
AWS CloudTrail

User Guide

Version 1.0



AWS CloudTrail: User Guide

Copyright © 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 CloudTrail works | 1 |
| CloudTrail workflow | 3 |
| CloudTrail concepts | 4 |
| What are CloudTrail events? | 5 |
| What is CloudTrail event history? | 6 |
| What are trails? | 7 |
| What are organization trails? | 7 |
| How do you manage CloudTrail? | 7 |
| How do you control access to CloudTrail? | 8 |
| How do you log management and data events? | 8 |
| How do you log CloudTrail Insights events? | 8 |
| How do you perform monitoring with CloudTrail? | 9 |
| How does CloudTrail behave regionally and globally? | 9 |
| Global service events | 10 |
| How does CloudTrail relate to other AWS monitoring services? | 11 |
| Partner solutions | 12 |
| CloudTrail supported regions | 12 |
| CloudTrail log file examples | 14 |
| CloudTrail log file name format | 14 |
| Log file examples | 14 |
| CloudTrail supported services and integrations | 20 |
| AWS service integrations with CloudTrail Logs | 20 |
| CloudTrail integration with AWS Organizations | 21 |
| AWS service topics for CloudTrail | 21 |
| CloudTrail unsupported services | 31 |
| Quotas in AWS CloudTrail | 32 |
| CloudTrail tutorial | 35 |
| Prerequisites | 35 |
| Step 1: Review AWS account activity in event history | 35 |
| Step 2: Create your first trail | 37 |
| Step 3: View your log files | 40 |
| Step 4: Plan for next steps | 42 |
| Working with CloudTrail | 44 |
| Viewing events with CloudTrail Event history | 44 |
| Viewing CloudTrail events in the CloudTrail console | 45 |
| Viewing CloudTrail events with the AWS CLI | 49 |
| Viewing CloudTrail Insights events | 55 |
| Viewing CloudTrail Insights events in the CloudTrail console | 56 |
| Viewing CloudTrail Insights events with the AWS CLI | 61 |
| Creating a trail for your AWS account | 69 |
| Creating and updating a trail with the console | 69 |
| Creating, updating, and managing trails with the AWS Command Line Interface | 87 |
| Creating a trail for an organization | 112 |
| Event history and organization trails | 114 |
| Best practices for moving from member account trails to organization trails | 114 |
| Prepare for creating a trail for your organization | 114 |
| Creating a trail for your organization in the console | 116 |
| Creating a trail for an organization with the AWS Command Line Interface | 121 |
| Getting and viewing your CloudTrail log files | 124 |
| Finding your CloudTrail log files | 125 |
| Downloading your CloudTrail log files | 126 |
| Configuring Amazon SNS notifications for CloudTrail | 127 |
| Configuring CloudTrail to send notifications | 127 |

| | |
|---|-----|
| Controlling user permissions for CloudTrail | 128 |
| Tips for managing trails | 129 |
| Managing CloudTrail costs | 129 |
| CloudTrail trail naming requirements | 130 |
| Amazon S3 bucket naming requirements | 131 |
| AWS KMS alias naming requirements | 131 |
| Using AWS CloudTrail with interface VPC endpoints | 131 |
| Availability | 132 |
| Create a VPC endpoint for CloudTrail | 132 |
| Working with CloudTrail log files | 133 |
| Create multiple trails | 134 |
| Logging management events for trails | 136 |
| Management events | 136 |
| Read and write events | 137 |
| Logging events with the AWS Command Line Interface | 138 |
| Logging events with the AWS SDKs | 140 |
| Sending events to Amazon CloudWatch Logs | 140 |
| Logging data events for trails | 140 |
| Data events | 140 |
| Read-only and write-only events | 148 |
| Logging data events with the AWS Command Line Interface | 149 |
| Logging data events for AWS Config compliance | 153 |
| Logging events with the AWS SDKs | 153 |
| Sending events to Amazon CloudWatch Logs | 153 |
| Logging Insights events for trails | 154 |
| Understanding CloudTrail Insights | 154 |
| Logging Insights events with the AWS Management Console | 157 |
| Logging Insights events with the AWS Command Line Interface | 157 |
| Logging events with the AWS SDKs | 158 |
| Sending events to Amazon CloudWatch Logs | 158 |
| Receiving CloudTrail log files from multiple regions | 158 |
| Monitoring CloudTrail log files with Amazon CloudWatch Logs | 159 |
| Sending events to CloudWatch Logs | 160 |
| Creating CloudWatch alarms with an AWS CloudFormation template | 164 |
| Creating CloudWatch alarms for CloudTrail events: examples | 176 |
| Configuring notifications for CloudWatch Logs alarms | 181 |
| Stopping CloudTrail from sending events to CloudWatch Logs | 181 |
| CloudWatch log group and log stream naming for CloudTrail | 182 |
| Role policy document for CloudTrail to use CloudWatch Logs for monitoring | 182 |
| Receiving CloudTrail log files from multiple accounts | 183 |
| Setting bucket policy for multiple accounts | 184 |
| Turning on CloudTrail in additional accounts | 185 |
| Sharing CloudTrail log files between AWS accounts | 186 |
| Scenario 1: Granting access to the account that generated the log files | 187 |
| Scenario 2: Granting access to all logs | 188 |
| Creating a role | 189 |
| Creating an access policy to grant access to accounts you own | 191 |
| Creating an access policy to grant access to a third party | 192 |
| Assuming a role | 193 |
| Stop sharing CloudTrail log files between AWS accounts | 195 |
| Validating CloudTrail log file integrity | 196 |
| Why use it? | 196 |
| How it works | 196 |
| Enabling log file integrity validation for CloudTrail | 197 |
| Validating CloudTrail log file integrity with the AWS CLI | 197 |
| CloudTrail digest file structure | 202 |
| Custom implementations of CloudTrail log file integrity validation | 207 |

| | |
|--|-----|
| Using the CloudTrail Processing Library | 215 |
| Minimum requirements | 215 |
| Processing CloudTrail logs | 215 |
| Advanced topics | 219 |
| Additional resources | 223 |
| Security | 224 |
| Data protection | 224 |
| Identity and Access Management | 225 |
| Audience | 226 |
| Authenticating with identities | 226 |
| Managing access using policies | 228 |
| How AWS CloudTrail works with IAM | 229 |
| Identity-based policy examples | 233 |
| Amazon S3 bucket policy for CloudTrail | 243 |
| Amazon SNS topic policy for CloudTrail | 246 |
| Troubleshooting | 251 |
| Using service-linked roles | 252 |
| Compliance validation | 254 |
| Resilience | 255 |
| Infrastructure security | 255 |
| Security best practices | 255 |
| CloudTrail detective security best practices | 256 |
| CloudTrail preventative security best practices | 257 |
| Encrypting CloudTrail log files with AWS KMS–managed keys (SSE-KMS) | 259 |
| Enabling log file encryption | 260 |
| Granting permissions to create a KMS key | 260 |
| Configure AWS KMS key policies for CloudTrail | 261 |
| Updating a trail to use your KMS key | 267 |
| Enabling and disabling CloudTrail log file encryption with the AWS CLI | 268 |
| CloudTrail log event reference | 270 |
| CloudTrail record contents | 272 |
| Record fields for Insights events | 279 |
| Example sharedEventID | 280 |
| CloudTrail userIdentity element | 281 |
| Examples | 281 |
| Fields | 282 |
| Values for AWS STS APIs with SAML and web identity federation | 285 |
| AWS STS source identity | 286 |
| Insights insightDetails element | 288 |
| Example insightDetails block | 292 |
| Non-API events captured by CloudTrail | 294 |
| AWS service events | 294 |
| AWS Management Console sign-in events | 295 |
| Document history | 301 |
| Earlier updates | 312 |
| AWS glossary | 324 |

What Is AWS CloudTrail?

AWS CloudTrail is an AWS service that helps you enable governance, compliance, and operational and risk auditing of your AWS account. Actions taken by a user, role, or an AWS service are recorded as events in CloudTrail. Events include actions taken in the AWS Management Console, AWS Command Line Interface, and AWS SDKs and APIs.

CloudTrail is enabled on your AWS account when you create it. When activity occurs in your AWS account, that activity is recorded in a CloudTrail event. You can easily view recent events in the CloudTrail console by going to Event history. For an ongoing record of activity and events in your AWS account, [create a trail \(p. 70\)](#). For more information about CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Visibility into your AWS account activity is a key aspect of security and operational best practices. You can use CloudTrail to view, search, download, archive, analyze, and respond to account activity across your AWS infrastructure. You can identify who or what took which action, what resources were acted upon, when the event occurred, and other details to help you analyze and respond to activity in your AWS account. Optionally, you can enable AWS CloudTrail Insights on a trail to help you identify and respond to unusual activity.

You can integrate CloudTrail into applications using the API, automate trail creation for your organization, check the status of trails you create, and control how users view CloudTrail events.

Topics

- [How CloudTrail works \(p. 1\)](#)
- [CloudTrail workflow \(p. 3\)](#)
- [CloudTrail concepts \(p. 4\)](#)
- [CloudTrail supported regions \(p. 12\)](#)
- [CloudTrail log file examples \(p. 14\)](#)
- [CloudTrail supported services and integrations \(p. 20\)](#)
- [Quotas in AWS CloudTrail \(p. 32\)](#)

How CloudTrail works

CloudTrail is enabled on your AWS account when you create it. When activity occurs in your AWS account, that activity is recorded in a CloudTrail event. You can easily view events in the CloudTrail console by going to **Event history**.

Event history allows you to view, search, and download the past 90 days of activity in your AWS account. In addition, you can create a CloudTrail trail to archive, analyze, and respond to changes in your AWS resources. A trail is a configuration that enables delivery of events to an Amazon S3 bucket that you specify. You can also deliver and analyze events in a trail with Amazon CloudWatch Logs and Amazon CloudWatch Events. You can create a trail with the CloudTrail console, the AWS CLI, or the CloudTrail API.

You can create two types of trails for an AWS account:

A trail that applies to all regions

When you create a trail that applies to all regions, CloudTrail records events in each region and delivers the CloudTrail event log files to an S3 bucket that you specify. If a region is added after

you create a trail that applies to all regions, that new region is automatically included, and events in that region are logged. Because creating a trail in all regions is a recommended best practice, so you capture activity in all regions in your account, an all-regions trail is the default option when you create a trail in the CloudTrail console. You can only update a single-region trail to log all regions by using the AWS CLI. For more information, see [Creating a trail in the console \(basic event selectors\)](#) (p. 70).

A trail that applies to one region

When you create a trail that applies to one region, CloudTrail records the events in that region only. It then delivers the CloudTrail event log files to an Amazon S3 bucket that you specify. You can only create a single-region trail by using the AWS CLI. If you create additional single trails, you can have those trails deliver CloudTrail event log files to the same Amazon S3 bucket or to separate buckets. This is the default option when you create a trail using the AWS CLI or the CloudTrail API. For more information, see [Creating, updating, and managing trails with the AWS Command Line Interface](#) (p. 87).

Note

For both types of trails, you can specify an Amazon S3 bucket from any region.

Beginning on April 12, 2019, trails will be viewable only in the AWS Regions where they log events. If you create a trail that logs events in all AWS Regions, it will appear in the console in all AWS Regions. If you create a trail that only logs events in a single AWS Region, you can view and manage it only in that AWS Region.

If you have created an organization in AWS Organizations, you can also create a trail that will log all events for all AWS accounts in that organization. This is referred to as an *organization trail*. Organization trails can apply to all AWS Regions or one Region. Organization trails must be created in the management account, and when specified as applying to an organization, are automatically applied to all member accounts in the organization. Member accounts will be able to see the organization trail, but cannot modify or delete it. By default, member accounts will not have access to the log files for the organization trail in the Amazon S3 bucket.

You can change the configuration of a trail after you create it, including whether it logs events in one region or all regions. To change a single-region trail to an all-region trail, or vice-versa, you must run the AWS CLI [update-trail](#) (p. 90) command. You can also change whether it logs data or CloudTrail Insights events. Changing whether a trail logs events in one region or in all regions affects which events are logged. For more information, see [Managing trails with the AWS CLI](#) (p. 93) (AWS CLI), and [Working with CloudTrail log files](#) (p. 133).

By default, CloudTrail event log files are encrypted using Amazon S3 server-side encryption (SSE). You can also choose to encrypt your log files with an AWS Key Management Service (AWS KMS) key. You can store your log files in your bucket for as long as you want. You can also define Amazon S3 lifecycle rules to archive or delete log files automatically. If you want notifications about log file delivery and validation, you can set up Amazon SNS notifications.

CloudTrail publishes log files multiple times an hour, about every five minutes. These log files contain API calls from services in the account that support CloudTrail. For more information, see [CloudTrail supported services and integrations](#) (p. 20).

Note

CloudTrail typically delivers logs within an average of about 15 minutes of an API call. This time is not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information. CloudTrail captures actions made directly by the user or on behalf of the user by an AWS service. 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. You can identify if the action was taken by an AWS service with the `invokedby` field in the CloudTrail event.

To get started with CloudTrail, see [Getting started with AWS CloudTrail tutorial \(p. 35\)](#).

For CloudTrail pricing, see [AWS CloudTrail Pricing](#). For Amazon S3 and Amazon SNS pricing, see [Amazon S3 Pricing](#) and [Amazon SNS Pricing](#).

CloudTrail workflow

View event history for your AWS account

You can view and search the last 90 days of events recorded by CloudTrail in the CloudTrail console or by using the AWS CLI. For more information, see [Viewing events with CloudTrail Event history \(p. 44\)](#).

Download events

You can download a CSV or JSON file containing up to the past 90 days of CloudTrail events for your AWS account. For more information, see [Downloading events \(p. 48\)](#) or [Downloading Insights events \(p. 61\)](#).

Create a trail

A trail enables CloudTrail to deliver log files to your Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all regions. The trail logs events from all regions in the AWS partition and delivers the log files to the S3 bucket that you specify. For more information, see [Creating a trail for your AWS account \(p. 69\)](#).

Create and subscribe to an Amazon SNS topic

Subscribe to a topic to receive notifications about log file delivery to your bucket. Amazon SNS can notify you in multiple ways, including programmatically with Amazon Simple Queue Service. For information, see [Configuring Amazon SNS notifications for CloudTrail \(p. 127\)](#).

Note

If you want to receive SNS notifications about log file deliveries from all regions, specify only one SNS topic for your trail. If you want to programmatically process all events, see [Using the CloudTrail Processing Library \(p. 215\)](#).

View your log files

Use Amazon S3 to retrieve log files. For information, see [Getting and viewing your CloudTrail log files \(p. 124\)](#).

Manage user permissions

Use AWS Identity and Access Management (IAM) to manage which users have permissions to create, configure, or delete trails; start and stop logging; and access buckets that have log files. For more information, see [Controlling user permissions for CloudTrail \(p. 128\)](#).

Monitor events with CloudWatch Logs

You can configure your trail to send events to CloudWatch Logs. You can then use CloudWatch Logs to monitor your account for specific API calls and events. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).

Note

If you configure a trail that applies to all regions to send events to a CloudWatch Logs log group, CloudTrail sends events from all regions to a single log group.

Log management and data events

Configure your trails to log read-only, write-only, or all management and data events. By default, trails log management events. For more information, see [Working with CloudTrail log files \(p. 133\)](#).

Log CloudTrail Insights events

Configure your trails to log Insights events to help you identify and respond to unusual activity associated with `write` management API calls. If your trail is configured to log read-only or no management events, you cannot turn on CloudTrail Insights event logging. For more information, see [Logging Insights events for trails \(p. 154\)](#).

Enable log encryption

Log file encryption provides an extra layer of security for your log files. For more information, see [Encrypting CloudTrail log files with AWS KMS-managed keys \(SSE-KMS\) \(p. 259\)](#).

Enable log file integrity

Log file integrity validation helps you verify that log files have remained unchanged since CloudTrail delivered them. For more information, see [Validating CloudTrail log file integrity \(p. 196\)](#).

Share log files with other AWS accounts

You can share log files between accounts. For more information, see [Sharing CloudTrail log files between AWS accounts \(p. 186\)](#).

Aggregate logs from multiple accounts

You can aggregate log files from multiple accounts to a single bucket. For more information, see [Receiving CloudTrail log files from multiple accounts \(p. 183\)](#).

Work with partner solutions

Analyze your CloudTrail output with a partner solution that integrates with CloudTrail. Partner solutions offer a broad set of capabilities, such as change tracking, troubleshooting, and security analysis. For more information, see the [AWS CloudTrail](#) partner page.

CloudTrail concepts

This section summarizes basic concepts related to CloudTrail.

Contents

- [What are CloudTrail events? \(p. 5\)](#)
 - [What are management events? \(p. 5\)](#)
 - [What are data events? \(p. 5\)](#)
 - [What are Insights events? \(p. 6\)](#)
- [What is CloudTrail event history? \(p. 6\)](#)
- [What are trails? \(p. 7\)](#)
- [What are organization trails? \(p. 7\)](#)
- [How do you manage CloudTrail? \(p. 7\)](#)
 - [CloudTrail console \(p. 7\)](#)
 - [CloudTrail CLI \(p. 8\)](#)
 - [CloudTrail APIs \(p. 8\)](#)
 - [AWS SDKs \(p. 8\)](#)
 - [Why use tags for trails? \(p. 8\)](#)
- [How do you control access to CloudTrail? \(p. 8\)](#)
- [How do you log management and data events? \(p. 8\)](#)
- [How do you log CloudTrail Insights events? \(p. 8\)](#)
- [How do you perform monitoring with CloudTrail? \(p. 9\)](#)

- [CloudWatch Logs, CloudWatch Events, and CloudTrail \(p. 9\)](#)
- [How does CloudTrail behave regionally and globally? \(p. 9\)](#)
 - [What are the advantages of applying a trail to all Regions? \(p. 9\)](#)
 - [What happens when you apply a trail to all Regions? \(p. 10\)](#)
 - [Multiple trails per Region \(p. 10\)](#)
 - [AWS Security Token Service and CloudTrail \(p. 10\)](#)
- [Global service events \(p. 10\)](#)
- [How does CloudTrail relate to other AWS monitoring services? \(p. 11\)](#)
- [Partner solutions \(p. 12\)](#)

What are CloudTrail events?

An event in CloudTrail is the record of an activity in an AWS account. This activity can be an action taken by a user, role, or service that is monitorable by CloudTrail. CloudTrail events provide a history of both API and non-API account activity made through the AWS Management Console, AWS SDKs, command line tools, and other AWS services. There are three types of events that can be logged in CloudTrail: management events, data events, and CloudTrail Insights events. By default, trails log management events, but not data or Insights events.

All event types use the same CloudTrail JSON log format.

Note

CloudTrail does not log all AWS services and all events. For more information about which APIs are logged for a specific service, see documentation for that service in [CloudTrail supported services and integrations \(p. 20\)](#).

What are management events?

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as *control plane operations*. Example management events include:

- Configuring security (for example, AWS Identity and Access Management `AttachRolePolicy` API operations).
- Registering devices (for example, Amazon EC2 `CreateDefaultVpc` API operations).
- Configuring rules for routing data (for example, Amazon EC2 `CreateSubnet` API operations).
- Setting up logging (for example, AWS CloudTrail `CreateTrail` API operations).

Management events can also include non-API events that occur in your account. For example, when a user signs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see [Non-API events captured by CloudTrail \(p. 294\)](#). For a list of management events that CloudTrail logs for AWS services, see [CloudTrail supported services and integrations \(p. 20\)](#).

What are data events?

Data events provide information about the resource operations performed on or in a resource. These are also known as *data plane operations*. Data events are often high-volume activities. The following data types are recorded:

- Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on buckets and objects in buckets
- AWS Lambda function execution activity (the `Invoke` API)

- Amazon DynamoDB object-level API activity on tables (for example, `PutItem`, `DeleteItem`, and `UpdateItem` API operations)
- Amazon S3 on Outposts object-level API activity
- Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, such as `eth_getBalance` or `eth_getBlockByNumber`
- Amazon S3 Object Lambda access points API activity, such as calls to `CompleteMultipartUpload` and `GetObject`
- Amazon Elastic Block Store (EBS) direct APIs, such as `PutSnapshotBlock`, `GetSnapshotBlock`, and `ListChangedBlocks` on Amazon EBS snapshots
- Amazon S3 API activity on access points
- Amazon DynamoDB API activity on streams

Data events are not logged by default when you create a trail. To record CloudTrail data events, you must explicitly add to a trail the supported resources or resource types for which you want to collect activity. For more information, see [Creating a trail \(p. 70\)](#) and [Data events \(p. 140\)](#).

Additional charges apply for logging data events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

What are Insights events?

CloudTrail Insights events capture unusual activity in your AWS account. If you have Insights events enabled, and CloudTrail detects unusual activity, Insights events are logged to a different folder or prefix in the destination S3 bucket for your trail. You can also see the type of insight and the incident time period when you view Insights events on the CloudTrail console. Insights events provide relevant information, such as the associated API, incident time, and statistics, that help you understand and act on unusual activity. Unlike other types of events captured in a CloudTrail trail, Insights events are logged only when CloudTrail detects changes in your account's API usage that differ significantly from the account's typical usage patterns. Examples of activity that might generate Insights events include:

- Your account typically logs no more than 20 Amazon S3 `deleteBucket` API calls per minute, but your account starts to log an average of 100 `deleteBucket` API calls per minute. An Insights event is logged at the start of the unusual activity, and another Insights event is logged to mark the end of the unusual activity.
- Your account typically logs 20 calls per minute to the Amazon EC2 `AuthorizeSecurityGroupIngress` API, but your account starts to log zero calls to `AuthorizeSecurityGroupIngress`. An Insights event is logged at the start of the unusual activity, and ten minutes later, when the unusual activity ends, another Insights event is logged to mark the end of the unusual activity.

These examples are provided for illustration purposes only. Your results may vary depending on your use case.

Insights events are disabled by default when you create a trail. To record CloudTrail Insights events, you must explicitly enable Insights event collection on a new or existing trail. For more information, see [Creating a trail \(p. 70\)](#) and [Logging Insights events for trails \(p. 154\)](#).

Additional charges apply for logging CloudTrail Insights events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

What is CloudTrail event history?

CloudTrail event history provides a viewable, searchable, and downloadable record of the past 90 days of CloudTrail events. You can use this history to gain visibility into actions taken in your AWS account in the AWS Management Console, AWS SDKs, command line tools, and other AWS services. You can customize

your view of event history in the CloudTrail console by selecting which columns are displayed. For more information, see [Viewing events with CloudTrail Event history \(p. 44\)](#).

What are trails?

A trail is a configuration that enables delivery of CloudTrail events to an Amazon S3 bucket, CloudWatch Logs, and CloudWatch Events. You can use a trail to filter the CloudTrail events you want delivered, encrypt your CloudTrail event log files with an AWS KMS key, and set up Amazon SNS notifications for log file delivery. For more information about how to create and manage a trail, see [Creating a trail for your AWS account \(p. 69\)](#).

What are organization trails?

An organization trail is a configuration that enables delivery of CloudTrail events in the management account and all member accounts in an AWS Organizations organization to the same Amazon S3 bucket, CloudWatch Logs, and CloudWatch Events. Creating an organization trail helps you define a uniform event logging strategy for your organization.

When you create an organization trail, a trail with the name that you give it will be created in every AWS account that belongs to your organization. Users with CloudTrail permissions in member accounts will be able to see this trail (including the trail ARN) when they log into the AWS CloudTrail console from their AWS accounts, or when they run AWS CLI commands such as `describe-trails` (although member accounts must use the ARN for the organization trail, and not the name, when using the AWS CLI). However, users in member accounts will not have sufficient permissions to delete the organization trail, turn logging on or off, change what types of events are logged, or otherwise alter the organization trail in any way. For more information about AWS Organizations, see [Organizations Terminology and Concepts](#). For more information about creating and working with organization trails, see [Creating a trail for an organization \(p. 112\)](#).

