developer creates a trail in the EU (Frankfurt) Region and configures CloudWatch alarms to receive notifications for specific API activity.
- An IT auditor creates a trail in the EU (Ireland) Region and configures log file integrity validation to ensure the log files have not changed since CloudTrail delivered them.
- Another developer creates a trail in the EU (Frankfurt) Region and configures SNS. The log files are delivered to a separate S3 bucket.

The following image illustrates this example.



Note

You can create up to five trails per region. A trail that logs activity from all regions counts as one trail per region.

You can use resource-level permissions to manage a user's ability to perform specific operations on CloudTrail.

For example, you might grant one user permission to view trail activity, but restrict the user from starting or stopping logging for a trail. You might grant another user full permission to create and delete trails. This gives you granular control over your trails and user access.

For more information about resource-level permissions, see [Controlling User Permissions for Actions on Specific Trails](#) (p. 85).

For more information about multiple trails, see the following resources:

- [How Does CloudTrail Behave Regionally and Globally?](#) (p. 5)
- [CloudTrail FAQs](#)

Receiving CloudTrail Log Files from Multiple Regions

You can configure CloudTrail to deliver log files from multiple regions to a single S3 bucket for a single account. For example, you have a trail in the US West (Oregon) Region that is configured to deliver log files to a S3 bucket, and a CloudWatch Logs log group. When you apply the trail to all regions, CloudTrail creates a new trail in all other regions. This trail has the original trail configuration. CloudTrail delivers log files to the same S3 bucket and CloudWatch Logs log group.

To receive CloudTrail log files from multiple regions

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
2. Choose **Trails**, and then choose a trail name.
3. Click the pencil icon next to **Apply trail to all regions**, and then choose **Yes**.
4. Choose **Save**. The original trail is now replicated across all regions. CloudTrail delivers log files from all regions to the specified S3 bucket.

Note

When a new region launches in the [aws partition](#), CloudTrail automatically creates a trail for you in the new region with the same settings as your original trail.

For more information, see the following resources:

- [How Does CloudTrail Behave Regionally and Globally? \(p. 5\)](#)
- [CloudTrail FAQs](#)

Monitoring CloudTrail Log Files with Amazon CloudWatch Logs

Topics

- [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#)
- [Using an AWS CloudFormation Template to Create CloudWatch Alarms \(p. 99\)](#)
- [Creating CloudWatch Alarms for CloudTrail Events: Examples \(p. 108\)](#)
- [Creating CloudWatch Alarms for CloudTrail Events: Additional Examples \(p. 131\)](#)
- [Configuring Notifications for CloudWatch Logs Alarms \(p. 137\)](#)
- [Stopping CloudTrail from Sending Events to CloudWatch Logs \(p. 137\)](#)
- [CloudWatch Log Group and Log Stream Naming for CloudTrail \(p. 138\)](#)
- [Role Policy Document for CloudTrail to Use CloudWatch Logs for Monitoring \(p. 138\)](#)

One of the ways that you can work with CloudTrail logs is to monitor them in real time by sending them to CloudWatch Logs. For a trail that is enabled in all regions in your account, CloudTrail sends log files from all those regions to a CloudWatch Logs log group. You define CloudWatch Logs metric filters that will evaluate your CloudTrail log events for matches in terms, phrases, or values. You assign CloudWatch metrics to the metric filters. You also create CloudWatch alarms that are triggered according to thresholds and time periods that you specify. You can configure an alarm to send a notification when the alarm is triggered so that you can take immediate action. You can also configure CloudWatch to automatically perform an action in response to an alarm. CloudTrail events are protected by SSL encryption as they are delivered from CloudTrail to the CloudWatch Logs log group.

CloudWatch Logs is supported in the following regions:

- US East (N. Virginia) (us-east-1)
- US West (N. California) (us-west-1)
- US West (Oregon) (us-west-2)
- Asia Pacific (Mumbai) (ap-south-1)
- Asia Pacific (Seoul) (ap-northeast-2)
- Asia Pacific (Singapore) (ap-southeast-1)

- Asia Pacific (Sydney) (ap-southeast-2)
- Asia Pacific (Tokyo) (ap-northeast-1)
- EU (Frankfurt) (eu-central-1)
- EU (Ireland) (eu-west-1)
- South America (São Paulo) (sa-east-1)

Standard pricing for Amazon CloudWatch and Amazon CloudWatch Logs apply. For more information, see [Amazon CloudWatch Pricing](#).

Sending CloudTrail Events to CloudWatch Logs

This topic describes how to configure CloudTrail to send logs to CloudWatch Logs so that you can monitor CloudTrail log events. Sending CloudTrail events to a CloudWatch Logs log group requires you to do a minimum of the following:

- Create a log group or specify an existing one.
- Specify or create an IAM role.
- Attach a role policy, or use the default.

Note

You must create a trail before you can configure CloudTrail to send log events to CloudWatch Logs. The procedures in this section assume you have already created a trail. For more information, see [Creating a Trail for the First Time \(p. 28\)](#) or [Creating and Updating a Trail with the AWS CLI \(p. 34\)](#). Also make sure to configure permissions for your IAM user to create a log group, change the log group, and assume the role by creating an action policy. For more information, see [Granting Custom Permissions for CloudTrail Users \(p. 83\)](#).

This topic includes procedures for the AWS Management Console and the AWS Command Line Interface (AWS CLI).

Configuring CloudWatch Logs Monitoring Using the Console

You can use the AWS Management Console to configure CloudTrail to send log events to CloudWatch Logs for monitoring.

Creating a Log Group or Specifying an Existing Log Group

CloudTrail uses a CloudWatch Logs log group as a delivery endpoint for log events. You can create a new log group or specify an existing one.

To specify a log group using the console

1. Navigate to the **CloudTrail Trails** page.
2. Choose the name of the trail that you want to configure.
3. In the **CloudWatch Logs (Optional)** box, do one of the following:
 - a. If you do not yet have any CloudWatch logs configured, choose **Configure**.
 - b. If you already have one or more CloudWatch logs configured, choose the **Edit** (pencil) icon to the right of **CloudWatch Logs (Optional)**.
4. In the **New or existing log group** box, type a log group name to organize CloudTrail events for you to review using CloudWatch Logs, and then choose **Continue**.

Note

For recommended log group naming conventions, see [Log Group and Log Stream Names](#).

Next, specify a role for CloudTrail to assume to deliver events to the log stream.

Specify a Role

To specify a role using the console

1. By default, the **CloudTrail_CloudWatchLogs_Role** is selected for you. To verify this, choose **View Details** and look at the **IAM Role** box. The default role policy contains the permissions required for creating a CloudWatch Logs log stream in a log group that you specify and for delivering CloudTrail events to that log stream. To see the contents of the role policy, choose **View Policy Document**.

Note

You can specify another role, but you must attach the appropriate role policy to the existing role if you want to use it to send log events to CloudWatch Logs.

2. Choose **Allow**.

When you are finished with these steps in the console, the CloudTrail trail will be set up to use the log group and role you specified to send events to CloudWatch Logs. If the trail you configured to use CloudWatch Logs receives log files across regions, events from all regions will be sent to the CloudWatch Logs log group that you specified.

Configuring CloudWatch Logs Monitoring Using the CLI

You can use the AWS CLI to configure CloudTrail to send log events to CloudWatch Logs for monitoring.

Creating a Log Group

1. If you do not already have an existing log group, create a CloudWatch Logs log group as a delivery endpoint for log events using the CloudWatch Logs command `create-log-group`.

```
aws logs create-log-group --log-group-name name
```

For example:

```
aws logs create-log-group --log-group-name CloudTrail/logs
```

2. Retrieve the log group Amazon Resource Name (ARN).

```
aws logs describe-log-groups
```

Creating a Role

Next, you will use the AWS CLI to create a role for CloudTrail that enables it to send events to the CloudWatch Logs log group. The IAM `create-role` command takes two parameters: a role name and a file path to an assume role policy document in JSON format. The policy document that you will use gives Assume Role permissions to CloudTrail. The `create-role` command will create the role with the proper assume role permissions.

To create the JSON file that will contain the policy document, open a text editor and save the following policy contents in a file called `assume_role_policy_document.json`.

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

Run the following command to create the role with Assume Role permissions for CloudTrail.

```
aws iam create-role --role-name role_name --assume-role-policy-document
file://<path to assume_role_policy_document>.json
```

When the command completes, take a note of the role ARN in the output.

Create a Policy Document

Next, create the following role policy document for CloudTrail. This document grants CloudTrail the required permissions to create a CloudWatch Logs log stream in the log group that you specify and to deliver CloudTrail events to that log stream.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailCreateLogStream20141110",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
        "arn:aws:logs:region:accountID:log-group:log_group_name:log-stream:ac
countID_CloudTrail_region*"
      ]
    },
    {
      "Sid": "AWSCloudTrailPutLogEvents20141101",
      "Effect": "Allow",
      "Action": [
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:region:accountID:log-group:log_group_name:log-stream:ac
countID_CloudTrail_region*"
      ]
    }
  ]
}
```

```
}  
  ]  
}
```

Save the policy document in a file called `role-policy-document.json`.

When you are done, run the following command to apply the policy to the role.

```
aws iam put-role-policy --role-name role_name --policy-name cloudtrail-policy  
--policy-document file://<path to role-policy-document>.json
```

Updating the Trail

Update the trail with the log group and role information using the CloudTrail `update-trail` command.