How do you manage CloudTrail?

CloudTrail console

You can use and manage the CloudTrail service with the AWS CloudTrail console. The console provides a user interface for performing many CloudTrail tasks such as:

- Viewing recent events and event history for your AWS account.
- Downloading a filtered or complete file of the last 90 days of events.
- Creating and editing CloudTrail trails.
- Configuring CloudTrail trails, including:
 - Selecting an Amazon S3 bucket.
 - Setting a prefix.
 - Configuring delivery to CloudWatch Logs.
 - Using AWS KMS keys for encryption.
 - Enabling Amazon SNS notifications for log file delivery.
 - Adding and managing tags for your trails.

Beginning on April 12, 2019, trails will be viewable only in the AWS Regions where they log events. If you create a trail that logs events in all AWS Regions, it will appear in the console in all AWS Regions. If you create a trail that only logs events in a single AWS Region, you can view and manage it only in that AWS Region.

For more information about the AWS Management Console, see [AWS Management Console](#).

CloudTrail CLI

The AWS Command Line Interface is a unified tool that you can use to interact with CloudTrail from the command line. For more information, see the [AWS Command Line Interface User Guide](#). For a complete list of 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, you can use the SDKs to sign requests cryptographically, manage errors, and retry requests automatically. For more information, see the [Tools for Amazon Web Services](#) page.

Why use tags for trails?

A tag is a customer-defined key and optional value that can be assigned to AWS resources, such as CloudTrail trails, Amazon S3 buckets used to store CloudTrail log files, AWS Organizations organizations and organizational units, and many more. By adding the same tags to trails and to the Amazon S3 buckets you use to store log files for trails, you can make it easier to manage, search for, and filter these resources with [AWS Resource Groups](#). You can implement tagging strategies to help you consistently, effectively, and easily find and manage your resources. For more information, see [AWS Tagging Strategies](#).

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. Use IAM to create individual users for anyone who needs access to AWS CloudTrail. Create an IAM user for yourself, give that IAM user administrative privileges, and use that IAM user for all of your work. By creating individual IAM users for people accessing your account, you can give each IAM user a unique set of security credentials. You can also grant different permissions to each IAM user. If necessary, you can change or revoke an IAM user's permissions at any time. For more information, see [Controlling user permissions for CloudTrail \(p. 128\)](#).

How do you log management and data events?

By default, trails log all management events for your AWS account and don't include data events. You can choose to create or update trails to log data events. Only events that match your trail settings are delivered to your Amazon S3 bucket, and optionally to an Amazon CloudWatch Logs log group. If the event doesn't match the settings for a trail, the trail doesn't log the event. For more information, see [Working with CloudTrail log files \(p. 133\)](#).

How do you log CloudTrail Insights events?

AWS CloudTrail Insights helps AWS users identify and respond to unusual volumes of API calls by continuously analyzing CloudTrail management events. An Insights event is a record of unusual levels of write management API activity. The details page of an Insights event shows the event as a graph of unusual activity, and shows the start and end times of the unusual activity, along with the baseline that is used to determine whether the activity is unusual. By default, trails don't log CloudTrail Insights events. In the console, you can choose to log Insights events when you create or update a trail. When

you use the CloudTrail API, you can log Insights events by editing the settings of an existing trail with the **PutInsightSelectors** API. Additional charges apply for logging CloudTrail Insights events. For more information, see [Logging Insights events for trails \(p. 154\)](#) and [AWS CloudTrail Pricing](#).

How do you perform monitoring with CloudTrail?

CloudWatch Logs, CloudWatch Events, and CloudTrail

Amazon CloudWatch is a web service that collects and tracks metrics to monitor 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 \(p. 159\)](#).

Amazon CloudWatch Events is an AWS service that delivers a near real-time stream of system events that describe changes in AWS resources. In CloudWatch Events, you can create rules that trigger on any event recorded by CloudTrail. For more information, see [Creating a CloudWatch Events Rule That Triggers on an AWS API Call Using AWS CloudTrail](#).

Insights events are integrated with CloudWatch. You can deliver events that you are subscribed to on your trail, including Insights events, to CloudWatch Events and CloudWatch Logs. To configure CloudWatch Events with the CloudWatch console or API, choose the **AWS Insight via CloudTrail** event type on the **Create rule** page in the CloudWatch console.

Sending data that is logged by CloudTrail to CloudWatch Logs or CloudWatch Events requires that you have at least one trail. For more information about how to create a trail, see [Creating a Trail \(p. 70\)](#).

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 delivery of CloudTrail event log files to an Amazon S3 bucket. 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 AWS Regions has the following advantages:

- The configuration settings for the trail apply consistently across all AWS Regions.
- You receive CloudTrail events from all AWS Regions in a single Amazon S3 bucket and, optionally, in a CloudWatch Logs log group.
- You manage trail configuration for all AWS Regions from one location.
- You immediately receive events from a new AWS Region. When a new AWS Region is launched, CloudTrail automatically creates a copy of all of your Region trails for you in the new Region with the same settings as your original trail.
- You don't need to create trails in AWS Regions that you don't use often in order to monitor for unusual activity. Any activity in any AWS Region is logged in a trail that applies to all AWS Regions.

What happens when you apply a trail to all Regions?

When you apply a trail to all AWS Regions, CloudTrail uses the trail that you create in a particular Region to create trails with identical configurations in all other Regions in your account.

This has the following effects:

- CloudTrail delivers log files for account activity from all AWS 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 AWS Regions are sent to that single SNS topic.
- If you enabled it, log file integrity validation is enabled for the trail in all AWS Regions. For information, see [Validating CloudTrail log file integrity \(p. 196\)](#).

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 AWS Regions counts as one trail in every Region.

The following example is a Region with five trails:

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

Trails appear in the AWS Region where they exist. Trails that log events in all AWS Regions appear in every Region. You can view a list of trails in an AWS Region in the **Trails** page of the CloudTrail console. For more information, see [Updating a trail \(p. 79\)](#). For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

AWS Security Token Service and CloudTrail

AWS Security Token Service (AWS STS) is a service that has a global endpoint and 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 and Deactivating AWS STS in an AWS Region](#) in the *IAM User Guide*.

For a complete list of AWS regional endpoints, see [AWS Regions and Endpoints](#) in the *AWS General Reference*. For details about events from the global AWS STS endpoint, see [Global service events \(p. 10\)](#).

Global service events

Important

As of November 22, 2021, AWS CloudTrail will change how trails can be used to capture global service events. After the change, events created by CloudFront, IAM, and AWS STS will be recorded in the region in which they were created, the US East (N. Virginia) region, us-east-1.

This makes CloudTrail's treatment of these services consistent with that of other AWS global services.

To continue receiving global service events outside of US East (N. Virginia), be sure to convert *single-region trails* using global service events outside of US East (N. Virginia) into *multi-region trails*. Also update the region of your lookup-events API calls to view global service events. For more information about using the CLI to update or create trails for global service events and update lookup events, see [Viewing CloudTrail events with the AWS CLI \(p. 49\)](#) and [Using update-trail \(p. 90\)](#).

For most services, events are recorded in the region where the action occurred. For global services such as AWS Identity and Access Management (IAM), AWS STS, and Amazon CloudFront, events are delivered to any trail that includes global services.

For most global services, events are logged as occurring in US East (N. Virginia) Region, but some global service events are logged as occurring in other regions, such as US East (Ohio) Region or US West (Oregon) Region.

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

- Global service events are delivered by default to trails that are created using the CloudTrail console. Events are delivered to the bucket for the trail.
- If you have multiple single region trails, consider configuring your trails so that global service events are delivered in only one of the trails. For more information, see [Enabling and disabling global service event logging \(p. 92\)](#).
- If you change the configuration of a trail from logging all regions to logging a single region, global service event logging is turned off automatically for that trail. Similarly, if you change the configuration of a trail from logging a single region to logging all regions, global service event logging is turned on automatically for that trail.

For more information about changing global service event logging for a trail, see [Enabling and disabling global service event logging \(p. 92\)](#).

Example:

1. You create a trail in the CloudTrail console. By default, this trail logs global service events.
2. You have multiple single region trails.
3. You do not need to include global services for the single region trails. Global service events are delivered for the first trail. For more information, see [Creating, updating, and managing trails with the AWS Command Line Interface \(p. 87\)](#).

Note

When you create or update a trail with the AWS CLI, AWS SDKs, or CloudTrail API, you can specify whether to include or exclude global service events for trails. You cannot configure global service event logging from the CloudTrail console.

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. Although CloudTrail does not report on system performance or health, you can use CloudTrail with CloudWatch alarms to notify you about activity that you might be interested in.

Partner solutions

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

CloudTrail supported regions

| Region Name | Region | Endpoint | Protocol | AWS Account ID | Support Date |
|--------------------------|----------------|---|----------|----------------|--------------|
| US East (Ohio) | us-east-2 | cloudtrail.us-east-2.amazonaws.com | HTTPS | 475085895292 | 10/17/2016 |
| US East (N. Virginia) | us-east-1 | cloudtrail.us-east-1.amazonaws.com | HTTPS | 086441151436 | 11/13/2013 |
| US West (N. California) | us-west-1 | cloudtrail.us-west-1.amazonaws.com | HTTPS | 388731089494 | 05/13/2014 |
| US West (Oregon) | us-west-2 | cloudtrail.us-west-2.amazonaws.com | HTTPS | 113285607260 | 11/13/2013 |
| Canada (Central) | ca-central-1 | cloudtrail.ca-central-1.amazonaws.com | HTTPS | 819402241893 | 12/08/2016 |
| Africa (Cape Town) | af-south-1 | cloudtrail.af-south-1.amazonaws.com | HTTPS | 525921808201 | 04/22/2020 |
| Asia Pacific (Hong Kong) | ap-east-1 | cloudtrail.ap-east-1.amazonaws.com | HTTPS | 119688915426 | 04/24/2019 |
| Asia Pacific (Mumbai) | ap-south-1 | cloudtrail.ap-south-1.amazonaws.com | HTTPS | 977081816279 | 06/27/2016 |
| Asia Pacific (Osaka) | ap-northeast-3 | cloudtrail.ap-northeast-3.amazonaws.com | HTTPS | 765225791966 | 02/12/2018 |
| Asia Pacific (Seoul) | ap-northeast-2 | cloudtrail.ap-northeast-2.amazonaws.com | HTTPS | 492519147666 | 01/06/2016 |
| Asia Pacific (Singapore) | ap-southeast-1 | cloudtrail.ap-southeast-1.amazonaws.com | HTTPS | 903692715234 | 06/30/2014 |
| Asia Pacific (Sydney) | ap-southeast-2 | cloudtrail.ap-southeast-2.amazonaws.com | HTTPS | 284668455005 | 05/13/2014 |

| Region Name | Region | Endpoint | Protocol | AWS Account ID | Support Date |
|---------------------------|----------------|--|----------|----------------|--------------|
| Asia Pacific (Tokyo) | ap-northeast-1 | cloudtrail.ap-northeast-1.amazonaws.com | HTTPS | 216624486486 | 06/30/2014 |
| China (Beijing) | cn-north-1 | cloudtrail.cn-north-1.amazonaws.com.cn | HTTPS | 193415116832 | 03/01/2014 |
| China (Ningxia) | cn-northwest-1 | cloudtrail.cn-northwest-1.amazonaws.com.cn | HTTPS | 681348832753 | 12/11/2017 |
| Europe (Frankfurt) | eu-central-1 | cloudtrail.eu-central-1.amazonaws.com | HTTPS | 035351147821 | 10/23/2014 |
| Europe (Stockholm) | eu-north-1 | cloudtrail.eu-north-1.amazonaws.com | HTTPS | 829690693026 | 12/11/2018 |
| Europe (Ireland) | eu-west-1 | cloudtrail.eu-west-1.amazonaws.com | HTTPS | 859597730677 | 05/13/2014 |
| Europe (London) | eu-west-2 | cloudtrail.eu-west-2.amazonaws.com | HTTPS | 282025262664 | 12/13/2016 |
| Europe (Paris) | eu-west-3 | cloudtrail.eu-west-3.amazonaws.com | HTTPS | 262312530599 | 12/18/2017 |
| Europe (Milan) | eu-south-1 | cloudtrail.eu-south-1.amazonaws.com | HTTPS | 669305197877 | 04/27/2020 |
| Middle East (Bahrain) | me-south-1 | cloudtrail.me-south-1.amazonaws.com | HTTPS | 034638983726 | 07/29/2019 |
| AWS GovCloud (US-East) | us-gov-east-1 | cloudtrail.us-gov-east-1.amazonaws.com | HTTPS | 886388586500 | 11/12/2018 |
| AWS GovCloud (US-West) | us-gov-west-1 | cloudtrail.us-gov-west-1.amazonaws.com | HTTPS | 608710470296 | 08/16/2011 |
| South America (São Paulo) | sa-east-1 | cloudtrail.sa-east-1.amazonaws.com | HTTPS | 814480443879 | 06/30/2014 |

For more information about using CloudTrail in the AWS GovCloud (US-East) Region, see [AWS GovCloud \(US-East\) Endpoints](#) in the *AWS GovCloud (US) User Guide*.

For more information about using CloudTrail in the AWS GovCloud (US-West) Region, see [AWS GovCloud \(US-West\) Endpoints](#) in the *AWS GovCloud (US) User Guide*.

For more information about using CloudTrail in the China (Beijing) Region, see [China \(Beijing\) Region Endpoints](#) in the *Amazon Web Services General Reference*.

CloudTrail log file examples

CloudTrail monitors events for your account. If you create a trail, it delivers those events as log files to your Amazon S3 bucket. See the following to learn more about log files.

Topics

- [CloudTrail log file name format \(p. 14\)](#)
- [Log file examples \(p. 14\)](#)

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_us-east-2_20150801T0210Z_Mu0KsOhtH1ar15ZZ.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.

Contents

- [Amazon EC2 log examples \(p. 14\)](#)
- [IAM log examples \(p. 16\)](#)
- [Error code and message log example \(p. 18\)](#)
- [CloudTrail Insights event log example \(p. 18\)](#)

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 for Linux Instances](#).

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-east-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"
              }
            }
          ]
        }
      }
    }
  ]
}
```

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` command.

```
{
  "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-east-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": "pending"
              }
            }
          ]
        }
      }
    }
  ]
}
```

```
        "name": "running"
      }
    }
  }
}
```

The following example shows that the Amazon EC2 console backend 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-east-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"
      }
    }
  ]
}
```

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-2",

```

```
"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-2",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "AWSConsole",
    "requestParameters": {
      "userName": "Bob",
      "groupName": "admin"
    },
    "responseElements": null
  ]
}
```

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-2",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-cli/1.3.2 Python/2.7.5 Windows/7",
    "requestParameters": {

```


field shows whether the event was logged at the start or end of the period of unusual activity. The event name, `UpdateInstanceInformation`, is the same name as the AWS Systems Manager API for which CloudTrail analyzed management events to determine that unusual activity occurred. Although the start and end events have unique `eventID` values, they also have a `sharedEventID` value that is used by the pair. The Insights event shows the baseline, or the normal pattern of activity, the `insight`, or average unusual activity that triggered the start Insights event, and in the end event, the `insight` value for the average unusual activity over the duration of the Insights event. For more information about CloudTrail Insights, see [Logging Insights events for trails \(p. 154\)](#).

```
{
  "Records": [
    {
      "eventVersion": "1.07",
      "eventTime": "2019-11-14T00:51:00Z",
      "awsRegion": "us-east-1",
      "eventID": "EXAMPLE8-9621-4d00-b913-beca2EXAMPLE",
      "eventType": "AwsCloudTrailInsight",
      "recipientAccountId": "123456789012",
      "sharedEventID": "EXAMPLE2-1729-42f1-b735-5d8c0EXAMPLE",
      "insightDetails": {
        "state": "Start",
        "eventSource": "ssm.amazonaws.com",
        "eventName": "UpdateInstanceInformation",
        "insightType": "ApiCallRateInsight",
        "insightContext": {
          "statistics": {
            "baseline": {
              "average": 85.4202380952
            },
            "insight": {
              "average": 664
            }
          }
        }
      },
      "eventCategory": "Insight"
    },
    {
      "eventVersion": "1.07",
      "eventTime": "2019-11-14T00:52:00Z",
      "awsRegion": "us-east-1",
      "eventID": "EXAMPLEc-28be-486c-8928-49ce6EXAMPLE",
      "eventType": "AwsCloudTrailInsight",
      "recipientAccountId": "123456789012",
      "sharedEventID": "EXAMPLE2-1729-42f1-b735-5d8c0EXAMPLE",
      "insightDetails": {
        "state": "End",
        "eventSource": "ssm.amazonaws.com",
        "eventName": "UpdateInstanceInformation",
        "insightType": "ApiCallRateInsight",
        "insightContext": {
          "statistics": {
            "baseline": {
              "average": 85.4202380952
            },
            "insight": {
              "average": 664
            },
            "insightDuration": 1
          }
        }
      },
      "eventCategory": "Insight"
    }
  ]
}
```



```
} ]
```

CloudTrail supported services and integrations

CloudTrail supports logging events for many AWS services. You can find the specifics for each supported service in that service's guide. Links to those service-specific topics are provided below. In addition, some AWS services can be used to analyze and act upon data collected in CloudTrail logs. You can browse an overview of those service integrations here.

Note

To see the list of supported regions for each service, see [Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

Topics

- [AWS service integrations with CloudTrail Logs](#) (p. 20)
- [CloudTrail integration with AWS Organizations](#) (p. 21)
- [AWS service topics for CloudTrail](#) (p. 21)
- [CloudTrail unsupported services](#) (p. 31)

AWS service integrations with CloudTrail Logs

You can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following topics.

| AWS Service | Topic | Description |
|---------------|--|---|
| Amazon Athena | Querying AWS CloudTrail Logs | <p>Using Athena with CloudTrail logs is a powerful way to enhance your analysis of AWS service activity. For example, you can use queries to identify trends and further isolate activity by attribute, such as source IP address or user.</p> <p>You can automatically create tables for querying logs directly from the CloudTrail console, and use those tables to run queries in Athena. For more information, see Creating a Table for CloudTrail Logs in the CloudTrail Console in the <i>Amazon Athena User Guide</i>.</p> <p>Note Running queries in Amazon Athena incurs additional costs. For more information, see Amazon Athena Pricing.</p> |

| AWS Service | Topic | Description |
|------------------------|--|---|
| Amazon CloudWatch Logs | Monitoring CloudTrail Log Files with Amazon CloudWatch Logs (p. 159) | <p>You can configure CloudTrail with CloudWatch Logs to monitor your trail logs and be notified when specific activity occurs. For example, you can define CloudWatch Logs metric filters that will trigger CloudWatch alarms and send notifications to you when those alarms are triggered.</p> <p>Note Standard pricing for Amazon CloudWatch and Amazon CloudWatch Logs applies. For more information, see Amazon CloudWatch Pricing.</p> |

CloudTrail integration with AWS Organizations

You can create a trail in the management account for an organization that collects all event data for all AWS accounts in an organization in AWS Organizations. This is called an organization trail. Creating an organization trail helps you define a uniform event logging strategy for your organization. An organization trail is applied automatically to each AWS account in your organization. Users in member accounts can see these trails but cannot modify them, and by default cannot see the log files created for the organization trail. For more information, see [Creating a trail for an organization \(p. 112\)](#).

AWS service topics for CloudTrail

You can learn more about how the events for individual AWS services are recorded in CloudTrail logs, including example events for that service in log files. For more information about how specific AWS services integrate with CloudTrail, see the topic about integration in the individual guide for that service.

| AWS Service | CloudTrail Topics | Support began |
|--------------------|--|---------------|
| Alexa for Business | Logging Alexa for Business Administration Calls Using AWS CloudTrail | 11/29/2017 |
| AWS Amplify | Logging Amplify API calls using AWS CloudTrail | 11/30/2020 |
| AWS Audit Manager | Logging AWS Audit Manager API calls with AWS CloudTrail | 12/07/2020 |
| Amazon API Gateway | Log API management calls to Amazon API Gateway Using AWS CloudTrail | 07/09/2015 |
| Amazon Connect | Logging Amazon Connect API Calls with AWS CloudTrail | 12/11/2019 |

| AWS Service | CloudTrail Topics | Support began |
|---|---|---------------|
| Application Auto Scaling | Logging Application Auto Scaling API calls with AWS CloudTrail | 10/31/2016 |
| AWS Application Discovery Service | Logging Application Discovery Service API Calls with AWS CloudTrail | 05/12/2016 |
| Amazon AppFlow | Logging Amazon AppFlow API calls with AWS CloudTrail | 04/22/2020 |
| AWS App Mesh | Logging App Mesh API Calls with AWS CloudTrail | |
| AWS App Runner | Logging App Runner API calls with AWS CloudTrail | 05/18/2021 |
| Amazon AppStream 2.0 | Logging Amazon AppStream 2.0 API Calls with AWS CloudTrail | 04/25/2019 |
| AWS AppSync | Logging AWS AppSync API Calls with AWS CloudTrail | 02/13/2018 |
| Amazon Athena | Logging Amazon Athena API Calls with AWS CloudTrail | 05/19/2017 |
| AWS Auto Scaling | Logging AWS Auto Scaling API Calls By Using CloudTrail | 08/15/2018 |
| AWS Backup | Logging AWS Backup API Calls with AWS CloudTrail | 02/04/2019 |
| AWS Batch | Logging AWS Batch API Calls with AWS CloudTrail | 1/10/2018 |
| AWS Billing and Cost Management | Logging AWS Billing and Cost Management API Calls with AWS CloudTrail | 06/07/2018 |
| AWS BugBust | Logging BugBust API calls using CloudTrail | 06/24/2021 |
| AWS Certificate Manager | Using AWS CloudTrail | 03/25/2016 |
| AWS Certificate Manager Private Certificate Authority | Using CloudTrail | 06/06/2019 |
| Amazon Chime | Log Amazon Chime Administration Calls Using AWS CloudTrail | 09/27/2017 |
| Amazon Cloud Directory | Logging Amazon Cloud Directory API Calls Using AWS CloudTrail | 01/26/2017 |
| AWS Cloud9 | Logging AWS Cloud9 API Calls with AWS CloudTrail | 01/21/2019 |
| AWS CloudFormation | Logging AWS CloudFormation API Calls in AWS CloudTrail | 04/02/2014 |
| Amazon CloudFront | Using AWS CloudTrail to Capture Requests Sent to the CloudFront API | 05/28/2014 |
| AWS CloudHSM | Logging AWS CloudHSM API Calls By Using AWS CloudTrail | 01/08/2015 |
| AWS Cloud Map | Logging AWS Cloud Map API Calls with AWS CloudTrail | 11/28/2018 |