```
aws cloudtrail update-trail --name trail_name --cloud-watch-logs-log-group-arn  
log_group_arn --cloud-watch-logs-role-arn role_arn
```

For more information about the AWS CLI commands in these procedures, see the [AWS CloudTrail Command Line Reference](#).

Limitation

Because CloudWatch Logs has an [event size limitation of 256KB](#), CloudTrail does not send events larger than 256KB to CloudWatch Logs. For example, a call to the EC2 `RunInstances` API to launch 500 instances will exceed the 256KB limit. CloudTrail does not send the event to CloudWatch Logs. To ensure that CloudTrail sends events to CloudWatch Logs, break large requests into smaller batches.

Using an AWS CloudFormation Template to Create CloudWatch Alarms

You can create CloudWatch metric filters and alarms that monitor the CloudTrail events that you specify and send you notifications when the events occur. You can create your filters and alarms separately, or by using an AWS CloudFormation template to define them all at once.

This topic describes an example CloudFormation template from AWS that you can use as is, or as a starting point or as a reference for creating your own templates. For information on creating CloudWatch metric filters and alarms individually, see [Creating CloudWatch Alarms for CloudTrail Events: Examples](#) (p. 108).

The Example CloudFormation Template

The downloadable and editable example CloudFormation template from AWS contains predefined CloudWatch metric filters and alarms that enable you to receive email notifications when certain security-related API calls are made in your AWS account. You can download the template directly from the following link: https://s3-us-west-2.amazonaws.com/awscloudtrail/cloudwatch-alarms-for-cloudtrail-api-activity/CloudWatch_Alarms_for_CloudTrail_API_Activity.json.

The example template defines metric filters that monitor creation and deletion of, or updates to, security groups, network ACLs, internet gateways, Amazon EC2 instances, and IAM policies. For each filter, the template describes a corresponding alarm that enables you to receive email notifications when a call to one of the APIs being monitored by the filter is made.

By default, most of the filters in the template trigger an alarm when one monitored event occurs within a five-minute period. You can modify these alarm thresholds for your own requirements. For example, you could monitor for 3 events in a 10-minute period. To make the changes, you can edit the template directly or, after following the steps in the section that follows ([Using the CloudFormation template \(p. 100\)](#)), you can alter the thresholds in the [CloudWatch console](#).

Note

Because CloudTrail typically delivers log files every five minutes, it is highly recommended that you specify alarm periods of five minutes or more.

For a description of each of the metric filters and alarms in the template, and the API calls for which email notifications are triggered, see the section [CloudFormation Template Contents \(p. 104\)](#) later in this document.

Using the CloudFormation template

To use the template:

1. Configure CloudTrail log file delivery to CloudWatch Logs. See [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#).

Note

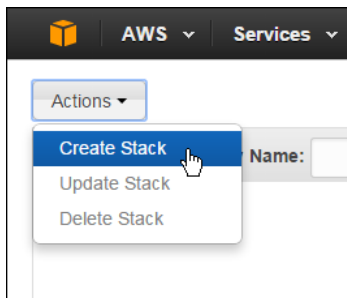
If you change the default log group name provided by CloudTrail, note it so that you can use it in the next step.

2. Create a AWS CloudFormation stack by using the template. A CloudFormation stack is a collection of related resources that you provision and update as a single unit.

The next section shows you how to create the stack and validate the email address that will receive any notifications that are generated.

Create a CloudFormation Stack

1. Download the CloudFormation template from https://s3-us-west-2.amazonaws.com/awscloudtrail/cloudwatch-alarms-for-cloudtrail-api-activity/CloudWatch_Alarms_for_CloudTrail_API_Activity.json.
2. Go to the [CloudFormation console](#) and create a new stack.



3. On the **Select Template** page, give the stack a name. This example uses `CloudWatchAlarmsForCloudTrail`.

AWS CloudTrail User Guide
Using an AWS CloudFormation Template to Create
CloudWatch Alarms

Select Template

Specify a stack name and then select the template that describes the stack that you want to create.

Stack

An AWS CloudFormation stack is a collection of related resources that you provision and update as a single unit.

Name

4. Under **Template**, select **Upload a template to Amazon S3**.

Template

A template is a JSON-formatted text file that describes your stack's resources and their properties. AWS CloudFormation stores the stack's template in an Amazon S3 bucket. [Learn more](#).

Source

Select a sample template

Upload a template to Amazon S3

Specify an Amazon S3 template URL

No file chosen

5. Click **Choose File**, and then browse to and select the CloudFormation template that you downloaded.
6. Click **Next**.
7. On the **Specify Parameters** page, provide the email address that will receive notifications, and the enter name of the log group name that you used when you configured CloudTrail log file delivery to CloudWatch Logs.

AWS CloudTrail User Guide
Using an AWS CloudFormation Template to Create
CloudWatch Alarms

Specify Parameters

Specify values or use the default values for the parameters that are associated with your AWS CloudFormation template.

Parameters

Email
Enter the email address where you would like to receive notifications

LogGroupName
Enter CloudWatch Logs log group name. Default is CloudTrail/DefaultLogGroup

[Cancel](#) [Previous](#) [Next](#)

8. Click **Next**.
9. On the **Options** page, you can create tags or configure other advanced options. These are not required.

Options

Tags

You can specify tags (key-value pairs) for resources in your stack. You can add up to 10 unique key-value pairs for each stack. [Learn more.](#)

	Key (127 characters maximum)	Value (255 characters maximum)	
1	<input type="text"/>	<input type="text"/>	+

► **Advanced**

You can set additional options for your stack, like notification options and a stack policy. [Learn more.](#)

[Cancel](#) [Previous](#) [Next](#)

10. Click **Next**.
11. On the **Review** page, verify that the template, email, log group, and other options, if any, are correct.

AWS CloudTrail User Guide
Using an AWS CloudFormation Template to Create
CloudWatch Alarms

Review

Template

Name	CloudWatchAlarmsForCloudTrail
Template URL	https://s3-us-west-2.amazonaws.com/cf-templates-1kf4fpqkb57z-us-west-2/2015060P4H-CloudWatch_Alarms_for_CloudTrail_API_Activity.json
Description	AWS CloudTrail API Activity Alarm Template for CloudWatch Logs
Estimate cost	Cost

Parameters

Email	
LogGroupName	CloudTrail/DefaultLogGroup
Create IAM resources	False

Options

Tags

No tags provided

Advanced

Notification Timeout	none
Rollback on failure	Yes

12. Click **Create**. The stack will be created in a few minutes.

Filter: Active ▾	By Name: <input type="text"/>		
Stack Name	Created Time	Status	Description
<input type="checkbox"/> CloudWatchAlarmsForCloudTrail	2015-02-28 17:18:55 UTC-0800	CREATE_COMPLETE	AWS CloudTrail API Activ

13. After the CloudFormation stack has been created, you will receive an email at the address that you specified to validate it.
14. In the email, click **Confirm subscription**.

You have chosen to subscribe to the topic:
arn:aws:sns:us-west-2:111222333444:CloudWatchAlarmsForCloudTrail-AlarmNotificationTopic-22ABC2DEFGHI2


To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
[Confirm subscription](#)

Please do not reply directly to this e-mail. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send email to [sns-opt-out](#)

You will now receive email notifications when the alarms specified by the template are triggered.

AWS CloudTrail User Guide

Using an AWS CloudFormation Template to Create CloudWatch Alarms



Sat 2/28/2015 7:11 PM
AWS Notifications <no-reply@sns.amazonaws.com>
ALARM: "CloudTrailIAMPolicyChanges" in US-West-2

To

You are receiving this email because your Amazon CloudWatch Alarm "CloudTrailIAMPolicyChanges" in the US-West-2 region has entered the ALARM state, because "Threshold Crossed: 1 datapoint (7.0) was greater than or equal to the threshold (1.0)." at "Sunday 01 March, 2015 03:10:34 UTC".

View this alarm in the AWS Management Console:
<https://console.aws.amazon.com/cloudwatch/home?region=us-west-2#s=Alarms&alarm=CloudTrailIAMPolicyChanges>

Alarm Details:

- Name: CloudTrailIAMPolicyChanges
- Description: Alarms when an API call is made to change an IAM policy.
- State Change: INSUFFICIENT_DATA -> ALARM
- Reason for State Change: Threshold Crossed: 1 datapoint (7.0) was greater than or equal to the threshold (1.0).
- Timestamp: Sunday 01 March, 2015 03:10:34 UTC
- AWS Account: 111122223333

Threshold:

- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 1.0 for 300 seconds.

Monitored Metric:

- MetricNamespace: CloudTrailMetrics
- MetricName: IAMPolicyEventCount
- Dimensions:
- Period: 300 seconds
- Statistic: Sum
- Unit: not specified

State Change Actions:

- OK:
- ALARM: [arn:aws:sns:us-west-2:111122223333:CWLAlarms1234-AlarmNotificationTopic-ABC1DEFG234H]
- INSUFFICIENT_DATA:

--

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
<https://sns.us-west-2.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-west-2:111122223333:CWLAlarms1234-AlarmNotificationTopic-ABC1DEFG234H:78080d51-8221-4ff5-b4b5-0366898a7d0a&Endpoint=janedoe@amazon.com>

Please do not reply directly to this e-mail. If you have any questions or comments regarding this email, please contact us at <https://aws.amazon.com/support>

You can review the metric filter or alarm definitions in the [CloudWatch console](#).

CloudFormation Template Contents

The following tables show each of the metric filters and alarms in the template, their purpose, and the API calls for which email notifications are triggered. Notifications are triggered when one or more of the API calls for a listed filter are made.

AWS CloudTrail User Guide
Using an AWS CloudFormation Template to Create
CloudWatch Alarms

Amazon S3 Bucket Events

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of
S3BucketChangesMetricFilter S3BucketChangesAlarm	API calls that change bucket policy, lifecycle, replication, or ACLs	PutBucketAcl PutBucketPolicy PutBucketCors PutBucketLifecycle PutBucketReplication DeleteBucketPolicy DeleteBucketCors DeleteBucketLifecycle DeleteBucketReplication

Network Events

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of
SecurityGroupChangesMetricFilter SecurityGroupChangesAlarm	API calls that create, update and delete Security Groups	AuthorizeSecurityGroupIngress AuthorizeSecurityGroupEgress RevokeSecurityGroupIngress RevokeSecurityGroupEgress CreateSecurityGroup DeleteSecurityGroup
NetworkAclChangesMetricFilter NetworkAclChangesAlarm	API calls that create, update and delete Network ACLs	CreateNetworkAcl CreateNetworkAclEntry DeleteNetworkAcl DeleteNetworkAclEntry ReplaceNetworkAclAssociation ReplaceNetworkAclEntry

AWS CloudTrail User Guide
Using an AWS CloudFormation Template to Create
CloudWatch Alarms

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of
GatewayChangesMetricFilter GatewayChangesAlarm	API calls that create, update and delete customer and Internet gateways	CreateCustomerGateway DeleteCustomerGateway AttachInternetGateway CreateInternetGateway DeleteInternetGateway DetachInternetGateway
VpcChangesMetricFilter VpcChangesAlarm	API calls that create, update and delete Virtual Private Clouds (VPCs), VPC peering connections and VPC connections to classic EC2 instances using ClassicLink	CreateVpc DeleteVpc ModifyVpcAttribute AcceptVpcPeeringConnection CreateVpcPeeringConnection DeleteVpcPeeringConnection RejectVpcPeeringConnection AttachClassicLinkVpc DetachClassicLinkVpc DisableVpcClassicLink EnableVpcClassicLink

Amazon EC2 Events

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of
EC2InstanceChangesMetricFilter EC2InstanceChangesAlarm	The creation, termination, start, stop, and reboot of EC2 instances	RebootInstances RunInstances StartInstances StopInstances TerminateInstances

AWS CloudTrail User Guide
Using an AWS CloudFormation Template to Create
CloudWatch Alarms

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of
EC2LargeInstanceChangesMetricFilter EC2LargeInstanceChangesAlarm	The creation, termination, start, stop, and reboot of 4x and 8x large EC2 instances	<u>At least one of</u> RebootInstances RunInstances StartInstances StopInstances TerminateInstances <u>and at least one of:</u> instancetype=*.4xlarge instancetype=*.8xlarge

CloudTrail and IAM Events

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of these calls (or activity)
CloudTrailChangesMetricFilter CloudTrailChangesAlarm	The creation or deletion of trails, or updates to trails. The occurrence of start and stop logging events for a trail.	CreateTrail DeleteTrail StartLogging StopLogging UpdateTrail
ConsoleSignInFailuresMetricFilter ConsoleSignInFailuresAlarm	Console login failures	eventName is ConsoleLogin <i>and</i> errorMessage is "Failed authentication"
AuthorizationFailuresMetricFilter AuthorizationFailuresAlarm	Authorization failures	Any API call which results in an error code of AccessDenied <i>or</i> *UnauthorizedOperation.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Metric Filter and Alarm	Monitor and Send Notifications for	Notifications triggered by one or more of these calls (or activity)
IAMPolicyChangesMetricFilter IAMPolicyChangesAlarm	Changes to IAM policies	DeleteGroupPolicy DeleteRolePolicy DeleteUserPolicy PutGroupPolicy PutRolePolicy PutUserPolicy CreatePolicy DeletePolicy CreatePolicyVersion DeletePolicyVersion AttachRolePolicy DetachRolePolicy AttachUserPolicy DetachUserPolicy AttachGroupPolicy DetachGroupPolicy

Creating CloudWatch Alarms for CloudTrail Events: Examples

This topic describes how to configure alarms for CloudTrail events using example scenarios.

Prerequisites

Before you can use the examples in this topic, you must:

- Create a trail with the CloudTrail console or CLI
- Create a log group
- Specify or create an IAM role that grants CloudTrail the permissions to create a CloudWatch Logs log stream in the log group that you specify and to deliver CloudTrail events to that log stream (the default CloudTrail_CloudWatchLogs_Role does this for you).

For more information, see [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#).

Create a metric filter, create an alarm

To create an alarm, you must first create a metric filter and then configure an alarm based on the filter. These steps are shown for every example in this topic.

Note

Instead of creating the metric filters and alarms that are presented here manually, you can use an AWS CloudFormation template to create them all at once. For more information, see [Using an AWS CloudFormation Template to Create CloudWatch Alarms \(p. 99\)](#).

Topics

- [Example: Amazon S3 Bucket Activity \(p. 109\)](#)
- [Example: Security Group Configuration Changes \(p. 111\)](#)
- [Example: Network Access Control List \(ACL\) Changes \(p. 113\)](#)
- [Example: Network Gateway Changes \(p. 115\)](#)
- [Example: Amazon Virtual Private Cloud \(VPC\) Changes \(p. 117\)](#)
- [Example: Amazon EC2 Instance Changes \(p. 119\)](#)
- [Example: EC2 Large Instance Changes \(p. 121\)](#)
- [Example: CloudTrail Changes \(p. 123\)](#)
- [Example: Console Sign-In Failures \(p. 125\)](#)
- [Example: Authorization Failures \(p. 127\)](#)
- [Example: IAM Policy Changes \(p. 129\)](#)

Example: Amazon S3 Bucket Activity

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an Amazon S3 API call is made to `PUT` or `DELETE` bucket policy, bucket lifecycle, bucket replication, or to `PUT` a bucket ACL. The alarm also is triggered for the CORS (cross-origin resource sharing) `PUT` bucket and `DELETE` bucket events. For information about cross-origin resource sharing, see [Cross-Origin Resource Sharing](#).

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($eventSource = s3.amazonaws.com) && (($eventName = PutBucketAcl) ||
($eventName = PutBucketPolicy) || ($eventName = PutBucketCors) ||
($eventName = PutBucketLifecycle) || ($eventName = PutBucketReplication)
|| ($eventName = DeleteBucketPolicy) || ($eventName = DeleteBucketCors)
|| ($eventName = DeleteBucketLifecycle) || ($eventName = DeleteBucketRep
lication)) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **S3BucketActivity**
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **S3BucketActivityEventCount**.
9. Click **Metric Value**, and then type **1**. If **Metric Value** does not appear, click **Advanced** first.

10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the **S3BucketActivity** filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

Create Alarm

[1. Select Metric](#) **[2. Define Alarm](#)**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 **Name:**

Description:

Whenever: S3BucketActivityEventCount

2 **is:**

3 **for:** consecutive period(s)

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm:

6 **Send notification to:** [Select list](#) ⓘ

7 **Email list:**

[+ Notification](#) [+ AutoScaling Action](#) [+ EC2 Action](#)

Alarm Preview

This alarm will trigger when the b to or above the red line for a dura








4 **Period:**

5 **Statistic:**

Namespace: CloudTrailM
Metric Name:

[Cancel](#) [Back](#) [N](#)

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Setting	Value
	S3 Bucket Activity
	1
	1
	5 Minutes
	Sum
	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

- When you are finished, click **Create Alarm**.

Example: Security Group Configuration Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when any configuration changes happen involving security groups.

Create a Metric Filter

- Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
- In the navigation pane, click **Logs**.
- In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
- Click **Create Metric Filter**.
- On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = AuthorizeSecurityGroupIngress) || ($.eventName = AuthorizeSecurityGroupEgress) || ($.eventName = RevokeSecurityGroupIngress) || ($.eventName = RevokeSecurityGroupEgress) || ($.eventName = CreateSecurityGroup) || ($.eventName = DeleteSecurityGroup) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

- Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **SecurityGroupEvents**.
- Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
- In the **Metric Name** field, enter **SecurityGroupEventCount**.
- Click **Metric Value**, and then type **1**.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

Create Alarm
✕

1. [Select Metric](#)
2. [Define Alarm](#)

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: SecurityGroupEventCount

2 Is: >= 1

3 for: 1 consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics

Metric Name:

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm: State is ALARM

6 Send notification to: [Select list](#) ⓘ


7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel
Back
Next
Create Alarm

Setting	Value
1	Security Group Configuration Changes
2	>=1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Setting	Value
	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

- When you are finished, click **Create Alarm**.

Example: Network Access Control List (ACL) Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when any configuration changes happen involving network ACLs.

Create a Metric Filter

- Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
- In the navigation pane, click **Logs**.
- In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
- Click **Create Metric Filter**.
- On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = CreateNetworkAcl) || ($.eventName = CreateNetworkAclEntry)
  || ($.eventName = DeleteNetworkAcl) || ($.eventName = DeleteNetworkAclEntry)
  || ($.eventName = ReplaceNetworkAclEntry) || ($.eventName = ReplaceNetwork
AclAssociation) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

- Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **NetworkACLEvents**.
- Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
- In the **Metric Name** box, enter **NetworkACLEventCount**.
- Click **Metric Value**, and then type **1**.
- When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

- On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
- On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. Select Metric
2. Define Alarm

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: NetworkACLEventCount

2 is: >= 1

3 for: 1 consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics

Metric Name:

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm: State is ALARM

6 Send notification to: [Select list](#) ⓘ

7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel
Back
Next
Create Alarm

Setting	Value
1	Network ACL Configuration Changes
2	>=1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: Network Gateway Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an API call is made to create, update, or delete a customer or Internet gateway.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = CreateCustomerGateway) || ($.eventName = DeleteCustomerGateway) || ($.eventName = AttachInternetGateway) || ($.eventName = CreateInternetGateway) || ($.eventName = DeleteInternetGateway) || ($.eventName = DetachInternetGateway) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **GatewayChanges**.
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **GatewayEventCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Example: Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. Select Metric
2. Define Alarm

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: GatewayEventCount

2 is: >= 1

3 for: 1 consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics
Metric Name: GatewayEventCount

4 Period: 5 Minutes

5 Statistic: Sum

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm: State is ALARM

6 Send notification to: NotifyMe Select list ⓘ

7 Email list: user1@example.net.

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel Back Next Create Alarm

Setting	Value
1	Network Gateway Changes
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: Amazon Virtual Private Cloud (VPC) Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an API call is made to create, update or delete an Amazon VPC, an Amazon VPC peering connection or an Amazon VPC connection to classic Amazon EC2 instances.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = CreateVpc) || ($.eventName = DeleteVpc) || ($.eventName = ModifyVpcAttribute) || ($.eventName = AcceptVpcPeeringConnection) || ($.eventName = CreateVpcPeeringConnection) || ($.eventName = DeleteVpcPeeringConnection) || ($.eventName = RejectVpcPeeringConnection) || ($.eventName = AttachClassicLinkVpc) || ($.eventName = DetachClassicLinkVpc) || ($.eventName = DisableVpcClassicLink) || ($.eventName = EnableVpcClassicLink) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **VpcChanges**.
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **VpcEventCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. [Select Metric](#)
2. **Define Alarm**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: VpcEventCount

2 is:

3 for: consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics

Metric Name:

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel
Back
Next
Create Alarm

Setting	Value
1	VPC Changes
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: Amazon EC2 Instance Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an API call is made to create, terminate, start, stop or reboot an Amazon EC2 instance.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = RunInstances) || ($.eventName = RebootInstances) ||  
  ($.eventName = StartInstances) || ($.eventName = StopInstances) ||  
  ($.eventName = TerminateInstances) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **EC2InstanceChanges**
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **EC2InstanceEventCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. [Select Metric](#)
2. [Define Alarm](#)

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: EC2InstanceEventCount

2 is:

3 for: consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics
Metric Name: EC2InstanceEventCount

4 Period:

5 Statistic:

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel Back Next Create Alarm

Setting	Value
1	EC2 Instance Changes
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: EC2 Large Instance Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an API call is made to create a 4x or 8x-large EC2 instance.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = RunInstances) && (($.requestParameters.instanceType = *.8xlarge) || ($.requestParameters.instanceType = *.4xlarge)) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **EC2LargeInstanceChanges**
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **EC2LargeInstanceEventCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. Select Metric
2. Define Alarm

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: EC2LargeInstanceEventCount

2 is:

3 for: consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics
Metric Name: EC2LargeInstanceEven

4 Period:

5 Statistic:

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

Cancel Back Next Create Alarm

Setting	Value
1	EC2 Large Instance Changes
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: CloudTrail Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an API call is made to create, update or delete a CloudTrail trail, or to start or stop logging to a trail.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = CreateTrail) || ($.eventName = UpdateTrail) || ($.eventName = DeleteTrail) || ($.eventName = StartLogging) || ($.eventName = StopLogging) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **CloudTrailChanges**.
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **CloudTrailEventCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. [Select Metric](#)
2. **Define Alarm**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: CloudTrailEventCount

2 is: >= 1

3 for: 1 consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

CloudTrailEventCount >= 1

Namespace: CloudTrailMetrics

Metric Name:

4 Period: 5 Minutes

5 Statistic: Sum

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm: State is ALARM

6 Send notification to: [Select list](#) ⓘ

7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel Back Next Create Alarm

Setting	Value
1	CloudTrail Changes
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: Console Sign-In Failures

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when there are three or more sign-in failures during a five minute period.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName = ConsoleLogin) && ($.errorMessage = "Failed authentication")
}
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **ConsoleSignInFailures**
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** box, enter **ConsoleSigninFailureCount**.
9. Click **Show advanced metric settings**.
10. Click **Metric Value**, and then type **1**.
11. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. [Select Metric](#)
2. **Define Alarm**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: ConsoleSigninFailureCount

2 is: >= 3

3 for: 1 consecutive period(s)

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm: State is ALARM

6 Send notification to: NotifyMe [Select list](#) ⓘ

7 Email list: user1@example.net

+ Notification
+ AutoScaling Action
+ EC2 Action

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

ConsoleSigninFailureCount >= 3

Namespace: CloudTrailMetrics

Metric Name: ConsoleSigninFailureCc

4 Period: 5 Minutes

5 Statistic: Sum

Cancel
Back
Next
Create Alarm

Setting	Value
1	Console Sign-in Failures
2	>=3
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: Authorization Failures

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an unauthorized API call is made.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.errorCode = "*UnauthorizedOperation") || ($.errorCode = "AccessDenied*")
}
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **AuthorizationFailures**
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **AuthorizationFailureCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. [Select Metric](#)
2. **Define Alarm**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: AuthorizationFailureCount

2 is:

3 for: consecutive period(s)

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

AuthorizationFailureCount >= 1

Namespace: CloudTrailMetrics

Metric Name: AuthorizationFailureCou

4 Period:

5 Statistic:

Cancel
Back
Next
Create Alarm

Setting	Value
1	Authorization Failures
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Example: IAM Policy Changes

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when an API call is made to change an IAM policy.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ ($.eventName=DeleteGroupPolicy) || ($.eventName=DeleteRolePolicy) || ($.eventName=DeleteUserPolicy) || ($.eventName=PutGroupPolicy) || ($.eventName=PutRolePolicy) || ($.eventName=PutUserPolicy) || ($.eventName=CreatePolicy) || ($.eventName=DeletePolicy) || ($.eventName=CreatePolicyVersion) || ($.eventName=DeletePolicyVersion) || ($.eventName=AttachRolePolicy) || ($.eventName=DetachRolePolicy) || ($.eventName=AttachUserPolicy) || ($.eventName=DetachUserPolicy) || ($.eventName=AttachGroupPolicy) || ($.eventName=DetachGroupPolicy) }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **IAMPolicyChanges**.
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **IAMPolicyEventCount**.
9. Click **Metric Value**, and then type **1**.
10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Examples

Create Alarm
✕

1. [Select Metric](#)
2. [Define Alarm](#)

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: IAMPolicyEventCount

2 is:

3 for: consecutive period(s)

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line for a duration of 5 minutes

Namespace: CloudTrailMetrics

Metric Name:

Actions

Define what actions are taken when your alarm changes state.

Notification
Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

+ Notification
+ AutoScaling Action
+ EC2 Action

Cancel
Back
Next
Create Alarm

Setting	Value
1	IAM Policy Changes
2	1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Creating CloudWatch Alarms for CloudTrail Events: Additional Examples

[AWS Identity and Access Management \(IAM\) best practices](#) recommend that you do not use your root account credentials to access AWS. Instead, you should [create individual IAM users](#) so that you can give each user a unique set of security credentials. The IAM Best Practices also recommend that you [enable multi-factor authentication \(MFA\)](#) for IAM users who are allowed access to sensitive resources or APIs.

You can monitor whether activity in your AWS account adheres to these best practices by creating the CloudWatch alarms that notify you when root account credentials have been used to access AWS, or when API activity or console sign-ins without MFA have occurred. These alarms are described in this document.

Configuring an alarm involves two main steps:

- Create a metric filter
- Create an alarm based on the filter

Topics

- [Example: Monitor for Root Usage \(p. 131\)](#)
- [Example: Monitor for API Activity Without Multi-factor Authentication \(MFA\) \(p. 133\)](#)
- [Example: Monitor for Console Sign In Without Multi-factor Authentication \(MFA\) \(p. 135\)](#)

Example: Monitor for Root Usage

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when root (account) credentials are used.

Create a Metric Filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, click **Logs**.
3. In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
4. Click **Create Metric Filter**.
5. On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ $.userIdentity.type = "Root" && $.userIdentity.invokedBy NOT EXISTS && $.eventType != "AwsServiceEvent" }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

6. Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **RootAccountUsage**
7. Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
8. In the **Metric Name** field, enter **RootAccountUsageCount**.
9. Click **Metric Value**, and then type **1**.

Note

If **Metric Value** does not appear, click **Show advanced metric settings** first.

10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

Create Alarm

1. Select Metric **2. Define Alarm**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: RootAccountUsageCount

2 is:

3 for: consecutive period(s)

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

[+ Notification](#) [+ AutoScaling Action](#) [+ EC2 Action](#)

Alarm Preview

This alarm will trigger when the b to or above the red line for a dura

RootAccountUsageCount >

Namespace: CloudTrail

Metric Name:

4 Period:

5 Statistic:

[Cancel](#) [Back](#) [Next](#)

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Additional Examples

Setting	Value
1	Root Account Usage
2	>=1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

- When you are finished, click **Create Alarm**.

Example: Monitor for API Activity Without Multi-factor Authentication (MFA)

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when API calls are made without the use of multi-factor authentication (MFA).

Create a Metric Filter

- Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
- In the navigation pane, click **Logs**.
- In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
- Click **Create Metric Filter**.
- On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ $.userIdentity.sessionContext.attributes.mfaAuthenticated != "true" }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

- Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **ApiActivityWithoutMFA**.
- Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
- In the **Metric Name** box, enter **ApiActivityWithoutMFACount**.
- Click **Metric Value**, and then type **1**.

Note

If **Metric Value** does not appear, click **Show advanced metric settings** first.

10. When you are finished, click **Create Filter**.

Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

Create Alarm

1. Select Metric **2. Define Alarm**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 Name:

Description:

Whenever: ApiActivityWithoutMFACount

2 is:

3 for: consecutive period(s)

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm:

6 Send notification to: [Select list](#) ⓘ

7 Email list:

[+ Notification](#) [+ AutoScaling Action](#) [+ EC2 Action](#)

Alarm Preview

This alarm will trigger when the b to or above the red line for a dura

ApiActivityWithoutMFACount

Namespace: CloudTrail

Metric Name:

4 Period:

5 Statistic:

[Cancel](#) [Back](#) [Next](#)

AWS CloudTrail User Guide
Creating CloudWatch Alarms for CloudTrail Events:
Additional Examples

Setting	Value
1	Api Activity Without MFA
2	>=1
3	1
4	5 Minutes
5	Sum
6	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
7	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

- When you are finished, click **Create Alarm**.

Example: Monitor for Console Sign In Without Multi-factor Authentication (MFA)

This scenario walks you through how to use the AWS Management Console to create an Amazon CloudWatch alarm that is triggered when a console sign in is made without multi-factor authentication.

Create a Metric Filter

- Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
- In the navigation pane, click **Logs**.
- In the list of log groups, select the check box next to the log group that you created for CloudTrail log events.
- Click **Create Metric Filter**.
- On the **Define Logs Metric Filter** screen, click **Filter Pattern** and then type the following:

```
{ $.eventName = "ConsoleLogin" && $.additionalEventData.MFAUsed != "No" }
```

Note

For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of [Filter and Pattern Syntax](#) in the Amazon CloudWatch User Guide.

- Click **Assign Metric**, and then on the **Create Metric Filter and Assign a Metric** screen, in the **Filter Name** box, enter **ConsoleSignInWithoutMfa**
- Under **Metric Details**, in the **Metric Namespace** box, enter **CloudTrailMetrics**.
- In the **Metric Name** field, enter **ConsoleSignInWithoutMfaCount**.
- Click **Metric Value**, and then type 1.

Note

If **Metric Value** does not appear, click **Show advanced metric settings** first.

10. When you are finished, click **Create Filter**.

Example: Create an Alarm

These steps are a continuation of the previous steps for creating a metric filter.

1. On the **Filters for *Log_Group_Name*** page, next to the filter name, click **Create Alarm**.
2. On the **Create Alarm** page, provide the following values.

Create Alarm

[1. Select Metric](#) **[2. Define Alarm](#)**

Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

1 **Name:**

2 **Description:**

Whenever: ConsoleSignInWithoutMfaCount

2 **is:**

3 **for:** consecutive period(s)

Actions

Define what actions are taken when your alarm changes state.

Notification Delete

Whenever this alarm:

6 **Send notification to:** [Select list](#) **i**

7 **Email list:**

[+ Notification](#) [+ AutoScaling Action](#) [+ EC2 Action](#)

Alarm Preview

This alarm will trigger when the metric value is at or above the red line for a duration of 5 minutes.








Namespace: CloudTrail

Metric Name:

4 **Period:**

5 **Statistic:**

[Cancel](#) [Back](#) [Next](#)

Setting	Value
	Console Sign In Without MFA
	1
	1
	5 Minutes
	Sum
	Near the Select a notification list box, click New list , and then type a unique topic name for the list.
	Click Email list , and then type the email address to which you want notifications sent. (You will receive an email at this address to confirm that you created this alarm.)

3. When you are finished, click **Create Alarm**.

Configuring Notifications for CloudWatch Logs Alarms

You can configure CloudWatch Logs to send a notification whenever an alarm is triggered for CloudTrail. Doing so enables you to respond quickly to critical operational events captured in CloudTrail events and detected by CloudWatch Logs. CloudWatch uses Amazon Simple Notification Service (SNS) to send email. For more information, see [Set Up Amazon SNS](#) in the CloudWatch Developer Guide.

Stopping CloudTrail from Sending Events to CloudWatch Logs

You can stop sending events to CloudWatch Logs by deleting the delivery endpoint.

AWS Management Console

To remove the CloudWatch Logs delivery endpoint using the AWS Management Console

1. Sign in to the AWS Management Console.
2. Navigate to the CloudTrail console.
3. In the navigation pane, click **Configuration**.
4. In the **CloudWatch Logs (optional)** section, click the **Delete** (trash can) icon.
5. Click **Continue** to confirm.

AWS Command Line Interface (CLI)

You can remove the CloudWatch Logs log group as a delivery endpoint using the `update-trail` command. The following command clears the log group and role from the trail configuration.

```
aws cloudtrail update-trail --name trailname --cloud-watch-logs-log-group-arn=" "
--cloud-watch-logs-role-arn=" "
```

CloudWatch Log Group and Log Stream Naming for CloudTrail

Amazon CloudWatch will display the log group that you created for CloudTrail events alongside any other log groups you have in a region. We recommend that you use a log group name that helps you easily distinguish the log group from others. For example, `CloudTrail/logs`. Log group names can be between 1 and 512 characters long. Allowed characters include a-z, A-Z, 0-9, '_' (underscore), '-' (hyphen), '/' (forward slash), and '.' (period).

When CloudTrail creates the log stream for the log group, it names the log stream according to the following format: `account_ID_CloudTrail_source_region`.

Note

If the volume of CloudTrail logs is large, multiple log streams may be created to deliver log data to your log group.

Role Policy Document for CloudTrail to Use CloudWatch Logs for Monitoring

This section describes the trust policy required for the CloudTrail role to send log events to CloudWatch Logs. You can attach a policy document to a role when you configure CloudTrail to send events, as described in [Sending CloudTrail Events to CloudWatch Logs \(p. 96\)](#). You can also create a role using IAM. For more information, see [Creating a Role for an AWS Service \(AWS Management Console\)](#) or [Creating a Role \(CLI and API\)](#).

The following policy document contains the permissions required to create a CloudWatch log stream in the log group that you specify and to deliver CloudTrail events to that log stream. (This is the default policy for the default IAM role `CloudTrail_CloudWatchLogs_Role`.)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailCreateLogStream2014110",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:accountID:log-group:log_group_name:log-
stream:CloudTrail_log_stream_name_prefix*"
      ]
    }
  ],
}
```

```
{
  "Sid": "AWSCloudTrailPutLogEvents20141101",
  "Effect": "Allow",
  "Action": [
    "logs:PutLogEvents"
  ],
  "Resource": [
    "arn:aws:logs:us-east-1:accountID:log-group:log_group_name:log-
stream:CloudTrail_log_stream_name_prefix*"
  ]
}
```

Receiving CloudTrail Log Files from Multiple Accounts

You can have CloudTrail deliver log files from multiple AWS accounts into a single Amazon S3 bucket. For example, you have four AWS accounts with account IDs 111111111111, 222222222222, 333333333333, and 444444444444, and you want to configure CloudTrail to deliver log files from all four of these accounts to a bucket belonging to account 111111111111. To accomplish this, complete the following steps in order:

1. Turn on CloudTrail in the account where the destination bucket will belong (111111111111 in this example). Do not turn on CloudTrail in any other accounts yet.

For instructions, see [Creating a Trail for the First Time \(p. 28\)](#).

2. Update the bucket policy on your destination bucket to grant cross-account permissions to CloudTrail.

For instructions, see [Setting Bucket Policy for Multiple Accounts \(p. 139\)](#).

3. Turn on CloudTrail in the other accounts you want (222222222222, 333333333333, and 444444444444 in this example). Configure CloudTrail in these accounts to use the same bucket belonging to the account that you specified in step 1 (111111111111 in this example).

For instructions, see [Turning on CloudTrail in Additional Accounts \(p. 140\)](#).

Topics

- [Setting Bucket Policy for Multiple Accounts \(p. 139\)](#)
- [Turning on CloudTrail in Additional Accounts \(p. 140\)](#)

Setting Bucket Policy for Multiple Accounts

For a bucket to receive log files from multiple accounts, its bucket policy must grant CloudTrail permission to write log files from all the accounts you specify. This means that you must modify the bucket policy on your destination bucket to grant CloudTrail permission to write log files from each specified account.

To modify bucket permissions so that files can be received from multiple accounts

1. Sign in to the AWS Management Console using the account that owns the bucket (111111111111 in this example) and open the Amazon S3 console.
2. Choose the bucket where CloudTrail delivers your log files and then choose **Properties**.

3. Choose **Permissions**.
4. Choose **Edit Bucket Policy**.
5. Modify the existing policy to add a line for each additional account whose log files you want delivered to this bucket. See the following example policy and note the underlined `Resource` line specifying a second account ID.

Note

An AWS account ID is a twelve-digit number, and leading zeros must not be omitted.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailAclCheck20131101",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::myBucketName"
    },
    {
      "Sid": "AWSCloudTrailWrite20131101",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": [
        "arn:aws:s3:::myBucketName/[optional] myLogFilePrefix/AWS
Logs/111111111111/*",
        "arn:aws:s3:::myBucketName/[optional] myLogFilePrefix/AWS
Logs/222222222222/*"
      ],
      "Condition": {
        "StringEquals": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    }
  ]
}
```

Turning on CloudTrail in Additional Accounts

You can use the console or the command line interface to turn on CloudTrail in additional AWS accounts.

Using the Console to Turn on CloudTrail in Additional AWS Accounts

You can use the CloudTrail console to turn on CloudTrail in additional accounts.

1. Sign into the AWS management console using account 222222222222 credentials and open the AWS CloudTrail console. In the navigation bar, select the region where you want to turn on CloudTrail.

2. Choose **Get Started Now**.
3. On the following page, type a name for your trail in the **Trail name** box.
4. For **Create a new S3 bucket?**, choose **No**. Use the text box to enter the name of the bucket you created previously for storing log files when you signed in using account 111111111111 credentials. CloudTrail displays a warning asking you if you are sure that you want to specify an S3 bucket in another account. Verify the name of the bucket you entered.
5. Choose **Advanced**.
6. In the **Log file prefix** field, enter the same prefix you entered for storing log files when you turned on CloudTrail using account 111111111111 credentials. If you choose to use a prefix that is different from the one you entered when you turned on CloudTrail in the first account, you must edit the bucket policy on your destination bucket to allow CloudTrail to write log files to your bucket using this new prefix.
7. (Optional) Choose **Yes** or **No** for **SNS notification for every log file delivery?**. If you chose **Yes**, type a name for your Amazon SNS topic in the **SNS topic (new)** field.

Note

Amazon SNS is a regional service, so if you choose to create a topic, that topic will exist in the same region in which you turn on CloudTrail. If you have a trail that applies to all regions, you can pick an Amazon SNS topic in any region as long as you have the correct policy applied to the topic. For more information, see [CloudTrail Permissions for SNS Notifications \(p. 92\)](#).

8. Choose **Turn On**.

In about 15 minutes, CloudTrail starts publishing log files that show the AWS calls made in your accounts in this region since you completed the preceding steps.

Using the CLI to Turn on CloudTrail in Additional AWS Accounts

You can use the AWS command line tools to turn on CloudTrail in additional accounts and aggregate their log files to one Amazon S3 bucket. For more information about these tools, see the [AWS Command Line Interface User Guide](#).

Turn on CloudTrail in your additional accounts by using the `create-subscription` command. Use the following options to specify additional settings:

- `--name` specifies the name of the trail.
- `--s3-use-bucket` specifies the existing Amazon S3 bucket, created when you turned on CloudTrail in your first account (111111111111 in this example).
- `--s3-prefix` specifies a prefix for the log file delivery path (optional).
- `--sns-new-topic` specifies the name of the Amazon SNS topic to which you can subscribe for notification of log file delivery to your bucket (optional).

In contrast to trails that you create using the console, you must give every trail you create with the AWS CLI a name. You can create one trail for each region in which an account is running AWS resources.

The following example command shows how to create a trail for your additional accounts by using the AWS CLI. To have log files for these account delivered to the bucket you created in your first account (111111111111 in this example), specify the bucket name in the `--s3-new-bucket` option. Amazon S3 bucket names are globally unique.

```
aws cloudtrail create-subscription --name AWSCloudTrailExample --s3-use-bucket
MyBucketBelongingToAccount111111111111 --s3-prefix AWSCloudTrailPrefixExample
--sns-new-topic AWSCloudTrailLogDeliveryTopicExample
```

When you run the command, you will see output similar to the following:

```
CloudTrail configuration:
{
  "trailList": [
    {
      "S3KeyPrefix": "AWSCloudTrailPrefixExample",
      "IncludeGlobalServiceEvents": true,
      "Name": "AWSCloudTrailExample",
      "SnsTopicName": "AWSCloudTrailLogDeliveryTopicExample",
      "S3BucketName": "MyBucketBelongingToAccount111111111111"
    }
  ]
}
```

For more information about using CloudTrail from the AWS command line tools, see the [CloudTrail command line reference](#).

Sharing CloudTrail Log Files Between AWS Accounts

This section explains how to share CloudTrail log files between multiple AWS accounts. We will assume that the log files have all been received in a single Amazon S3 bucket, which is the default setting for a trail created in the CloudTrail console. In the first scenario, you will learn how to grant read-only access to the accounts that generated the log files that have been placed into your Amazon S3 bucket. In the second scenario, you will learn how to grant access to all of the log files to a third-party account that can analyze the files for you.

To share log files between multiple AWS accounts, you must perform the following general steps. These steps are explained in detail later in this section.

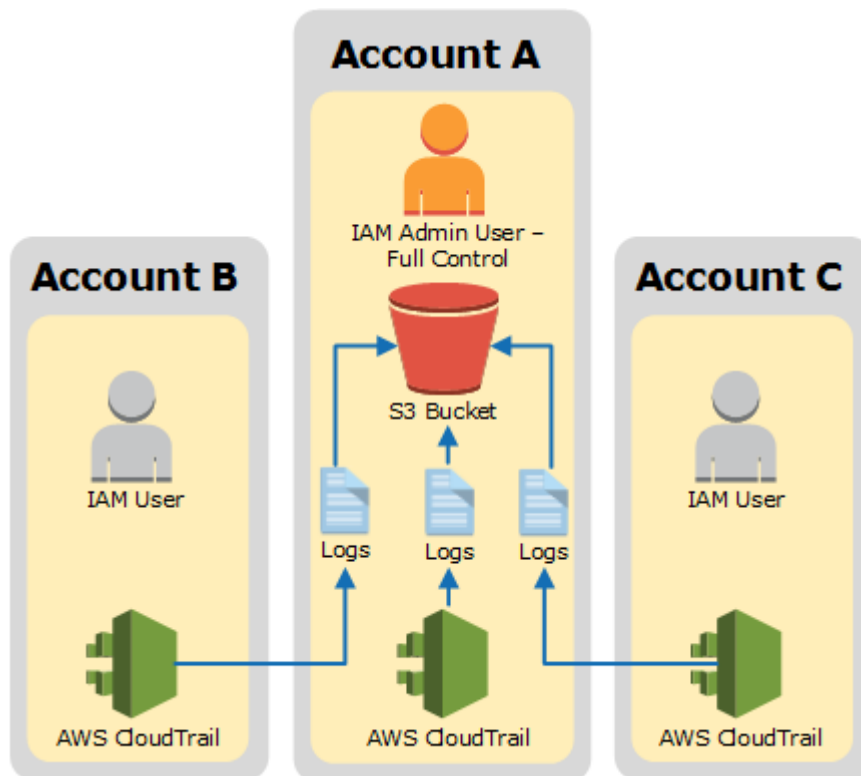
- Create an IAM role for each account that you want to share log files with.
- For each of these IAM roles, create an access policy that grants read-only access to the account you want to share the log files with.
- Have an IAM user in each account programmatically assume the appropriate role and retrieve the log files.

This section walks you through the preceding steps in the context of two different sharing scenarios: granting access to the log files to each account that generated those files, and sharing log files with a third party. Most of the steps you take for the two scenarios are the same; the important difference is in what kind of permissions the IAM role grants to each account. That is, you can grant permission for an account to read only its own log files, or you can grant an account permission to read all log files. For details about permissions management for IAM roles, see [Roles \(Delegation and Federation\)](#) in *IAM User Guide*.

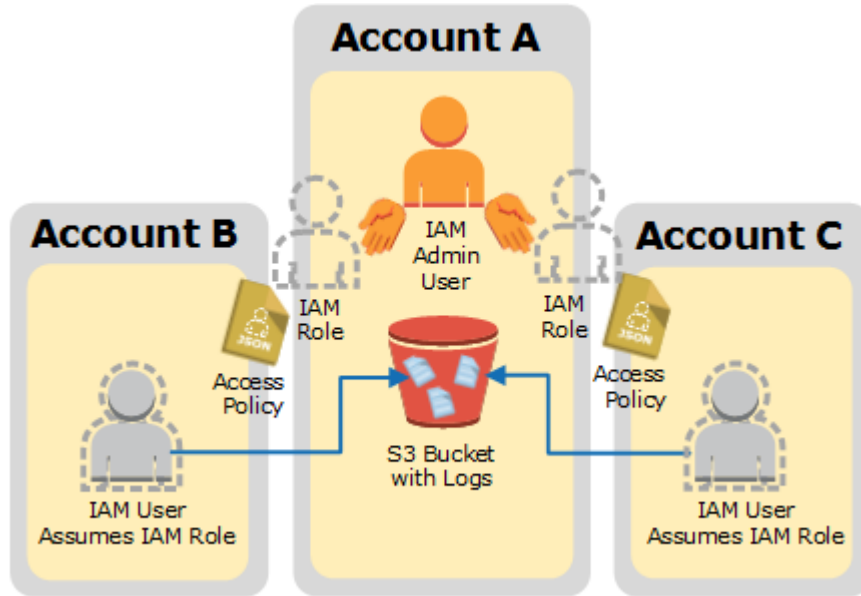
Scenario 1: Granting Access to the Account that Generated the Log Files

In this scenario, we'll assume that your enterprise is made up of two business units and that it maintains three AWS accounts. The first account, Account A, is the top-level account. For example, it might be managed by your enterprise's IT department and therefore be responsible for collecting log files from all other departments and business units into a single bucket. The other two accounts, B and C, correspond to your enterprise's business units.

This scenario assumes that you have already configured the log files from all three accounts to be delivered to a single Amazon S3 bucket, and that account A has full control over that bucket, as shown in the following illustration.



Although the Amazon S3 bucket contains log files that were generated by Accounts A, B and C, accounts B and C do not initially have access to the log files that accounts B and C generated. You will give each business unit read-only access to the log files that it generated, as shown in the following illustration.



To grant read-only access to the log files generated by accounts B and C, you must do the following in the account Account A. Remember that Account A has full control of the Amazon S3 bucket.

- Create an IAM role for account B and another IAM role for account C. How: [Creating a Role \(p. 145\)](#)
- For the IAM role created for account B, create an access policy that grants read-only access to the log files generated by account B. For the IAM role created for account C, create an access policy that grants read-only access to the log files generated by account C. How: [Creating an Access Policy to Grant Access to Accounts You Own \(p. 147\)](#)
- Have an IAM user in account B programmatically assume the role created for account B. Have an IAM user in account C programmatically assume the role created for account C. Each IAM user must be given permission to assume the role by the respective account owner. How: [Creating permissions policies for IAM users \(p. 150\)](#).
- Finally, the account owner who grants the permission must be an administrator, and must know the ARN of the role in account A that is being assumed. How: [Calling AssumeRole \(p. 151\)](#).

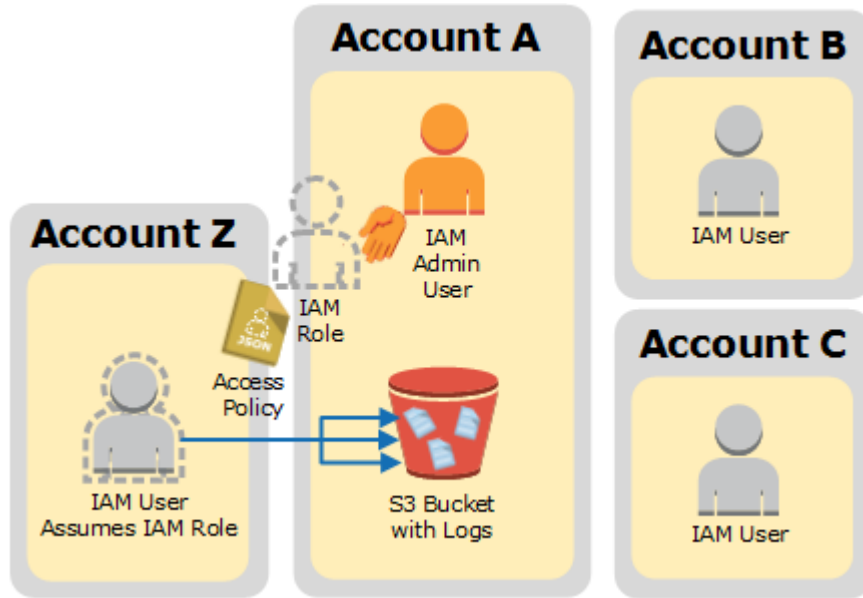
The IAM users in accounts B and C can then programmatically retrieve their own log files, but not the log files of any other account.

Scenario 2: Granting Access to All Logs

In this scenario, we'll assume that your enterprise is structured as it was in the previous scenario, that is, it is made up of two business units and it maintains three AWS accounts. The first account, Account A, is the top-level account. For example, it might be managed by your enterprise's IT department and therefore be responsible for placing all other log files into a single bucket. The other two accounts, B and C, correspond to each of your enterprise's business units.

Like the previous scenario, this scenario assumes that you have already placed the log files from all three accounts into a single Amazon S3 bucket, and that account A has full control over that bucket.

Finally, we'll also assume that your enterprise wants to share all the log files from all accounts (A, B, and C) with a third party. We'll say that the third party has an AWS account called Account Z, as shown in the following illustration.



To share all of the log files from your enterprise with Account Z, you must do the following in the Account A, the account that has full control over the Amazon S3 bucket.

- Create an IAM role for Account Z. How: [Creating a Role \(p. 145\)](#)
- For the IAM role created for Account Z, create an access policy that grants read-only access to the log files generated by accounts A, B, and C. How: [Creating an Access Policy to Grant Access to a Third Party \(p. 148\)](#)
- Have an IAM user in Account Z programmatically assume the role and then retrieve the appropriate log files. The IAM user must be given permission to assume the role by the owner of Account Z. How: [Creating permissions policies for IAM users \(p. 150\)](#). Further, the account owner who grants the permission must be an administrator and know the ARN of the role in Account A that is being assumed. How: [Calling AssumeRole \(p. 151\)](#).

Topics

- [Creating a Role \(p. 145\)](#)
- [Creating an Access Policy to Grant Access to Accounts You Own \(p. 147\)](#)
- [Creating an Access Policy to Grant Access to a Third Party \(p. 148\)](#)
- [Assuming a Role \(p. 150\)](#)
- [Stop Sharing CloudTrail Log Files Between AWS Accounts \(p. 152\)](#)

Creating a Role

When you aggregate log files from multiple accounts into a single Amazon S3 bucket, only the account that has full control of the bucket, Account A in our example, has full read access to all of the log files in the bucket. Accounts B, C, and Z in our example do not have any rights until granted. Therefore, to share your AWS CloudTrail log files from one account to another (that is, to complete either Scenario 1 or Scenario 2 described previously in this section), you must *enable cross-account access*. You can do this by creating IAM roles and their associated access policies.

Roles

Create an IAM *role* for each account to which you want to give access. In our example, you will have three roles, one each for accounts B, C, and Z. Each IAM role defines an access or permissions policy that enables the accounts to access the resources (log files) owned by account A. The permissions are attached to each role and are associated with each account (B, C, or Z) only when the role is assumed. For details about permissions management for IAM roles, see [Roles \(Delegation and Federation\)](#) . For more information about how to assume a role, see [Assuming a Role](#) (p. 150).

Policies

There are two policies for each IAM role you create. The *trust policy* specifies a *trusted entity* or *principal*. In our example, accounts B, C, and Z are trusted entities, and an IAM user with the proper permissions in those accounts can assume the role.

The *trust policy* is automatically created when you use the console to create the role. If you use the SDK to create the role, you must supply the trust policy as a parameter to the `CreateRole` API. If you use the CLI to create the role, you must specify the trust policy in the `create-role` CLI command.

The *role access (or permissions) policy* that you must create as the owner of Account A defines what actions and resources the principal or trusted entity is allowed access to (in this case, the CloudTrail log files). For Scenario 1 that grants log file access to the account that generated the log files, as discussed in [Creating an Access Policy to Grant Access to Accounts You Own](#) (p. 147). For Scenario 2 that grants read access to all log files to a third party, as discussed in [Creating an Access Policy to Grant Access to a Third Party](#) (p. 148).

For further details about creating and working with IAM policies, see [Permissions and Policies](#) in *IAM User Guide*.

Creating a Role

To Create a Role by Using the Console

1. Sign into the AWS Management Console as an administrator of Account A.
2. Navigate to the IAM console.
3. In the navigation pane, click **Roles**.
4. Click **Create New Role**.
5. Enter a name for the new role, and then click **Next Step**.
6. Click **Role for Cross-Account Access**.
7. For Scenario 1, do the following to provide access between accounts you own:
 - a. Select **Provide access between AWS accounts you own**.
 - b. Enter the twelve-digit account ID of the account (B, C, or Z) to be granted access.
 - c. Check the **Require MFA** box if you want the user to provide multi-factor authentication before assuming the role.

For Scenario 2, do the following to provide access to a third-party account. In our example, you would perform these steps for Account Z, the third-party log analyzer:

- a. Select **Allows IAM users from a 3rd party AWS account to access this account**.
 - b. Enter the twelve-digit account ID of the account (Account Z) to be granted access.
 - c. Enter an external ID that provides additional control over who can assume the role. For more information, see [About the External ID](#) in the [AWS Security Token Service User Guide](#) .
8. Click **Next Step** to attach a policy that sets the permissions for this role.
 9. Under **Attach Policy**, find and select the **AmazonS3ReadOnlyAccess** policy.

Note

By default, the **AmazonS3ReadOnlyAccess** policy grants retrieval and list rights to all Amazon S3 buckets within your account.

- To grant an account access to only that account's log files (Scenario 1), see [Creating an Access Policy to Grant Access to Accounts You Own](#) (p. 147).
- To grant an account access to all of the log files in the Amazon S3 bucket (Scenario 2), see [Creating an Access Policy to Grant Access to a Third Party](#) (p. 148).

10. Click **Next Step**

11. Review the role information.

Note

You can edit the role name at this point if you wish, but if you do so, you will be taken back to the **Step 2: Select Role Type** page where you must reenter the information for the role.

12. Click **Create Role**. When the role creation process completes, the role you created appears in the role list.

Creating an Access Policy to Grant Access to Accounts You Own

In Scenario 1, as an administrative user in Account A, you have full control over the Amazon S3 bucket to which CloudTrail writes log files for accounts B and C. You want to share each business unit's log files back to business unit that created them. But, you don't want a unit to be able to read any other unit's log files.

For example, to share Account B's log files with Account B but not with Account C, you must create a new IAM role in Account A that specifies that Account B is a trusted account. This role trust policy specifies that Account B is trusted to assume the role created by Account A, and should look like the following example. The trust policy is automatically created if you create the role by using the console. If you use the SDK to create the role, you must supply the trust policy as a parameter to the `CreateRole` API. If you use the CLI to create the role, you must specify the trust policy in the `create-role` CLI command.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-B-id:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

You must also create an access policy to specify that Account B can read from only the location to which B wrote its log files. The access policy will look something like the following. Note that the **Resource** ARN includes the twelve-digit account ID for Account B, and the prefix you specified, if any, when you turned on CloudTrail for Account B during the aggregation process. For more information about specifying a prefix, see [Turning on CloudTrail in Additional Accounts](#) (p. 140).

Important

You must ensure that the prefix in the access policy is exactly the same as the prefix that you specified when you turned on CloudTrail for Account B. If it is not, then you must edit the IAM role access policy in Account A to incorporate the actual prefix for Account B. If the prefix in the role access policy is not exactly the same as the prefix you specified when you turned on CloudTrail in Account B, then Account B will not be able to access its log files.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3:::bucket-name/prefix/AWSLogs/account-B-id/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3:::bucket-name"
    }
  ]
}
```

The role you create for Account C will be nearly identical to the one you created for Account B. The access policy for each role must include the appropriate account ID and prefix so that each account can read from only the location to which CloudTrail wrote that account's log files.

After you have created roles for each account and specified the appropriate trust and access policies, and after an IAM user in each account has been granted access by the administrator of that account, an IAM user in accounts B or C can programmatically assume the role.

After you have created roles for each account and specified the appropriate trust and access policies, an IAM user in one of the newly trusted accounts (B or C) must programmatically assume the role in order to read log files from the Amazon S3 bucket.

For more information, see [Assuming a Role \(p. 150\)](#).

Creating an Access Policy to Grant Access to a Third Party

Account A must create a separate IAM role for Account Z, the third-party analyzer in Scenario 2. The trust relationship, automatically created by AWS when you create the role, specifies that Account Z will be trusted to assume the role. The access policy for the role specifies what actions Account Z can take. For more information about creating roles and role policies, see [Creating a Role \(p. 145\)](#).

For example, the trust relationship created by AWS will specify that Account Z is trusted to assume the role created by Account A, and will look something like the following.

AWS CloudTrail User Guide
Creating an Access Policy to Grant Access to a Third Party

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-Z-id:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

If you specified an external ID when you created the role for Account Z, your access policy contains an added **Condition** element that tests the unique ID assigned by Account Z. The test is performed when the role is assumed. The **Condition** element is shown in the following example access policy. For more information, see [About the External ID](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-Z-id:root"
      },
      "Action": "sts:AssumeRole"
      "Condition": {
        "StringEquals": {
          "sts:ExternalId": "external ID-issued-by-account-Z"
        }
      }
    }
  ]
}
```

You must also create an access policy for the Account A role to specify that Account Z can read all logs from the Amazon S3 bucket. The access policy should look something like the following. The wild card (*) at the end of the **Resource** value indicates that Account Z can access any log file in the Amazon S3 bucket to which it has been granted access.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3::bucket-name/*"
    }
  ]
}
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3:::bucket-name"
    }
  ]
}
```

After you have created a role for Account Z and specified the appropriate trust relationship and access policy, an IAM user in Account Z must programmatically assume the role to be able to read log files from the Amazon S3 bucket. For more information, see [Assuming a Role \(p. 150\)](#).