| AWS Service | CloudTrail Topics | Support began |
|--|---|---------------|
| Amazon CloudSearch | Logging Amazon CloudSearch Configuration Service Calls Using AWS CloudTrail | 10/16/2014 |
| AWS CloudTrail | AWS CloudTrail API Reference (All CloudTrail API calls are logged by CloudTrail.) | 11/13/2013 |
| Amazon CloudWatch | Logging Amazon CloudWatch API Calls in AWS CloudTrail | 04/30/2014 |
| CloudWatch Events | Logging Amazon CloudWatch Events API Calls in AWS CloudTrail | 01/16/2016 |
| CloudWatch Logs | Logging Amazon CloudWatch Logs API Calls in AWS CloudTrail | 03/10/2016 |
| AWS CodeBuild | Logging AWS CodeBuild API Calls with AWS CloudTrail | 12/01/2016 |
| AWS CodeCommit | Logging AWS CodeCommit API Calls with AWS CloudTrail | 01/11/2017 |
| AWS CodeDeploy | Monitoring Deployments with AWS CloudTrail | 12/16/2014 |
| Amazon CodeGuru Reviewer | Logging Amazon CodeGuru Reviewer API Calls with AWS CloudTrail | 12/02/2019 |
| AWS CodePipeline | Logging CodePipeline API Calls By Using AWS CloudTrail | 07/09/2015 |
| AWS CodeStar | Logging AWS CodeStar API Calls with AWS CloudTrail | 06/14/2017 |
| AWS CodeStar Notifications | Logging AWS CodeStar Notifications API Calls with AWS CloudTrail | 11/05/2019 |
| Amazon Cognito | Logging Amazon Cognito API Calls with AWS CloudTrail | 02/18/2016 |
| Amazon Comprehend | Logging Amazon Comprehend API Calls with AWS CloudTrail | 01/17/2018 |
| Amazon Comprehend Medical | Logging Amazon Comprehend Medical API Calls by Using AWS CloudTrail | 11/27/2018 |
| AWS Config | Logging AWS Config API Calls By with AWS CloudTrail | 02/10/2015 |
| AWS Control Tower | Logging AWS Control Tower Actions with AWS CloudTrail | 08/12/2019 |
| Amazon Data Lifecycle Manager | Logging Amazon Data Lifecycle Manager API Calls Using AWS CloudTrail | 07/24/2018 |
| AWS Data Pipeline | Logging AWS Data Pipeline API Calls by using AWS CloudTrail | 12/02/2014 |
| AWS Database Migration Service (AWS DMS) | Logging AWS Database Migration Service API Calls Using AWS CloudTrail | 02/04/2016 |

| AWS Service | CloudTrail Topics | Support began |
|--|---|---|
| AWS DataSync | Logging AWS DataSync API Calls with AWS CloudTrail | 11/26/2018 |
| Amazon Detective | Logging Amazon Detective API calls with AWS CloudTrail | 03/31/2020 |
| AWS Device Farm | Logging AWS Device Farm API Calls By Using AWS CloudTrail | 07/13/2015 |
| AWS Direct Connect | Logging AWS Direct Connect API Calls in AWS CloudTrail | 03/08/2014 |
| AWS Directory Service | Logging AWS Directory Service API Calls by Using CloudTrail | 05/14/2015 |
| Amazon DocumentDB (with MongoDB compatibility) | Logging Amazon DocumentDB API Calls with AWS CloudTrail | 01/09/2019 |
| Amazon DynamoDB | Logging DynamoDB Operations By Using AWS CloudTrail | 05/28/2015 |
| Amazon Elastic Container Registry (Amazon ECR) | Logging Amazon ECR API Calls By Using AWS CloudTrail | 12/21/2015 |
| Amazon Elastic Container Service (Amazon ECS) | Logging Amazon ECS API Calls By Using AWS CloudTrail | 04/09/2015 |
| AWS Elastic Beanstalk (Elastic Beanstalk) | Using Elastic Beanstalk API Calls with AWS CloudTrail | 03/31/2014 |
| Amazon Elastic Block Store (Amazon EBS) EBS direct APIs | Logging API Calls Using AWS CloudTrail Log API Calls for the EBS direct APIs with AWS CloudTrail | Amazon EBS: 11/13/2013 EBS direct APIs: 06/30/2020 |
| Amazon Elastic Compute Cloud (Amazon EC2) | Logging API Calls Using AWS CloudTrail | 11/13/2013 |
| Amazon EC2 Auto Scaling | Logging Auto Scaling API Calls By Using CloudTrail | 07/16/2014 |
| Amazon EC2 Image Builder | Logging EC2 Image Builder API calls using CloudTrail | 12/02/2019 |
| Amazon Elastic File System (Amazon EFS) | Logging Amazon EFS API Calls with AWS CloudTrail | 06/28/2016 |
| Amazon Fraud Detector | Logging Amazon Fraud Detector API Calls with AWS CloudTrail | 01/09/2020 |
| Amazon Elastic Kubernetes Service (Amazon EKS) | Logging Amazon EKS API Calls with AWS CloudTrail | 06/05/2018 |
| Elastic Load Balancing | AWS CloudTrail Logging for Your Classic Load Balancer and AWS CloudTrail Logging for Your Application Load Balancer | 04/04/2014 |

| AWS Service | CloudTrail Topics | Support began |
|-------------------------------------|--|---------------|
| Amazon Elastic Transcoder | Logging Amazon Elastic Transcoder API Calls with AWS CloudTrail | 10/27/2014 |
| Amazon ElastiCache | Logging Amazon ElastiCache API Calls Using AWS CloudTrail | 09/15/2014 |
| Amazon OpenSearch Service | Auditing Amazon OpenSearch Service Domains with AWS CloudTrail | 10/01/2015 |
| AWS Elemental MediaConnect | Logging AWS Elemental MediaConnect API Calls with AWS CloudTrail | 11/27/2018 |
| AWS Elemental MediaConvert | Logging AWS Elemental MediaConvert API Calls with CloudTrail | 11/27/2017 |
| AWS Elemental MediaLive | Logging MediaLive API Calls with AWS CloudTrail | 01/19/2019 |
| AWS Elemental MediaPackage | Logging AWS Elemental MediaPackage API Calls with AWS CloudTrail | 12/21/2018 |
| AWS Elemental MediaStore | Logging AWS Elemental MediaStore API Calls with CloudTrail | 11/27/2017 |
| AWS Elemental MediaTailor | Logging AWS Elemental MediaTailor API Calls with AWS CloudTrail | 02/11/2019 |
| Amazon EMR | Logging Amazon EMR API Calls in AWS CloudTrail | 04/04/2014 |
| Amazon EMR on EKS | Logging Amazon EMR on EKS API calls using AWS CloudTrail | 12/09/2020 |
| AWS Fault Injection Simulator | Log API calls with AWS CloudTrail | 03/15/2021 |
| AWS Firewall Manager | Logging AWS Firewall Manager API Calls with AWS CloudTrail | 04/05/2018 |
| Amazon Forecast | Logging Amazon Forecast API Calls with AWS CloudTrail | 11/28/2018 |
| FreeRTOS Over-the-Air Updates (OTA) | Logging AWS IoT OTA API Calls with AWS CloudTrail | 05/22/2019 |
| Amazon FSx for Lustre | Logging Amazon FSx for Lustre API Calls with AWS CloudTrail | 01/11/2019 |
| Amazon FSx for Windows File Server | Monitoring with AWS CloudTrail | 11/28/2018 |
| Amazon GameLift | Logging Amazon GameLift API Calls with AWS CloudTrail | 01/27/2016 |
| Amazon S3 Glacier | Logging S3 Glacier API Calls By Using AWS CloudTrail | 12/11/2014 |
| AWS Global Accelerator | Logging AWS Global Accelerator API Calls with AWS CloudTrail | 11/26/2018 |

| AWS Service | CloudTrail Topics | Support began |
|--|--|---------------|
| AWS Glue | Logging AWS Glue Operations Using AWS CloudTrail | 11/07/2017 |
| AWS Ground Station | Logging AWS Ground Station API Calls with AWS CloudTrail | 05/31/2019 |
| Amazon GuardDuty | Logging Amazon GuardDuty API Calls with AWS CloudTrail | 02/12/2018 |
| AWS Health | Logging AWS Health API Calls with AWS CloudTrail | 11/21/2016 |
| Amazon HealthLake | Logging Amazon HealthLake API calls with AWS CloudTrail | 12/07/2020 |
| Amazon Honeycode | Logging Amazon Honeycode API Calls with AWS CloudTrail | 06/24/2020 |
| Amazon Inspector | Logging Amazon Inspector API calls with AWS CloudTrail | 04/20/2016 |
| Amazon Interactive Video Service | Logging Amazon IVS API Calls with AWS CloudTrail | 07/15/2020 |
| AWS IoT | Logging AWS IoT API Calls with AWS CloudTrail | 04/11/2016 |
| AWS IoT Analytics | Logging AWS IoT Analytics API calls with AWS CloudTrail | 04/23/2018 |
| AWS IoT 1-Click | Logging AWS IoT 1-Click API Calls with AWS CloudTrail | 05/14/2018 |
| AWS IoT Events | Logging AWS IoT Events API Calls with AWS CloudTrail | 06/11/2019 |
| AWS IoT Greengrass | Logging AWS IoT Greengrass API Calls with AWS CloudTrail | 10/29/2018 |
| AWS IoT Greengrass V2 | Log AWS IoT Greengrass V2 API calls with AWS CloudTrail | 12/14/2020 |
| AWS IoT SiteWise | Logging AWS IoT SiteWise API calls with AWS CloudTrail | 04/29/2020 |
| AWS IoT Things Graph | Logging AWS IoT Things Graph API Calls with AWS CloudTrail | 05/31/2019 |
| AWS Identity and Access Management (IAM) | Logging IAM Events with AWS CloudTrail | 11/13/2013 |
| Amazon Kendra | Logging Amazon Kendra API calls with AWS CloudTrail | 05/11/2020 |
| AWS Key Management Service (AWS KMS) | Logging AWS KMS API Calls using AWS CloudTrail | 11/12/2014 |

| AWS Service | CloudTrail Topics | Support began |
|---------------------------------------|--|--|
| Amazon Kinesis Data Analytics | Monitoring Amazon Kinesis Data Analytics with AWS CloudTrail (SQL Applications) and Monitoring Amazon Kinesis Data Analytics with AWS CloudTrail (Apache Flink Applications) | 03/22/2019 |
| Amazon Kinesis Data Firehose | Monitoring Amazon Kinesis Data Firehose API Calls with AWS CloudTrail | 03/17/2016 |
| Amazon Kinesis Data Streams | Logging Amazon Kinesis Data Streams API Calls Using AWS CloudTrail | 04/25/2014 |
| Amazon Kinesis Video Streams | Logging Kinesis Video Streams API Calls with AWS CloudTrail | 05/24/2018 |
| AWS Lake Formation | Logging AWS Lake Formation API Calls Using AWS CloudTrail | 08/09/2019 |
| AWS Lambda | Logging AWS Lambda API Calls By Using AWS CloudTrail Using Lambda with AWS CloudTrail | Management events: 04/09/2015 Data events: 11/30/2017 |
| Amazon Lex | Logging Amazon Lex API Calls with CloudTrail | 08/15/2017 |
| AWS License Manager | Logging AWS License Manager API Calls with AWS CloudTrail | 03/01/2019 |
| Amazon Lightsail | Logging Lightsail API Calls with AWS CloudTrail | 12/23/2016 |
| Amazon Location Service | Logging and monitoring with AWS CloudTrail | 12/15/2020 |
| Amazon Lookout for Vision | Logging Amazon Lookout for Vision calls with AWS CloudTrail | 12/01/2020 |
| Amazon Lookout for Equipment | Monitoring Amazon Lookout for Equipment calls with AWS CloudTrail | 12/01/2020 |
| Amazon Lookout for Metrics | Viewing Amazon Lookout for Metrics API activity in AWS CloudTrail | 12/08/2020 |
| Amazon Machine Learning | Logging Amazon ML API Calls By Using AWS CloudTrail | 12/10/2015 |
| Amazon Macie | Log Amazon Macie API calls using AWS CloudTrail | 05/13/2020 |
| Amazon Managed Blockchain | Logging Amazon Managed Blockchain API calls using AWS CloudTrail Logging Ethereum for Managed Blockchain API calls using AWS CloudTrail (Preview) | 04/01/2019 |
| Amazon Managed Grafana | Logging Amazon Managed Grafana API calls using AWS CloudTrail | 12/15/2020 |
| Amazon Managed Service for Prometheus | Logging Amazon Managed Service for Prometheus API calls using AWS CloudTrail | 12/15/2020 |

| AWS Service | CloudTrail Topics | Support began |
|---|--|---------------|
| Amazon Keyspaces (for Apache Cassandra) | Logging Amazon Keyspaces API calls with AWS CloudTrail | 01/13/2020 |
| AWS Managed Services | AWS Managed Services | 12/21/2016 |
| Amazon Managed Streaming for Apache Kafka | Logging Amazon MSK API Calls with AWS CloudTrail | 12/11/2018 |
| Amazon Managed Workflows for Apache Airflow | Monitoring Amazon MWAA API activity with AWS CloudTrail | 11/24/2020 |
| AWS Marketplace | Logging AWS Marketplace API Calls with AWS CloudTrail | 05/02/2017 |
| AWS Marketplace Metering Service | Logging AWS Marketplace API Calls with AWS CloudTrail | 08/22/2018 |
| AWS Migration Hub | Logging AWS Migration Hub API Calls with AWS CloudTrail | 08/14/2017 |
| AWS Mobile Hub | Logging AWS Mobile CLI API Calls with AWS CloudTrail | 06/29/2018 |
| Amazon MQ | Logging Amazon MQ API Calls Using AWS CloudTrail | 07/19/2018 |
| Amazon Neptune | Logging Amazon Neptune API Calls Using AWS CloudTrail | 05/30/2018 |
| AWS Network Firewall | Logging calls to the AWS Network Firewall API with AWS CloudTrail | 11/17/2020 |
| AWS OpsWorks for Chef Automate | Logging AWS OpsWorks for Chef Automate API Calls with AWS CloudTrail | 07/16/2018 |
| AWS OpsWorks for Puppet Enterprise | Logging OpsWorks for Puppet Enterprise API Calls with AWS CloudTrail | 07/16/2018 |
| AWS OpsWorks Stacks | Logging AWS OpsWorks Stacks API Calls with AWS CloudTrail | 06/04/2014 |
| AWS Organizations | Logging AWS Organizations Events with AWS CloudTrail | 02/27/2017 |
| AWS Outposts | Logging AWS Outposts API calls with AWS CloudTrail | 02/04/2020 |
| AWS Personal Health Dashboard | Logging AWS Health API Calls with AWS CloudTrail | 12/01/2016 |
| Amazon Personalize | Logging Amazon Personalize API Calls with AWS CloudTrail | 11/28/2018 |
| Amazon Pinpoint | Logging Amazon Pinpoint API Calls with AWS CloudTrail | 02/06/2018 |
| Amazon Pinpoint SMS and Voice API | Logging Amazon Pinpoint API Calls with AWS CloudTrail | 11/16/2018 |

| AWS Service | CloudTrail Topics | Support began |
|---|---|---------------|
| Amazon Polly | Logging Amazon Polly API Calls with AWS CloudTrail | 11/30/2016 |
| Amazon Quantum Ledger Database (Amazon QLDB) | Logging Amazon QLDB API Calls with AWS CloudTrail | 09/10/2019 |
| AWS Certificate Manager Private Certificate Authority | Using CloudTrail | 04/04/2018 |
| Amazon QuickSight | Logging Operations with CloudTrail | 04/28/2017 |
| Amazon Redshift | Logging Amazon Redshift API Calls with AWS CloudTrail | 06/10/2014 |
| Amazon Rekognition | Logging Amazon Rekognition API Calls Using AWS CloudTrail | 04/6/2018 |
| Amazon Relational Database Service (Amazon RDS) | Logging Amazon RDS API Calls Using AWS CloudTrail | 11/13/2013 |
| Amazon RDS Performance Insights | Logging Amazon RDS API Calls Using AWS CloudTrail The Amazon RDS Performance Insights API is a subset of the Amazon RDS API. | 06/21/2018 |
| AWS Resource Access Manager (AWS RAM) | Logging AWS RAM API Calls with AWS CloudTrail | 11/20/2018 |
| AWS Resource Groups | Logging AWS Resource Groups API Calls with AWS CloudTrail | 06/29/2018 |
| AWS RoboMaker | Logging AWS RoboMaker API Calls with AWS CloudTrail | 01/16/2019 |
| Amazon Route 53 | Using AWS CloudTrail to Capture Requests Sent to the Route 53 API | 02/11/2015 |
| Amazon Route 53 Application Recovery Controller | Logging Amazon Route 53 Application Recovery Controller API calls using AWS CloudTrail | 07/27/2021 |
| Amazon SageMaker | Logging Amazon SageMaker API Calls with AWS CloudTrail | 01/11/2018 |
| AWS Secrets Manager | Monitor the Use of Your AWS Secrets Manager Secrets | 04/05/2018 |
| AWS Security Hub | Logging AWS Security Hub API Calls with AWS CloudTrail | 11/27/2018 |
| AWS Security Token Service (AWS STS) | Logging IAM Events with AWS CloudTrail The IAM topic includes information for AWS STS. | 11/13/2013 |
| AWS Server Migration Service | AWS SMS API Reference | 11/14/2016 |
| AWS Serverless Application Repository | Logging AWS Serverless Application Repository API Calls with AWS CloudTrail | 02/20/2018 |

| AWS Service | CloudTrail Topics | Support began |
|---|--|--|
| AWS Service Catalog | Logging AWS Service Catalog API Calls with AWS CloudTrail | 07/06/2016 |
| Service Quotas | | 06/24/2019 |
| AWS Shield | Logging Shield Advanced API Calls with AWS CloudTrail | 02/08/2018 |
| Amazon Simple Email Service (Amazon SES) | Logging Amazon SES API Calls By Using AWS CloudTrail | 05/07/2015 |
| Amazon Simple Notification Service (Amazon SNS) | Logging Amazon Simple Notification Service API Calls By Using AWS CloudTrail | 10/09/2014 |
| Amazon Simple Queue Service (Amazon SQS) | Logging Amazon SQS API Actions Using AWS CloudTrail | 07/16/2014 |
| Amazon Simple Storage Service | Logging Amazon S3 API Calls By Using AWS CloudTrail | Management events: 09/01/2015 Data events: 11/21/2016 |
| Amazon Simple Workflow Service (Amazon SWF) | Logging Amazon Simple Workflow Service API Calls with AWS CloudTrail | 05/13/2014 |
| AWS Single Sign-On (AWS SSO) | Logging AWS SSO API Calls with AWS CloudTrail | 12/07/2017 |
| AWS Snowball | Logging AWS Snowball API Calls with AWS CloudTrail | 01/25/2019 |
| AWS Snowball Edge | Logging AWS Snowball Edge API Calls with AWS CloudTrail | 01/25/2019 |
| AWS Step Functions | Logging AWS Step Functions API Calls with AWS CloudTrail | 12/01/2016 |
| Storage Gateway | Logging Storage Gateway API Calls by Using AWS CloudTrail | 12/16/2014 |
| AWS Support | Logging AWS Support API Calls with AWS CloudTrail | 04/21/2016 |
| AWS Systems Manager | Logging AWS Systems Manager API Calls with AWS CloudTrail | 11/13/2013 |
| AWS Systems Manager Incident Manager | Logging AWS Systems Manager Incident Manager API calls using AWS CloudTrail | 05/10/2021 |
| Amazon Textract | Logging Amazon Textract API Calls with AWS CloudTrail | 05/29/2019 |
| Amazon Transcribe | Logging Amazon Transcribe API Calls with AWS CloudTrail | 06/28/2018 |
| AWS Transfer for SFTP | Logging AWS Transfer for SFTP API Calls with AWS CloudTrail | 01/08/2019 |

| AWS Service | CloudTrail Topics | Support began |
|---|---|---------------|
| Amazon Translate | Logging Amazon Translate API Calls with AWS CloudTrail | 04/04/2018 |
| AWS Transit Gateway | Logging API Calls for Your Transit Gateway Using AWS CloudTrail | 11/26/2018 |
| AWS Trusted Advisor | Logging AWS Trusted Advisor console actions with AWS CloudTrail | 10/22/2020 |
| Amazon Virtual Private Cloud (Amazon VPC) | Logging API Calls Using AWS CloudTrail The Amazon VPC API is a subset of the Amazon EC2 API. | 11/13/2013 |
| AWS WAF | Logging AWS WAF API Calls with AWS CloudTrail | 04/28/2016 |
| AWS Well-Architected Tool | Logging AWS Well-Architected Tool API Calls with AWS CloudTrail | 12/15/2020 |
| Amazon WorkDocs | Logging Amazon WorkDocs API Calls By Using AWS CloudTrail | 08/27/2014 |
| Amazon WorkLink | Logging Amazon WorkLink API Calls with AWS CloudTrail | 01/23/2019 |
| Amazon WorkMail | Logging Amazon WorkMail API Calls Using AWS CloudTrail | 12/12/2017 |
| WorkSpaces | Logging Amazon WorkSpaces API Calls by Using CloudTrail | 04/09/2015 |
| AWS X-Ray | Logging AWS X-Ray API Calls With CloudTrail | 04/25/2018 |

CloudTrail unsupported services

The following AWS services do not support logging events with AWS CloudTrail. The reasons why a service does not support CloudTrail logging can vary. For example, a service that is still in preview, or not yet released for general availability (GA), is not considered supported for CloudTrail logging.

For a list of supported AWS services, see [CloudTrail supported services and integrations \(p. 20\)](#).

| AWS service | Launch date |
|------------------------|-------------------|
| AWS Import/Export | June 17, 2020 |
| Amazon Macie Classic | August 14, 2017 |
| AWS Price List Service | December 17, 2018 |

The following AWS services do not have public API operations.

| AWS service | Launch date |
|---------------------------------------|-------------------|
| AWS Deep Learning AMI | November 15, 2017 |
| Amazon WorkSpaces Application Manager | April 9, 2015 |
| AWS Artifact | November 30, 2016 |
| AWS DeepComposer | December 2, 2019 |
| AWS DeepLens | November 29, 2017 |
| AWS DeepRacer | April 29, 2019 |
| AWS Snowmobile | November 30, 2016 |
| Amazon Sumerian | May 15, 2018 |

Quotas in AWS CloudTrail

The following table describes quotas, or limits, within CloudTrail. CloudTrail has no adjustable quotas. For information about other quotas in AWS, see [AWS service quotas](#).

| Resource | Default Limit | Comments |
|------------------------------|--|--|
| Trails per region | 5 | This limit cannot be increased. |
| Get, describe, and list APIs | 10 transactions per second (TPS) | The maximum number of operation requests you can make per second without being throttled. The <code>LookupEvents</code> API is not included in this category. This limit cannot be increased. |
| LookupEvents API | 2 transactions per second (TPS) | The maximum number of operation requests you can make per second without being throttled. This limit cannot be increased. |
| All other APIs | 1 transaction per second (TPS) | The maximum number of operation requests you can make per second without being throttled. This limit cannot be increased. |
| Event selectors | 5 per trail | This limit cannot be increased. |
| Advanced event selectors | 500 conditions across all advanced event selectors | If a trail uses advanced event selectors, a maximum of 500 total values for all conditions in all advanced event selectors is allowed. Unless a trail logs data |

| Resource | Default Limit | Comments |
|-----------------------------------|---|---|
| | | <p>events on all resources, such as all S3 buckets or all Lambda functions, a trail is limited to 250 data resources. Data resources can be distributed across event selectors, but the overall total cannot exceed 250.</p> <p>This limit cannot be increased.</p> |
| Data resources in event selectors | 250 across all event selectors in a trail | <p>If you choose to limit data events by using event selectors or advanced event selectors, the total number of data resources cannot exceed 250 across all event selectors in a trail. The limit of number of resources on an individual event selector is configurable up to 250. This upper limit is allowed only if the total number of data resources does not exceed 250 across all event selectors.</p> <p>Examples:</p> <ul style="list-style-type: none"> A trail with 5 event selectors, each configured with 50 data resources, is allowed. $(5 \times 50 = 250)$ A trail with 5 event selectors, 3 of which are configured with 50 data resources, 1 of which is configured with 99 data resources, and 1 of which is configured with 1 data resource, is also allowed. $((3 \times 50) + 1 + 99 = 250)$ A trail configured with 5 event selectors, all of which are configured with 100 data resources, is not allowed. $(5 \times 100 = 500)$ <p>This limit cannot be increased.</p> <p>The limit does not apply if you choose to log data events on all resources, such as all S3 buckets or all Lambda functions.</p> |

| Resource | Default Limit | Comments |
|--|--|--|
| Event size | All event versions: events over 256 KB cannot be sent to CloudWatch Logs Event version 1.05 and newer: total event size limit of 256 KB | Amazon CloudWatch Logs and Amazon CloudWatch Events each allow a maximum event size of 256 KB. CloudTrail does not send events over 256 KB to CloudWatch Logs or CloudWatch Events. Starting with event version 1.05, events have a maximum size of 256 KB. This is to help prevent exploitation by malicious actors, and allow events to be consumed by other AWS services, such as CloudWatch Logs and CloudWatch Events. |
| CloudTrail file size sent to Amazon S3 | 50 MB ZIP file, after compression | For both management and data events, CloudTrail sends events to S3 in maximum 50 MB (compressed) ZIP files. If enabled on the trail, log delivery notifications are sent by Amazon SNS after CloudTrail sends ZIP files to S3. |

Getting started with AWS CloudTrail tutorial

If you're new to AWS CloudTrail, this tutorial helps you learn how to use its features. In this tutorial, you review your recent AWS account activity in the CloudTrail console and examine an event. You then create a trail, which is an ongoing record of management event activity that is stored in an Amazon S3 bucket. Unlike **Event history**, this ongoing record is not limited to 90 days, logs events in all AWS Regions, and can help you meet your security and auditing needs over time.