Assuming a Role

You must designate a separate IAM user to assume each role you've created in each account, and ensure that each IAM user has appropriate permissions.

IAM Users and Roles

After you have created the necessary roles and policies in Account A for scenarios 1 and 2, you must designate an IAM user in each of the accounts B, C, and Z. Each IAM user will programmatically assume the appropriate role to access the log files. That is, the user in account B will assume the role created for account B, the user in account C will assume the role created for account C, and the user in account Z will assume the role created for account Z. When a user assumes a role, AWS returns temporary security credentials that can be used to make requests to list, retrieve, copy, or delete the log files depending on the permissions granted by the access policy associated with the role.

For more information about working with IAM users, see [Working with IAM Users and Groups](#).

The primary difference between scenarios 1 and 2 is in the access policy that you create for each IAM role in each scenario.

- In scenario 1, the access policies for accounts B and C limit each account to reading only its own log files. For more information, see [Creating an Access Policy to Grant Access to Accounts You Own \(p. 147\)](#).
- In scenario 2, the access policy for Account Z allows it to read all the log files that are aggregated in the Amazon S3 bucket. For more information, see [Creating an Access Policy to Grant Access to a Third Party \(p. 148\)](#).

Creating permissions policies for IAM users

To perform the actions permitted by the roles, the IAM user must have permission to call the AWS STS [AssumeRole](#) API. You must edit the *user-based policy* for each IAM user to grant them the appropriate permissions. That is, you set a **Resource** element in the policy that is attached to the IAM user. The following example shows a policy for an IAM user in Account B that allows the user to assume a role named "Test" created earlier by Account A.

To attach the required policy to the IAM role

1. Sign in to the AWS Management Console and open the IAM console.
2. Select the user whose permissions you want to modify.
3. Select the **Permissions** tab.

4. Select **Custom Policy**.
5. Select **Use the policy editor to customize your own set of permissions**.
6. Give the policy a name.
7. Copy the following policy into the space provided for the policy document.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole"],
      "Resource": "arn:aws:iam::account-A-id:role/Test"
    }
  ]
}
```

Important

Only IAM users can assume a role. If you attempt to use AWS root account credentials to assume a role, access will be denied.

Calling AssumeRole

A user in accounts B, C, or Z can assume a role by creating an application that calls the AWS STS [AssumeRole](#) API and passes the role session name, the Amazon Resource Number (ARN) of the role to assume, and an optional external ID. The role session name is defined by Account A when it creates the role to assume. The external ID, if any, is defined by Account Z and passed to Account A for inclusion during role creation. For more information, see [About the External ID](#). You can retrieve the ARN from the Account A by opening the IAM console.

To Find the ARN Value in Account A

- Select **Roles**
- Select the role you want to examine.
- Look for the Role ARN in the **Summary** tab

The AssumeRole API returns temporary credentials that a user in accounts B, C, or Z can use to access resources in Account A. In this example, the resources you want to access are the Amazon S3 bucket and the log files that the bucket contains. The temporary credentials have the permissions that you defined in the role access policy.

The following Python example (using the [AWS SDK for Python \(Boto\)](#)) shows how to call AssumeRole and how to use the temporary security credentials returned to list all Amazon S3 buckets controlled by Account A.

```
import boto
from boto.sts import STSConnection
from boto.s3.connection import S3Connection

# The calls to AWS STS AssumeRole must be signed using the access key ID and
# secret
# access key of an IAM user or using existing temporary credentials. (You cannot
# call
# AssumeRole using the access key for an account.) The credentials can be in
```

```
# environment variables or in a configuration file and will be discovered
automatically
# by the STSConnection() function. For more information, see the Python SDK
# documentation: http://boto.readthedocs.org/en/latest/boto_config_tut.html

sts_connection = STSConnection()
assumedRoleObject = sts_connection.assume_role(
    role_arn="arn:aws:iam::account-of-role-to-assume:role/name-of-role",
    role_session_name="AssumeRoleSession1"
)

# Use the temporary credentials returned by AssumeRole to call Amazon S3
# and list the bucket in the account that owns the role (the trusting account)
s3_connection = S3Connection(
    aws_access_key_id=assumedRoleObject.credentials.access_key,
    aws_secret_access_key=assumedRoleObject.credentials.secret_key,
    security_token=assumedRoleObject.credentials.session_token
)
bucket = s3_connection.get_bucket(bucketname)
print bucket.name
```

Stop Sharing CloudTrail Log Files Between AWS Accounts

To stop sharing log files to another AWS account, simply delete the role that you created for that account in [Creating a Role \(p. 145\)](#).

1. Sign in to the AWS Management Console as an IAM user with administrative-level permissions for Account A.
2. Navigate to the IAM console.
3. In the navigation pane, click **Roles**.
4. Select the role you want to delete.
5. Right-click and select **Delete Role** from the context menu.

Encrypting CloudTrail Log Files with AWS KMS–Managed Keys (SSE-KMS)

By default, the log files delivered by CloudTrail to your bucket are encrypted by Amazon [server-side encryption with Amazon S3-managed encryption keys \(SSE-S3\)](#). To provide a security layer that is directly manageable, you can instead use [server-side encryption with AWS KMS–managed keys \(SSE-KMS\)](#) for your CloudTrail log files.

To use SSE-KMS with CloudTrail, you create and manage a KMS key, also known as a [customer master key \(CMK\)](#). You attach a policy to the key that determines which users can use the key for encrypting and decrypting CloudTrail log files. The decryption is seamless through S3. When authorized users of the key read CloudTrail log files, S3 manages the decryption, and the authorized users are able to read log files in unencrypted form.

This approach has the following advantages:

- You can create and manage the CMK encryption keys yourself.

- You can use a single CMK to encrypt and decrypt log files for multiple accounts across all regions.
- You have control over who can use your key for encrypting and decrypting CloudTrail log files. You can assign permissions for the key to the users in your organization according to your requirements.
- You have enhanced security. With this feature, in order to read log files, you now need to meet two conditions: 1) you must have S3 read permission on the bucket, and 2) you must be granted decrypt permission by the CMK policy.
- Because S3 automatically decrypts the log files for requests from users authorized to use the CMK, SSE-KMS encryption for CloudTrail log files is backward compatible with existing applications that read CloudTrail log data.

Note

The key that you choose must be in the same region as the S3 bucket that receives your log files. To verify the region that an S3 bucket belongs to, inspect its properties in the S3 console.

Enabling log file encryption

Note

If you create a CMK in the CloudTrail console, CloudTrail adds the required CMK policy sections for you. Follow these procedures if you created a key in the IAM console or AWS CLI and you need to manually add the required policy sections.

To enable SSE-KMS encryption for CloudTrail log files, perform the following high-level steps:

1. Create a CMK.
 - For information on creating a CMK with the AWS Management Console, see [Creating Keys](#) in the *AWS Key Management Service Developer Guide*.
 - For information on creating a CMK with the AWS CLI, see [create-key](#).

Note

The CMK that you choose must be in the same region as the S3 bucket that receives your log files. To verify the region that an S3 bucket belongs to, inspect the bucket's properties in the S3 console.

2. Add policy sections to the key that enable CloudTrail to encrypt and users to decrypt log files.
 - For information about what to include in the policy, see [AWS KMS Key Policy for CloudTrail \(p. 154\)](#).

Caution

Be sure to include decrypt permissions in the policy for all users who need to read log files. If you do not perform this step before adding the key to your trail configuration, users without decrypt permissions will not be able to read encrypted files.

- For information on editing a policy with the IAM console, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.
 - For information on attaching a policy to a CMK with the AWS CLI, see [put-key-policy](#).
3. Update your trail to use the CMK whose policy you modified for CloudTrail.
 - To update your trail configuration by using the CloudTrail console, see [Updating a Trail to Use Your CMK \(p. 161\)](#).
 - To update your trail configuration by using the AWS CLI, see [Enabling and disabling CloudTrail log file encryption with the AWS CLI \(p. 162\)](#).

The next section describes the policy sections that your CMK policy requires for use with CloudTrail.

Topics

- [Granting Permissions to Create a CMK \(p. 154\)](#)
- [AWS KMS Key Policy for CloudTrail \(p. 154\)](#)
- [Updating a Trail to Use Your CMK \(p. 161\)](#)
- [Enabling and disabling CloudTrail log file encryption with the AWS CLI \(p. 162\)](#)

Granting Permissions to Create a CMK

You can grant users permission to create a customer master key (CMK) with the `AWSKeyManagementServicePowerUser` policy.

To grant permission to create a CMK

1. Open the IAM console at <https://console.aws.amazon.com/iam>.
2. Choose the group or user that you want to give permission.
3. Choose **Permissions**, and then choose **Attach Policy**.
4. Search for **AWSKeyManagementServicePowerUser**, choose the policy, and then choose **Attach policy**.

The user now has permission to create a CMK. If you want to create custom policies for your users, see [Creating Customer Managed Policies](#) in the *IAM User Guide*.

AWS KMS Key Policy for CloudTrail

You can create a customer master key (CMK) in three ways:

- The CloudTrail console
- The IAM console
- The AWS CLI

If you create a CMK in the CloudTrail console, CloudTrail adds the required CMK policy sections for you. You do not need to complete the following steps.

If you create a CMK in the IAM console or the AWS CLI, you need to add policy sections to the key so that you can use it with CloudTrail. The policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form.

See the following resources:

- To create a CMK with the AWS CLI, see [create-key](#).
- To edit a CMK policy for CloudTrail, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.
- For technical details on how CloudTrail uses AWS KMS, see [How AWS CloudTrail Uses AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Required CMK policy sections for use with CloudTrail

If you created a CMK in the CloudTrail console, CloudTrail adds the required CMK policy for you. You do not need to manually add the policy statements. See [Default Key Policy Created in CloudTrail Console \(p. 159\)](#).

If you created a CMK with the IAM console or the AWS CLI, then you must, at minimum, add three statements to your CMK policy for it to work with CloudTrail.

1. Enable CloudTrail log encrypt permissions. See [Granting encrypt permissions \(p. 155\)](#).
2. Enable CloudTrail log decrypt permissions. See [Granting decrypt permissions \(p. 156\)](#).
3. Enable CloudTrail to describe CMK properties. See [Enable CloudTrail to describe CMK properties \(p. 159\)](#).

Caution

When you add the new sections to your CMK policy, do not change any existing sections in the policy.

Warning

If encryption is enabled on a trail and the CMK is disabled or the CMK policy is not correctly configured for CloudTrail, CloudTrail will not deliver logs until the CMK issue is corrected.

Granting encrypt permissions

Allow CloudTrail to encrypt logs on behalf of specific accounts

CloudTrail needs explicit permission to use the CMK to encrypt logs on behalf of specific accounts. To specify an account, add the following required statement to your CMK policy, modifying *aws-account-id* as necessary. You can add additional account IDs to the `EncryptionContext` section to enable those accounts to use CloudTrail to use your CMK to encrypt log files.

```
{
  "Sid": "Allow CloudTrail to encrypt logs",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "kms:GenerateDataKey*",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:EncryptionContext:aws:cloudtrail:arn": [
        "arn:aws:cloudtrail:*:aws-account-id:trail/*"
      ]
    }
  }
}
```

Example

The following example policy statement illustrates how another account can use your CMK to encrypt CloudTrail logs.

Scenario

- Your CMK is in account 111111111111.

- Both you and account 222222222222 will encrypt logs.

In the policy, you add one or more accounts that will encrypt with your key to the CloudTrail **EncryptionContext**. This restricts CloudTrail to using your key to encrypt logs only for those accounts that you specify. Giving the root of account 222222222222 permission to encrypt logs delegates the administrator of that account to allocate encrypt permissions as required to other users in account 222222222222 by changing their IAM user policies.

CMK policy statement:

```
{
  "Sid": "Enable CloudTrail Encrypt Permissions",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "kms:GenerateDataKey*",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:EncryptionContext:aws:cloudtrail:arn": [
        "arn:aws:cloudtrail:*:111111111111:trail/*",
        "arn:aws:cloudtrail:*:222222222222:trail/*"
      ]
    }
  }
}
```

For steps on editing a CMK policy for use with CloudTrail, see [Editing a Key Policy](#) in the AWS Key Management Service Developer Guide.

Granting decrypt permissions

Before you add your CMK to your CloudTrail configuration, it is important to give decrypt permissions to all users who require them. Users who have encrypt permissions but no decrypt permissions will not be able to read encrypted logs.

Enable CloudTrail log decrypt permissions

Users of your key must be given explicit permissions to read the log files that CloudTrail has encrypted. To enable users to read encrypted logs, add the following required statement to your CMK policy, modifying the `Principal` section to add a line for every principal (role or user) that you want to be able decrypt by using your CMK.

```
{
  "Sid": "Enable CloudTrail log decrypt permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::aws-account-id:user/username"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"
    }
  }
}
```



```
}  
}
```

Allow users in your account to decrypt with your CMK

Example

This policy statement illustrates how to allow an IAM user or role in your account to use your key to read the encrypted logs in your account's S3 bucket.

Scenario

- Your CMK, S3 bucket, and IAM user Bob are in account 111111111111.
- You give IAM user Bob permission to decrypt CloudTrail logs in the S3 bucket.

In the key policy, you enable CloudTrail log decrypt permissions for IAM user Bob.

CMK policy statement:

```
{  
  "Sid": "Enable CloudTrail log decrypt permissions",  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "arn:aws:iam::111111111111:user/Bob"  
  },  
  "Action": "kms:Decrypt",  
  "Resource": "*",  
  "Condition": {  
    "Null": {  
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"  
    }  
  }  
}
```

Topics

Allow users in other accounts to decrypt with your CMK

You can allow users in other accounts to use your CMK to decrypt logs. The changes required to your key policy depend on whether the S3 bucket is in your account or in another account.

Allow users of a bucket in a different account to decrypt logs

Example

This policy statement illustrates how to allow an IAM user or role in another account to use your key to read encrypted logs from an S3 bucket in the other account.

Scenario

- Your CMK is in account 111111111111.
- The IAM user Alice and S3 bucket are in account 222222222222.

In this case, you give CloudTrail permission to decrypt logs under account 222222222222, and you give Alice's IAM user policy permission to use your key `keyA`, which is in account 111111111111.

CMK policy statement:

```
{
  "Sid": "Enable encrypted CloudTrail log read access",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::222222222222:root"
    ]
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"
    }
  }
}
```

Alice's IAM user policy statement:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "arn:aws:kms:us-east-1:111111111111:key/keyA"
    }
  ]
}
```

Allow users in a different account to decrypt logs from your bucket

Example

This policy illustrates how another account can use your key to read encrypted logs from your S3 bucket.

Scenario

- Your CMK and S3 bucket are in account 111111111111.
- The user who will read logs from your bucket is in account 222222222222.

To enable this scenario, you enable decrypt permissions for the IAM role **CloudTrailReadRole** in your account, and then give the other account permission to assume that role.

CMK policy statement:

```
{
  "Sid": "Enable encrypted CloudTrail log read access",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::111111111111:role/CloudTrailReadRole"
    ]
  }
}
```

```
    },  
    "Action": "kms:Decrypt",  
    "Resource": "*",  
    "Condition": {  
      "Null": {  
        "kms:EncryptionContext:aws:cloudtrail:arn": "false"  
      }  
    }  
  }  
}
```

CloudTrailReadRole trust entity policy statement:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::222222222222:root"  
      },  
      "Action": "sts:AssumeRole"  
    }  
  ]  
}
```

For steps on editing a CMK policy for use with CloudTrail, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.

Enable CloudTrail to describe CMK properties

CloudTrail requires the ability to describe the properties of the CMK. To enable this functionality, add the following required statement as is to your CMK policy. This statement does not grant CloudTrail any permissions beyond the other permissions that you specify.

```
{  
  "Sid": "Allow CloudTrail access",  
  "Effect": "Allow",  
  "Principal": {  
    "Service": "cloudtrail.amazonaws.com"  
  },  
  "Action": "kms:DescribeKey",  
  "Resource": "*"  
}
```

For steps on editing a CMK policy for use with CloudTrail, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.

Default Key Policy Created in CloudTrail Console

If you create a customer master key (CMK) in the CloudTrail console, the following policy is automatically created for you. The policy allows these permissions:

- Allows AWS account (root) permissions for the CMK
- Allows CloudTrail to encrypt log files under the CMK and describe the CMK

- Allows all users in the specified accounts to decrypt log files
- Allows all users in the specified account to create a KMS alias for the CMK

```
{
  "Version": "2012-10-17",
  "Id": "Key policy created by CloudTrail",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": { "AWS": [
        "arn:aws:iam::aws-account-id:root",
        "arn:aws:iam::aws-account-id:user/username"
      ]},
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow CloudTrail to encrypt logs",
      "Effect": "Allow",
      "Principal": { "Service": ["cloudtrail.amazonaws.com"] },
      "Action": "kms:GenerateDataKey*",
      "Resource": "*",
      "Condition": { "StringLike": { "kms:EncryptionContext:aws:cloudtrail:arn": "arn:aws:cloudtrail:*:aws-account-id:trail/*" } }
    },
    {
      "Sid": "Allow CloudTrail to describe key",
      "Effect": "Allow",
      "Principal": { "Service": ["cloudtrail.amazonaws.com"] },
      "Action": "kms:DescribeKey",
      "Resource": "*"
    },
    {
      "Sid": "Allow principals in the account to decrypt log files",
      "Effect": "Allow",
      "Principal": { "AWS": "*" },
      "Action": [
        "kms:Decrypt",
        "kms:ReEncryptFrom"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": { "kms:CallerAccount": "aws-account-id" },
        "StringLike": { "kms:EncryptionContext:aws:cloudtrail:arn": "arn:aws:cloudtrail:*:aws-account-id:trail/*" }
      }
    },
    {
      "Sid": "Allow alias creation during setup",
      "Effect": "Allow",
      "Principal": { "AWS": "*" },
      "Action": "kms:CreateAlias",
      "Resource": "*",
      "Condition": { "StringEquals": {
        "kms:ViaService": "ec2.region.amazonaws.com",
        "kms:CallerAccount": "aws-account-id"
      } }
    }
  ]
}
```

```
    }  
  },  
  {  
    "Sid": "Enable cross account log decryption",  
    "Effect": "Allow",  
    "Principal": {"AWS": "*"},  
    "Action": [  
      "kms:Decrypt",  
      "kms:ReEncryptFrom"  
    ],  
    "Resource": "*",  
    "Condition": {  
      "StringEquals": {"kms:CallerAccount": "aws-account-id"},  
      "StringLike": {"kms:EncryptionContext:aws:cloudtrail:arn":  
"arn:aws:cloudtrail:*:aws-account-id:trail/*"}  
    }  
  }  
]  
}
```

Note

The policy's final statement allows cross accounts to decrypt log files with the CMK.

Updating a Trail to Use Your CMK

To update a trail to use the customer master key (CMK) that you modified for CloudTrail, complete the following steps in the CloudTrail console.

To update a trail using the AWS CLI, see [Enabling and disabling CloudTrail log file encryption with the AWS CLI \(p. 162\)](#).

To update a trail to use your CMK

1. Sign in to the AWS Management Console and open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
2. Navigate to the **Configuration** page for your trail.
3. Click the pencil icon to the right of **S3**.
4. Choose **Advanced**.
5. For **Encrypt log files**, choose **Yes** to have CloudTrail encrypt your log files with the CMK.
6. For **Create a new KMS key**, choose **No**.
7. For **KMS key**, choose the CMK alias whose policy you modified for use with CloudTrail.

Note

Choose a CMK that is in the same region as the S3 bucket that receives your log files. To verify the region that an S3 bucket belongs to, inspect its properties in the S3 console.

Encrypt log files Yes No

Create a new KMS key Yes No

KMS key* ⓘ

KMS key and S3 bucket must be in the same region.

You can type the alias name, ARN, or the globally unique key ID. If the CMK belongs to another account, verify that the key policy has permissions that enable you to use it. The value can be one of the following formats:

- **Alias Name:** `alias/MyAliasName`
- **Alias ARN:** `arn:aws:kms:us-east-1:123456789012:alias/MyAliasName`
- **Key ARN:**
`arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012`
- **Globally unique key ID:** `12345678-1234-1234-1234-123456789012`

8. Choose **Save**.

Note

If the CMK that you chose is disabled or is pending deletion, you won't be able to save the trail with that CMK. You can enable the CMK or choose another one. For more information, see [How Key State Affects Use of a Customer Master Key](#).

Enabling and disabling CloudTrail log file encryption with the AWS CLI

This topic describes how to enable and disable SSE-KMS log file encryption for CloudTrail by using the AWS CLI. For background information, see [Encrypting CloudTrail Log Files with AWS KMS–Managed Keys \(SSE-KMS\) \(p. 152\)](#).

Enabling CloudTrail log file encryption by using the AWS CLI

1. Create a key with the AWS CLI. The key that you create must be in the same region as the S3 bucket that receives your CloudTrail log files. For this step, you use the KMS [create-key](#) command.
2. Get the existing key policy so that you can modify it for use with CloudTrail. You can retrieve the key policy with the KMS [get-key-policy](#) command.
3. Add the necessary sections to the key policy so that CloudTrail can encrypt and users can decrypt your log files. Make sure that all users who will read the log files are granted decrypt permissions. Do not modify any existing sections of the policy. For information on the policy sections to include, see [AWS KMS Key Policy for CloudTrail \(p. 154\)](#).
4. Attach the modified .json policy file to the key by using the KMS [put-key-policy](#) command.
5. Run the CloudTrail `create-trail` or `update-trail` command with the `--kms-key-id` parameter. This command will enable log encryption.

```
aws cloudtrail update-trail --name Default --kms-key-id alias/MyKmsKey
```

The `--kms-key-id` parameter specifies the key whose policy you modified for CloudTrail. It can be any one of the following four formats:

- **Alias Name.** Example: `alias/MyAliasName`
- **Alias ARN.** Example: `arn:aws:kms:us-east-1:123456789012:alias/MyAliasName`
- **Key ARN.** Example:
`arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012`
- **Globally unique key ID.** Example: `12345678-1234-1234-1234-123456789012`

The response will look like the following:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "Default",
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Default",

  "LogFileValidationEnabled": false,
  "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012",
  "S3BucketName": "my-bucket-name"
}
```

The presence of the `KmsKeyId` element indicates that log file encryption has been enabled. The encrypted log files should appear in your bucket in about 10 minutes.

Disabling CloudTrail log file encryption by using the AWS CLI

To stop encrypting logs, call `update-trail` and pass an empty string to the `kms-key-id` parameter:

```
aws cloudtrail update-trail --name Default --kms-key-id ""
```

The response will look like the following:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "Default",
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Default",
  "LogFileValidationEnabled": false,
  "S3BucketName": "my-bucket-name"
}
```

The absence of the `KmsKeyId` element indicates that log file encryption is no longer enabled.

Validating CloudTrail Log File Integrity

To determine whether a log file was modified, deleted, or unchanged after CloudTrail delivered it, you can use CloudTrail log file integrity validation. This feature is built using industry standard algorithms: SHA-256 for hashing and SHA-256 with RSA for digital signing. This makes it computationally infeasible to modify, delete or forge CloudTrail log files without detection. You can use the AWS CLI to validate the files in the location where CloudTrail delivered them.

Why Use It?

Validated log files are invaluable in security and forensic investigations. For example, a validated log file enables you to assert positively that the log file itself has not changed, or that particular user credentials performed specific API activity. The CloudTrail log file integrity validation process also lets you know if a log file has been deleted or changed, or assert positively that no log files were delivered to your account during a given period of time.

How It Works

When you enable log file integrity validation, CloudTrail creates a hash for every log file that it delivers. Every hour, CloudTrail also creates and delivers a file that references the log files for the last hour and contains a hash of each. This file is called a digest file. CloudTrail signs each digest file using the private key of a public and private key pair. After delivery, you can use the public key to validate the digest file. CloudTrail uses different key pairs for each AWS region.

The digest files are delivered to the same Amazon S3 bucket associated with your trail as your CloudTrail log files. If your log files are delivered from all regions or from multiple accounts into a single Amazon S3 bucket, CloudTrail will deliver the digest files from those regions and accounts into the same bucket.

The digest files are put into a folder separate from the log files. This separation of digest files and log files enables you to enforce granular security policies and permits existing log processing solutions to continue to operate without modification. Each digest file also contains the digital signature of the previous digest file if one exists. The signature for the current digest file is in the metadata properties of the digest file Amazon S3 object. For more information about digest file contents, see [CloudTrail Digest File Structure](#) (p. 170).

Storing log and digest files

You can store the CloudTrail log files and digest files in Amazon S3 or Amazon Glacier securely, durably and inexpensively for an indefinite period of time. To enhance the security of the digest files stored in Amazon S3, you can use [Amazon S3 MFA Delete](#).

Enabling Validation and Validating Files

To enable log file integrity validation, you can use the AWS Management Console, the AWS CLI, or CloudTrail API. For more information, see [Enabling Log File Integrity Validation for CloudTrail](#) (p. 164).

To validate the integrity of CloudTrail log files, you can use the AWS CLI or create your own solution. The AWS CLI will validate files in the location where CloudTrail delivered them. If you want to validate logs that you have moved to a different location, either in Amazon S3 or elsewhere, you can create your own validation tools.

For information on validating logs by using the AWS CLI, see [Validating CloudTrail Log File Integrity with the AWS CLI](#) (p. 165). For information on developing custom implementations of CloudTrail log file validation, see [Custom Implementations of CloudTrail Log File Integrity Validation](#) (p. 175).

Enabling Log File Integrity Validation for CloudTrail

You can enable log file integrity validation by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or CloudTrail API. CloudTrail starts delivering digest files in about an hour.

AWS Management Console

To enable log file integrity validation with the CloudTrail console, choose **Yes** for the **Enable log file validation** option when you create or update a trail. By default, this feature is enabled for new trails. For more information, see [Creating and Updating a Trail with the CloudTrail Console](#) (p. 28).

AWS CLI

To enable log file integrity validation with the AWS CLI, use the `--enable-log-file-validation` option with the `create-trail` or `update-trail` commands. To disable log file integrity validation, use the `--no-enable-log-file-validation` option.

Example

The following `update-trail` command enables log file validation and starts delivering digest files to the Amazon S3 bucket for the specified trail.

```
aws cloudtrail update-trail --name your-trail-name --enable-log-file-validation
```

CloudTrail API

To enable log file integrity validation with the CloudTrail API, set the `EnableLogFileValidation` request parameter to `true` when calling `CreateTrail` or `UpdateTrail`.

For more information, see [CreateTrail](#) and [UpdateTrail](#) in the [AWS CloudTrail API Reference](#).

Validating CloudTrail Log File Integrity with the AWS CLI

To validate logs with the AWS Command Line Interface, use the CloudTrail `validate-logs` command. The command uses the digest files delivered to your Amazon S3 bucket to perform the validation. For information about digest files, see [CloudTrail Digest File Structure \(p. 170\)](#).

The AWS CLI allows you to detect the following types of changes:

- Modification or deletion of CloudTrail log files
- Modification or deletion of CloudTrail digest files
- Modification or deletion of both of the above

Note

The AWS CLI validates only log files that are referenced by digest files. For more information, see [Checking Whether a Particular File was Delivered by CloudTrail \(p. 170\)](#).

Prerequisites

To validate log file integrity with the AWS CLI, the following conditions must be met:

- You must have online connectivity to AWS.
- You must have read access to the Amazon S3 bucket that contains the digest and log files.
- The digest and log files must not have been moved from the original Amazon S3 location where CloudTrail delivered them.

Note

Log files that have been downloaded to local disk cannot be validated with the AWS CLI. For guidance on creating your own tools for validation, see [Custom Implementations of CloudTrail Log File Integrity Validation \(p. 175\)](#).

validate-logs

Syntax

The following is the syntax for `validate-logs`. Optional parameters are shown in brackets.

AWS CloudTrail User Guide
Validating CloudTrail Log File Integrity with the AWS CLI

```
aws cloudtrail validate-logs --trail-arn <trailARN> --start-time <start-time>
[--end-time <end-time>] [--s3-bucket <bucket-name>] [--s3-prefix <prefix>]
[--verbose]
```

Options

The following are the command-line options for `validate-logs`. The `--trail-arn` and `--start-time` options are required.

`--start-time`

Specifies that log files delivered on or after the specified UTC timestamp value will be validated.
Example: `2015-01-08T05:21:42Z`.

`--end-time`

Optionally specifies that log files delivered on or before the specified UTC timestamp value will be validated. The default value is the current UTC time (`Date.now()`). Example:
`2015-01-08T12:31:41Z`.

Note

For the time range specified, the `validate-logs` command checks only the log files that are referenced in their corresponding digest files. No other log files in the Amazon S3 bucket are checked. For more information, see [Checking Whether a Particular File was Delivered by CloudTrail](#) (p. 170).

`--bucket-name`

Optionally specifies the Amazon S3 bucket where the digest files are stored. If a bucket name is not specified, the AWS CLI will retrieve it by calling `DescribeTrails()`.

`--prefix`

Optionally specifies the Amazon S3 prefix where the digest files are stored. If not specified, the AWS CLI will retrieve it by calling `DescribeTrails()`.

Note

You should use this option only if your current prefix is different from the prefix that was in use during the time range that you specify.

`--trailARN`

Specifies the Amazon Resource Name (ARN) of the trail to be validated. The format of a trail ARN follows.

```
arn:aws:cloudtrail:us-east-1:111111111111:trail/MyTrailName
```

Tip

To obtain the trail ARN for a trail, you can use the `describe-trails` command before running `validate-logs`.

Note

You may want to specify the bucket name and prefix in addition to the trail ARN if log files have been delivered to more than one bucket in the time range that you specified, and you want to restrict the validation to the log files in only one of the buckets.

`--verbose`

Optionally outputs validation information for every log or digest file in the specified time range. The output indicates whether the file remains unchanged or has been modified or deleted. In non-verbose mode (the default), information is returned only for those cases in which there was a validation failure.

Example

The following example validates log files from the specified start time to the present, using the Amazon S3 bucket configured for the current trail and specifying verbose output.

```
aws cloudtrail validate-logs --start-time 2015-08-27T00:00:00Z --end-time 2015-08-28T00:00:00Z --trail-arn arn:aws:cloudtrail:us-west-2:111111111111:trail/my-trail-name --verbose
```

How `validate-logs` Works

The `validate-logs` command starts by validating the most recent digest file in the specified time range. First, it verifies that the digest file has been downloaded from the location to which it claims to belong. In other words, if the CLI downloads digest file `df1` from the S3 location `p1`, `validate-logs` will verify that `p1 == df1.digestS3Bucket + '/' + df1.digestS3Object`.

If the signature of the digest file is valid, it checks the hash value of each of the logs referenced in the digest file. The command then goes back in time, validating the previous digest files and their referenced log files in succession. It continues until the specified value for `start-time` is reached, or until the digest chain ends. If a digest file is missing or not valid, the time range that cannot be validated is indicated in the output.

Validation Results

Validation results begin with a summary header in the following format:

```
Validating log files for trail trail_ARN between time_stamp and time_stamp
```

Each line of the main output contains the validation results for a single digest or log file in the following format:

```
<Digest file | Log file> <S3 path> <Validation Message>
```

The following table describes the possible validation messages for log and digest files.

File Type	Validation Message	Description
Digest file	valid	The digest file signature is valid. The log files it references can be checked. This message is included only in verbose mode.
Digest file	INVALID: has been moved from its original location	The S3 bucket or S3 object from which the digest file was retrieved do not match the S3 bucket or S3 object locations that are recorded in the digest file itself.
Digest file	INVALID: invalid format	The format of the digest file is invalid. The log files corresponding to the time range that the digest file represents cannot be validated.
Digest file	INVALID: not found	The digest file was not found. The log files corresponding to the time range that the digest file represents cannot be validated.

AWS CloudTrail User Guide
Validating CloudTrail Log File Integrity with the AWS CLI

File Type	Validation Message	Description
Digest file	INVALID: public key not found for fingerprint <i>finger- print</i>	The public key corresponding to the fingerprint recorded in the digest file was not found. The digest file cannot be validated.
Digest file	INVALID: signature verification failed	The digest file signature is not valid. Because the digest file is not valid, the log files it references cannot be validated, and no assertions can be made about the API activity in them.
Digest file	INVALID: Unable to load PKCS #1 key with fingerprint <i>finger- print</i>	Because the DER encoded public key in PKCS #1 format having the specified fingerprint could not be loaded, the digest file cannot be validated.
Log file	valid	The log file has been validated and has not been modified since the time of delivery. This message is included only in verbose mode.
Log file	INVALID: hash value doesn't match	The hash for the log file does not match. The log file has been modified after delivery by CloudTrail.
Log file	INVALID: invalid format	The format of the log file is invalid. The log file cannot be validated.
Log file	INVALID: not found	The log file was not found and cannot be validated.

The output includes summary information about the results returned.

Example Outputs

Verbose

The following example `validate-logs` command uses the `--verbose` flag and produces the sample output that follows. [...] indicates the sample output has been abbreviated.

```
aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-west-2:111111111111:trail/example-trail-name --start-time 2015-08-31T22:00:00Z --end-time 2015-09-01T19:17:29Z --verbose
```

```
Validating log files for trail arn:aws:cloudtrail:us-west-2:111111111111:trail/example-trail-name between 2015-08-31T22:00:00Z and 2015-09-01T19:17:29Z
```

```
Digest file      s3://example-bucket/AWSLogs/111111111111/CloudTrail-Digest/us-west-2/2015/09/01/111111111111_CloudTrail-Digest_us-west-2_example-trail-name_us-west-2_20150901T201728Z.json.gz valid
Log file        s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-2/2015/09/01/111111111111_CloudTrail_us-west-2_20150901T1925Z_WZZw1RymnjCRjxXc.json.gz valid
Log file        s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-2/2015/09/01/111111111111_CloudTrail_us-west-2_20150901T1915Z_POuvV87nu6pfAV2W.json.gz valid
Log file        s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
```

AWS CloudTrail User Guide
Validating CloudTrail Log File Integrity with the AWS
CLI

```
2/2015/09/01/111111111111_CloudTrail_us-west-2_20150901T1930Z_l2QgXhAK
Vm1QXiIA.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/09/01/111111111111_CloudTrail_us-west-2_20150901T1920Z_eQJteBBrfpB
CqOqw.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/09/01/111111111111_CloudTrail_us-west-
2_20150901T1950Z_9g5A6qlR2B5KaRdq.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/09/01/111111111111_CloudTrail_us-west-
2_20150901T1920Z_i4DNCC12BuXd6Ru7.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/09/01/111111111111_CloudTrail_us-west-
2_20150901T1915Z_Sg5caf2RH6Jdx0EJ.json.gz valid
Digest file   s3://example-bucket/AWSLogs/111111111111/CloudTrail-Digest/us-
west-2/2015/09/01/111111111111_CloudTrail-Digest_us-west-2_example-trail-name_us-
west-2_20150901T191728Z.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/09/01/111111111111_CloudTrail_us-west-2_20150901T1910Z YYSFiuFQk4nrt
nEW.json.gz valid
[...]
Log file      s3://example-bucket/AWSLogs/144218288521/CloudTrail/us-west-
2/2015/09/01/144218288521_CloudTrail_us-west-2_20150901T1055Z_0Sfy6m9f6iBz
moPF.json.gz valid
Log file      s3://example-bucket/AWSLogs/144218288521/CloudTrail/us-west-
2/2015/09/01/144218288521_CloudTrail_us-west-2_20150901T1040Z_lLa3QzVL
pOed7igR.json.gz valid

Digest file   s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
west-2/2015/09/01/144218288521_CloudTrail-Digest_us-west-2_example-trail-name_us-
west-2_20150901T101728Z.json.gz INVALID: signature verification failed

Digest file   s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
west-2/2015/09/01/144218288521_CloudTrail-Digest_us-west-2_example-trail-name_us-
west-2_20150901T091728Z.json.gz valid
Log file      s3://example-bucket/AWSLogs/144218288521/CloudTrail/us-west-
2/2015/09/01/144218288521_CloudTrail_us-west-2_20150901T0830Z_eaFvO3dwHo4NC
qqc.json.gz valid
Digest file   s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
west-2/2015/09/01/144218288521_CloudTrail-Digest_us-west-2_example-trail-name_us-
west-2_20150901T081728Z.json.gz valid
Digest file   s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
west-2/2015/09/01/144218288521_CloudTrail-Digest_us-west-2_example-trail-name_us-
west-2_20150901T071728Z.json.gz valid
[...]
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/08/31/111111111111_CloudTrail_us-west-2_20150831T2245Z_mbJkEO5kNcDn
VhGh.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/08/31/111111111111_CloudTrail_us-west-
2_20150831T2225Z_IQ6kXy8sKU03RSPR.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/08/31/111111111111_CloudTrail_us-west-2_20150831T2230Z_eRPVRTx
HQ5498ROA.json.gz valid
Log file      s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-west-
2/2015/08/31/111111111111_CloudTrail_us-west-2_20150831T2255Z_1lWawYZGvT
WB5vYN.json.gz valid
Digest file   s3://example-bucket/AWSLogs/111111111111/CloudTrail-Digest/us-
```

```
west-2/2015/08/31/111111111111_CloudTrail-Digest_us-west-2_example-trail-name_us-  
west-2_20150831T221728Z.json.gz valid
```

```
Results requested for 2015-08-31T22:00:00Z to 2015-09-01T19:17:29Z  
Results found for 2015-08-31T22:17:28Z to 2015-09-01T20:17:28Z:
```

```
22/23 digest files valid, 1/23 digest files INVALID  
63/63 log files valid
```

Non-verbose

The following example `validate-logs` command does not use the `--verbose` flag. In the sample output that follows, one error was found. Only the header, error, and summary information are returned.

```
aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-west-  
2:111111111111:trail/example-trail-name --start-time 2015-08-31T22:00:00Z --  
end-time 2015-09-01T19:17:29Z
```

```
Validating log files for trail arn:aws:cloudtrail:us-west-  
2:111111111111:trail/example-trail-name between 2015-08-31T22:00:00Z and 2015-  
09-01T19:17:29Z
```

```
Digest file s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-west-  
2/2015/09/01/144218288521_CloudTrail-Digest_us-west-2_example-trail-name_us-  
west-2_20150901T101728Z.json.gz INVALID: signature verification failed
```

```
Results requested for 2015-08-31T22:00:00Z to 2015-09-01T19:17:29Z  
Results found for 2015-08-31T22:17:28Z to 2015-09-01T20:17:28Z:
```

```
22/23 digest files valid, 1/23 digest files INVALID  
63/63 log files valid
```

Checking Whether a Particular File was Delivered by CloudTrail

To check if a particular file in your bucket was delivered by CloudTrail, run `validate-logs` in verbose mode for the time period that includes the file. If the file appears in the output of `validate-logs`, then the file was delivered by CloudTrail.

CloudTrail Digest File Structure

Each digest file contains the names of the log files that were delivered to your Amazon S3 bucket during the last hour, the hash values for those log files, and the digital signature of the previous digest file. The signature for the current digest file is stored in the metadata properties of the digest file object. The digital signatures and hashes are used for validating the integrity of the log files and of the digest file itself.

Digest File Location

Digest files are delivered to an Amazon S3 bucket location that follows this syntax.

```
s3://S3-bucket-name/AWSLogs/your-aws-account-id/CloudTrail-Digest/  
AWSregionname/digest-end-year/digest-end-month/digest-end-date/
```

```
your-aws-account-id_CloudTrail-Digest_region_trail-name_region_digest_end_timestamp.json.gz
```

Sample Digest File Contents

The following example digest file contains information for a CloudTrail log.

```
{
  "awsAccountId": "111122223333",
  "digestStartTime": "2015-08-17T14:01:31Z",
  "digestEndTime": "2015-08-17T15:01:31Z",
  "digestS3Bucket": "S3-bucket-name",
  "digestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-east-1/2015/08/17/111122223333_CloudTrail-Digest_us-east-1_your-trail-name_us-east-1_20150817T150131Z.json.gz",
  "digestPublicKeyFingerprint": "31e8b5433410dfb61a9dc45cc65b22ff",
  "digestSignatureAlgorithm": "SHA256withRSA",
  "newestEventTime": "2015-08-17T14:52:27Z",
  "oldestEventTime": "2015-08-17T14:42:27Z",
  "previousDigestS3Bucket": "S3-bucket-name",
  "previousDigestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-east-1/2015/08/17/111122223333_CloudTrail-Digest_us-east-1_your-trail-name_us-east-1_20150817T140131Z.json.gz",
  "previousDigestHashValue": "97fb791cf91ffc440d274f8190dbdd9aa09c34432aba82739df18b6d3c13df2d",
  "previousDigestHashAlgorithm": "SHA-256",
  "previousDigestSignature": "50887ccf812dc805bb3eeff6657db32a6cb48d2d096404eb76181877bc6ebb8cd0b23f823200155b2fd8848d428e46e8456328a",
  "logFiles": [
    {
      "s3Bucket": "S3-bucket-name",
      "s3Object": "AWSLogs/111122223333/CloudTrail/us-east-1/2015/08/17/111122223333_CloudTrail_us-east-1_20150817T1445Z_9nYN7gp2eWAJHIFT.json.gz",
      "hashValue": "9bb6196fc6b84d6f075a56548fec262bd99ba3c2de41b618e5b6e22c1fc71f6",
      "hashAlgorithm": "SHA-256",
      "newestEventTime": "2015-08-17T14:52:27Z",
      "oldestEventTime": "2015-08-17T14:42:27Z"
    }
  ]
}
```

Digest File Field Descriptions

The following are descriptions for each field in the digest file:

`awsAccountId`

The AWS account ID for which the digest file has been delivered.

`digestStartTime`

The starting UTC time range that the digest file covers, taking as a reference the time in which log files have been delivered by CloudTrail. This means that if the time range is [Ta, Tb], the digest will contain all the log files delivered to the customer between Ta and Tb.

`digestEndTime`

The ending UTC time range that the digest file covers, taking as a reference the time in which log files have been delivered by CloudTrail. This means that if the time range is [Ta, Tb], the digest will contain all the log files delivered to the customer between Ta and Tb.

`digestS3Bucket`

The name of the Amazon S3 bucket to which the current digest file has been delivered.

`digestS3Object`

The Amazon S3 object key (that is, the Amazon S3 bucket location) of the current digest file. The first two regions in the string show the region from which the digest file was delivered. The last region (after `your-trail-name`) is the home region of the trail. The home region is the region in which the trail was created. In the case of a multi-region trail, this can be different from the region from which the digest file was delivered.

`newestEventTime`

The UTC time of the most recent event among all of the events in the log files in the digest.

`oldestEventTime`

The UTC time of the oldest event among all of the events in the log files in the digest.

Note

If the digest file is delivered late, the value of `oldestEventTime` will be earlier than the value of `digestStartTime`.

`previousDigestS3Bucket`

The Amazon S3 bucket to which the previous digest file was delivered.

`previousDigestS3Object`

The Amazon S3 object key (that is, the Amazon S3 bucket location) of the previous digest file.

`previousDigestHashValue`

The hexadecimal encoded hash value of the uncompressed contents of the previous digest file.

`previousDigestHashAlgorithm`

The name of the hash algorithm that was used to hash the previous digest file.

`publicKeyFingerprint`

The hexadecimal encoded fingerprint of the public key that matches the private key used to sign this digest file. You can retrieve the public keys for the time range corresponding to the digest file by using the AWS CLI or the CloudTrail API. Of the public keys returned, the one whose fingerprint matches this value can be used for validating the digest file. For information about retrieving public keys for digest files, see the AWS CLI [list-public-keys](#) command or the CloudTrail [ListPublicKeys](#) API.

Note

CloudTrail uses different private/public key pairs per region. Each digest file is signed with a private key unique to its region. Therefore, when you validate a digest file from a particular region, you must look in the same region for its corresponding public key.

`digestSignatureAlgorithm`

The algorithm used to sign the digest file.

`logFiles.s3Bucket`

The name of the Amazon S3 bucket for the log file.

`logFiles.s3Object`

The Amazon S3 object key of the current log file.

`logFiles.newestEventTime`

The UTC time of the most recent event in the log file. This time also corresponds to the time stamp of the log file itself.

`logFiles.oldestEventTime`

The UTC time of the oldest event in the log file.

`logFiles.hashValue`

The hexadecimal encoded hash value of the uncompressed log file content.

`logFiles.hashAlgorithm`

The hash algorithm used to hash the log file.

Starting Digest File

When log file integrity validation is started, a starting digest file will be generated. A starting digest file will also be generated when log file integrity validation is restarted (by either disabling and then reenabling log file integrity validation, or by stopping logging and then restarting logging with validation enabled). In a starting digest file, the following fields relating to the previous digest file will be null:

- `previousDigestS3Bucket`
- `previousDigestS3Object`
- `previousDigestHashValue`
- `previousDigestHashAlgorithm`
- `previousDigestSignature`

'Empty' Digest Files

CloudTrail will deliver a digest file even when there has been no API activity in your account during the one hour period that the digest file represents. This can be useful when you need to assert that no log files were delivered during the hour reported by the digest file.

The following example shows the contents of a digest file that recorded an hour when no API activity occurred. Note that the `logFiles:[]` field at the end of the digest file contents is empty.