Topics

- [Prerequisites \(p. 35\)](#)
- [Step 1: Review AWS account activity in event history \(p. 35\)](#)
- [Step 2: Create your first trail \(p. 37\)](#)
- [Step 3: View your log files \(p. 40\)](#)
- [Step 4: Plan for next steps \(p. 42\)](#)

Prerequisites

Before you begin, you must complete the following prerequisites and setup:

- Create an AWS account, if you do not already have one.

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

To sign up for an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

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

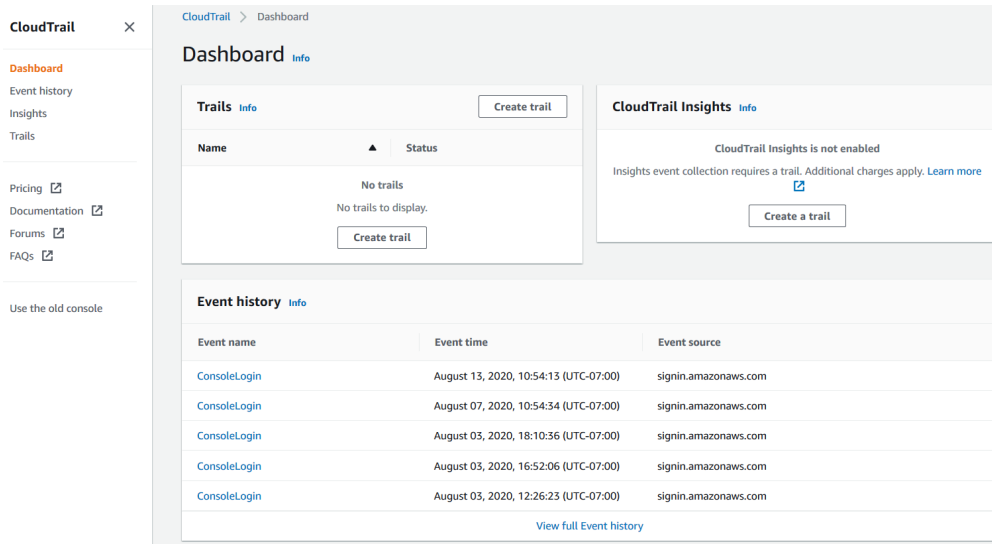
- Create an IAM user for administering CloudTrail. For more information, see [Granting permissions for CloudTrail administration \(p. 236\)](#).

Step 1: Review AWS account activity in event history

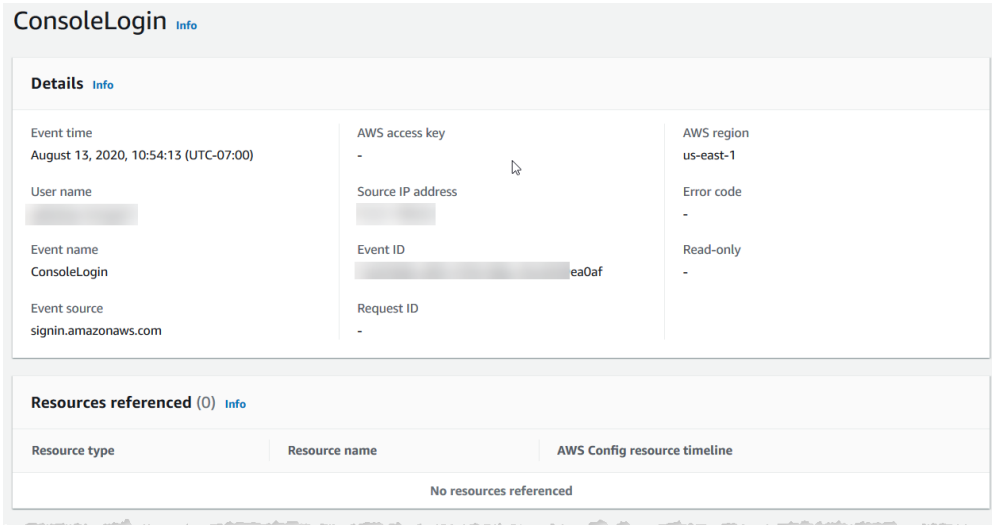
CloudTrail is enabled on your AWS account when you create the account. When activity occurs in any AWS service that supports CloudTrail, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. In other words, you can view, search, and download recent events in your AWS account before creating a trail, though creating a trail is important for long-term records and

auditing of your AWS account activity. Unlike a trail, **Event history** only shows events that have occurred over the last 90 days.

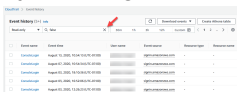
1. Sign in to the AWS Management Console using the IAM user you configured for CloudTrail administration. Open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/home/>.
2. Review the information in your dashboard about the most recent events that have occurred in your AWS account. A recent event should be a `ConsoleLogin` event, showing that you just signed in to the AWS Management Console.



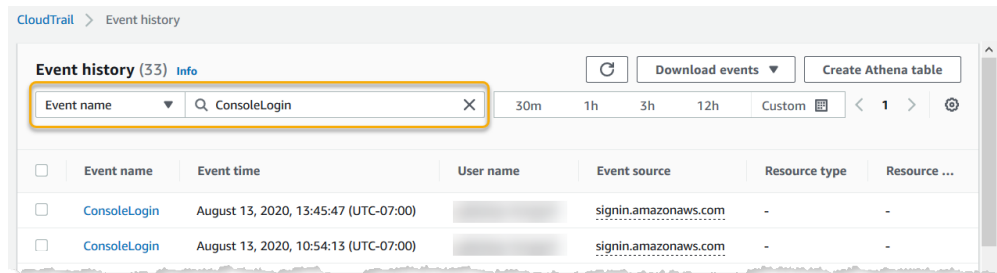
3. To see more information about an event, expand it.



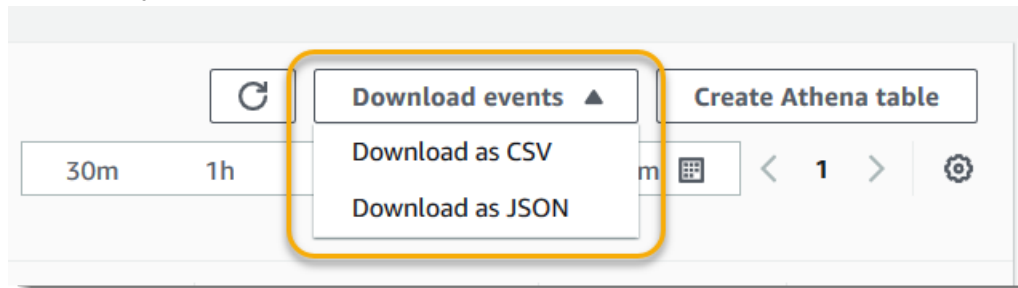
4. In the navigation pane, choose **Event history**. You see a filtered list of events, with the most recent events showing first. The default filter for events is **Read only**, set to **false**. You can clear that filter by choosing **X** at the right of the filter.



5. Many more events are shown without the default filter. You can filter events in many ways. For example, to view all console login events, you could choose the **Event name** filter, and specify **ConsoleLogin**. The choice of filters is up to you.



6. You can save event history by downloading it as a file in CSV or JSON format. Downloading your event history can take a few minutes.



For more information, see [Viewing events with CloudTrail Event history \(p. 44\)](#).

Step 2: Create your first trail

While the events provided in **Event history** in the CloudTrail console are useful for reviewing recent activity, they are limited to recent activity, and they do not include all possible events that can be recorded by CloudTrail. Additionally, your view of events in the console is limited to the AWS Region where you are signed in. To create an ongoing record of activity in your AWS account that captures information for all AWS Regions, create a trail. By default, when you create a trail in the CloudTrail console, the trail logs events in all Regions. Logging events in all Regions in your account is a recommended best practice.

For your first trail, we recommend creating a trail that logs all [management events \(p. 5\)](#) in all AWS Regions, and does not log any [data events \(p. 5\)](#). Examples of management events include security events such as IAM `CreateUser` and `AttachRolePolicy` events, resource events such as `RunInstances` and `CreateBucket`, and many more. You will create an Amazon S3 bucket where you will store the log files for the trail as part of creating the trail in the CloudTrail console.

Note

This tutorial assumes you are creating your first trail. Depending on the number of trails you have in your AWS account, and how those trails are configured, the following procedure might or might not incur expenses. CloudTrail stores log files in an Amazon S3 bucket, which incurs costs. For more information about pricing, see [AWS CloudTrail Pricing](#) and [Amazon S3 Pricing](#).

1. Sign in to the AWS Management Console using the IAM user you configured for CloudTrail administration. Open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/home/>. In the **Region** selector, choose the AWS Region where you want your trail to be created. This is the home Region for the trail.

Note

The home Region is the only AWS Region where you can view and update the trail after it is created, even if the trail logs events in all AWS Regions.

2. On the CloudTrail service home page, the **Trails** page, or the **Trails** section of the **Dashboard** page, choose **Create trail**.
3. In **Trail name**, give your trail a name, such as *My-Management-Events-Tail*. As a best practice, use a name that quickly identifies the purpose of the trail. In this case, you're creating a trail that logs management events.
4. Leave default settings for AWS Organizations organization trails. This option won't be available to change unless you have accounts configured in Organizations.
5. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies. Give your bucket a name, such as *my-bucket-for-storing-cloudtrail-logs*.

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

Note

The name of your Amazon S3 bucket must be globally unique. For more information, see [Amazon S3 bucket naming requirements \(p. 131\)](#).

Choose trail attributes

General details

Trail name
Enter a display name for your trail.

My-management-events-trail

3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.

☐ Enable for all accounts in my organization
To review accounts in your organization, open AWS Organizations. [See all accounts](#)

Storage location [Info](#)

☒ Create new S3 bucket
Create a bucket to store logs for the trail.

☐ Use existing S3 bucket
Choose an existing bucket to store logs for this trail.

Trail log bucket and folder
Enter a new S3 bucket name and folder (prefix) to store your logs. Bucket names must be globally unique.

aws-cloudtrail-logs-08132020-my-trail

Logs will be stored in aws-cloudtrail-logs-08132020-my-trail/AWSLogs/840881077363

Log file SSE-KMS encryption [Info](#)

☐ Enabled

► Additional settings

6. Clear the check box to disable **Log file SSE-KMS encryption**. By default, your log files are encrypted with SSE-S3 encryption. For more information about this setting, see [Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys \(SSE-S3\)](#).
7. Leave default settings in **Additional settings**.
8. For now, do not send logs to Amazon CloudWatch Logs.
9. In **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify your CloudTrail trails and other resources, such as the Amazon S3 buckets that contain CloudTrail log files. For example, you could attach a tag with the name **Compliance** and the value **Auditing**.

Note

Though you can add tags to trails when you create them in the CloudTrail console, and you can create an Amazon S3 bucket to store your log files in the CloudTrail console, you cannot

add tags to the Amazon S3 bucket from the CloudTrail console. For more information about viewing and changing the properties of an Amazon S3 bucket, including adding tags to a bucket, see the [Amazon S3 User Guide](#).

The screenshot shows a section of the AWS CloudTrail console. It has two main panels. The top panel is titled 'CloudWatch Logs - optional' and contains a checkbox for 'Enabled' which is currently unchecked, and a link for 'Policy document'. The bottom panel is titled 'Tags - optional' and contains a table for adding tags. The table has two columns: 'Key' and 'Value - optional'. There are two rows of tags: one with 'Compliance' as the key and 'Auditing' as the value, and another with 'Compliance' as the key and 'Auditing' as the value. There is a 'Remove' button next to the second tag. Below the table is an 'Add tag' button and a note that says 'You can add 49 more tags'. At the bottom right of the console, there are 'Cancel' and 'Next' buttons.

CloudWatch Logs - optional
You can enable SNS notifications in CloudWatch Logs for specific API actions. Standard CloudWatch and CloudWatch Logs charges apply.

CloudWatch Logs [Info](#)

☐ Enabled

► Policy document

Tags - optional [Info](#)
You can add one or more tags to help you manage and organize your resources, including trails.

| Key | Value - optional | |
|---|---------------------------------------|---------------------------------------|
| <input type="text" value="Compliance"/> | <input type="text" value="Auditing"/> | <input type="button" value="Remove"/> |
| <input type="button" value="Add tag"/> | | |

You can add 49 more tags

Cancel **Next**

When you are finished creating tags, choose **Next**.

10. On the **Choose log events** page, select event types to log. For this trail, keep the default, **Management events**. In the **Management events** area, choose to log both **Read** and **Write** events, if they are not already selected. Leave the check boxes for **Exclude AWS KMS events** and **Exclude Amazon RDS Data API events** empty, to log all events.

Choose log events

Events [Info](#)

Record API activity for individual resources, or for all current and future resources in AWS account. [Additional charges apply](#)

Event type
Choose the type of events that you want to log.

☒ **Management events**
Capture management operations performed on your AWS resources.

☐ **Data events**
Log the resource operations performed on or within a resource.

☐ **Insights events**
Identify unusual activity, errors, or user behavior in your account.

Management events [Info](#)

Management events show information about management operations performed on resources in your AWS account.

Charges apply to log management events on this trail because you are logging at least one other copy of management events in your account.

API activity
Choose the activities you want to log.

☒ **Read** ☒ **Write**

☐ **Exclude AWS KMS events**

☐ **Exclude Amazon RDS Data API events**

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

11. Leave default settings for **Data events** and Insights events. This trail will not log any data or CloudTrail Insights events. Choose **Next**.
12. On the **Review and create** page, review the settings you've chosen for your trail. Choose **Edit** for a section to go back and make changes. When you are ready to create your trail, choose **Create trail**.
13. The **Trails** page shows your new trail in the table. Note that the trail is set to **Multi-region trail** by default, and that logging is turned on for the trail by default.

| | Name | Home region | Multi-region trail | Insights | Organization trail | S3 bucket | Log file prefix | CloudWatch Logs log group | Status |
|-----------------------|----------------------------|--------------------|--------------------|----------|--------------------|--------------------------------------|-----------------|---------------------------|---------|
| <input type="radio"/> | My-Management-Events-Trail | Europe (Frankfurt) | Yes | Disabled | No | aws-cloudtrail-logs-08132020-mytrail | | | Logging |

Step 3: View your log files

Within an average of about 15 minutes of creating your first trail, CloudTrail delivers the first set of log files to the Amazon S3 bucket for your trail. You can look at these files and learn about the information they contain.

Note

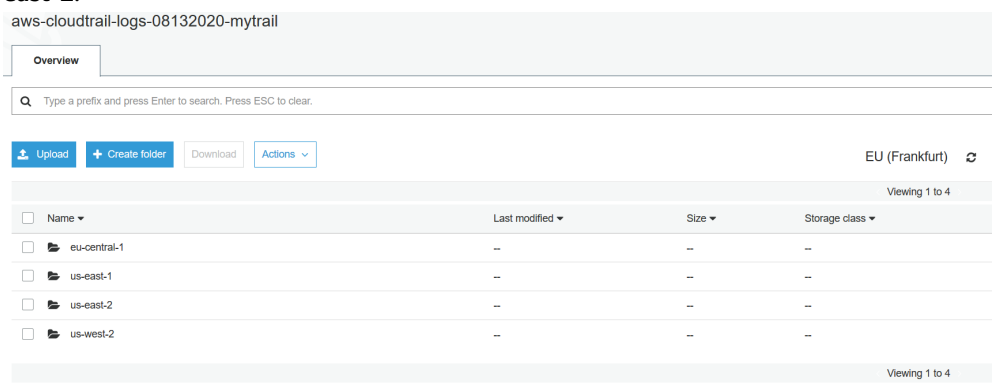
CloudTrail typically delivers logs within an average of about 15 minutes of an API call. This time is not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information.

1. In the navigation pane, choose **Trails**. On the **Trails** page, find the name of the trail you just created (in the example, *My-Management-Events-Trail*).

Note

Be sure you are still signed in using the IAM user you configured for CloudTrail administration. Otherwise you might not have sufficient permissions to view trails in the CloudTrail console or the Amazon S3 bucket that contains log files for that trail.

2. In the row for the trail, choose the value for the S3 bucket (in the example, `aws-cloudtrail-logs-08132020-mytrail`).
3. The Amazon S3 console opens and shows that bucket, at the top level for log files. Because you created a trail that logs events in all AWS Regions, the display opens at the level that shows you each Region folder. The hierarchy of the Amazon S3 bucket navigation at this level is `bucket-name/AWSLogs/account-id/CloudTrail`. Choose the folder for the AWS Region where you want to review log files. For example, if you want to review the log files for the US East (Ohio) Region, choose **us-east-2**.



4. Navigate the bucket folder structure to the year, the month, and the day where you want to review logs of activity in that Region. In that day, there are a number of files. The name of the files begin with your AWS account ID, and end with the extension `.gz`. For example, if your account ID is `123456789012`, you would see files with names similar to this: `123456789012_CloudTrail_us-east-2_20190610T1255abcdeEXAMPLE.json.gz`.

To view these files, you can download them, unzip them, and then view them in a plain-text editor or a JSON file viewer. Some browsers also support viewing `.gz` and JSON files directly. We recommend using a JSON viewer, as it makes it easier to parse the information in CloudTrail log files.

As you're browsing through the file content, you might start to wonder about what you're seeing. CloudTrail logs events for every AWS service that experienced activity in that AWS Region at the time that event occurred. In other words, events for different AWS services are mixed together, based solely on time. To learn more about what a specific AWS service logs with CloudTrail, including examples of log file entries for API calls for that service, see the [list of supported services for CloudTrail \(p. 21\)](#), and read the CloudTrail integration topic for that service. You can also learn more about the content and structure of CloudTrail log files by reviewing the [CloudTrail log event reference \(p. 270\)](#).

You might also notice what you're not seeing in log files in US East (Ohio). Specifically, you won't see any console sign-in events, even though you know you logged into the console. That's because console sign-in and IAM events are [global service events \(p. 10\)](#), which are usually logged in a specific AWS Region. In this case, they are logged in US East (N. Virginia), and found in the folder **us-east-1**. Open that folder, and open the year, month, and day you're interested in. Browse the log files, and you find `ConsoleLogin` events that look similar to the following:

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
```

```
{
  "arn": "arn:aws:iam::123456789012:user/Mary_Major",
  "accountId": "123456789012",
  "userName": "Mary_Major"
},
{
  "eventTime": "2019-06-10T17:14:09Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.67",
  "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "2681fc29-EXAMPLE",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "123456789012"
}
```

This log file entry tells you more than just the identity of the IAM user who logged in (**Mary_Major**), the date and time she logged in, and that the login was successful. You can also learn the IP address she logged in from, the operating system and browser software of the computer she used, and that she was not using multi-factor authentication.

Step 4: Plan for next steps

Now that you have a trail, you have access to an ongoing record of events and activities in your AWS account. This ongoing record helps you meet accounting and auditing needs for your AWS account. However, there is a lot more you can do with CloudTrail and CloudTrail data.

- **Add additional security for your trail data.** CloudTrail automatically applies a certain level of security when you create a trail. However, there are additional steps you can take to help keep your data secure.
 - By default, the Amazon S3 bucket you created as part of creating a trail has a policy applied that allows CloudTrail to write log files to that bucket. The bucket is not publicly accessible, but it might be accessible to other users in your AWS account if they have permissions to read and write to buckets in your AWS account. Review the policy for your bucket and if necessary, make changes to restrict access to a specific set of IAM users. For more information, see the [Amazon S3 security documentation](#) and the [example walkthrough for securing a bucket](#).
 - 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 an [AWS KMS key](#). For more information, see [Encrypting CloudTrail log files with AWS KMS-managed keys \(SSE-KMS\)](#) (p. 259).
 - For additional security planning, review the [security best practices for CloudTrail](#) (p. 255).
- **Create a trail to log data events.** If you are interested in logging when objects are added, retrieved, and deleted in one or more Amazon S3 buckets, when items are added, changed, or deleted in DynamoDB tables, or when one or more AWS Lambda functions are invoked, these are data events. The management event trail you created earlier in this tutorial doesn't log these types of events. You

can create a separate trail specifically to log data events for some or all of supported resources. For more information, see [Data events \(p. 140\)](#).

Note

Additional charges apply for logging data events. For more information, see [AWS CloudTrail Pricing](#).

- **Log CloudTrail Insights events on your trail.** CloudTrail Insights helps you identify and respond to unusual or anomalous activity associated with `write` API calls by continuously analyzing CloudTrail management events. CloudTrail Insights uses mathematical models to determine the normal levels of API and service event activity for an account. It identifies behavior that is outside normal patterns, generates Insights events, and delivers those events to a `/CloudTrail-Insight` folder in the chosen destination S3 bucket for your trail. For more information about CloudTrail Insights, see [Logging Insights events for trails \(p. 154\)](#).

Note

Additional charges apply for logging Insights events. For more information, see [AWS CloudTrail Pricing](#).

- **Set up CloudWatch Logs alarms to alert you when certain events occur.** CloudWatch Logs lets you monitor and receive alerts for specific events captured by CloudTrail. For example, you can monitor key security and network-related management events, such as [security group changes \(p. 177\)](#), [failed AWS Management Console sign-in events \(p. 178\)](#), or [changes to IAM policies \(p. 179\)](#). For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).
- **Use analysis tools to identify trends in your CloudTrail logs.** While the filters in Event history can help you find specific events or event types in your recent activity, it does not provide the ability to search through activity over longer time periods. For deeper and more sophisticated analysis, you can use Amazon Athena. For more information, see [Querying AWS CloudTrail Logs](#) in the Amazon Athena User Guide.

Working with CloudTrail

CloudTrail is enabled by default for your AWS account. You can use **Event history** in the CloudTrail console to view, search, download, archive, analyze, and respond to account activity across your AWS infrastructure. This includes activity made through the AWS Management Console, AWS Command Line Interface, and AWS SDKs and APIs.

For an ongoing record of events in your AWS account, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs.

If you have created an organization in AWS Organizations, you can create a trail that will log all events for all AWS accounts in that organization. Creating an organization trail helps you define a uniform event logging strategy for your organization.

Topics

- [Viewing events with CloudTrail Event history \(p. 44\)](#)
- [Viewing CloudTrail Insights events \(p. 55\)](#)
- [Creating a trail for your AWS account \(p. 69\)](#)
- [Creating a trail for an organization \(p. 112\)](#)
- [Getting and viewing your CloudTrail log files \(p. 124\)](#)
- [Configuring Amazon SNS notifications for CloudTrail \(p. 127\)](#)
- [Controlling user permissions for CloudTrail \(p. 128\)](#)
- [Tips for managing trails \(p. 129\)](#)
- [Using AWS CloudTrail with interface VPC endpoints \(p. 131\)](#)

Viewing events with CloudTrail Event history

You can troubleshoot operational and security incidents over the past 90 days in the CloudTrail console by viewing **Event history**. You can look up events related to creation, modification, or deletion of resources (such as IAM users or Amazon EC2 instances) in your AWS account on a per-region basis. Events can be viewed and downloaded by using the AWS CloudTrail console. You can customize the view of event history in the console by selecting which columns are displayed and which are hidden. You can programmatically look up events by using the AWS SDKs or AWS Command Line Interface. You can also compare the details of events in **Event history** side-by-side.

Note

Over time, AWS services might add additional events. CloudTrail will record these events in **Event history**, but a full 90-day record of activity that includes added events will not be available until 90 days after the events are added.

This section describes how to look up events by using the CloudTrail console and the AWS CLI. It also describes how to download a file of events. For information on using the `LookupEvents` API to retrieve information from CloudTrail events, see the [AWS CloudTrail API Reference](#).

For information on creating a trail so that you have a record of events that extends past 90 days, see [Creating a trail \(p. 70\)](#) and [Getting and viewing your CloudTrail log files \(p. 124\)](#).

Topics

- [Viewing CloudTrail events in the CloudTrail console \(p. 45\)](#)
- [Viewing CloudTrail events with the AWS CLI \(p. 49\)](#)

Viewing CloudTrail events in the CloudTrail console

You can use the CloudTrail console to view the last 90 days of recorded API activity (management events) in an AWS Region. You can also download a file with that information, or a subset of information based on the filter and time range you choose. You can customize your view of **Event history** by selecting which columns are displayed in the console. You can also look up and filter events by the resource types available for a particular service. You can select up to five events in **Event history** and compare their details side-by-side.

Event history does not show data events. To view data events, [create a trail \(p. 69\)](#).

After 90 days, events are no longer shown in **Event history**. You cannot manually delete events from **Event history**. When you [create a trail \(p. 70\)](#), you can view events that are logged to your trail for as long as you store them in the S3 bucket that is configured in your trail settings.

CloudTrail logging varies between AWS services. While most AWS services support CloudTrail logging of all events, some services only support logging a subset of APIs and events, and a few services are unsupported. You can learn more about the specifics of how CloudTrail logs events for a specific service by consulting the documentation for that service. For more information, see [CloudTrail supported services and integrations \(p. 20\)](#).

Note

For an ongoing record of activity and events, [create a trail \(p. 70\)](#). Creating a trail also enables you to take advantage of the following integrations:

- A trail lets you log CloudTrail Insights events, which can help you identify and respond to unusual activity associated with `write` management API calls. For more information, see [Logging Insights events for trails \(p. 154\)](#).
- Analyze your AWS service activity with queries in Amazon Athena. For more information, see [Creating a Table for CloudTrail Logs in the CloudTrail Console](#) in the [Amazon Athena User Guide](#), or choose the option to create a table directly from **Event history** in the CloudTrail console.
- Monitor your trail logs and be notified when specific activity occurs with Amazon CloudWatch Logs. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).
- A trail lets you exclude AWS Key Management Service (AWS KMS) or Amazon Relational Database Service Data API events. AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. Events cannot be excluded from **Event history**; you can only exclude events if you create or update a trail to log management events.

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 **Event history**.

A filtered list of events appears in the content pane with the latest event first. Scroll down to see more events.

3. To compare events, select up to five events by filling their check boxes in the left margin of the **Event history** table. View details for selected events side-by-side in the **Compare event details** table.

The default view of events in **Event history** has a filter applied so that it does not display read-only events. To remove this filter, or to apply other filters, change the filter settings. For more information, see [Filtering CloudTrail events \(p. 46\)](#).

Contents

- [Displaying CloudTrail events \(p. 46\)](#)
- [Filtering CloudTrail events \(p. 46\)](#)
- [Viewing details for an event \(p. 48\)](#)
- [Downloading events \(p. 48\)](#)
- [Viewing resources referenced with AWS Config \(p. 48\)](#)

Displaying CloudTrail events

You can customize the display of **Event history** by selecting which columns to display in the CloudTrail console. By default, the following columns are displayed:

- **Event name**
- **Event time**
- **User name**
- **Event source**
- **Resource type**
- **Resource name**

Note

You cannot change the order of the columns, or manually delete events from **Event history**.

To customize the columns displayed in Event history

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 **Event history**.
3. Choose the gear icon.
4. In **Select visible columns**, select the columns you want to display. Turn off columns you do not want to display. When you have finished, choose **Confirm**.

Filtering CloudTrail events

The default display of events in **Event history** uses an attribute filter to exclude read-only events from the list of displayed events. This attribute filter is named **Read-only**, and it is set to **false**. You can remove this filter to display both read and write events. To view only **Read** events, you can change the filter value to **true**. You can also filter events by other attributes. You can additionally filter by time range.

Note

You can only apply one attribute filter and a time range filter. You cannot apply multiple attribute filters.

AWS access key

The AWS 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.

Event ID

The CloudTrail ID of the event. Each event has a unique ID.

Event name

The name of the event. For example, you can filter on IAM events, such as `CreatePolicy`, or Amazon EC2 events, such as `RunInstances`.

Event source

The AWS service to which the request was made, such as `iam.amazonaws.com` or `s3.amazonaws.com`. You can scroll through a list of event sources after you choose the **Event source** filter.

Read only

The read type of the event. Events are categorized as read events or write events. If set to **false**, read events are not included in the list of displayed events. By default, this attribute filter is applied and the value is set to **false**.

Resource name

The name or ID of the resource referenced by the event. For example, the resource name might be "auto-scaling-test-group" for an Auto Scaling group or "i-12345678910" for an EC2 instance.

Resource type

The type of resource referenced by the event. For example, a resource type can be `Instance` for EC2 or `DBInstance` for RDS. Resource types vary for each AWS service.

Time range

The time range in which you want to filter events. You can filter events for the last 90 days.

User name

The identity of the user referenced by the event. For example, this can be an IAM user, an IAM role name, or a service role.

If there are no events logged for the attribute or time that you choose, the results list is empty. You can apply only one attribute filter in addition to the time range. If you choose a different attribute filter, your specified time range is preserved.

The following steps describe how to filter by attribute.

To filter by attribute

1. To filter the results by an attribute, choose an attribute from the **Lookup attributes** drop-down list, and then type or choose a value for the attribute in the text box.
2. To remove an attribute filter, choose the **X** at the right of the attribute filter box.

The following steps describe how to filter by a start and end date and time.

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 a time range in the time range bar. Preset values are 30 minutes, 1 hour, 3 hours, or 12 hours. To specify a custom time range, choose **Custom**.
2. To remove a time range filter, choose **Clear** in the time range bar.

Viewing details for an event

1. Choose an event in the results list to show its details.
2. Resources referenced in the event are shown in the **Resources referenced** table on the event details page.
3. Some referenced resources have links. Choose the link to open the console for that resource.
4. Scroll to **Event record** on the details page to see the JSON event record, also called the event *payload*.
5. Choose **Event history** in the page breadcrumb to close the event details page and return to **Event history**.

Downloading events

You can download recorded event history as a file in CSV or JSON format. Use filters and time ranges to reduce the size of the file you download.

Note

CloudTrail event history files are data files that contain information (such as resource names) that can be configured by individual users. Some data can potentially be interpreted as commands in programs used to read and analyze this data (CSV injection). For example, when CloudTrail events are exported to CSV and imported to a spreadsheet program, that program might warn you about security concerns. You should choose to disable this content to keep your system secure. Always disable links or macros from downloaded event history files.

1. Add a filter and time range for events in **Event history** that you want to download. For example, you can specify the event name, `StartInstances`, and specify a time range for the last three days of activity.
2. Choose **Download events**, and then choose **Download as CSV** or **Download as JSON**. The download starts immediately.

Note


Your download might take some time to complete. For faster results, before you start the download process, use a more specific filter or a shorter time range to narrow the results. You can cancel a download. If you cancel a download, a partial download including only some event data might be on your local computer. To download the full event history, restart the download.

3. After your download is complete, open the file to view the events that you specified.
4. To cancel your download, choose **Cancel**, and then confirm by choosing **Cancel download**. If you need to restart a download, wait until the earlier download is finished canceling.

Viewing resources referenced with AWS Config

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

On the **Resources Referenced** pane, choose the  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. Choose 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 **Link not available** appears in the column, the resource can't be viewed for one of the following reasons:

- AWS Config doesn't support the resource type. For more information, see [Supported Resources, Configuration Items, and Relationships](#) in the *AWS Config Developer Guide*.
- AWS Config recently added support for the resource type, but it's not yet available from the CloudTrail console. You can look up the resource in the AWS Config console to see the timeline for the resource.
- 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.
- 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 **Event history** page shows the `CreateUser` event and **Bob-user** as an IAM resource. You can choose 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 **Event history** page shows the `UpdateUser` event and **Bob-admin** as the updated IAM resource.
5. You can choose the icon to view the **Bob-admin** IAM resource in the timeline. However, you can't choose 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 \(p. 242\)](#).

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

Viewing CloudTrail events with the AWS CLI

Important

As of November 22, 2021, AWS CloudTrail will change how trails can be used to capture global service events. After the change, events created by CloudFront, IAM, and AWS STS will be recorded in the region in which they were created, the US East (N. Virginia) region, `us-east-1`. This makes CloudTrail's treatment of these services consistent with that of other AWS global services.

To continue receiving global service events outside of US East (N. Virginia), be sure to convert *single-region trails* using global service events outside of US East (N. Virginia) into *multi-region trails*. Also update the region of your `lookup-events` API calls. For more information about updating lookup events, including an example CLI command, see [Looking up events by attribute \(p. 52\)](#) later in this section.

You can look up CloudTrail events for the last 90 days using the **`aws cloudtrail lookup-events`** command. The `lookup-events` command 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 general information on using the AWS Command Line Interface, see the [AWS Command Line Interface User Guide](#).

Contents

- [Prerequisites \(p. 50\)](#)
- [Getting command line help \(p. 50\)](#)
- [Looking up events \(p. 50\)](#)
- [Specifying the number of events to return \(p. 51\)](#)
- [Looking up events by time range \(p. 51\)](#)
 - [Valid <timestamp> formats \(p. 51\)](#)
- [Looking up events by attribute \(p. 52\)](#)
 - [Attribute lookup examples \(p. 52\)](#)
- [Specifying the next page of results \(p. 53\)](#)
- [Getting JSON input from a file \(p. 53\)](#)
- [Lookup output fields \(p. 54\)](#)

Prerequisites

- To run AWS CLI commands, you must 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+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSqq66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juy3CIZW8=",
  "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-2\",
    \"sourceIPAddress\": \"203.0.113.4\",
    \"userAgent\": \"Mozilla/5.0\",
    \"requestParameters\": null,
    \"responseElements\": {\"ConsoleLogin\": \"Success\"},
    \"additionalEventData\": {
        \"MobileVersion\": \"No\",
        \"LoginTo\": \"https://console.aws.amazon.com/console/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 \(p. 54\)](#) later in this document. For an explanation of the fields in the CloudTrail event, see [CloudTrail record contents \(p. 272\)](#).

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 90 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 90 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>
```

You can specify only one attribute key/value pair for each **lookup-events** command. The following are values for `AttributeKey`. Value names are case sensitive.

- AccessKeyId
- EventId
- EventName
- EventSource
- ReadOnly
- ResourceName
- ResourceType
- Username

Attribute lookup examples

The following example command returns events in region US East (N. Virginia) region, us-east-1 which allows you to view global service events. Replace *ConsoleLogin*, and *gseService* with appropriate values for your configuration.

```
aws cloudtrail --region us-east-1 lookup-events --lookup-
attributes AttributeKey=EventName,AttributeValue=ConsoleLogin
  AttributeKey=EventSource,AttributeValue=gseService
```

The following example command returns events in which the value of `AccessKeyId` is `AKIAIOSFODNN7EXAMPLE`.

```
aws cloudtrail lookup-events --lookup-attributes
  AttributeKey=AccessKeyId,AttributeValue=AKIAIOSFODNN7EXAMPLE
```

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 `EventSource` is `iam.amazonaws.com`.

```
aws cloudtrail lookup-events --lookup-attributes  
AttributeKey=EventSource,AttributeValue=iam.amazonaws.com
```

The following example command returns write events. It excludes read events such as `GetBucketLocation` and `DescribeStream`.

```
aws cloudtrail lookup-events --lookup-attributes AttributeKey=ReadOnly,AttributeValue=false
```

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
```

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 `Username` is `root`.

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

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+  
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSgg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juy3CIZW8=
```

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

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.

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](#).

EventId

A string that contains the GUID of the event returned.

EventName

A string that contains the name of the event returned.

EventSource

The AWS service that the request was made to.

EventTime

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

Resources

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

ResourceName

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

ResourceType

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

Username

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

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.

Viewing CloudTrail Insights events

After you enable CloudTrail Insights on a trail, you can view up to 90 days of Insights events by using the CloudTrail console or the AWS CLI. This section describes how to view, look up, and download a file of Insights events. For information about using the `LookupEvents` API to retrieve information from CloudTrail events, see the [AWS CloudTrail API Reference](#). For more information about CloudTrail Insights, see [Logging Insights events for trails \(p. 154\)](#) in this guide.

For information about how to create a trail so that you have a record of events that extends past 90 days, see [Creating a trail \(p. 70\)](#) and [Getting and viewing your CloudTrail log files \(p. 124\)](#).

Topics

- [Viewing CloudTrail Insights events in the CloudTrail console \(p. 56\)](#)
- [Viewing CloudTrail Insights events with the AWS CLI \(p. 61\)](#)

Viewing CloudTrail Insights events in the CloudTrail console

After you enable CloudTrail Insights events on a trail, when CloudTrail detects unusual write management API activity, CloudTrail generates Insights events and displays them on the **Dashboard** and **Insights** pages in the AWS Management Console. You can view the Insights events in the console and troubleshoot the unusual activity. The most recent 90 days of Insights events are shown in the console. You can also download Insights events by using the AWS CloudTrail console. You can programmatically look up events by using the AWS SDKs or AWS Command Line Interface. For more information about CloudTrail Insights events, see [Logging Insights events for trails \(p. 154\)](#) in this guide.

After Insights events are logged, the events are shown on the **Insights** page for 90 days. You cannot manually delete events from the **Insights** page. Because you must [create a trail \(p. 70\)](#) before you can enable CloudTrail Insights, you can view Insights events that are logged to your trail for as long as you store them in the S3 bucket that is configured in your trail settings.

Monitor your trail logs and be notified when specific Insights events activity occurs with Amazon CloudWatch Logs. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).

To view Insights events

CloudTrail Insights events must be enabled on your trail to see Insights events in the console. Allow up to 36 hours for CloudTrail to deliver the first Insights events, if unusual activity is detected.

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 **Dashboard** to see the five most recent Insights events, or **Insights** to see all Insights events logged in your account in the last 90 days.

On the **Insights** page, you can filter Insights events by criteria including event API source, event name, and event ID, and limit the events displayed to those occurring within a specific time range. For more information about filtering Insights events, see [Filtering Insights events \(p. 56\)](#).

Contents

- [Filtering Insights events \(p. 56\)](#)
- [Viewing Insights events details \(p. 48\)](#)
- [Zoom, pan, and download graph \(p. 59\)](#)
- [Change graph time span settings \(p. 59\)](#)
- [Downloading Insights events \(p. 61\)](#)

Filtering Insights events

The default display of events in **Insights** shows events in reverse chronological order. The newest Insights events, sorted by event start time, are at the top. The following list describes the available attributes.

Event name

The name of the event, typically the AWS API on which unusual levels of activity were recorded.

Event source

The AWS service to which the request was made, such as `iam.amazonaws.com` or `s3.amazonaws.com`. You can scroll through a list of event sources after you choose the **Event source** filter.

Event ID

The ID of the Insights event. Event IDs are not shown in the **Insights** page table, but they are an attribute on which you can filter Insights events. The event IDs of management events that are analyzed to generate Insights events are different from the event IDs of Insights events.

Event start time

The start time of the Insights event, measured as the first minute in which unusual API activity was recorded. This attribute is shown in the **Insights** table, but you cannot filter on event start time in the console.

If there are no events logged for the attribute or time that you choose, the results list is empty. You can apply only one attribute filter in addition to the time range. If you choose a different attribute filter, your specified time range is preserved.

The following steps describe how to filter by attribute.

To filter by attribute

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

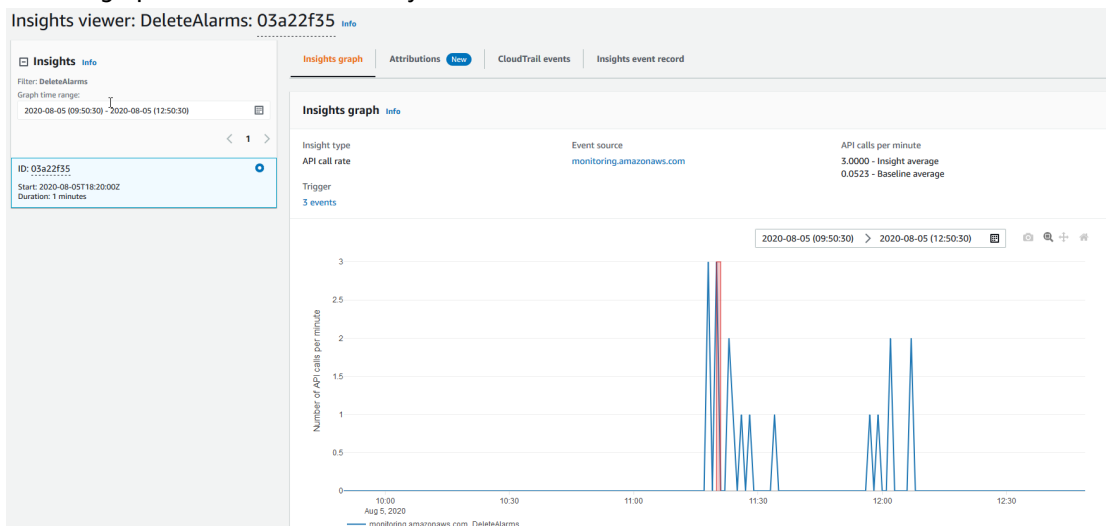
The following steps describe how to filter by a start and end date and time.

To filter by a start and end date and time

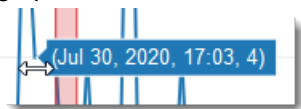
1. To narrow the time range for the events that you want to see, choose a time range on the time span bar at the top of the table.. Preset time ranges include 30 minutes, 1 hour, 3 hours, or 12 hours. To specify a custom time range, choose **Custom**.
2. Choose one of the following tabs.
 - **Absolute** - Lets you choose a specific time. Go on to the next step.
 - **Relative to selected event** - Selected by default. Lets you choose a time period relative to the start time of an Insights event. Go on to step 4.
3. To set an **Absolute** time range, do the following.
 - a. On the **Absolute** tab, choose the day that you want the time range to start. Enter a start time on the selected day. To enter a date manually, type the date in the format `yyyy/mm/dd`. The start and end times use a 24-hour clock, and values must be in the format `hh:mm:ss`. For example, to indicate a 6:30 p.m. start time, enter **18:30:00**.
 - b. Choose an end date for the range on the calendar, or specify an end date and time below the calendar. Choose **Apply**.
4. To set a **Relative to selected event** time range, do the following.
 - a. Choose a preset time period relative to the start time of Insights events. Preset values are available in minutes, hours, days, or weeks. The maximum relative time period is 12 weeks.
 - b. If needed, customize the preset value in the boxes below the presets. Choose **Clear** to reset your changes if needed. When you have set the relative time that you want, choose **Apply**.
5. In **To**, choose the day and specify the time that you want to be the end of the time range. Choose **Apply**.
6. To remove a time range filter, choose the calendar icon on the right of the **Time range** box, and then choose **Remove**.

Viewing Insights events details

1. Choose an Insights event in the results list to show its details. The details page for an Insights event shows a graph of the unusual activity timeline.



2. Hover over the highlighted bands to show the start time and duration of each Insights event in the graph.



The following information is shown in the graph legend:

- **Insight type.** Currently, this is API call rate.
 - **Trigger.** This is a link to the **Cloudtrail events** tab, which lists the management events that were analyzed to determine that unusual activity occurred.
 - **API calls per minute**
 - **Baseline average** - The typical rate of calls per minute to this API, as measured within approximately the preceding seven days, in a specific Region in your account.
 - **Insights average** - The rate of calls per minute to this API that triggered the Insights event. The CloudTrail Insights average for the start event is the rate of calls per minute to the API that triggered the Insights event. Typically, this is the first minute of unusual activity. The Insights average for the end event is the rate of API calls per minute over the duration of the unusual activity, between the start Insights event and the end Insights event.
 - **Event source.** The AWS service endpoint on which the unusual number of API calls was made. In the preceding image, the source is `monitoring.amazonaws.com`, which is the service endpoint for Amazon CloudWatch.
 - Details about the Insights event
 - **Event start time** - The minute during which unusual activity was first recorded.
 - **Event duration** - The duration, in minutes, of the Insights event.
3. Choose the **Attributions** tab to view information about the user identities, user agents, and error codes correlated with unusual and baseline activity. A maximum of five user identities, five user agents, and five error codes are shown in tables on the **Attributions** tab, sorted by an average of the count of activity, in descending order from highest to lowest. For more information about

the **Attributions** tab, see [Attributions tab \(p. 156\)](#) and [CloudTrail Insights insightDetails element \(p. 288\)](#) in this guide.

4. On the **CloudTrail events** tab, view related events that CloudTrail analyzed to determine that unusual activity occurred. By default, a filter is already applied for the Insights event name, which is also the name of the related API. The **CloudTrail events** tab shows CloudTrail management events related to the subject API that occurred between the start time (minus one minute) and end time (plus one minute) of the Insights event.

As you select other Insights events in the graph, the events shown in the **CloudTrail events** table change. These events help you perform deeper analysis to determine the probable cause of an Insights event and reasons for unusual API activity.

To show all CloudTrail events that were logged during the Insights event duration, and not only those for the related API, turn off the filter.

5. Choose the **Insights event record** tab to view the Insights start and end events in JSON format.
6. Choosing the linked **Event source** returns you to the **Insights** page, filtered by that event source.

Zoom, pan, and download graph

You can zoom, pan, and reset the axes of the graph on the Insights event details page by using a toolbar in the upper right corner.



From left to right, the command buttons on the graph toolbar do the following:

- **Download plot as a PNG** - Download the graph image shown on the details page, and save it in PNG format.
- **Zoom** - Drag to select an area on the graph that you want to enlarge and see in greater detail.
- **Pan** - Shift the graph to see adjacent dates or times.
- **Reset axes** - Change graph axes back to the original, clearing zoom and pan settings.

Change graph time span settings

You can change the time span—the selected duration of the events shown on the x axis—that is shown in the graph by choosing a setting in the graph's upper right corner.

2020-08-05 (09:50:30) > 2020-08-05 (12:50:30)

The default time span that is shown in the graph depends on the duration of the selected Insights event.

| Duration of Insights event | Default time span |
|----------------------------|-------------------------|
| Less than 4 hours | 3h (three hours) |
| Between 4 and 12 hours | 12h (12 hours) |
| Between 12 and 24 hours | 1d (one day) |
| Between 24 and 72 hours | 3d (three days) |

| Duration of Insights event | Default time span |
|----------------------------|-------------------|
| Longer than 72 hours | 1w (one week) |

You can choose presets of five minutes, 30 minutes, one hour, three hours, 12 hours, or **Custom**. The following image shows **Relative to selected event** time periods you can choose in **Custom** settings. Relative time periods are approximate time periods surrounding the start and end of the selected Insights event that is displayed on an Insights event details page.

Absolute **Relative to selected event** Local time zone ▼

Minutes 5 10 15 30 45

Hours 1 2 3 6 8 12

Days 1 2 3 4 5 6

Weeks 1 2 3 4

45 Minutes ▼

To customize a selected preset, specify a number and time unit in the boxes below the presets.

To specify an exact date and time range, choose the **Absolute** tab. If you set an absolute date and time range, start and end times are required. For information about how to set the time, see [the section called “Filtering Insights events” \(p. 56\)](#) in this topic.

Absolute **Relative to selected event** Local time zone ▼

< **August 2020** **September 2020** >

Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

2020/08/05 09:50:30 2020/08/05 12:50:30

Downloading Insights events

You can download recorded Insights event history as a file in CSV or JSON format. Use filters and time ranges to reduce the size of the file you download.

Note

CloudTrail event history files are data files that contain information (such as resource names) that can be configured by individual users. Some data can potentially be interpreted as commands in programs used to read and analyze this data (CSV injection). For example, when CloudTrail events are exported to CSV and imported to a spreadsheet program, that program might warn you about security concerns. As a security best practice, disable links or macros from downloaded event history files.

1. Specify the filter and time range for events you want to download. For example, you can specify the event name, `StartInstances`, and specify a time range for the last three days of activity.
2. Choose **Download events**, and then choose **Download CSV** or **Download JSON**. You are prompted to choose a location to save the file.

Note

Your download might take some time to complete. For faster results, before you start the download process, use a more specific filter or a shorter time range to narrow the results.

3. After your download is complete, open the file to view the events that you specified.
4. To cancel your download, choose **Cancel download**. If you cancel a download before it is finished, a CSV or JSON file on your local computer might contain only part of your events.

Viewing CloudTrail Insights events with the AWS CLI

You can look up CloudTrail Insights events for the last 90 days by running the **aws cloudtrail lookup-events** command. The `lookup-events` command has the following options:

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

These options are explained in this topic. For general information about using the AWS Command Line Interface, see the [AWS Command Line Interface User Guide](#).

Contents

- [Prerequisites \(p. 62\)](#)
- [Getting command line help \(p. 62\)](#)
- [Looking up Insights events \(p. 62\)](#)
- [Specifying the number of Insights events to return \(p. 65\)](#)
- [Looking up Insights events by time range \(p. 65\)](#)
 - [Valid <timestamp> formats \(p. 65\)](#)
- [Looking up Insights events by attribute \(p. 66\)](#)

- [Attribute lookup examples \(p. 66\)](#)
- [Specifying the next page of results \(p. 66\)](#)
- [Getting JSON input from a file \(p. 67\)](#)
- [Lookup output fields \(p. 68\)](#)

Prerequisites

- To run AWS CLI commands, you must install the AWS CLI. For more 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 Insights events

To see the ten latest Insights events, type the following command.

```
aws cloudtrail lookup-events --event-category insight
```

A returned event looks similar to the following example,

```
{
  "NextToken": "kBot5LlZe+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSgg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juEXAMPLE=",
  "Events": [
    {
      "eventVersion": "1.07",
      "eventTime": "2019-10-15T21:13:00Z",
      "awsRegion": "us-east-1",
      "eventID": "EXAMPLE-9b6f-45f8-bc6b-9b41c052ebc7",
      "eventType": "AwsCloudTrailInsight",
      "recipientAccountId": "123456789012",
      "sharedEventID": "EXAMPLE8-02b2-4e93-9aab-08ed47ea5fd3",
      "insightDetails": {
        "state": "Start",
        "eventSource": "autoscaling.amazonaws.com",
        "eventName": "CompleteLifecycleAction",
        "insightType": "ApiCallRateInsight",
        "insightContext": {
          "statistics": {
            "baseline": {
              "average": 0.0000882145
            },
            "insight": {
```

```
        "average": 0.6
      },
      "insightDuration": 5,
      "baselineDuration": 11336
    },
    "attributions": [
      {
        "attribute": "userIdentityArn",
        "insight": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole2",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole3",
            "average": 0.2
          }
        ],
        "baseline": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "userAgent",
        "insight": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.6
          }
        ],
        "baseline": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "errorCode",
        "insight": [
          {
            "value": "null",
            "average": 0.6
          }
        ],
        "baseline": [
          {
            "value": "null",
            "average": 0.0000882145
          }
        ]
      }
    ]
  },
  "eventCategory": "Insight"
},
{
  "eventVersion": "1.07",
```

```
"eventTime": "2019-10-15T21:14:00Z",
"awsRegion": "us-east-1",
"eventID": "EXAMPLEc-9eac-4af6-8e07-26a5ae8786a5",
"eventType": "AwsCloudTrailInsight",
"recipientAccountId": "123456789012",
"sharedEventID": "EXAMPLE8-02b2-4e93-9aab-08ed47ea5fd3",
"insightDetails": {
  "state": "End",
  "eventSource": "autoscaling.amazonaws.com",
  "eventName": "CompleteLifecycleAction",
  "insightType": "ApiCallRateInsight",
  "insightContext": {
    "statistics": {
      "baseline": {
        "average": 0.0000882145
      },
      "insight": {
        "average": 0.6
      },
      "insightDuration": 5,
      "baselineDuration": 11336
    },
    "attributions": [
      {
        "attribute": "userIdentityArn",
        "insight": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole2",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole3",
            "average": 0.2
          }
        ],
        "baseline": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "userAgent",
        "insight": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.6
          }
        ],
        "baseline": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "errorCode",
        "insight": [
          {
            "value": "null",
```

```
        "average": 0.6
      }
    ],
    "baseline": [
      {
        "value": "null",
        "average": 0.0000882145
      }
    ]
  }
},
"eventCategory": "Insight"
}
]
```

For an explanation of the lookup-related fields in the output, see the section [Lookup output fields](#) (p. 68) later in this document. For an explanation of the fields in the Insights event, see [CloudTrail record contents](#) (p. 272).

Specifying the number of Insights events to return

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

```
aws cloudtrail lookup-events --event-category insight --max-results <integer>
```

The default value for *<integer>*, if it is not specified, is 10. Possible values are 1 through 50. The following example returns one result.

```
aws cloudtrail lookup-events --event-category insight --max-results 1
```

Looking up Insights events by time range

Insights events from the past 90 days are available for lookup. To specify a time range, type the following command.

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

--start-time <timestamp> specifies that only Insights 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 Insights 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 90 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 Insights events by attribute

To filter by an attribute, type the following command.

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

You can specify only one attribute key-value pair for each **lookup-events** command. The following are valid Insights event values for **AttributeKey**. Value names are case sensitive.

- EventId
- EventName
- EventSource

Attribute lookup examples

The following example command returns Insights events in which the value of **EventName** is **PutRule**.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
AttributeKey=EventName, AttributeValue=PutRule
```

The following example command returns Insights events in which the value of **EventId** is **b5cc8c40-12ba-4d08-a8d9-2bceb9a3e002**.

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

The following example command returns Insights events in which the value of **EventSource** is **iam.amazonaws.com**.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
AttributeKey=EventSource, AttributeValue=iam.amazonaws.com
```

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 --event-category insight <same parameters as previous command>
--next-token=<token>
```

In this command, 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 --event-category insight --lookup-attributes
  AttributeKey=EventName, AttributeValue=PutRule
```

To get the next page of results, your next command would look like the following.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
  AttributeKey=EventName,AttributeValue=PutRule --next-token=EXAMPLEZe+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSgg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YA1ju3oXd12juEXAMPLE=
```

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 information, see [Generate CLI Skeleton and CLI Input JSON Parameters](#).

To look up Insights 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 --event-category insight --generate-cli-skeleton >
  LookupEvents.txt
```

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

```
{
  "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 --event-category insight --cli-input-json file://  
LookupEvents.txt
```

Note

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

Lookup output fields

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.

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](#).

EventId

A string that contains the GUID of the event returned.

EventName

A string that contains the name of the event returned.

EventSource

The AWS service that the request was made to.

EventTime

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

Resources

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

ResourceName

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

ResourceType

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

Username

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

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.

For more information about CloudTrail Insights events, see [Logging Insights events for trails \(p. 154\)](#) in this guide.

Creating a trail for your AWS account

When you create a trail, you enable ongoing delivery of events as log files to an Amazon S3 bucket that you specify. Creating a trail has many benefits, including:

- A record of events that extends past 90 days.
- The option to automatically monitor and alarm on specified events by sending log events to Amazon CloudWatch Logs.
- The option to query logs and analyze AWS service activity with Amazon Athena.

Beginning on April 12, 2019, you can view trails only in the AWS Regions where they log events. If you create a trail that logs events in all AWS Regions, it appears in the console in all Regions. If you create a trail that only logs events in a single Region, you can view and manage it only in that Region. Creating a multi-region trail is the default option if you create a trail by using the AWS CloudTrail console, and is a recommended best practice. To create a single-region trail, you must use the AWS CLI.

If you use AWS Organizations, you can create a trail that will log events for all AWS accounts in the organization. A trail with the same name will be created in each member account, and events from each trail will be delivered to the Amazon S3 bucket that you specify.

Note

Only the management account for an organization can create a trail for the organization. Creating a trail for an organization automatically enables integration between CloudTrail and Organizations. For more information, see [Creating a trail for an organization \(p. 112\)](#).

Topics

- [Creating and updating a trail with the console \(p. 69\)](#)
- [Creating, updating, and managing trails with the AWS Command Line Interface \(p. 87\)](#)

Creating and updating a trail with the console

You can create, update, or delete your trails with the CloudTrail console. When you create a trail in the CloudTrail console, you create a multi-region trail. To create a trail that logs events in only one region, [use the AWS CLI \(p. 90\)](#).

You can create up to five trails for each region. After you create a trail, CloudTrail automatically starts logging API calls and related events in your account to the Amazon S3 bucket that you specify. To stop logging, you can turn off logging for the trail or delete it.

Creating and updating trails in the CloudTrail console has several features not available when creating a trail using the AWS CLI. For example, if you are configuring a trail to log data events for Amazon S3 buckets or AWS Lambda functions in your AWS account, you can view a list of these resources while creating the trail in the console experience. If this is your first time creating a trail, creating a trail using the console will help you understand the features and options available for the trail.

For information specific to creating a trail for an organization in AWS Organizations, see [Creating a trail for an organization \(p. 112\)](#).

Topics

- [Creating a trail \(p. 70\)](#)
- [Updating a trail \(p. 79\)](#)

- [Deleting a trail \(p. 86\)](#)
- [Turning off logging for a trail \(p. 87\)](#)

Creating a trail

Follow the procedure to create a trail that applies to all Regions. A trail that applies to all Regions delivers log files from all Regions to an S3 bucket. After you create the trail, AWS CloudTrail automatically starts logging the events that you specified.

Note

After you create a trail, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see [CloudTrail supported services and integrations \(p. 20\)](#).

Creating a trail in the console (basic event selectors)

In the console, you create a trail that logs events in all AWS Regions [that you have enabled](#). This is a recommended best practice. To log events in a single region (not recommended), [use the AWS CLI \(p. 90\)](#).

Use the following procedure if you have not yet enabled advanced event selectors. If you have advanced event selectors enabled, see [Creating a trail in the console \(advanced event selectors\) \(p. 74\)](#) to configure data event logging on your trail.

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. On the CloudTrail service home page, the **Trails** page, or the **Trails** section of the **Dashboard** page, choose **Create trail**.
3. On the **Create Trail** page, for **Trail name**, type a name for your trail. For more information, see [CloudTrail trail naming requirements \(p. 130\)](#).
4. If this is an AWS Organizations organization trail, you can choose to enable the trail for all accounts in your organization. You only see this option if you are signed in to the console with an IAM user or role in the management account. To successfully create an organization trail, be sure that the user or role has [sufficient permissions \(p. 115\)](#). For more information, see [Creating a trail for an organization \(p. 112\)](#).
5. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies.

Note

If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket. The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

6. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files with SSE-KMS instead of SSE-S3. The default is **Enabled**. For more information about this encryption type, see [Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys \(SSE-S3\)](#).

If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see [Using multi-Region keys](#) in the *AWS Key Management Service Developer Guide*.

Note

You can also type the ARN of a key from another account. 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 [Configure AWS KMS key policies for CloudTrail \(p. 261\)](#).

7. In **Additional settings**, configure the following.
 - a. For **Log file validation**, choose **Enabled** 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 \(p. 196\)](#).
 - b. For **SNS notification delivery**, choose **Enabled** 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 more information, see [Configuring Amazon SNS notifications for CloudTrail \(p. 127\)](#).

If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions, SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see [Amazon SNS topic policy for CloudTrail \(p. 246\)](#).

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](#).

8. Optionally, configure CloudTrail to send log files to CloudWatch Logs by choosing **Enabled in CloudWatch Logs**. For more information, see [Sending events to CloudWatch Logs \(p. 160\)](#).
 - a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.
 - b. If you choose **Existing**, choose a log group from the drop-down list.
 - c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see [Role policy document for CloudTrail to use CloudWatch Logs for monitoring \(p. 182\)](#).

Note

When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

9. For **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see [AWS Resource Groups and Why use tags for trails? \(p. 8\)](#).
10. On the **Choose log events** page, choose the event types that you want to log. For **Management events**, do the following.
 - a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see [Management events \(p. 136\)](#).

- b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

- c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see [Logging Data API calls with AWS CloudTrail](#) in the *Amazon RDS User Guide for Aurora*.
11. For **Data events**, you can specify logging data events for Amazon S3 buckets, AWS Lambda functions, Amazon DynamoDB tables, or a combination of these resource types. By default, trails don't log data events. Additional charges apply for logging data events. For more information, see [Data events \(p. 140\)](#). For CloudTrail pricing, see [AWS CloudTrail Pricing](#). More data event types are available if you use advanced event selectors; for more information, see [Creating a trail in the console \(advanced event selectors\) \(p. 74\)](#) in this topic.

For Amazon S3 buckets:

- a. For **Data event source**, choose **S3**.
- b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

Note

Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If you are creating a trail for a single Region (done by using the AWS CLI), choosing **All current and future S3 buckets** enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

- c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.
- d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. Find specific buckets by typing a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

To remove a bucket from logging, choose **X**.

12. To add another data type on which to log data events, choose **Add data event type**.

13. For Lambda functions:

- a. For **Data event source**, choose **Lambda**.
- b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

Note

If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

- c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

Note

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN.

You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI \(p. 93\)](#).

14. For DynamoDB tables:

- a. For **Data event source**, choose **DynamoDB**.
- b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN uses the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

15. Choose **Insights events** if you want your trail to log CloudTrail Insights events.

In **Event type**, select **Insights events**. You must be logging **Write** management events to log Insights events.

CloudTrail Insights analyzes management **Write** events for unusual activity, and logs events when anomalies are detected. By default, trails don't log Insights events. For more information about Insights events, see [Logging Insights events for trails \(p. 154\)](#). Additional charges apply for logging Insights events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Insights events are delivered to a different folder named `/CloudTrail-Insight` of the same S3 bucket that is specified in the **Storage location** area of the trail details page. CloudTrail creates the new prefix for you. For example, if your current destination S3 bucket is named `S3bucketName/AWSLogs/CloudTrail/`, the S3 bucket name with a new prefix is named `S3bucketName/AWSLogs/CloudTrail-Insight/`.

16. When you are finished choosing event types to log, choose **Next**.

17. On the **Review and create** page, review your choices. Choose **Edit** in a section to change the trail settings shown in that section. When you are ready to create the trail, choose **Create trail**.
18. The new trail appears on the **Trails** page. The **Trails** page shows the trails in your account from all Regions. In about 15 minutes, CloudTrail publishes log files that show the AWS API calls made in your account. You can see the log files in the S3 bucket that you specified. It can take up to 36 hours for CloudTrail to deliver the first Insights event, if you have enabled Insights event logging, and unusual activity is detected.

Note

CloudTrail typically delivers logs within an average of about 15 minutes of an API call. This time is not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information.

Creating a trail in the console (advanced event selectors)

In the console, you create a trail that logs events in all AWS Regions. This is a recommended best practice. To log events in a single region (not recommended), [use the AWS CLI \(p. 90\)](#).

Use the following procedure if you have enabled advanced event selectors. If you prefer to use basic data event selectors, see [Creating a trail in the console \(basic event selectors\) \(p. 70\)](#) to configure data event logging on your trail.

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. On the CloudTrail service home page, the **Trails** page, or the **Trails** section of the **Dashboard** page, choose **Create trail**.
3. On the **Create Trail** page, for **Trail name**, type a name for your trail. For more information, see [CloudTrail trail naming requirements \(p. 130\)](#).
4. If this is an AWS Organizations organization trail, you can choose to enable the trail for all accounts in your organization. You only see this option if you are signed in to the console with an IAM user or role in the management account. To successfully create an organization trail, be sure that the user or role has [sufficient permissions \(p. 115\)](#). For more information, see [Creating a trail for an organization \(p. 112\)](#).
5. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies.

Note

If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket. The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

6. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files with SSE-KMS instead of SSE-S3. The default is **Enabled**. For more information about this encryption type, see [Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys \(SSE-S3\)](#).

If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see [Using multi-Region keys](#) in the *AWS Key Management Service Developer Guide*.

Note

You can also type the ARN of a key from another account. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). 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 [Configure AWS KMS key policies for CloudTrail \(p. 261\)](#).

7. In **Additional settings**, configure the following.
 - a. For **Log file validation**, choose **Enabled** 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 \(p. 196\)](#).
 - b. For **SNS notification delivery**, choose **Enabled** 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 more information, see [Configuring Amazon SNS notifications for CloudTrail \(p. 127\)](#).

If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions, SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see [Amazon SNS topic policy for CloudTrail \(p. 246\)](#).

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](#).

8. Optionally, configure CloudTrail to send log files to CloudWatch Logs by choosing **Enabled in CloudWatch Logs**. For more information, see [Sending events to CloudWatch Logs \(p. 160\)](#).
 - a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.
 - b. If you choose **Existing**, choose a log group from the drop-down list.
 - c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see [Role policy document for CloudTrail to use CloudWatch Logs for monitoring \(p. 182\)](#).

Note

When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

9. For **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see [AWS Resource Groups and Why use tags for trails? \(p. 8\)](#).
10. On the **Choose log events** page, choose the event types that you want to log. For **Management events**, do the following.

- a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see [Management events \(p. 136\)](#).
- b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

- c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see [Logging Data API calls with AWS CloudTrail](#) in the *Amazon RDS User Guide for Aurora*.
11. To log data events, choose **Data events**.
 12. For **Data event type**, choose the resource type on which you want to log data events.
 13. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

Note

Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail.

If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

14. If you want to apply a predefined log selector template, and you do not want to add another data event resource type, go on to Step 18. To apply a custom log selector template, go on to the next step.
15. To create a custom log selector template, in the **Log selector template** drop-down list, choose **Custom**.
16. Optionally, enter a name for your custom log selector template.
17. In **Advanced event selectors**, build an expression for the specific resources on which you want to log data events.

- a. Choose from the following fields. For fields that accept an array (more than one value), CloudTrail adds an OR between values.

- **readOnly** - readOnly can be set to **Equals** a value of true or false. Read-only data events are events that do not change the state of a resource, such as Get* or Describe* events. Write events add, change, or delete resources, attributes, or artifacts, such as Put*, Delete*, or Write* events. To log both read and write events, don't add a readOnly selector.
- **eventName** - eventName can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as PutBucket, PutItem, or GetSnapshotBlock. You can have multiple values for this field, separated by commas.
- **resources.type** - In the AWS Management Console, this field doesn't occur, because it's already populated by your choice of data event type from the **Data event type** drop-down list. In the AWS CLI and SDKs, resources.type can only use the **Equals** operator, and the value can be one of the following:
 - AWS::S3::Object
 - AWS::Lambda::Function
 - AWS::DynamoDB::Table
 - AWS::S3Outposts::Object
 - AWS::ManagedBlockchain::Node
 - AWS::S3ObjectLambda::AccessPoint
 - AWS::EC2::Snapshot
 - AWS::S3::AccessPoint
 - AWS::DynamoDB::Stream
- **resources.ARN** - You can use any operator with resources.ARN, but if you use **Equals** or **NotEquals**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of resources.type.

For example, when resources.type equals **AWS::S3::Object**, the ARN must be in one of the following formats. To log all data events for all objects in a specific S3 bucket, use the StartsWith operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

```
arn:partition:s3::bucket_name/  
arn:partition:s3::bucket_name/object_or_file_name/
```

When resources.type equals **AWS::Lambda::Function**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:lambda:region:account_ID:function:function_name
```

When resources.type equals **AWS::DynamoDB::Table**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

When resources.type equals **AWS::S3Outposts::Object**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:s3-outposts:region:account_ID:object_path
```

When `resources.type` equals **AWS::ManagedBlockchain::Node**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:managedblockchain:region:account_ID:nodes/node_ID
```

When `resources.type` equals **AWS::S3ObjectLambda::AccessPoint**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name
```

When `resources.type` equals **AWS::EC2::Snapshot**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:ec2:region::snapshot/snapshot_ID
```

When `resources.type` equals **AWS::S3::AccessPoint**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in one of the following formats. To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

```
arn:partition:s3:region:account_ID:accesspoint/access_point_name  
arn:partition:s3:region:account_ID:accesspoint/access_point_name/  
object/object_path
```

When `resources.type` equals **AWS::DynamoDB::Stream**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name/stream/date_time
```

For more information about the ARN formats of data event resources, see [Actions, resources, and condition keys](#) in the *AWS Identity and Access Management User Guide*.

- b. For each field, choose **+ Conditions** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **NotEquals**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

To add the second S3 bucket, choose **+ Conditions**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

Note

You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still log all functions with a predefined selector template, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI](#) (p. 93).

- c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.
 - d. Save changes to your custom selector template by choosing **Next**. Do not choose another log selector template, or leave this page, or your custom selectors will be lost.
 - e. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 12 through this step to configure advanced event selectors for the data event type.
18. Choose **Insights events** if you want your trail to log CloudTrail Insights events.

In **Event type**, select **Insights events**. You must be logging **Write** management events to log Insights events.

CloudTrail Insights analyzes management **Write** events for unusual activity, and logs events when anomalies are detected. By default, trails don't log Insights events. For more information about Insights events, see [Logging Insights events for trails \(p. 154\)](#). Additional charges apply for logging Insights events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Insights events are delivered to a different folder named `/CloudTrail-Insight` of the same S3 bucket that is specified in the **Storage location** area of the trail details page. CloudTrail creates the new prefix for you. For example, if your current destination S3 bucket is named `S3bucketName/AWSLogs/CloudTrail/`, the S3 bucket name with a new prefix is named `S3bucketName/AWSLogs/CloudTrail-Insight/`.