```
{
  "awsAccountId": "111122223333",
  "digestStartTime": "2015-08-20T17:01:31Z",
  "digestEndTime": "2015-08-20T18:01:31Z",
  "digestS3Bucket": "example-bucket-name",
  "digestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-east-1/2015/08/20/111122223333_CloudTrail-Digest_us-east-1_example-trail-name_us-east-1_20150820T180131Z.json.gz",
  "digestPublicKeyFingerprint": "31e8b5433410dfb61a9dc45cc65b22ff",
  "digestSignatureAlgorithm": "SHA256withRSA",
  "newestEventTime": null,
  "oldestEventTime": null,
  "previousDigestS3Bucket": "example-bucket-name",
  "previousDigestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-east-1/2015/08/20/111122223333_CloudTrail-Digest_us-east-1_example-trail-name_us-east-1_20150820T170131Z.json.gz",
  "previousDigestHashValue":
"ed96c4bac9eaa8fe9716ca0e515da51938be651b1db31d781956416a9d05cdfa",
  "previousDigestHashAlgorithm": "SHA-256",
  "previousDigestSignature":
"82705525fb0fe7f919f9434e5b7138cb41793c776c7414f3520c0242902daa8cc8286b29263d2627f2f259471c
bb3a4e5d8c5f17673celf989dff82d4becf24e452f20d3bcac94ad50131f93e57f10155536acb54c60ef
be9d57228c2b930bc6082b2318e3ccd36834a8e835b8d112dbf32145f445c11",
  "logFiles": []
}
```

Signature of the Digest File

The signature information for a digest file is located in two object metadata properties of the Amazon S3 digest file object. Each digest file has the following metadata entries:

- x-amz-meta-signature

The hexadecimal encoded value of the digest file signature. The following is an example signature:

```
3e47236a289ef341b31f85f9400ef32f600823f4682966b9949e4f98d53d150f54c63862b308504c55541b5294899b523db7b
af9e204430f3877135e9fce5918571e538fce895ca0ef3cd726e7c460336ce2ff6b739e98aef09eb53bba8fc4b59469579c30fc58d
c34619e5874523abb289e3180ff4c4d17809e022df5d1794146a8f0d0a1e7f98a08c6953ada0c4b95599443a6740809208ae07ef
28f1cc237e372264e51b611c1c42956cdf703539f4e71009051769469231bc22232fa260f02740047af53229885ea20e95acd3533267104941e0cb
d
d6520ff299f47828a5916c2406f2b5894d782d678537f54491312f923f627048a1d0282af47f5241a1545a526388c36a6f9f
dd6952987c31ae871650e130bd2e63bfe145b22bbd39ea192210f6df64d49b888a321e02d3fc4cf126ac
cae30d2857ccd6b2286a7c9feba6c35c44161b24147d645e6ca26844ba
05d3ffcb5d2dd5dc28f8bb5b7993938e8a5f912a82b448a367ec
cb2ec0f198ba71e23eb0b97278cf65f3c8d1e652c6de33a22ca8428821ffc95bf8b726ba9f37cf
bc20c54dc5bd6159c0ealc4d951b68db8e0528852c55db0c5e499ea60560f7c2bb3af7f694407da863a2594f7a2f2838db09254af
baf8003587746ef719a0437f85eef
f52289e9a375061495245e9269e8792e4f05a76c1e97fa8802468c382a037006701374774e28970524579260ca98460
```

- x-amz-meta-signature-algorithm

The following shows an example value of the algorithm used to generate the digest signature:

SHA256withRSA

Digest File Chaining

The fact that each digest file contains a reference to its previous digest file enables a "chaining" that permits validation tools like the AWS CLI to detect if a digest file has been deleted. It also allows the digest files in a specified time range to be successively inspected, starting with the most recent first.

Note

When you disable log file integrity validation, the chain of digest files is broken after one hour. CloudTrail will not create digest files for log files that were delivered during a period in which log file integrity validation was disabled. For example, if you enable log file integrity validation at noon on January 1, disable it at noon on January 2, and re-enable it at noon on January 10, digest files will not be created for the log files delivered from noon on January 2 to noon on January 10. The same applies whenever you stop CloudTrail logging or delete a trail.

If logging is stopped or the trail is deleted, CloudTrail will deliver a final digest file. This digest file can contain information for any remaining log files that cover events up to and including the `StopLogging` event.

Custom Implementations of CloudTrail Log File Integrity Validation

Because CloudTrail uses industry standard, openly available cryptographic algorithms and hash functions, you can create your own tools to validate the integrity of CloudTrail log files. When log file integrity validation is enabled, CloudTrail delivers digest files to your Amazon S3 bucket. You can use these files to implement your own validation solution. For more information about digest files, see [CloudTrail Digest File Structure \(p. 170\)](#).

This topic describes how digest files are signed, and then details the steps that you will need to take to implement a solution that validates the digest files and the log files that they reference.

Understanding How CloudTrail Digest Files are Signed

CloudTrail digest files are signed with RSA digital signatures. For each digest file, CloudTrail does the following:

1. Creates a string for data signing based on designated digest file fields (described in the next section).
2. Gets a private key unique to the region.
3. Passes the SHA-256 hash of the string and the private key to the RSA signing algorithm, which produces a digital signature.
4. Encodes the byte code of the signature into hexadecimal format.
5. Puts the digital signature into the `x-amz-meta-signature` metadata property of the Amazon S3 digest file object.

Contents of the Data Signing String

The following CloudTrail objects are included in the string for data signing:

- The ending timestamp of the digest file in UTC extended format (for example, `2015-05-08T07:19:37Z`)
- The current digest file S3 path
- The hexadecimal-encoded SHA-256 hash of the current digest file
- The hexadecimal-encoded signature of the previous digest file

The format for calculating this string and an example string are provided later in this document.

Custom Validation Implementation Steps

When implementing a custom validation solution, you will need to validate the digest file first, and then the log files that it references.

Validate the Digest File

To validate a digest file, you need its signature, the public key whose private key was used to signed it, and a data signing string that you compute.

1. Get the digest file.
2. Verify that the digest file has been retrieved from its original location.
3. Get the hexadecimal-encoded signature of the digest file.
4. Get the hexadecimal-encoded fingerprint of the public key whose private key was used to sign the digest file.
5. Retrieve the public keys for the time range corresponding to the digest file.
6. From among the public keys retrieved, choose the public key whose fingerprint matches the fingerprint in the digest file.
7. Using the digest file hash and other digest file fields, recreate the data signing string used to verify the digest file signature.
8. Validate the signature by passing in the SHA-256 hash of the string, the public key, and the signature as parameters to the RSA signature verification algorithm. If the result is true, the digest file is valid.

Validate the Log Files

If the digest file is valid, validate each of the log files that the digest file references.

1. To validate the integrity of a log file, compute its SHA-256 hash value on its uncompressed content and compare the results with the hash for the log file recorded in hexadecimal in the digest. If the hashes match, the log file is valid.
2. By using the information about the previous digest file that is included in the current digest file, validate the previous digest files and their corresponding log files in succession.

The following sections describe these steps in detail.

A. Get the Digest File

The first steps are to get the most recent digest file, verify that you have retrieved it from its original location, verify its digital signature, and get the fingerprint of the public key.

1. Using `S3 Get` or the `AmazonS3Client` class (for example), get the most recent digest file from your Amazon S3 bucket for the time range that you want to validate.
2. Check that the S3 bucket and S3 object used to retrieve the file match the S3 bucket S3 object locations that are recorded in the digest file itself.
3. Next, get the digital signature of the digest file from the `x-amz-meta-signature` metadata property of the digest file object in Amazon S3.
4. In the digest file, get the fingerprint of the public key whose private key was used to sign the digest file from the `digestPublicKeyFingerprint` field.

B. Retrieve the Public Key for Validating the Digest File

To get the public key to validate the digest file, you can use either the AWS CLI or the CloudTrail API. In both cases, you specify a time range (that is, a start time and end time) for the digest files that you want

AWS CloudTrail User Guide
Custom Implementations of CloudTrail Log File Integrity
Validation

to validate. One or more public keys may be returned for the time range that you specify. The returned keys may have validity time ranges that overlap.

Note

Because CloudTrail uses different private/public key pairs per region, each digest file is signed with a private key unique to its region. Therefore, when you validate a digest file from a particular region, you must retrieve its public key from the same region.

Use the AWS CLI to Retrieve Public Keys

To retrieve public keys for digest files by using the AWS CLI, use the `cloudtrail list-public-keys` command. The command has the following format:

```
aws cloudtrail list-public-keys [--start-time <start-time>] [--end-time <end-time>]
```

The start-time and end-time parameters are UTC timestamps and are optional. If not specified, the current time is used, and the currently active public key or keys are returned.

Sample Response

The response will be a list of JSON objects representing the key (or keys) returned:

```
{
  "publicKeyList": [
    {
      "ValidityStartTime": "1436317441.0",
      "ValidityEndTime": "1438909441.0",
      "Value": "MIIBCgKCAQEAn1lL2YZ9h7onug2ILilMWyHiMR
sTQjfWE+pHVRLk1QjfWhirG+lpOa8NrwQ/r7Ah5bNL6HepznOU9XTDSfmmnP97mqyc7z/upfZdS/AH
hYcGaz7n6Wc/RRBU6VmiPCrAUojuSk6/GjvA8iOPFsYDuBtviXarvuLPlrT9kAd4Lb+rFfr5peEg
BEkhlzc5HuW07S0y+KunqxX6jQBnXGMtxmPBPP0FylgWGNdFtks/4YSKcgqwH0YDcawP9GGDAeCIqP
WIXDLG1jOjRRzWfCmd0iJUkz8vTsn4hq/5ZxRFE7UBAUiVcGbdnDdvVfhF9C3dQiDq3k7adQIz
iLT0cShgQIDAQAB",
      "Fingerprint": "8eba5db5bea9b640d1c96a77256fe7f2"
    },
    {
      "ValidityStartTime": "1434589460.0",
      "ValidityEndTime": "1437181460.0",
      "Value": "MIIBCgKCAQEApfYL2FiZhpN74LNWVUzhr+VheYh
whYm8w0n5Gf6i95ylw5kBAWKVEmnAQG7BvS5g9SMqFDQx52fW7NWW44IvfJ2xGXT+wT+Dgr6ZQ+6yx
skQnqv5YcXj4Aa5Zz4jJfsYjDuO2MDTZNIzNvBNza
BJ+r2WIWAJ/Xq54kyF63B6WE38vKuDE7nSdlFqQuEoNBFLPIn
vgggYe2Ym1Refe2z7lwNcJ2kY+q0h1BSHrSM8RWuJIw7MXwF9iQncg9jYzU1NJomozQzAG5wSRf
bplcCNY40xvGd/aAm00m+Y+XFMrKwtLCwseHPvJ843qVno6x4BJN9bpWnoPo9sdsbGoik3QIDAQAB",
      "Fingerprint": "8933b39ddc64d26d8e14ffbf6566fee4"
    },
    {
      "ValidityStartTime": "1434589370.0",
      "ValidityEndTime": "1437181370.0",
      "Value": "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqlzPjBvZJ42Ud
cmLfPUqXYnfOs6I8lCfao/tOs8CmzPOEdtLWugB9xoIUz78qVHdKIqxbaG4jWHfJBioSSf
BM0lt8cdVo4TnRa7oG9io5pysS6DJhBBAeXsicufsi
FJR+wrUnH8RSLxL4k6G1+BhLX20tJkZ/erT97tDGBujaelqseGg3vPZb
Tx9SMfOLN65PdLFudLP7GatOZ9p5jw/rjpc1Kfo9Bfc3heeBxWGKwBBOKnFAAn9V57pOaosCvPKm
Hd9bg7jsQkI9Xp22IzGLsTFJZYVA3KiTAE1DMu80iFXPHEq9hKNbt9e4URFam+lutKVEiLkr2disd
CmPTK0VQIDAQAB",
      "Fingerprint": "31e8b5433410dfb61a9dc45cc65b22ff"
    }
  ]
}
```

```
}  
  ]  
}
```

Use the CloudTrail API to Retrieve Public Keys

To retrieve public keys for digest files by using the CloudTrail API, pass in start time and end time values to the `ListPublicKeys` API. The `ListPublicKeys` API returns the public keys whose private keys were used to sign digest files within the specified time range. For each public key, the API also returns the corresponding fingerprint.

ListPublicKeys

This section describes the request parameters and response elements for the `ListPublicKeys` API.

Note

The encoding for the binary fields for `ListPublicKeys` is subject to change.

Request Parameters

Name	Description
Start-Time	Optionally specifies, in UTC, the start of the time range to look up public keys for CloudTrail digest files. If <code>StartTime</code> is not specified, the current time is used, and the current public key is returned. Type: DateTime
EndTime	Optionally specifies, in UTC, the end of the time range to look up public keys for CloudTrail digest files. If <code>EndTime</code> is not specified, the current time is used. Type: DateTime

Response Elements

`PublicKeyList`, an array of `PublicKey` objects that contains:

Name	Description
Value	The DER encoded public key value in PKCS #1 format. Type: Blob
ValidityStart-Time	The starting time of validity of the public key. Type: DateTime
ValidityEnd-Time	The ending time of validity of the public key. Type: DateTime
Fingerprint	The fingerprint of the public key. The fingerprint can be used to identify the public key that you must use to validate the digest file. Type: String

C. Choose the Public Key to Use for Validation

From among the public keys retrieved by `list-public-keys` or `ListPublicKeys`, choose the public key returned whose fingerprint matches the fingerprint recorded in the `digestPublicKeyFingerprint` field of the digest file. This is the public key that you will use to validate the digest file.

D. Recreate the Data Signing String

Now that you have the signature of the digest file and associated public key, you need to calculate the data signing string. After you have calculated the data signing string, you will have the inputs needed to verify the signature.

The data signing string has the following format:

```
Data_To_Sign_String =  
  Digest_End_Timestamp_in_UTC_Extended_format + '\n' +  
  Current_Digest_File_S3_Path + '\n' +  
  Hex(Sha256(current-digest-file-content)) + '\n' +  
  Previous_digest_signature_in_hex
```

An example `Data_To_Sign_String` follows.

```
2015-08-12T04:01:31Z  
S3-bucket-name/AWSLogs/111122223333/CloudTrail-Digest/us-east-  
1/2015/08/12/111122223333_us-east-1_CloudTrail-Digest_us-east-  
1_20150812T040131Z.json.gz  
4ff08d7c6ecd6eb313257e839645d20363ee3784a2328a7d76b99b53cc9bcacd  
6e8540b83c3ac86a0312d971a225361d28ed0af20d70c211a2d405e32abf529a8145c2966e3bb47362383a52441545ed091fb81  
d4c7c09d1152b84e79099ce7a9ec35d2b264eb92ab6e090f1e5ec5d40ec8a0729c02ff57f9e30b5343a8591638f8b794972ce15db3063a01972  
9a218f6c725f221e02967fafa7455a0547154932206580c026f999a50b20f707205f698895f12827022b7028b74544345
```

After you recreate this string, you can validate the digest file.

E. Validate the Digest File

Pass the SHA-256 hash of the recreated data signing string, digital signature, and public key to the RSA signature verification algorithm. If the output is true, the signature of the digest file is verified and the digest file is valid.

F. Validate the Log Files

After you have validated the digest file, you can validate the log files it references. The digest file contains the SHA-256 hashes of the log files. If one of the log files was modified after CloudTrail delivered it, the SHA-256 hashes will change, and the signature of digest file will not match.

The following shows how validate the log files:

1. Do an `S3 Get` of the log file using the `S3` location information in the digest file's `logFiles.s3Bucket` and `logFiles.s3Object` fields.
2. If the `S3 Get` operation is successful, iterate through the log files listed in the digest file's `logFiles` array using the following steps:
 - a. Retrieve the original hash of the file from the `logFiles.hashValue` field of the corresponding log in the digest file.

- b. Hash the uncompressed contents of the log file with the hashing algorithm specified in `logFiles.hashAlgorithm`.
- c. Compare the hash value that you generated with the one for the log in the digest file. If the hashes match, the log file is valid.

G. Validate Additional Digest and Log Files

In each digest file, the following fields provide the location and signature of the previous digest file:

- `previousDigestS3Bucket`
- `previousDigestS3Object`
- `previousDigestSignature`

Use this information to visit previous digest files sequentially, validating the signature of each and the log files that they reference by using the steps in the previous sections. The only difference is that for previous digest files, you do not need to retrieve the digital signature from the digest file object's Amazon S3 metadata properties. The signature for the previous digest file is provided for you in the `previousDigestSignature` field.

You can go back until the starting digest file is reached, or until the chain of digest files is broken, whichever comes first.

Validating Digest and Log Files Offline

When validating digest and log files offline, you can generally follow the procedures described in the previous sections. However, you must take into account the following areas:

Handling the Most Recent Digest File

The digital signature of the most recent (that is, "current") digest file is in the Amazon S3 metadata properties of the digest file object. In an offline scenario, the digital signature for the current digest file will not be available.

Two possible ways of handling this are:

- Since the digital signature for the previous digest file is in the current digest file, start validating from the next-to-last digest file. With this method, the most recent digest file cannot be validated.
- As a preliminary step, obtain the signature for the current digest file from the digest file object's metadata properties (for example, by calling the Amazon S3 [getObjectMetadata](#) API) and then store it securely offline. This would allow the current digest file to be validated in addition to the previous files in the chain.

Path Resolution

Fields in the downloaded digest files like `s3Object` and `previousDigestS3Object` will still be pointing to Amazon S3 online locations for log files and digest files. An offline solution must find a way to reroute these to the current path of the downloaded log and digest files.

Public Keys

In order to validate offline, all of the public keys that you need for validating log files in a given time range must first be obtained online (by calling `ListPublicKeys`, for example) and then stored securely offline. This step must be repeated whenever you want to validate additional files outside the initial time range that you specified.

Sample Validation Snippet

The following sample snippet provides skeleton code for validating CloudTrail digest and log files. The skeleton code is online/offline agnostic; that is, it is up to you to decide whether to implement it with or without online connectivity to AWS. The suggested implementation uses the [Java Cryptography Extension \(JCE\)](#) and [Bouncy Castle](#) as a security provider.

The sample snippet shows:

- How to create the data signing string used to validate the digest file signature.
- How to verify the digest file signature.
- How to verify the log file hashes.
- A code structure for validating a chain of digest files.

```
import java.util.Arrays;
import java.security.MessageDigest;
import java.security.KeyFactory;
import java.security.PublicKey;
import java.security.Security;
import java.security.Signature;
import java.security.spec.X509EncodedKeySpec;
import org.json.JSONObject;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.apache.commons.codec.binary.Hex;

public class DigestFileValidator {

    public void validateDigestFile(String digestS3Bucket, String digestS3Object,
String digestSignature) {

        // Using the Bouncy Castle provider as a JCE security provider - ht
tp://www.bouncycastle.org/
        Security.addProvider(new BouncyCastleProvider());

        // Load the digest file from S3 (using Amazon S3 Client) or from your
local copy
        JSONObject digestFile = loadDigestFileInMemory(digestS3Bucket, digestS3Ob
ject);

        // Check that the digest file has been retrieved from its original
location
        if (!digestFile.getString("digestS3Bucket").equals(digestS3Bucket) ||
            !digestFile.getString("digestS3Object").equals(digestS3Object))
        {
            System.err.println("Digest file has been moved from its original
location.");
        } else {
            // Compute digest file hash
            MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");

            messageDigest.update(convertToByteArray(digestFile));
            byte[] digestFileHash = messageDigest.digest();
            messageDigest.reset();

            // Compute the data to sign
```

AWS CloudTrail User Guide
Custom Implementations of CloudTrail Log File Integrity
Validation

```
String dataToSign = String.format("%s%n%s/%s%n%s%n%s",
    digestFile.getString("digestEndTime"),
    digestFile.getString("digestS3Bucket"), digest
File.getString("digestS3Object"), // Constructing the S3 path of the digest
file as part of the data to sign
    Hex.encodeHexString(digestFileHash),
    digestFile.getString("previousDigestSignature"));

byte[] signatureContent = Hex.decodeHex(digestSignature);

/*
NOTE:
To find the right public key to verify the signature, call
CloudTrail ListPublicKey API to get a list
of public keys, then match by the publicKeyFingerprint in the
digest file. Also, the public key bytes
returned from ListPublicKey API are DER encoded in PKCS#1 format:

PublicKeyInfo ::= SEQUENCE {
    algorithm      AlgorithmIdentifier,
    PublicKey      BIT STRING
}

AlgorithmIdentifier ::= SEQUENCE {
    algorithm      OBJECT IDENTIFIER,
    parameters    ANY DEFINED BY algorithm OPTIONAL
}
*/
pkcs1PublicKeyBytes = getPublicKey(digestFile.getString("digestPub
licKeyFingerprint"));

// Transform the PKCS#1 formatted public key to x.509 format.
RSAPublicKey rsaPublicKey = RSAPublicKey.getInstance(pkcs1PublicK
eyBytes);
AlgorithmIdentifier rsaEncryption = new AlgorithmIdentifier(PKCSOb
jectIdentifiers.rsaEncryption, null);
SubjectPublicKeyInfo publicKeyInfo = new SubjectPublicKeyInfo(rsaEn
cryption, rsaPublicKey);

// Create the PublicKey object needed for the signature validation

PublicKey publicKey = KeyFactory.getInstance("RSA", "BC").generate
Public(new X509EncodedKeySpec(publicKeyInfo.getEncoded()));

// Verify signature
Signature signature = Signature.getInstance("SHA256withRSA", "BC");

signature.initVerify(publicKey);
signature.update(dataToSign.getBytes("UTF-8"));

if (signature.verify(signatureContent)) {
    System.out.println("Digest file signature is valid, validating
log files...");
    for (int i = 0; i < digestFile.getJSONArray("logFiles").length();
i++) {
```

AWS CloudTrail User Guide
Custom Implementations of CloudTrail Log File Integrity
Validation

```
        JSONObject logFileMetadata = digestFile.getJSONArray("log
Files").getJSONObject(i);

        // Compute log file hash
        byte[] logFileContent = loadUncompressedLogFileInMemory(
            logFileMetadata.getString("s3Buck
et"),
            logFileMetadata.getString("s3Ob
ject")
        );
        messageDigest.update(logFileContent);
        byte[] logFileHash = messageDigest.digest();
        messageDigest.reset();

        // Retrieve expected hash for the log file being processed
        byte[] expectedHash = Hex.decodeHex(logFileMetadata.get
String("hashValue"));

        boolean signaturesMatch = Arrays.equals(expectedHash, log
FileHash);
        if (!signaturesMatch) {
            System.err.println(String.format("Log file: %s/%s hash
doesn't match.\tExpected: %s Actual: %s",
            logFileMetadata.getString("s3Bucket"), logFile
Metadata.getString("s3Object"),
            Hex.encodeHexString(expectedHash), Hex.encodeHex
String(logFileHash)));
        } else {
            System.out.println(String.format("Log file: %s/%s hash
match",
            logFileMetadata.getString("s3Bucket"), logFile
Metadata.getString("s3Object")));
        }
    } else {
        System.err.println("Digest signature failed validation.");
    }

    System.out.println("Digest file validation completed.");

    if (chainValidationIsEnabled()) {
        // This enables the digests' chain validation
        validateDigestFile(
            digestFile.getString("previousDigestS3Bucket"),
            digestFile.getString("previousDigestS3Object"),
            digestFile.getString("previousDigestSignature"));
    }
}
}
```

Using the CloudTrail Processing Library

The CloudTrail Processing Library is a Java library that provides an easy way to process AWS CloudTrail logs in a fault-tolerant, scalable and flexible way. You provide configuration details about your CloudTrail SQS queue and write code to process events. The CloudTrail Processing Library does the rest, polling your Amazon SQS queue, reading and parsing queue messages, downloading CloudTrail log files, parsing events in the log files and passing them to your code as Java objects. The CloudTrail Processing Library is highly scalable and fault-tolerant, handling parallel processing of log files so that you can process as many logs as necessary, and robustly handling network failures related to network timeouts and inaccessible resources.

This chapter provides information about how to use the CloudTrail Processing Library to process CloudTrail logs in your Java projects. The library is provided as an Apache-licensed open-source project, available on GitHub:

- <https://github.com/aws/aws-cloudtrail-processing-library>

The library source includes sample code that you can use as a base for your own projects.

Topics

- [Minimum requirements \(p. 184\)](#)
- [Processing CloudTrail Logs with the CloudTrail Processing Library \(p. 184\)](#)
- [Advanced Topics \(p. 188\)](#)
- [Additional Resources \(p. 191\)](#)

Minimum requirements

To use the CloudTrail Processing Library, you must have the following:

- [AWS SDK for Java 1.10.27](#)
- [Java 1.7](#)

Processing CloudTrail Logs with the CloudTrail Processing Library

To use the CloudTrail Processing Library to process CloudTrail logs in your Java application:

1. [Add the CloudTrail Processing Library to your Project \(p. 184\)](#)
2. [Configure the CloudTrail Processing Library \(p. 186\)](#)
3. [Implement the Events Processor \(p. 187\)](#)
4. [Instantiate and Run the Processing Executor \(p. 188\)](#)

Add the CloudTrail Processing Library to your Project

To use the CloudTrail Processing Library you must add it to your Java project's classpath.

Topics

- [Adding the Library to an Apache Ant Project \(p. 185\)](#)
- [Adding the Library to an Apache Maven Project \(p. 185\)](#)

- [Adding the CloudTrail Processing Library to an Eclipse Project \(p. 185\)](#)

Adding the Library to an Apache Ant Project

To add the CloudTrail Processing Library to an Ant project

1. Download or clone the CloudTrail Processing Library source code from GitHub at:

- <https://github.com/aws/aws-cloudtrail-processing-library>

2. Build the .jar file from source as described in the [README](#):

```
mvn clean install -Dpgp.skip=true
```

3. Copy the resulting .jar file into your project and add it to your project's `build.xml` file. For example:

```
<classpath>
  <pathelement path="${classpath}"/>
  <pathelement location="lib/aws-cloudtrail-processing-library-1.0.1.jar"/>
</classpath>
```

Adding the Library to an Apache Maven Project

The CloudTrail Processing Library is available for [Apache Maven](#), so adding it to your project is as easy as writing a single dependency in your project's `pom.xml` file.

To add the CloudTrail Processing Library to a Maven project

- Using your favorite text editor, open your Maven project's `pom.xml` file and add the following dependency:

```
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-cloudtrail-processing-library</artifactId>
  <version>1.0.1</version>
</dependency>
```

Adding the CloudTrail Processing Library to an Eclipse Project

To add the CloudTrail Processing Library to an Eclipse project

1. Download or clone the CloudTrail Processing Library source code from GitHub at:

- <https://github.com/aws/aws-cloudtrail-processing-library>

2. Build the .jar file from source as described in the [README](#):

```
mvn clean install -Dgpg.skip=true
```

3. Copy the built `aws-cloudtrail-processing-library-1.0.1.jar` to a directory in your project (typically `lib`).
4. Right-click your project's name in the Eclipse **Project Explorer**, and select **Build Path > Configure**
5. In the **Java Build Path** window, click the **Libraries** tab.
6. Click **Add JARs...** and navigate to the path where you copied `aws-cloudtrail-processing-library-1.0.1.jar`.
7. Click **OK** to complete adding the `.jar` to your project.

Configure the CloudTrail Processing Library

You can configure the CloudTrail Processing Library by creating a classpath properties file that is loaded at runtime, or by creating a `ClientConfiguration` object and setting options manually.

Providing a Properties File

You can write a classpath properties file that provides configuration options to your application. Here is an example file that demonstrates the options you can set:

```
# AWS access key. (Required)
accessKey = your_access_key

# AWS secret key. (Required)
secretKey = your_secret_key

# The SQS URL used to pull CloudTrail notification from. (Required)
sqsUrl = your_sqs_queue_url

# The SQS end point specific to a region.
sqsRegion = us-east-1

# A period of time during which Amazon SQS prevents other consuming components
# from receiving and processing that message.
visibilityTimeout = 60

# The S3 region to use.
s3Region = us-east-1

# Number of threads used to download S3 files in parallel. Callbacks can be
# invoked from any thread.
threadCount = 1

# The time allowed, in seconds, for threads to shut down after
# AWScloudTrailEventProcessingExecutor.stop() is called. If they are still
# running beyond this time, they will be forcibly terminated.
threadTerminationDelaySeconds = 60

# The maximum number of AWScloudTrailClientEvents sent to a single invocation
# of processEvents().
maxEventsPerEmit = 10

# Whether to include raw event information in CloudTrailDeliveryInfo.
enableRawEventInfo = false
```

The required parameters are *sqsUrl*, *accessKey* and *secretKey*. The *sqsUrl* parameter provides the URL to pull your CloudTrail notifications from. If you don't provide this value, then an `IllegalStateException` will be thrown by the `AWSCloudTrailProcessingExecutor`. The *accessKey* and *secretKey* parameters provide your AWS credentials to the library, allowing it to access AWS on your behalf.

The other parameters have reasonable defaults that are set by the library. For information about the default values for each option, see the [AWS CloudTrail Processing Library Reference](#).

Creating a ClientConfiguration

Instead of setting options in the classpath properties, you can provide options to the `AWSCloudTrailProcessingExecutor` by initializing and setting options on a `ClientConfiguration` object.

For example:

```
ClientConfiguration basicConfig = new ClientConfiguration(
    "http://sqs.us-east-1.amazonaws.com/123456789012/queue2",
    new DefaultAWSCredentialsProviderChain());

basicConfig.setEnableRawEventInfo(true);
basicConfig.setThreadCount(4);
basicConfig.setnEventsPerEmit(20);
```

Implement the Events Processor

To process CloudTrail logs, you must implement a `EventsProcessor` that receives the CloudTrail log data. Here is an example implementation:

```
public class SampleEventsProcessor implements EventsProcessor {

    public void process(List<CloudTrailEvent> events) {
        int i = 0;
        for (CloudTrailEvent event : events) {
            System.out.println(String.format("Process event %d : %s", i++,
                event.getEventData()));
        }
    }
}
```

When implementing a `EventsProcessor`, you implement the `process()` callback that the `AWSCloudTrailProcessingExecutor` uses to send you CloudTrail events. Events are provided in a list of `CloudTrailClientEvent` objects.

The `CloudTrailClientEvent` object provides a `CloudTrailEvent` and `CloudTrailEventMetadata` that you can use to read the CloudTrail event and delivery information.

This simple example just prints the event information for each event passed to `SampleEventsProcessor`. In your own implementation, you can process logs as you see fit. The `AWSCloudTrailProcessingExecutor` will continue to send events to your `EventsProcessor` as long as it has events to send and is still running.

Instantiate and Run the Processing Executor

Once you have written a `EventsProcessor` and have set configuration values for the CloudTrail Processing Library (either in a properties file or by using the `ClientConfiguration` class), you can use these elements to initialize and use a `AWSCloudTrailProcessingExecutor`.

To use `AWSCloudTrailProcessingExecutor` to process CloudTrail events

1. Instantiate an `AWSCloudTrailProcessingExecutor.Builder` object. `Builder`'s constructor takes a `EventsProcessor` object and a classpath properties file name.
2. Call the `Builder`'s `build()` factory method to configure and obtain an `AWSCloudTrailProcessingExecutor` object.
3. Use the `AWSCloudTrailProcessingExecutor`'s `start()` and `stop()` methods to begin and end CloudTrail event processing.

```
public class SampleApp {
    public static void main(String[] args) throws InterruptedException {
        AWSCloudTrailProcessingExecutor executor = new
            AWSCloudTrailProcessingExecutor.Builder(new SampleEventsProcessor(),
                "/myproject/cloudtrailprocessing.properties").build();

        executor.start();
        Thread.sleep(24 * 60 * 60 * 1000); // let it run for a while (optional)
        executor.stop(); // optional
    }
}
```

Advanced Topics

Topics

- [Filtering the Events to Process \(p. 188\)](#)
- [Reporting Progress \(p. 190\)](#)
- [Handling Errors \(p. 191\)](#)

Filtering the Events to Process

By default, all of the logs in your Amazon SQS queue's S3 bucket and all of the events that they contain will be sent to your `EventsProcessor`. The CloudTrail Processing Library provides optional interfaces that you can implement to filter the sources used to obtain CloudTrail logs and to filter the events that you are interested in processing.

SourceFilter

You can implement the `SourceFilter` interface to choose whether or not you want to process logs from a provided source. `SourceFilter` declares a single callback method, `filterSource()`, that receives a `CloudTrailSource` object. To keep events from a source from being processed, return `false` from `filterSource()`.

The `filterSource()` method is called by the CloudTrail Processing Library after the library has polled for logs on the Amazon SQS queue, but before event filtering or processing has been done for those logs.

Here is an example implementation:


```
public class SampleSourceFilter implements SourceFilter{
    private static final int MAX_RECEIVED_COUNT = 3;

    private static List<String> accountIDs ;
    static {
        accountIDs = new ArrayList<>();
        accountIDs.add("123456789012");
        accountIDs.add("234567890123");
    }

    @Override
    public boolean filterSource(CloudTrailSource source) throws CallbackException {
        source = (SQSBasedSource) source;
        Map<String, String> sourceAttributes = source.getSourceAttributes();

        String accountId = sourceAttributes.get(
            SourceAttributeKeys.ACCOUNT_ID.getAttributeKey());

        String receivedCount = sourceAttributes.get(
            SourceAttributeKeys.APPROXIMATE_RECEIVE_COUNT.getAttributeKey());

        int approximateReceivedCount = Integer.parseInt(receivedCount);

        return approximateReceivedCount <= MAX_RECEIVED_COUNT && accountIDs.contains(accountId);
    }
}
```

If you don't provide your own `SourceFilter`, then `DefaultSourceFilter` will be used, which allows all sources to be processed (it always returns `true`).

EventFilter

You can implement the `EventFilter` interface to choose whether a CloudTrail event will be sent to your `EventsProcessor` or not. `EventFilter` declares a single callback method, `filterEvent()`, that receives a `CloudTrailEvent` object. To keep the event from being processed, return `false` from `filterEvent()`.

The `filterEvent()` method is called by the CloudTrail Processing Library after the library has polled for logs on the Amazon SQS queue and after source filtering, but before event processing has been done for those logs.

Here is an example implementation:

```
public class SampleEventFilter implements EventFilter{

    private static final String EC2_EVENTS = "ec2.amazonaws.com";

    @Override
    public boolean filterEvent(CloudTrailClientEvent clientEvent) throws CallbackException {
        CloudTrailEvent event = clientEvent.getEvent();

        String eventSource = event.getEventSource();
        String eventName = event.getEventName();

        return eventSource.equals(EC2_EVENTS) && eventName.startsWith("Delete");
    }
}
```

```
}  
}
```

If you don't provide your own `EventFilter`, then `DefaultEventFilter` will be used, which allows all events to be processed (it always returns `true`).

Reporting Progress

The `ProgressReporter` interface can be implemented to customize the reporting of CloudTrail Processing Library progress. `ProgressReporter` declares two methods: `reportStart()` and `reportEnd()`, which are called at the beginning and end of the following operations:

- polling messages from Amazon SQS.
- parsing messages from Amazon SQS.
- processing an Amazon SQS source for CloudTrail logs.
- deleting messages from Amazon SQS.
- downloading a CloudTrail log file.
- processing a CloudTrail log file.

Both methods receive a `ProgressStatus` object that contains information about the operation being performed (in the `progressState` member, which holds a member of the `ProgressState` enumeration that identifies the current operation) and can contain additional information (in the `progressInfo` member). Additionally, any object that you return from `reportStart()` will be passed to `reportEnd()`, so you can provide contextual information such as what time it was when the event began processing.

Here is an example implementation that provides information about how long an operation took to complete:

```
public class SampleProgressReporter implements ProgressReporter {  
    private static final Log logger =  
        LoggerFactory.getLog(DefaultProgressReporter.class);  
  
    @Override  
    public Object reportStart(ProgressStatus status) {  
        return new Date();  
    }  
  
    @Override  
    public void reportEnd(ProgressStatus status, Object startDate) {  
        System.out.println(status.getProgressState().toString() + " is " +  
            status.getProgressInfo().isSuccess() + " , and latency is " +  
            Math.abs(((Date) startDate).getTime()-new Date().getTime()) + "  
            milliseconds.");  
    }  
}
```

If you don't implement your own `ProgressReporter`, then `DefaultExceptionHandler`, which prints the name of the state being run, will be used instead.

Handling Errors

The `ExceptionHandler` interface allows you to provide special handling when an exception occurs during log processing. `ExceptionHandler` declares a single callback method, `handleException()`, which receives a `ProcessingLibraryException` object with context about the exception that occurred.

You can use the passed-in `ProcessingLibraryException`'s `getStatus()` method to find out what operation was being executed when the exception occurred and get additional information about the status of the operation. `ProcessingLibraryException` is derived from Java's standard `Exception` class, so you can also retrieve information about the exception by invoking any of the `Exception` methods, as well.

Here is an example implementation:

```
public class SampleExceptionHandler implements ExceptionHandler{
    private static final Log logger =
        LoggerFactory.getLog(DefaultProgressReporter.class);

    @Override
    public void handleException(ProcessingLibraryException exception) {
        ProgressStatus status = exception.getStatus();
        ProgressState state = status.getProgressState();
        ProgressInfo info = status.getProgressInfo();

        System.err.println(String.format(
            "Exception. Progress State: %s. Progress Information: %s.", state, info));
    }
}
```

If you don't provide your own `ExceptionHandler`, then `DefaultExceptionHandler`, which simply prints a standard error message, will be used instead.

Additional Resources

For more information about the CloudTrail Processing Library, see the following additional resources:

- The [CloudTrail Processing Library](#) GitHub project, which includes [sample](#) code that demonstrates how to implement a CloudTrail Processing Library application
- The [CloudTrail Processing Library Java Package Documentation](#)

CloudTrail Event Reference

A CloudTrail log is a record in JSON format that contains information about requests for resources in your account such as who made the request, the services used, the actions performed, and parameters for the action. The event data is enclosed in a `Records` array.

The following example shows a single log record at the beginning of a log file. The entry shows that an IAM user named Alice called the CloudTrail `StartLogging` API from the CloudTrail console to start the logging process.

```
{
  "Records": [{
    "eventVersion": "1.01",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "AIDAJDPLRKL7UEXAMPLE",
      "arn": "arn:aws:iam:123456789012:user/Alice",
      "accountId": "123456789012",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "userName": "Alice",
      "sessionContext": {
        "attributes": {
          "mfaAuthenticated": "false",
          "creationDate": "2014-03-18T14:29:23Z"
        }
      }
    },
    "eventTime": "2014-03-18T14:30:07Z",
    "eventSource": "cloudtrail.amazonaws.com",
    "eventName": "StartLogging",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "72.21.198.64",
    "userAgent": "AWSConsole, aws-sdk-java/1.4.5 Linux/x.xx.fleetxen Java_Hot
Spot(TM)_64-Bit_Server_VM/xx",
    "requestParameters": {
      "name": "Default"
    },
    "responseElements": null,
    "requestID": "cdc73f9d-aea9-11e3-9d5a-835b769c0d9c",
  }
]
```

```
"eventID": "3074414d-c626-42aa-984b-68ff152d6ab7"  
},  
  ... additional entries ...  
]
```

The following topics list the data fields that CloudTrail captures for each AWS API call and sign-in event.

Topics

- [CloudTrail Record Contents \(p. 193\)](#)
- [CloudTrail userIdentity Element \(p. 196\)](#)
- [Non-API Events Captured by CloudTrail \(p. 200\)](#)

CloudTrail Record Contents

The body of the record contains fields that help you determine the requested action as well as when and where the request was made.

eventTime

The date and time the request was made, in coordinated universal time (UTC).

eventVersion

The version of the log event format. The current version is 1.04.

userIdentity

Information about the user that made a request. For more information, see [CloudTrail userIdentity Element \(p. 196\)](#).

eventSource

The service that the request was made to. This name is typically a short form of the service name without spaces plus `.amazonaws.com`. For example:

- AWS CloudFormation is `cloudformation.amazonaws.com`.
- Amazon EC2 is `ec2.amazonaws.com`.
- Amazon Simple Workflow Service is `swf.amazonaws.com`.

This convention has some exceptions. For example, the `eventSource` for Amazon CloudWatch is `monitoring.amazonaws.com`.

eventName

The requested action, which is one of the actions in the API for that service.

awsRegion

The AWS region that the request was made to, such as `us-east-1`. See [CloudTrail Supported Regions \(p. 19\)](#).

sourceIPAddress

The IP address that the request was made from. For actions that originate from the service console, the address reported is for the underlying customer resource, not the console web server. For services in AWS, only the DNS name is displayed.

userAgent

The agent through which the request was made, such as the AWS Management Console or an AWS SDK. For example:

- `aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5` – The request was made with the AWS CLI installed on Linux.
- `AWSConsole` – The request was made through the AWS Management Console.
- `aws-sdk-java` – The request was made through the AWS SDK for Java.
- `aws-sdk-ruby` – The request was made through the AWS SDK for Ruby.

Note

For events originated by AWS, this field is usually `aws-internal/#` where `#` is a number used for internal purposes.

errorCode

The AWS service error if the request returns an error.

errorMessage

If the request returns an error, the description of the error. This message includes messages for authorization failures. CloudTrail captures the message logged by the service in its exception handling. For an example, see [Error Code and Message Log Example \(p. 26\)](#).

Note

Some AWS services provide the `errorCode` and `errorMessage` as top-level fields in the event. Other AWS services provide error information as part of `responseElements`.

requestParameters

The parameters, if any, that were sent with the request. These parameters are documented in the API reference documentation for the appropriate AWS service.

responseElements

The response element for actions that make changes (create, update, or delete actions). If an action does not change state (for example, a request to get or list objects), this element is omitted. These actions are documented in the API reference documentation for the appropriate AWS service.

requestID

The value that identifies the request. The service being called generates this value.

Support for this field begins with `eventVersion 1.01`.

eventID

GUID generated by CloudTrail to uniquely identify each event. You can use this value to identify a single event. For example, you can use the ID as a primary key to retrieve log data from a searchable database.

Support for this field begins with `eventVersion 1.01`.

eventType

Identifies the type of event that generated the event record. This can be the one of the following values:

- `AwsApiCall` – An API was called.
- `AwsServiceEvent` – The service generated an event related to your trail. For example, this can occur when another account made a call with a resource that you own.

- `ConsoleSignin` – A user in your account (root, IAM, federated, SAML, or `SwitchRole`) signed in to the AWS Management Console.
Support for this field begins with `eventVersion 1.02`.

apiVersion

Identifies the API version associated with the `AwsApiCall` `eventType` value.

Support for this field begins with `eventVersion 1.02`.

recipientAccountID

Represents the account ID that received this event. The `recipientAccountID` may be different from the [CloudTrail userIdentity Element \(p. 196\)](#) `accountId`. This can occur in cross-account resource access. For example, if a KMS key, also known as a [customer master key \(CMK\)](#), was used by a separate account to call the [Encrypt API](#), the `accountId` and `recipientAccountID` values will be the same for the event delivered to the account that made the call, but the values will be different for the event that is delivered to the account that owns the CMK.

Support for this field begins with `eventVersion 1.02`.

sharedEventID

GUID generated by CloudTrail to uniquely identify CloudTrail events from the same AWS action that is sent to different AWS accounts.

For example, when an account uses a KMS key, also known as a [customer master key \(CMK\)](#), that belongs to another account, the account that used the CMK and the account that owns the CMK receive separate CloudTrail events for the same action. Each CloudTrail event delivered for this AWS action shares the same `sharedEventID`, but also has a unique `eventID` and `recipientAccountID`.

For more information, see [sharedEventID Example \(p. 195\)](#).

Note

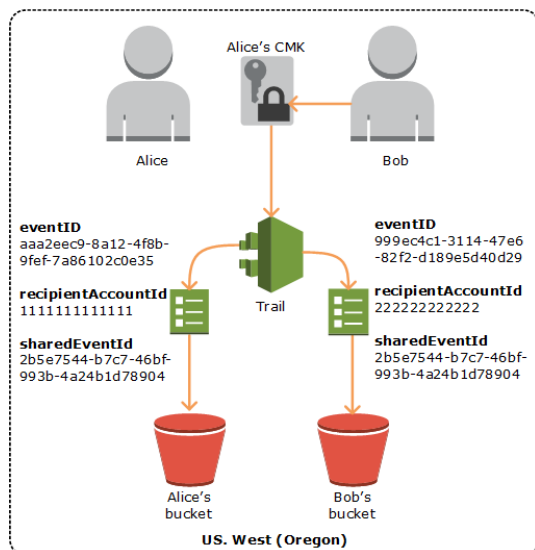
The `sharedEventID` field is present only when CloudTrail events are delivered to multiple accounts. If the caller and owner are the same AWS account, CloudTrail sends only one event, and the `sharedEventID` field is not present.

Support for this field begins with `eventVersion 1.03`.

sharedEventID Example

The following is an example that describes how CloudTrail delivers two events for the same action:

1. Alice has AWS account (111111111111) and creates a customer master key (CMK). She is the owner of this CMK.
2. Bob has AWS account (222222222222). Alice gives Bob permission to use the CMK.
3. Each account has a trail and a separate bucket.
4. Bob uses the CMK to call the `Encrypt` API.
5. CloudTrail sends two separate events.
 - One event is sent to Bob. The event shows that he used the CMK.
 - One event is sent to Alice. The event shows that Bob used the CMK.
 - The events have the same `sharedEventID`, but the `eventID` and `recipientAccountID` are unique.



CloudTrail userIdentity Element

AWS Identity and Access Management (IAM) provides different types of identities. The `userIdentity` element contains details about the type of IAM identity that made the request, and which credentials were used. If temporary credentials were used, the element shows how the credentials were obtained.

Contents

- [Examples \(p. 196\)](#)
- [Fields \(p. 197\)](#)
- [Values for AWS STS APIs with SAML and Web Identity Federation \(p. 199\)](#)

Examples

`userIdentity` with IAM user credentials

The following example shows the `userIdentity` element of a simple request made with the credentials of the IAM user named Alice.