19. When you are finished choosing event types to log, choose **Next**.
20. On the **Review and create** page, review your choices. Choose **Edit** in a section to change the trail settings shown in that section. When you are ready to create the trail, choose **Create trail**.
21. The new trail appears on the **Trails** page. The **Trails** page shows the trails in your account from all Regions. In about 15 minutes, CloudTrail publishes log files that show the AWS API calls made in your account. You can see the log files in the S3 bucket that you specified. It can take up to 36 hours for CloudTrail to deliver the first Insights event, if you have enabled Insights event logging, and unusual activity is detected.

Note

CloudTrail typically delivers logs within an average of about 15 minutes of an API call. This time is not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information.

Next steps

After you create your trail, you can return to the trail to make changes:

- If you haven't already, you can configure CloudTrail to send log files to CloudWatch Logs. For more information, see [Sending events to CloudWatch Logs \(p. 160\)](#).
- Create a table and use it to run a query in Amazon Athena to analyze your AWS service activity. For more information, see [Creating a Table for CloudTrail Logs in the CloudTrail Console](#) in the [Amazon Athena User Guide](#).
- Add custom tags (key-value pairs) to the trail.
- To create another trail, open the **Trails** page, and choose **Add new trail**.

Updating a trail

To change trail settings, use the following procedure.

To update a single-region trail to log events in all regions, or update an all-region trail to log events in only a single region, you must use the AWS CLI. For more information about how to update a single-

region trail to log events in all regions, see [Converting a trail that applies to one Region to apply to all Regions \(p. 91\)](#). For more information about how to update an all-region trail to log events in a single region, see [Converting a multi-region trail to a single-region trail \(p. 91\)](#).

To update 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. In the navigation pane, choose **Trails**, and then choose a trail name.
3. In **General details**, choose **Edit** to change the following settings. You cannot change the name of a trail.
 - **Apply trail to my organization** - Change whether this trail is an AWS Organizations organization trail.
 - **Trail log location** - Change the name of the S3 bucket or prefix in which you are storing logs for this trail.
 - **Log file SSE-KMS encryption** - Choose to enable or disable encrypting log files with SSE-KMS instead of SSE-S3.
 - **Log file validation** - Choose to enable or disable validation of the integrity of log files.
 - **SNS notification delivery** - Choose to enable or disable Amazon Simple Notification Service (Amazon SNS) notifications that log files have been delivered to the bucket specified for the trail.
- a. To change the trail to an AWS Organizations organization trail, you can choose to enable the trail for all accounts in your organization. For more information, see [Creating a trail for an organization \(p. 112\)](#).
- b. To change the specified bucket in **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies.

Note

If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket. The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

- c. For **Log file SSE-KMS encryption**, choose **Enabled** to encrypt your log files with SSE-KMS instead of SSE-S3. The default is **Enabled**. For more information about this encryption type, see [Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys \(SSE-S3\)](#).

If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see [Using multi-Region keys](#) in the *AWS Key Management Service Developer Guide*.

Note

You can also type the ARN of a key from another account. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). 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 [Configure AWS KMS key policies for CloudTrail \(p. 261\)](#).

- d. For **Log file validation**, choose **Enabled** 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 \(p. 196\)](#).

- e. For **SNS notification delivery**, choose **Enabled** 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 more information, see [Configuring Amazon SNS notifications for CloudTrail \(p. 127\)](#).

If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions, SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see [Amazon SNS topic policy for CloudTrail \(p. 246\)](#).

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](#).

4. In **CloudWatch Logs**, choose **Edit** to change settings for sending CloudTrail log files to CloudWatch Logs. Choose **Enabled in CloudWatch Logs** to enable sending log files. For more information, see [Sending events to CloudWatch Logs \(p. 160\)](#).
 - a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.
 - b. If you choose **Existing**, choose a log group from the drop-down list.
 - c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see [Role policy document for CloudTrail to use CloudWatch Logs for monitoring \(p. 182\)](#).

Note

When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

5. In **Tags**, choose **Edit** to change, add, or delete tags on the trail. Add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see [AWS Resource Groups](#) and [Why use tags for trails? \(p. 8\)](#).
6. In **Management events**, choose **Edit** to change management event logging settings.
 - a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see [Management events \(p. 136\)](#).
 - b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

- c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see [Logging Data API calls with AWS CloudTrail](#) in the *Amazon RDS User Guide for Aurora*.

7. **Important**

Steps 7-9 are for configuring data events on your trail if you are using basic event selectors. If you are using advanced event selectors, see [Updating data event settings with advanced event selectors \(p. 84\)](#), then return to this procedure for Step 10 and forward.

In **Data events**, choose **Edit** to change data event logging settings. With basic event selectors, you can specify logging data events for Amazon S3 buckets, AWS Lambda functions, DynamoDB tables, or a combination of those resources. Additional data event types are supported with [advanced event selectors \(p. 84\)](#). By default, trails don't log data events. Additional charges apply for logging data events. For more information, see [Data events \(p. 140\)](#). For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

For Amazon S3 buckets:

- a. For **Data event source**, choose **S3**.
- b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

Note

Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If the trail applies only to one Region, choosing **All current and future S3 buckets** enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

- c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.
- d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. To find specific buckets, type a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

To remove a bucket from logging, choose **X**.

8. To add another data type on which to log data events, choose **Add data event type**.
9. For Lambda functions:
 - a. For **Data event source**, choose **Lambda**.
 - b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

Note

If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

- c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

Note

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI \(p. 93\)](#).

10. To add another data type on which to log data events, choose **Add data event type**.

11. For DynamoDB tables:

- a. For **Data event source**, choose **DynamoDB**.
- b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN is in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

12. In **Insights events** choose **Edit** if you want your trail to log CloudTrail Insights events.

In **Event type**, select **Insights events**. You must be logging **Write** management events to log Insights events.

CloudTrail Insights analyzes management **Write** events for unusual activity, and logs events when anomalies are detected. By default, trails don't log Insights events. For more information about Insights events, see [Logging Insights events for trails \(p. 154\)](#). Additional charges apply for logging Insights events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Insights events are delivered to a different folder named `/CloudTrail-Insight` of the same S3 bucket that is specified in the **Storage location** area of the trail details page. CloudTrail creates the new prefix for you. For example, if your current destination S3 bucket is named `S3bucketName/AWSLogs/CloudTrail/`, the S3 bucket name with a new prefix is named `S3bucketName/AWSLogs/CloudTrail-Insight/`.

13. When you are finished changing settings on your trail, choose **Update trail**.

Updating data event settings with advanced event selectors

1. On the trail's details page, in **Data events**, choose **Edit**.
2. If you are not already logging data events, choose **Data events**.
3. For **Data event type**, choose the resource type on which you want to log data events.
4. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

Note

Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail.

If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

5. If you want to apply a predefined log selector template, and you do not want to add another data event resource type, choose **Save changes**, and skip to the last step of this procedure. To apply a custom log selector template, go on to the next step.
6. To create a custom log selector template, in the **Log selector template** drop-down list, choose **Custom**.
7. Optionally, enter a name for your custom log selector template.
8. In **Advanced event selectors**, build an expression for the specific resources on which you want to collect data events.
 - a. Choose from the following fields. For fields that accept an array (more than one value), CloudTrail adds an OR between values.
 - **readOnly** - readOnly can be set to **Equals** a value of true or false. To log both read and write events, don't add a readOnly selector.
 - **eventName** - eventName can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as PutBucket or GetSnapshotBlock. You can have multiple values for this field, separated by commas.
 - **resources.type** - In the AWS Management Console, this field doesn't occur, because it's already populated by your choice of data event type from the **Data event type** drop-down list. In the AWS CLI and SDKs, resources.type can only use the **Equals** operator, and the value can be one of the following:
 - `AWS::S3::Object`
 - `AWS::Lambda::Function`
 - `AWS::DynamoDB::Table`
 - `AWS::S3Outposts::Object`
 - `AWS::ManagedBlockchain::Node`

- `AWS::S3ObjectLambda::AccessPoint`
- `AWS::EC2::Snapshot`
- `AWS::S3::AccessPoint`
- `AWS::DynamoDB::Stream`
- **resources.ARN** - You can use any operator with `resources.ARN`, but if you use **Equals** or **NotEquals**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of `resources.type`.

For example, when `resources.type` equals **AWS::S3::Object**, the ARN must be in one of the following formats. To log all data events for all objects in a specific S3 bucket, use the **StartsWith** operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

```
arn:partition:s3::bucket_name/  
arn:partition:s3::bucket_name/object_or_file_name/
```

When `resources.type` equals **AWS::Lambda::Function**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:lambda:region:account_ID:function:function_name
```

When `resources.type` equals **AWS::DynamoDB::Table**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

When `resources.type` equals **AWS::S3Outposts::Object**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:s3-outposts:region:account_ID:object_path
```

When `resources.type` equals **AWS::ManagedBlockchain::Node**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:managedblockchain:region:account_ID:nodes/node_ID
```

When `resources.type` equals **AWS::S3ObjectLambda::AccessPoint**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name
```

When `resources.type` equals **AWS::EC2::Snapshot**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:ec2:region::snapshot/snapshot_ID
```

When `resources.type` equals **AWS::S3::AccessPoint**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in one of the following formats. To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the **StartsWith** or **NotStartsWith** operators.

```
arn:partition:s3:region:account_ID:accesspoint/access_point_name
```

```
arn:partition:s3:region:account_ID:accesspoint/access_point_name/  
object/object_path
```

When `resources.type` equals **AWS::DynamoDB::Stream**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name/stream/date_time
```

For more information about the ARN formats of data event resources, see [Actions, resources, and condition keys](#) in the *AWS Identity and Access Management User Guide*.

- b. For each field, choose **+ Conditions** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **NotEquals**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

To add the second S3 bucket, choose **+ Conditions**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

Note

You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still log all functions with a predefined selector template, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI](#) (p. 93).

- c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.
 - d. Save changes to your custom selector template by choosing **Next**. Do not choose another log selector template, or leave this page, or your custom selectors will be lost.
9. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 3 through this step to configure advanced event selectors for the data event type.
 10. After you choose **Next**, in **Step 2: Choose log events**, review the log selector template options you've chosen. Choose **Edit** to go back and make changes.
 11. To configure Insights events and other settings for your trail, go back to the preceding procedure in this topic, [Updating a trail](#) (p. 79).

Deleting a trail

You can delete trails with the CloudTrail console.

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. Open the **Trails** page of the CloudTrail console.
3. Choose the trail name.

4. At the top of the trail details page, choose **Delete**.
5. When you are prompted to confirm, choose **Delete** to delete the trail permanently. The trail is removed from the list of trails. Log files that were already delivered to the Amazon S3 bucket are not deleted.

Note

Content delivered to Amazon S3 buckets might contain customer content. For more information about removing sensitive data, see [How Do I Empty an S3 Bucket?](#) or [How Do I Delete an S3 Bucket?](#).

Turning off logging for a trail

When you create a trail, logging is turned on automatically. You can turn off logging for a trail. Existing logs are still stored in the trail's Amazon S3 bucket.

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 name of the trail.
3. At the top of the trail details page, choose **Stop logging** to turn off logging for the trail.
4. When you are prompted to confirm, choose **Stop logging**. CloudTrail stops logging activity for that trail.
5. To resume logging for that trail, choose **Start logging** on the trail configuration page.

Creating, updating, and managing trails with the AWS Command Line Interface

You can use the AWS CLI to create, update, and manage your trails. When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Note

You need the AWS command line tools to run the AWS Command Line Interface (AWS CLI) commands in this topic. Make sure you have a recent version of the AWS CLI installed. 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`.

Commonly used commands for trail creation, management, and status

Some of the more commonly used commands for creating and updating trails in CloudTrail include:

- [create-trail \(p. 88\)](#) to create a trail.
- [update-trail \(p. 90\)](#) to change the configuration of an existing trail.
- [add-tags \(p. 93\)](#) to add one or more tags (key-value pairs) to an existing trail.
- [remove-tags \(p. 94\)](#) to remove one or more tags from a trail.
- [list-tags \(p. 94\)](#) to return a list of tags associated with a trail.
- [put-event-selectors \(p. 97\)](#) to add or modify event selectors for a trail.
- [put-insight-selectors](#) to add or modify Insights event selectors for an existing trail, and enable or disable Insights events.

- [start-logging \(p. 89\)](#) to begin logging events with your trail.
- [stop-logging \(p. 112\)](#) to pause logging events with your trail.
- [delete-trail \(p. 112\)](#) to delete a trail. This command does not delete the Amazon S3 bucket that contains the log files for that trail, if any.
- [describe-trails \(p. 94\)](#) to return information about trails in an AWS Region.
- [get-trail \(p. 94\)](#) to return settings information for a trail.
- [get-trail-status \(p. 94\)](#) to return information about the current status of a trail.
- [get-event-selectors \(p. 97\)](#) to return information about event selectors configured for a trail.
- [get-insight-selectors](#) to return information about Insights event selectors configured for a trail.

Supported commands for creating and updating trails: create-trail and update-trail

The `create-trail` and `update-trail` commands offer a variety of functionality for creating and managing trails, including:

- Creating a trail that receives logs across Regions, or update a trail with the `--is-multi-region-trail` option. In most circumstances, you should create trails that log events in all AWS Regions.
- Creating a trail that receives logs for all AWS accounts in an organization with the `--is-organization-trail` option.
- Converting a multi-region trail to single-region trail with the `--no-is-multi-region-trail` option.
- Enabling or disabling log file encryption with the `--kms-key-id` option. The option specifies an AWS KMS key that you 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 \(p. 268\)](#).
- Enabling or disabling log file validation with the `--enable-log-file-validation` and `--no-enable-log-file-validation` options. For more information, see [Validating CloudTrail log file integrity \(p. 196\)](#).
- Specifying a CloudWatch Logs log group and role so that CloudTrail can deliver events to a CloudWatch Logs log group. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).

Deprecated commands: create-subscription and update-subscription

Important

The `create-subscription` and `update-subscription` commands were used to create and update trails, but are deprecated. Do not use these commands. They do not provide full functionality for creating and managing trails.

If you configured automation that uses one or both of these commands, we recommend that you update your code or scripts to use supported commands such as `create-trail`.

Using create-trail

You can run the `create-trail` command to create trails that are specifically configured to meet your business needs. When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the `--region` parameter with the command.

Creating a trail that applies to all Regions

To create a trail that applies to all Regions, use the `--is-multi-region-trail` option. By default, the `create-trail` command creates a trail that logs events only in the AWS Region where the trail was

created. To ensure that you log global service events and capture all management event activity in your AWS account, you should create trails that log events in all AWS Regions.

Note

When you create a trail, if you specify an Amazon S3 bucket that was not created with CloudTrail, you need to attach the appropriate policy. See [Amazon S3 bucket policy for CloudTrail](#) (p. 243).

The following example creates a trail with the name `my-trail` and a tag with a key named `Group` with a value of `Marketing` that delivers logs from all Regions to an existing bucket named `my-bucket`.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name my-bucket --is-multi-region-trail --tags-list [key=Group,value=Marketing]
```

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-east-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
  "IsOrganizationTrail": false,
  "S3BucketName": "my-bucket"
}
```

Note

Use the `start-logging` command to start logging for your trail.

Start logging for the trail

After the `create-trail` command completes, run the `start-logging` command to start logging for that trail.

Note

When you create a trail with the CloudTrail console, logging is turned on automatically.

The following example starts logging for a trail.

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

This command doesn't return an output, but you can use the `get-trail-status` command to verify that logging has started.

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

To confirm that the trail is logging, the `IsLogging` element in the output shows `true`.

```
{
  "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 single-region trail

The following command creates a single-region trail. The specified Amazon S3 bucket must already exist and have the appropriate CloudTrail permissions applied. For more information, see [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

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

For more information, see [CloudTrail trail naming requirements \(p. 130\)](#).

The following is example output.

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

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

To enable log file validation when using `create-trail`, use the `--enable-log-file-validation` option.

For information about log file validation, see [Validating CloudTrail log file integrity \(p. 196\)](#).

The following example creates a trail that delivers logs from all Regions to the specified bucket. The command uses the `--enable-log-file-validation` option.

```
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 is enabled, the `LogFileValidationEnabled` element in the output shows `true`.

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

Using update-trail

Important

As of November 22, 2021, AWS CloudTrail will change how trails can be used to capture global service events. After the change, events created by Amazon CloudFront, AWS Identity and Access Management, and AWS STS will be recorded in the region in which they were created, the US East (N. Virginia) region, `us-east-1`. This makes CloudTrail's treatment of these services consistent with that of other AWS global services.

To continue receiving global service events outside of US East (N. Virginia), be sure to convert *single-region trails* using global service events outside of US East (N. Virginia) into *multi-region trails*. For more information about capturing global service events, see [Enabling and disabling global service event logging \(p. 92\)](#) later in this section. Also update the region of your `lookup-events` API calls. For more information about updating lookup events, including an example CLI command, see [Looking up events by attribute \(p. 52\)](#) earlier in this user guide.

You can use the `update-trail` command to change the configuration settings for a trail. You can also use the `add-tags` and `remove-tags` commands to add and remove tags for a trail. You can only update trails from the AWS Region where the trail was created (its Home Region). When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the `--region` parameter with the command.

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, the policy must contain the full service name, `cloudtrail.amazonaws.com`. For more information, see [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

Converting a trail that applies to one Region to apply to all Regions

To change an existing trail so that it applies to all Regions, use the `--is-multi-region-trail` option.

```
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-east-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
  "IsOrganizationTrail": false,
  "S3BucketName": "my-bucket"
}
```

Converting a multi-region trail to a single-region trail

To change an existing multi-region trail so that it applies only to the Region in which it was created, use the `--no-is-multi-region-trail` option.

```
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-east-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": false,
  "IsOrganizationTrail": false,
  "S3BucketName": "my-bucket"
}
```


Enabling and disabling global service event logging

To change a trail so that it does not log global service events, use the `--no-include-global-service-events` option.

```
aws cloudtrail update-trail --name my-trail --no-include-global-service-events
```

To confirm that the trail no longer logs global service events, the `IncludeGlobalServiceEvents` element in the output shows false.

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

To change a trail so that it logs global service events, use the `--include-global-service-events` option.

Single-region trails will no longer receive global service events beginning November 22, 2021, unless the trail already appears in US East (N. Virginia) region, `us-east-1`. To continue capturing global service events, update the trail configuration to a multi-region trail. For example, this command updates a single-region trail in US East (Ohio), `us-east-2`, into a multi-region trail. Replace *myExistingSingleRegionTrailWithGSE* with the appropriate trail name for your configuration.

```
aws cloudtrail --region us-east-2 update-trail --name myExistingSingleRegionTrailWithGSE --is-multi-region-trail
```

Because global service events are only available in US East (N. Virginia) beginning November 22, 2021, you can also create a single-region trail to subscribe to global service events in the US East (N. Virginia) region, `us-east-1`. The following command creates a single-region trail in `us-east-1` to receive CloudFront, IAM, and AWS STS events:

```
aws cloudtrail --region us-east-1 create-trail --include-global-service-events --name myTrail --s3-bucket-name DOC-EXAMPLE-BUCKET
```

Enabling log file validation

To enable log file validation for a trail, use the `--enable-log-file-validation` option. Digest files are delivered to the Amazon S3 bucket for that 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-east-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": true,
  "IsMultiRegionTrail": false,
  "IsOrganizationTrail": false,
  "S3BucketName": "my-bucket"
}
```

```
}
```

Disabling log file validation

To disable log file validation for a trail, use the `--no-enable-log-file-validation` option.

```
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-east-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": false,
  "IsOrganizationTrail": false,
  "S3BucketName": "my-bucket"
}
```

To validate log files with the AWS CLI, see [Validating CloudTrail log file integrity with the AWS CLI](#) (p. 197).

Managing trails with the AWS CLI

The AWS CLI includes several other commands that help you manage your trails. These commands add tags to trails, get trail status, start and stop logging for trails, and delete a trail. You must run these commands from the same AWS Region where the trail was created (its Home Region). When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the `--region` parameter with the command.

Topics

- [Add one or more tags to a trail](#) (p. 93)
- [List tags for one or more trails](#) (p. 94)
- [Remove one or more tags from a trail](#) (p. 94)
- [Retrieving trail settings and the status of a trail](#) (p. 94)
- [Configuring CloudTrail Insights event selectors](#) (p. 96)
- [Configuring event selectors](#) (p. 97)
- [Configuring advanced event selectors](#) (p. 101)
- [Stopping and starting logging for a trail](#) (p. 112)
- [Deleting a trail](#) (p. 112)

Add one or more tags to a trail

To add one or more tags to an existing trail, run the **add-tags** command.

The following example adds a tag with the name *Owner* and the value of *Mary* to a trail with the ARN of *arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail* in the US East (Ohio) Region.

```
aws cloudtrail add-tags --resource-id arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail --tags-list Key=Owner,Value=Mary --region us-east-2
```

If successful, this command returns nothing.

List tags for one or more trails

To view the tags associated with one or more existing trails, use the **list-tags** command.

The following example lists the tags for *Trail1* and *Trail2*.

```
aws cloudtrail list-tags --resource-id-list arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1 arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail2
```

If successful, this command returns output similar to the following.

```
{
  "ResourceTagList": [
    {
      "ResourceId": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1",
      "TagsList": [
        {
          "Value": "Alice",
          "Key": "Name"
        },
        {
          "Value": "Ohio",
          "Key": "Location"
        }
      ]
    },
    {
      "ResourceId": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail2",
      "TagsList": [
        {
          "Value": "Bob",
          "Key": "Name"
        }
      ]
    }
  ]
}
```

Remove one or more tags from a trail

To remove one or more tags from an existing trail, run the **remove-tags** command.

The following example removes tags with the names *Location* and *Name* from a trail with the ARN of *arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1* in the US East (Ohio) Region.

```
aws cloudtrail remove-tags --resource-id arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1 --tags-list Key=Name Key=Location --region us-east-2
```

If successful, this command returns nothing.

Retrieving trail settings and the status of a trail

Run the **describe-trails** command to retrieve information about trails in an AWS Region. The following example returns information about trails configured in the US East (Ohio) Region.

```
aws cloudtrail describe-trails --region us-east-2
```

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

```
{
```

```
"trailList": [
  {
    "Name": "my-trail",
    "S3BucketName": "my-bucket",
    "S3KeyPrefix": "my-prefix",
    "IncludeGlobalServiceEvents": true,
    "IsMultiRegionTrail": true,
    "HomeRegion": "us-east-2",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "HasCustomEventSelectors": false,
    "SnsTopicName": "my-topic",
    "IsOrganizationTrail": false,
  },
  {
    "Name": "my-special-trail",
    "S3BucketName": "another-bucket",
    "S3KeyPrefix": "example-prefix",
    "IncludeGlobalServiceEvents": false,
    "IsMultiRegionTrail": false,
    "HomeRegion": "us-east-2",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-special-trail",
    "LogFileValidationEnabled": false,
    "HasCustomEventSelectors": true,
    "IsOrganizationTrail": false
  },
  {
    "Name": "my-org-trail",
    "S3BucketName": "my-bucket",
    "S3KeyPrefix": "my-prefix",
    "IncludeGlobalServiceEvents": true,
    "IsMultiRegionTrail": true,
    "HomeRegion": "us-east-1",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-org-trail",
    "LogFileValidationEnabled": false,
    "HasCustomEventSelectors": false,
    "SnsTopicName": "my-topic",
    "IsOrganizationTrail": true
  }
]
```

Run the `get-trail` command to retrieve settings information about a specific trail. The following example returns settings information for a trail named `my-trail`.

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

If successful, this command returns output similar to the following.