```
"userIdentity": {  
  "type": "IAMUser",  
  "principalId": "AIDAJ45Q7YFFAREXAMPLE",  
  "arn": "arn:aws:iam::123456789012:user/Alice",  
  "accountId": "123456789012",  
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",  
  "userName": "Alice"  
}
```

`userIdentity` with temporary security credentials

The following example shows a `userIdentity` element for a request made with temporary security credentials obtained by assuming an IAM role. The element contains additional details about the role that was assumed to get credentials.


```

"userIdentity": {
  "type": "AssumedRole",
  "principalId": "AROAI DPPEZS35WEXAMPLE:AssumedRoleSessionName",
  "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/MySessionName",
  "accountId": "123456789012",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
    "attributes": {
      "creationDate": "20131102T010628Z",
      "mfaAuthenticated": "false"
    },
    "sessionIssuer": {
      "type": "Role",
      "principalId": "AROAI DPPEZS35WEXAMPLE",
      "arn": "arn:aws:iam::123456789012:role/RoleToBeAssumed",
      "accountId": "123456789012",
      "userName": "RoleToBeAssumed"
    }
  }
}

```

Fields

The following fields can appear in a `userIdentity` element.

`type`

The type of the identity. The following values are possible:

- `Root` – The request was made with your AWS account credentials. If the `userIdentity` type is `Root` and you set an alias for your account, the `userName` field contains your account alias. For more information, see [Your AWS Account ID and Its Alias](#).
- `IAMUser` – The request was made with the credentials of an IAM user.
- `AssumedRole` – The request was made with temporary security credentials that were obtained with a role via a call to the AWS Security Token Service (AWS STS) `AssumeRole` API. This can include [roles for Amazon EC2](#) and [cross-account API access](#).
- `FederatedUser` – The request was made with temporary security credentials that were obtained via a call to the AWS STS `GetFederationToken` API. The `sessionIssuer` element indicates if the API was called with root or IAM user credentials.

For more information about temporary security credentials, see [Using Temporary Security Credentials](#).

`userName`

The friendly name of the identity that made the call. The value that appears in `userName` is based on the value in `type`. The following table shows the relationship between `type` and `userName`:

<code>type</code>	<code>userName</code>	Description
<code>Root</code> (no alias set)	Not present	If you have not set up an alias for your AWS account, the <code>userName</code> field does not appear. For more information about account aliases, see Your AWS Account ID and Its Alias . Note that the <code>userName</code> field will never contain <code>Root</code> because <code>Root</code> is an identity type, not a user name.
<code>Root</code> (alias set)	The account alias	For more information about AWS account aliases, see Your AWS Account ID and Its Alias .

type	userName	Description
IAMUser	The user name of the IAM user	
Assumed-Role	Not present	For <code>AssumedRole</code> type, you can find the <code>userName</code> field in <code>sessionContext</code> , as part of the <code>sessionIssuer</code> (p. ?) element. For an example entry, see Examples (p. 196).
Federated-User	Not present	The <code>sessionContext</code> and <code>sessionIssuer</code> section contains information about the identity that issued the session for the federated user.

Note

The `userName` field contains the string `HIDDEN_DUE_TO_SECURITY_REASONS` when the recorded event is a console sign-in failure caused by incorrect user name input. CloudTrail does not record the contents in this case because the text could contain sensitive information, as in the following examples:

- A user accidentally types a password in the user name field.
- A user clicks the link for one AWS account's sign-in page, but then types the account number for a different one.
- A user accidentally types the account name of a personal email account, a bank sign-in identifier, or some other private ID.

principalId

A unique identifier for the entity that made the call. For requests made with temporary security credentials, this value includes the session name that is passed to the `AssumeRole`, `AssumeRoleWithWebIdentity`, or `GetFederationToken` API call.

arn

The Amazon Resource Name (ARN) of the principal that made the call. The last section of the `arn` contains the user or role that made the call.

accountId

The account that owns the entity that granted permissions for the request. If the request was made with temporary security credentials, this is the account that owns the IAM user or role that was used to obtain credentials.

accessKeyId

The access key ID that was used to sign the request. If the request was made with temporary security credentials, this is the access key ID of the temporary credentials.

sessionContext

If the request was made with temporary security credentials, an element that provides information about the session that was created for those credentials. Sessions are created when any API is called that returns temporary credentials. Sessions are also created when users work in the console and when users make a request with APIs that include [multi-factor authentication](#). Attributes for this element are:

- `creationDate` – The date and time when the temporary security credentials were issued. Represented in ISO 8601 basic notation.
- `mfaAuthenticated` – The value is `true` if the root user or IAM user whose credentials were used for the request also was authenticated with an MFA device; otherwise, `false`.

invokedBy

The name of the AWS service if that made the request, such as Auto Scaling or AWS Elastic Beanstalk.

sessionIssuer

If the request was made with temporary security credentials, an element that provides information about how the credentials were obtained. For example, if the temporary security credentials were obtained by assuming a role, this element provides information about the assumed role. If the

AWS CloudTrail User Guide
Values for AWS STS APIs with SAML and Web Identity Federation

credentials were obtained with root or IAM user credentials to call AWS STS `GetFederationToken`, the element provides information about the root account or IAM user. Attributes for this element are:

- `type` – The source of the temporary security credentials, such as `Root`, `IAMUser`, or `Role`.
- `userName` – The friendly name of the user or role that issued the session. The value that appears depends on the `sessionIssuer` identity `type`. The following table shows the relationship between `sessionIssuer` `type` and `userName`:

<code>sessionIssuer type</code>	<code>userName</code>	Description
Root (no alias set)	Not present	If you have not set up an alias for your account, the <code>userName</code> field does not appear. For more information about AWS account aliases, see Your AWS Account ID and Its Alias . Note that the <code>userName</code> field will never contain <code>Root</code> because <code>Root</code> is an identity <code>type</code> , not a user name.
Root (alias set)	The account alias	For more information about AWS account aliases, see Your AWS Account ID and Its Alias .
<code>IAMUser</code>	The user name of the IAM user	This also applies when a federated user is using a session issued by <code>IAMUser</code> .
<code>Role</code>	The role name	A role assumed by an IAM user, AWS service, or web identity federated user in a role session.

- `principalId` – The internal ID of the entity that was used to get credentials.
- `arn` – The ARN of the source (account, IAM user, or role) that was used to get temporary security credentials.
- `accountId` – The account that owns the entity that was used to get credentials.

webIdFederationData

If the request was made with temporary security credentials obtained by [web identity federation](#), an element that lists information about the identity provider. Attributes for this element are:

- `federatedProvider` – The principal name of the identity provider (for example, `www.amazon.com` for Login with Amazon or `accounts.google.com` for Google).
- `attributes` – The application ID and user ID as reported by the provider (for example, `www.amazon.com:app_id` and `www.amazon.com:user_id` for Login with Amazon). For more information, see [Available Keys for Web Identity Federation](#) in the *IAM User Guide*.

Values for AWS STS APIs with SAML and Web Identity Federation

AWS CloudTrail supports logging AWS Security Token Service (AWS STS) API calls made with Security Assertion Markup Language (SAML) and web identity federation. When a call is made to the [AssumeRoleWithSAML](#) and [AssumeRoleWithWebIdentity](#) APIs, CloudTrail records the call and delivers the event to your Amazon S3 bucket.

The `userIdentity` element for these APIs contains the following values.

type

The identity type.

- `SAMLUser` – The request was made with SAML assertion.
- `WebIdentityUser` – The request was made by a web identity federation provider.

principalId

A unique identifier for the entity that made the call.

- For `SAMLUser`, this is a combination of the `saml:namequalifier` and `saml:sub` keys.
- For `WebIdentityUser`, this is a combination of the issuer, application ID, and user ID.

userName

The name of the identity that made the call.

- For `SAMLUser`, this is the `saml:sub` key. See [Available Keys for SAML-Based Federation](#).
- For `WebIdentityUser`, this is the user ID. See [Available Keys for Web Identity Federation](#).

identityProvider

The principal name of the external identity provider. This field appears only for `SAMLUser` or `WebIdentityUser` types.

- For `SAMLUser`, this is the `saml:namequalifier` key for the SAML assertion.
- For `WebIdentityUser`, this is the issuer name of the web identity federation provider. This can be a provider that you configured, such as the following:
 - `cognito-identity.amazon.com` for Amazon Cognito
 - `www.amazon.com` for Login with Amazon
 - `accounts.google.com` for Google
 - `graph.facebook.com` for Facebook

The following is an example `userIdentity` element for the `AssumeRoleWithWebIdentity` action.

```
"userIdentity": {
  "type": "WebIdentityUser",
  "principalId": "accounts.google.com:application-id.apps.googleusercontent.com:user-id",
  "userName": "user-id",
  "identityProvider": "accounts.google.com"
}
```

For example logs of how the `userIdentity` element appears for `SAMLUser` and `WebIdentityUser` types, see [Logging IAM Events with AWS CloudTrail](#).

Non-API Events Captured by CloudTrail

In addition to logging AWS API calls, CloudTrail also captures other related events that might have a security or compliance impact on your AWS account or that might help you troubleshoot operational problems.

Topics

- [AWS Console Sign-in Events \(p. 200\)](#)

AWS Console Sign-in Events

CloudTrail records attempts to sign into the AWS Management Console, the AWS Discussion Forums and the AWS Support Center. All IAM user sign-in attempts (successes and failures), all federated user sign-in events (successes and failures) and all successful AWS root account sign-in attempts generate records in CloudTrail log files. Note, however, that CloudTrail does not record root sign-in failures.

The following record shows that an IAM user named Alice successfully signed into the AWS console without using multi-factor authentication.

```
{
  "Records": [
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAELOPP77CWZEXAMPLE",
        "arn": "arn:aws:iam::12345679012:user/alice",
        "accountId": "12345679012",
        "userName": "alice"
      },
      "eventTime": "2014-07-08T17:35:32Z",
      "eventSource": "signin.amazonaws.com",
      "eventName": "ConsoleLogin",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.4b)
Gecko/20030516 Mozilla Firebird/0.6",
      "requestParameters": null,
      "responseElements": {
        "ConsoleLogin": "Success"
      },
      "additionalEventData": {
        "MobileVersion": "No",
        "LoginTo": "https://console.aws.amazon.com/sns",
        "MFAUsed": "No"
      },
      "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb",
      "eventType": "AwsConsoleSignin"
    }
  ]
}
```

The following record shows that an IAM user named Alice logged into the AWS console by using multi-factor authentication.

```
{
  "Records": [
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAEZ7VBM6PDZEXAMPLE",
        "arn": "arn:aws:iam::12345679012:user/Alice",
        "accountId": "12345679012",
        "userName": "Alice"
      },
      "eventTime": "2014-07-08T17:36:03Z",
      "eventSource": "signin.amazonaws.com",
      "eventName": "ConsoleLogin",
      "awsRegion": "us-east-1",
```

```
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.4b)
Gecko/20030516 Mozilla Firebird/0.6",
    "requestParameters": null,
    "responseElements": {
      "ConsoleLogin": "Success"
    },
    "additionalEventData": {
      "MobileVersion": "Yes",
      "LoginTo": "https://console.aws.amazon.com/sns",
      "MFAUsed": "Yes"
    },
    "eventID": "5d2c2f55-3d1e-4336-b940-dbf8e66f588c",
    "eventType": "AwsConsoleSignin"
  }
]
}
```

The following record shows an unsuccessful AWS console sign-in attempt because of an authentication failure.

```
{
  "Records": [
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAELOPP77CWZEXAMPLE",
        "accountId": "12345679012",
        "accessKeyId": "",
        "userName": "alice"
      },
      "eventTime": "2014-07-08T17:35:27Z",
      "eventSource": "signin.amazonaws.com",
      "eventName": "ConsoleLogin",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.4b)
Gecko/20030516 Mozilla Firebird/0.6",
      "errorMessage": "Failed authentication",
      "requestParameters": null,
      "responseElements": {
        "ConsoleLogin": "Failure"
      },
      "additionalEventData": {
        "MobileVersion": "No",
        "LoginTo": "https://console.aws.amazon.com/sns",
        "MFAUsed": "No"
      },
      "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf",
      "eventType": "AwsConsoleSignin"
    }
  ]
}
```

CloudTrail Document History

The following table describes the documentation release history of AWS CloudTrail.

- **API version:** 2013-11-01
- **Latest documentation update:** July 7, 2016

Change	Description	Release Date
Added functionality and documentation	This release supports using the CloudTrail console to view resource types that are supported by AWS Config. For more information, see Viewing Resources Referenced with AWS Config (p. 47) .	July 7, 2016
Added service support	This release supports AWS Service Catalog. See Management Tools (p. 12) .	July 6, 2016
Added region support	This release supports one additional region: ap-south-1 (Asia Pacific (Mumbai)). See CloudTrail Supported Regions (p. 19) .	June 27, 2016
Added service support	This release supports CloudWatch Logs in the South America (São Paulo) Region. For more information, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs (p. 95) .	May 6, 2016
Added service support	This release supports AWS WAF. See Security and Identity (p. 14) .	April 28, 2016
Added service support	This release supports AWS Support. See Support (p. 16) .	April 21, 2016
Added service support	This release supports Amazon Inspector. See Security and Identity (p. 14) .	April 20, 2016
Added service support	This release supports AWS IoT. See Internet of Things (p. 12) .	April 11, 2016

Change	Description	Release Date
Added functionality and documentation	This release supports logging AWS Security Token Service (AWS STS) API calls made with Security Assertion Markup Language (SAML) and web identity federation. For more information, see Values for AWS STS APIs with SAML and Web Identity Federation (p. 199).	March 28, 2016
Added service support	This release supports AWS Certificate Manager. See Security and Identity (p. 14).	March 25, 2016
Added service support	This release supports Amazon Kinesis Firehose. See Analytics (p. 8).	March 17, 2016
Added service support	This release supports Amazon CloudWatch Logs. See Management Tools (p. 12).	March 10, 2016
Added service support	This release supports Amazon Cognito. See Mobile Services (p. 13).	February 18, 2016
Added service support	This release supports AWS Database Migration Service. See Database (p. 10).	February 4, 2016
Added service support	This release supports Amazon GameLift (GameLift). See Game Development (p. 12).	January 27, 2016
Added service support	This release supports Amazon CloudWatch Events. See Management Tools (p. 12).	January 16, 2016
Added region support	This release supports one additional region: ap-northeast-2 (Asia Pacific (Seoul)). See CloudTrail Supported Regions (p. 19).	January 6, 2016
Added service support	This release supports Amazon EC2 Container Registry (Amazon ECR). See Compute (p. 9).	December 21, 2015
Added functionality and documentation	This release supports turning on CloudTrail across all regions and support for multiple trails per region. For more information, see How Does CloudTrail Behave Regionally and Globally? (p. 5).	December 17, 2015
Added service support	This release supports Amazon Machine Learning. See Analytics (p. 8).	December 10, 2015
Added functionality and documentation	This release supports log file encryption, log file integrity validation, and tagging. For more information, see Encrypting CloudTrail Log Files with AWS KMS–Managed Keys (SSE-KMS) (p. 152), Validating CloudTrail Log File Integrity (p. 163), and Updating a Trail (p. 32).	October 1, 2015
Added service support	This release supports Amazon Elasticsearch Service. See Analytics (p. 8).	October 1, 2015
Added service support	This release supports Amazon S3 bucket level events. See Storage and Content Delivery (p. 15).	September 1, 2015
Added service support	This release supports AWS Device Farm. See Mobile Services (p. 13).	July 13, 2015
Added service support	This release supports Amazon API Gateway. See Application Services (p. 9).	July 9, 2015

Change	Description	Release Date
Added service support	This release supports AWS CodePipeline. See Developer Tools (p. 11) .	July 9, 2015
Added service support	This release supports Amazon DynamoDB. See Data-base (p. 10) .	May 28, 2015
Added service support	This release supports CloudWatch Logs in the US West (N. California) region. See the CloudTrail release notes . For more information about CloudTrail support for CloudWatch Logs monitoring, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs (p. 95) .	May 19, 2015
Added service support	This release supports AWS Directory Service. See Security and Identity (p. 14) .	May 14, 2015
Added service support	This release supports Amazon Simple Email Service (Amazon SES). See Application Services (p. 9) .	May 7, 2015
Added service support	This release supports Amazon EC2 Container Service (see Compute (p. 9)).	April 9, 2015
Added service support	This release supports AWS Lambda (see Compute (p. 9)).	April 9, 2015
Added service support	This release supports Amazon WorkSpaces. See Enterprise Applications (p. 11) .	April 9, 2015
Added functionality and documentation	This release supports the lookup of AWS activity captured by CloudTrail (CloudTrail events). You can look up and filter events in your account related to creation, modification, or deletion. To look up these events, you can use the CloudTrail console, the AWS Command Line Interface (AWS CLI), or the AWS SDK. For more information, see Viewing Events with CloudTrail API Activity History (p. 45) .	March 12, 2015
Added service support and new documentation	This release supports Amazon CloudWatch Logs in the Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), and EU (Frankfurt) regions. Additional CloudWatch alarm examples have been added to Creating CloudWatch Alarms for CloudTrail Events , and a new page has been added: Using a AWS CloudFormation Template to Create CloudWatch Alarms .	March 5, 2015
Added API support	This release supports Amazon EC2 Simple Systems Manager (SSM). SSM lets you configure, manage and easily deploy custom Windows instance configurations. For more information about SSM, see Managing Windows Instance Configuration . For information about the SSM API calls logged by CloudTrail, see Logging SSM API Calls Using AWS CloudTrail .	February 17, 2015
New documentation	A new section that describes CloudTrail support for AWS Security Token Service (AWS STS) regional endpoints has been added to the CloudTrail Concepts page.	February 17, 2015
Added service support	This release supports Amazon Route 53. See Networking (p. 14) .	February 11, 2015

Change	Description	Release Date
Added service support	This release supports AWS Config. See Management Tools (p. 12) .	February 10, 2015
Added service support	This release supports AWS CloudHSM. See Security and Identity (p. 14) .	January 8, 2015
Added service support	This release supports AWS CodeDeploy. See Developer Tools (p. 11) .	December 17, 2014
Added service support	This release supports AWS Storage Gateway. See Storage and Content Delivery (p. 15) .	December 16, 2014
Added region support	This release supports one additional region: us-gov-west-1 (AWS GovCloud (US)). See CloudTrail Supported Regions (p. 19) .	December 16, 2014
Added service support	This release supports Amazon Glacier. See Storage and Content Delivery (p. 15) .	December 11, 2014
Added service support	This release supports AWS Data Pipeline. See Analytics (p. 8) .	December 2, 2014
Added service support	This release supports AWS Key Management Service. See Security and Identity (p. 14) .	November 12, 2014
New documentation	A new section, Monitoring CloudTrail Log Files with Amazon CloudWatch Logs (p. 95) , has been added to the guide. It describes how to use Amazon CloudWatch Logs to monitor CloudTrail log events.	November 10, 2014
New documentation	A new section, Using the CloudTrail Processing Library (p. 184) , has been added to the guide. It provides information about how to write a CloudTrail log processor in Java using the AWS CloudTrail Processing Library.	November 5, 2014
Added service support	This release supports Amazon Elastic Transcoder. See Application Services (p. 9) .	October 27, 2014
Added region support	This release supports one additional region: eu-central-1 (EU (Frankfurt)). See CloudTrail Supported Regions (p. 19) .	October 23, 2014
Added service support	This release supports Amazon CloudSearch. See Application Services (p. 9) .	October 16, 2014
Added service support	This release supports Amazon Simple Notification Service. See Mobile Services (p. 13) .	October 09, 2014
Added service support	This release supports Amazon ElastiCache. See Database (p. 10) .	September 15, 2014
Added service support	This release supports Amazon WorkDocs. See Enterprise Applications (p. 11) .	August 27, 2014
Added new content	This release includes a topic that discusses logging sign-in events. See AWS Console Sign-in Events (p. 200) .	July 24, 2014

Change	Description	Release Date
Added new content	The eventVersion element for this release has been upgraded to version 1.02 and three new fields have been added. See CloudTrail Record Contents (p. 193) .	July 18, 2014
Added service support	This release supports Auto Scaling (see Compute (p. 9)) and Amazon SQS (see Application Services (p. 9)).	July 17, 2014
Added region support	This release supports three additional regions: ap-south-east-1 (Asia Pacific (Singapore)), ap-northeast-1 (Asia Pacific (Tokyo)), sa-east-1 (South America (São Paulo)). See CloudTrail Supported Regions (p. 19) .	June 30, 2014
Additional service support	This release supports Amazon Redshift. See Analytics (p. 8) .	June 10, 2014
Added service support	This release supports AWS OpsWorks. See Management Tools (p. 12) .	June 5, 2014
Added service support	This release supports Amazon CloudFront. See Storage and Content Delivery (p. 15) .	May 28, 2014
Added region support	This release supports three additional regions: us-west-1 (US West (N. California)), eu-west-1 (EU (Ireland)), ap-southeast-2 (Asia Pacific (Sydney)). See CloudTrail Supported Regions (p. 19) .	May 13, 2014
Added service support	This release supports Amazon Simple Workflow Service. See Application Services (p. 9) .	May 9, 2014
Added new content	This release includes topics that discuss sharing log files between accounts. See Sharing CloudTrail Log Files Between AWS Accounts (p. 142) .	May 2, 2014
Added service support	This release supports Amazon CloudWatch. See Management Tools (p. 12) .	April 28, 2014
Added service support	This release supports Amazon Kinesis. See Analytics (p. 8) .	April 22, 2014
Added service support	This release supports AWS Direct Connect. See Networking (p. 14) .	April 11, 2014
Added service support	This release supports Amazon EMR. See Analytics (p. 8) .	April 4, 2014
Added service support	This release supports Elastic Beanstalk. See Compute (p. 9) .	April 2, 2014
Additional service support	This release supports AWS CloudFormation. See Management Tools (p. 12) .	March 7, 2014
New guide	This release introduces AWS CloudTrail.	November 13, 2013