```
{
  "Trail": {
    "Name": "my-trail",
    "S3BucketName": "my-bucket",
    "S3KeyPrefix": "my-prefix",
    "IncludeGlobalServiceEvents": true,
    "IsMultiRegionTrail": true,
    "HomeRegion": "us-east-2",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "HasCustomEventSelectors": false,
    "SnsTopicName": "my-topic",
    "IsOrganizationTrail": false,
  }
}
```

```
}
```

Run the `get-trail-status` command to retrieve the status of a trail. You must either run this command from the AWS Region where it was created (the Home Region), or you must specify that Region by adding the `--region` parameter.

Note

If the trail is an organization trail and you are a member account in the organization in AWS Organizations, you must provide the full ARN of that trail, and not just the name.

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

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.

Configuring CloudTrail Insights event selectors

Enable Insights events on a trail by running the **put-insight-selectors**, and specifying `ApiCallRateInsight` as the value of the `InsightType` attribute. To view the Insights selector settings for a trail, run the `get-insight-selectors` command. You must either run this command from the AWS Region where the trail was created (the Home Region), or you must specify that Region by adding the `--region` parameter to the command.

Example trail that logs Insights events

The following example uses **put-insight-selectors** to create an Insights event selector for a trail named *TrailName3*. This enables Insights event collection for the *TrailName3* trail.

```
aws cloudtrail put-insight-selectors --trail-name TrailName3 --insight-selectors
'{"InsightType": "ApiCallRateInsight"}'
```

The example returns the Insights event selector that is configured for the trail.

```
{
  "InsightSelectors": [
    {
      "InsightType": "ApiCallRateInsight"
    }
  ],
}
```

```
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName3"
}
```

Example: Turn off collection of Insights events

The following example uses **put-insight-selectors** to remove the Insights event selector for a trail named *TrailName3*. Clearing the JSON string of Insights selectors disables Insights event collection for the *TrailName3* trail.

```
aws cloudtrail put-insight-selectors --trail-name TrailName3 --insight-selectors '[]'
```

The example returns the now-empty Insights event selector that is configured for the trail.

```
{
  "InsightSelectors": [
    {
      "InsightType": ""
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName3"
}
```

Configuring event selectors

To view the event selector settings for a trail, run the `get-event-selectors` command. You must either run this command from the AWS Region where it was created (the Home Region), or you must specify that Region by using the **--region** parameter.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

Note

If the trail is an organization trail and you are a member account in the organization in AWS Organizations, you must provide the full ARN of that trail, and not just the name.

The following example returns the default settings for an event selector for a trail.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [],
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To create an event selector, run the `put-event-selectors` command. When an event occurs in your account, CloudTrail evaluates the configuration for your trails. If the event matches any event selector for a trail, the trail processes and logs the event. You can configure up to 5 event selectors for a trail and up to 250 data resources for a trail. For more information, see [Logging data events for trails \(p. 140\)](#).

Topics

- [Example trail with specific event selectors \(p. 98\)](#)
- [Example trail that logs all management and data events \(p. 98\)](#)
- [Example trail that does not log AWS Key Management Service events \(p. 99\)](#)
- [Example trail that logs relevant low-volume AWS Key Management Service events \(p. 100\)](#)

- [Example trail that does not log Amazon RDS data API events \(p. 100\)](#)

Example trail with specific event selectors

The following example creates an event selector for a trail named *TrailName* to include read-only and write-only management events, data events for two Amazon S3 bucket/prefix combinations, and data events for a single AWS Lambda function named *hello-world-python-function*.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors
'[{ "ReadWriteType": "All", "IncludeManagementEvents": true, "DataResources":
  [{ "Type": "AWS::S3::Object", "Values": [ "arn:aws:s3:::mybucket/
prefix", "arn:aws:s3:::mybucket2/prefix2" ] }, { "Type": "AWS::Lambda::Function", "Values":
  [ "arn:aws:lambda:us-west-2:999999999999:function:hello-world-python-function" ] } ] } ]'
```

The example returns the event selector configured for the trail.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [],
      "IncludeManagementEvents": true,
      "DataResources": [
        {
          "Values": [
            "arn:aws:s3:::mybucket/prefix",
            "arn:aws:s3:::mybucket2/prefix2"
          ],
          "Type": "AWS::S3::Object"
        },
        {
          "Values": [
            "arn:aws:lambda:us-west-2:123456789012:function:hello-world-python-
function"
          ],
          "Type": "AWS::Lambda::Function"
        }
      ],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Example trail that logs all management and data events

The following example creates an event selector for a trail named *TrailName2* that includes all events, including read-only and write-only management events, and all data events for all Amazon S3 buckets, AWS Lambda functions, and Amazon DynamoDB tables in the AWS account. Because this example uses basic event selectors, it cannot configure logging for S3 events on AWS Outposts, Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, or other advanced event selector resource types. You must use advanced event selectors to log data events for those resources. For more information, see [Configuring advanced event selectors \(p. 101\)](#).

Note

If the trail applies only to one Region, only events in that Region are logged, even though the event selector parameters specify all Amazon S3 buckets and Lambda functions. Event selectors apply only to the Regions where the trail is created.

```
aws cloudtrail put-event-selectors --trail-name TrailName2 --event-selectors
'[{ "ReadWriteType": "All", "IncludeManagementEvents": true, "DataResources":
```

```
[{"Type": "AWS::S3::Object", "Values": ["arn:aws:s3:::"]}, {"Type":  
"AWS::Lambda::Function", "Values": ["arn:aws:lambda:"]}, {"Type":  
"AWS::DynamoDB::Table", "Values": ["arn:aws:dynamodb"]}]]}'
```

The example returns the event selectors configured for the trail.

```
{  
  "EventSelectors": [  
    {  
      "ExcludeManagementEventSources": [],  
      "IncludeManagementEvents": true,  
      "DataResources": [  
        {  
          "Values": [  
            "arn:aws:s3:::"  
          ],  
          "Type": "AWS::S3::Object"  
        },  
        {  
          "Values": [  
            "arn:aws:lambda"  
          ],  
          "Type": "AWS::Lambda::Function"  
        },  
        {  
          "Values": [  
            "arn:aws:dynamodb"  
          ],  
          "Type": "AWS::DynamoDB::Table"  
        }  
      ],  
      "ReadWriteType": "All"  
    }  
  ],  
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName2"  
}
```

Example trail that does not log AWS Key Management Service events

The following example creates an event selector for a trail named *TrailName* to include read-only and write-only management events, but to exclude AWS Key Management Service (AWS KMS) events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. The user in this example has chosen to exclude AWS KMS events from every trail except for one. To exclude an event source, add `ExcludeManagementEventSources` to your event selectors, and specify an event source in the string value.

If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

To start logging AWS KMS events to a trail again, pass an empty string as the value of `ExcludeManagementEventSources`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-  
selectors '[{"ReadWriteType": "All", "ExcludeManagementEventSources":  
  ["kms.amazonaws.com"], "IncludeManagementEvents": true}]']
```

The example returns the event selector that is configured for the trail.

```
{  
  "EventSelectors": [  

```



```
{
  "ExcludeManagementEventSources": [ "kms.amazonaws.com" ],
  "IncludeManagementEvents": true,
  "DataResources": [],
  "ReadWriteType": "All"
},
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging AWS KMS events to a trail again, pass an empty string as the value of `ExcludeManagementEventSources`, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors
'[{ "ReadWriteType": "All", "ExcludeManagementEventSources": [], "IncludeManagementEvents":
true}]'
```

Example trail that logs relevant low-volume AWS Key Management Service events

The following example creates an event selector for a trail named *TrailName* to include write-only management events and AWS KMS events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. The user in this example has chosen to include AWS KMS **Write** events, which will include `Disable`, `Delete` and `ScheduleKey`, but no longer include high-volume actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` (these are now treated as **Read** events).

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "WriteOnly", "ExcludeManagementEventSources":
[], "IncludeManagementEvents": true}]'
```

The example returns the event selector that is configured for the trail. This logs write-only management events, including AWS KMS events.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [],
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "WriteOnly"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Example trail that does not log Amazon RDS data API events

The following example creates an event selector for a trail named *TrailName* to include read-only and write-only management events, but to exclude Amazon RDS Data API events. Because Amazon RDS Data API events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. The user in this example has chosen to exclude Amazon RDS Data API events from every trail except for one. To exclude an event source, add `ExcludeManagementEventSources` to your event selectors, and specify the Amazon RDS Data API event source in the string value: `rdsdata.amazonaws.com`.

If you choose not to log management events, Amazon RDS Data API events are not logged, and you cannot change event logging settings.

To start logging Amazon RDS Data API events to a trail again, pass an empty string as the value of `ExcludeManagementEventSources`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors '[{"ReadWriteType": "All", "ExcludeManagementEventSources": ["kms.amazonaws.com"], "IncludeManagementEvents": true}]'
```

The example returns the event selector that is configured for the trail.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [ "rdsdata.amazonaws.com" ],
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging Amazon RDS Data API events to a trail again, pass an empty string as the value of `ExcludeManagementEventSources`, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors '[{"ReadWriteType": "All", "ExcludeManagementEventSources": [], "IncludeManagementEvents": true}]'
```

Configuring advanced event selectors

To use advanced event selectors to include or exclude data events instead of basic event selectors, opt in to use advanced event selectors on a trail's details page. Advanced event selectors let you log data events on more resource types than basic event selectors. Basic selectors log S3 object activity, AWS Lambda function execution activity, and DynamoDB tables. Advanced event selectors log S3 object-level API activity on AWS Outposts, API calls on S3 Object on Lambda access points, Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, API activity on S3 Object Lambda access points, Amazon EBS direct APIs on EBS snapshots, API activity on S3 access points, and DynamoDB streams. For more information advanced event selectors, see [Configuring advanced event selectors \(p. 101\)](#).

To view the advanced event selector settings for a trail, run the following `get-event-selectors` command. You must either run this command from the AWS Region where the trail was created (the Home Region), or you must specify that Region by adding the `--region` parameter.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

Note

If the trail is an organization trail, and you are signed in with a member account in the organization in AWS Organizations, you must provide the full ARN of the trail, and not just the name.

The following example returns the default settings for advanced event selectors for a trail. By default, no advanced event selectors are configured for a trail.

```
{
  "AdvancedEventSelectors": [],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To create an advanced event selector, run the `put-event-selectors` command. When a data event occurs in your account, CloudTrail evaluates the configuration for your trails. If the event matches any advanced event selector for a trail, the trail processes and logs the event. You can configure up to 500 conditions on a trail, including all values specified for all advanced event selectors on your trail. For more information, see [Logging data events for trails \(p. 140\)](#).

Topics

- [Example trail with specific advanced event selectors \(p. 102\)](#)
- [Example trail that uses custom advanced event selectors to log all management and data events \(p. 104\)](#)
- [Example trail that uses custom advanced event selectors to log Amazon S3 on AWS Outposts data events \(p. 109\)](#)
- [Example trail that uses advanced event selectors to exclude AWS Key Management Service events \(p. 109\)](#)
- [Example trail that uses advanced event selectors to exclude Amazon RDS data API events \(p. 111\)](#)

Example trail with specific advanced event selectors

The following example creates custom advanced event selectors for a trail named *TrailName* to include read and write management events (by omitting the `readOnly` selector), `PutObject` and `DeleteObject` data events for all Amazon S3 bucket/prefix combinations except for a bucket named `sample_bucket_name` and data events for an AWS Lambda function named `MyLambdaFunction`. Because these are custom advanced event selectors, each set of selectors has a descriptive name. Note that a trailing slash is part of the ARN value for S3 buckets.

```
aws cloudtrail put-event-selectors --trail-name TrailName --advanced-event-selectors '[
  {
    "Name": "Log readOnly and writeOnly management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  },
  {
    "Name": "Log PutObject and DeleteObject events for all but one bucket",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "eventName", "Equals": ["PutObject", "DeleteObject"] },
      { "Field": "resources.ARN", "NotEquals": ["arn:aws:s3:::sample_bucket_name/"] }
    ]
  },
  {
    "Name": "Log data plane actions on MyLambdaFunction",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::Lambda::Function"] },
      { "Field": "resources.ARN", "Equals": ["arn:aws:lambda:us-east-2:111122223333:function/MyLambdaFunction"] }
    ]
  }
]
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log readOnly and writeOnly management events",
```

```

    "FieldSelectors": [
      {
        "Field": "eventCategory",
        "Equals": [ "Management" ],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
      }
    ]
  },
  {
    "Name": "Log PutObject and DeleteObject events for all but one bucket",
    "FieldSelectors": [
      {
        "Field": "eventCategory",
        "Equals": [ "Data" ],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
      },
      {
        "Field": "resources.type",
        "Equals": [ "AWS::S3::Object" ],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
      },
      {
        "Field": "resources.ARN",
        "Equals": [],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [ "arn:aws:s3:::sample_bucket_name/" ],
        "NotStartsWith": [],
        "NotEndsWith": []
      }
    ]
  },
  {
    "Name": "Log data plane actions on MyLambdaFunction",
    "FieldSelectors": [
      {
        "Field": "eventCategory",
        "Equals": [ "Data" ],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
      },
      {
        "Field": "resources.type",
        "Equals": [ "AWS::Lambda::Function" ],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
      }
    ]
  }
}

```

```
    "Field": "eventName",
    "Equals": [ "Invoke" ],
    "StartsWith": [],
    "EndsWith": [],
    "NotEquals": [],
    "NotStartsWith": [],
    "NotEndsWith": []
  },
  {
    "Field": "resources.ARN",
    "Equals": [ "arn:aws:lambda:us-east-2:111122223333:function/MyLambdaFunction" ],
    "StartsWith": [],
    "EndsWith": [],
    "NotEquals": [],
    "NotStartsWith": [],
    "NotEndsWith": []
  }
]
}
],
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Example trail that uses custom advanced event selectors to log all management and data events

The following example creates advanced event selectors for a trail named *TrailName2* that includes all events, including read-only and write-only management events, and all data events for all S3 buckets, Lambda functions, DynamoDB tables, S3 object-level API activity on AWS Outposts, Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, API activity on S3 Object Lambda access points, Amazon EBS direct API activity on Amazon EBS snapshots in the AWS account, S3 access points, and DynamoDB streams.

Note

If the trail applies only to one Region, only events in that Region are logged, even though the event selector parameters specify all Amazon S3 buckets and Lambda functions. In a single-region trail, event selectors apply only to the Region where the trail is created.

```
aws cloudtrail put-event-selectors --trail-name TrailName2 \
--advanced event-selectors '
[
  {
    "Name": "Log readOnly and writeOnly management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  },
  {
    "Name": "Log all events for all buckets",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] }
    ]
  },
  {
    "Name": "Log all events for Lambda functions",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::Lambda::Function"] }
    ]
  },
  {
    "Name": "Log all events for DynamoDB tables",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },

```

```
    { "Field": "resources.type", "Equals": ["AWS::DynamoDB::Table"] }
  ],
},
{
  "Name": "Log all events for S3 on Outposts",
  "FieldSelectors": [
    { "Field": "eventCategory", "Equals": ["Data"] },
    { "Field": "resources.type", "Equals": ["AWS::S3Outposts::Object"] }
  ]
},
{
  "Name": "Log all JSON-RPC calls for Ethereum nodes in AWS Managed Blockchain",
  "FieldSelectors": [
    { "Field": "eventCategory", "Equals": ["Data"] },
    { "Field": "resources.type", "Equals": ["AWS::ManagedBlockchain::Node"] }
  ]
},
{
  "Name": "Log all events for S3 Object Lambda access points",
  "FieldSelectors": [
    { "Field": "eventCategory", "Equals": ["Data"] },
    { "Field": "resources.type", "Equals": ["AWS::S3ObjectLambda::AccessPoint"] }
  ]
},
{
  "Name": "Log all Amazon EBS direct API calls on snapshots",
  "FieldSelectors": [
    { "Field": "eventCategory", "Equals": ["Data"] },
    { "Field": "resources.type", "Equals": ["AWS::EC2::Snapshot"] }
  ]
},
{
  "Name": "Log all events for S3 access points",
  "FieldSelectors": [
    { "Field": "eventCategory", "Equals": ["Data"] },
    { "Field": "resources.type", "Equals": ["AWS::S3::AccessPoint"] }
  ]
},
{
  "Name": "Log all events for DynamoDB streams",
  "FieldSelectors": [
    { "Field": "eventCategory", "Equals": ["Data"] },
    { "Field": "resources.type", "Equals": ["AWS::DynamoDB::Stream"] }
  ]
}
]'
```

The example returns the advanced event selectors configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log readOnly and writeOnly management events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ],
          "StartsWith": [],
          "EndsWith": [],
          "NotEquals": [],
          "NotStartsWith": [],
          "NotEndsWith": []
        }
      ]
    }
  ],
}
```

```
{
  "Name": "Log all events for all buckets",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::S3::Object" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
},
{
  "Name": "Log all events for Lambda functions",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::Lambda::Function" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
},
{
  "Name": "Log all events for DynamoDB tables",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::DynamoDB::Table" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
}
```

```
    }
  ],
},
{
  "Name": "Log all events for S3 objects on Outposts",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::S3Outposts::Object" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
},
{
  "Name": "Log all JSON-RPC call events for Ethereum on Managed Blockchain nodes",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::ManagedBlockchain::Node" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
},
{
  "Name": "Log all events for S3 Object Lambda access points",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::S3ObjectLambda::AccessPoint" ],
      "StartsWith": [],
      "EndsWith": [],
```



```
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
    }
]
},
{
    "Name": "Log all Amazon EBS direct API calls on snapshots",
    "FieldSelectors": [
        {
            "Field": "eventCategory",
            "Equals": [ "Data" ],
            "StartsWith": [],
            "EndsWith": [],
            "NotEquals": [],
            "NotStartsWith": [],
            "NotEndsWith": []
        },
        {
            "Field": "resources.type",
            "Equals": [ "AWS::EC2::Snapshot" ],
            "StartsWith": [],
            "EndsWith": [],
            "NotEquals": [],
            "NotStartsWith": [],
            "NotEndsWith": []
        }
    ]
},
{
    "Name": "Log all events for S3 access points",
    "FieldSelectors": [
        {
            "Field": "eventCategory",
            "Equals": [ "Data" ],
            "StartsWith": [],
            "EndsWith": [],
            "NotEquals": [],
            "NotStartsWith": [],
            "NotEndsWith": []
        },
        {
            "Field": "resources.type",
            "Equals": [ "AWS::S3::AccessPoint" ],
            "StartsWith": [],
            "EndsWith": [],
            "NotEquals": [],
            "NotStartsWith": [],
            "NotEndsWith": []
        }
    ]
},
{
    "Name": "Log all events for DynamoDB streams",
    "FieldSelectors": [
        {
            "Field": "eventCategory",
            "Equals": [ "Data" ],
            "StartsWith": [],
            "EndsWith": [],
            "NotEquals": [],
            "NotStartsWith": [],
            "NotEndsWith": []
        },
        {
            "Field": "resources.type",
```

```
        "Equals": [ "AWS::DynamoDB::Stream" ],
        "StartsWith": [],
        "EndsWith": [],
        "NotEquals": [],
        "NotStartsWith": [],
        "NotEndsWith": []
    }
  ]
},
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName2"
}
```

Example trail that uses custom advanced event selectors to log Amazon S3 on AWS Outposts data events

The following example shows how to configure your trail to include all data events for all Amazon S3 on AWS Outposts objects in your outpost. In this release, the supported value for S3 on AWS Outposts events for the `resources.type` field is `AWS::S3Outposts::Object`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --region region \
--advanced-event-selectors \
'[
  {
    "Name": "OutpostsEventSelector",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3Outposts::Object"] }
    ]
  }
]'
```

The command returns the following example output.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "OutpostsEventSelector",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [
            "Data"
          ]
        },
        {
          "Field": "resources.type",
          "Equals": [
            "AWS::S3Outposts::Object"
          ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:region:123456789012:trail/TrailName"
}
```

Example trail that uses advanced event selectors to exclude AWS Key Management Service events

The following example creates an advanced event selector for a trail named *TrailName* to include read-only and write-only management events (by omitting the `readOnly` selector), but to exclude AWS Key

Management Service (AWS KMS) events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. In this release, you can exclude events from `kms.amazonaws.com`.

If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

To start logging AWS KMS events to a trail again, remove the `eventSource` selector, and run the command again.

```
aws cloudtrail put-event-selectors --trail-name TrailName \  
--advanced event-selectors '  
[  
  {  
    "Name": "Log all management events except KMS events",  
    "FieldSelectors": [  
      { "Field": "eventCategory", "Equals": ["Management"] },  
      { "Field": "eventSource", "NotEquals": ["kms.amazonaws.com"] }  
    ]  
  }  
]
```

The example returns the advanced event selectors that are configured for the trail.

```
{  
  "AdvancedEventSelectors": [  
    {  
      "Name": "Log all management events except KMS events",  
      "FieldSelectors": [  
        {  
          "Field": "eventCategory",  
          "Equals": [ "Management" ],  
          "StartsWith": [],  
          "EndsWith": [],  
          "NotEquals": [],  
          "NotStartsWith": [],  
          "NotEndsWith": []  
        },  
        {  
          "Field": "eventSource",  
          "Equals": [],  
          "StartsWith": [],  
          "EndsWith": [],  
          "NotEquals": [ "kms.amazonaws.com" ],  
          "NotStartsWith": [],  
          "NotEndsWith": []  
        }  
      ]  
    }  
  ],  
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"  
}
```

To start logging excluded events to a trail again, remove the `eventSource` selector, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName \  
--advanced event-selectors '  
[  
  {  
    "Name": "Log all management events",  

```

```
"FieldSelectors": [
  { "Field": "eventCategory", "Equals": ["Management"] }
]
}'
```

Example trail that uses advanced event selectors to exclude Amazon RDS data API events

The following example creates an advanced event selector for a trail named *TrailName* to include read-only and write-only management events (by omitting the `readOnly` selector), but to exclude Amazon RDS Data API events. Because Amazon RDS Data API events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. To exclude Amazon RDS Data API events, specify the Amazon RDS Data API event source in the string value for the `eventSource` field: `rdsdata.amazonaws.com`.

If you choose not to log management events, Amazon RDS Data API events are not logged, and you cannot change Amazon RDS Data API event logging settings.

To start logging Amazon RDS Data API events to a trail again, remove the `eventSource` selector, and run the command again.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced event-selectors '
[
  {
    "Name": "Log all management events except Amazon RDS data events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] },
      { "Field": "eventSource", "NotEquals": ["rdsdata.amazonaws.com"] }
    ]
  }
]
'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log all management events except Amazon RDS data events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ],
          "StartsWith": [],
          "EndsWith": [],
          "NotEquals": [],
          "NotStartsWith": [],
          "NotEndsWith": []
        },
        {
          "Field": "eventSource",
          "Equals": [],
          "StartsWith": [],
          "EndsWith": [],
          "NotEquals": [ "rdsdata.amazonaws.com" ],
          "NotStartsWith": [],
          "NotEndsWith": []
        }
      ]
    }
  ],
}
```

```
}
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging excluded events to a trail again, remove the eventSource selector, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced event-selectors '
[
  {
    "Name": "Log all management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  }
]
```

Stopping and starting logging for a trail

The following commands start and stop CloudTrail logging.

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

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

Note

Before deleting a bucket, run the stop-logging command to stop delivering events to the bucket. If you don't stop logging, CloudTrail attempts to deliver log files to a bucket with the same name for a limited period of time.

If you stop logging or delete a trail, CloudTrail Insights is disabled on that trail.

Deleting a trail

You can delete a trail with the following command. You can delete a trail only in the Region it was created (the Home Region).

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

When you delete a trail, you do not delete the Amazon S3 bucket or the Amazon SNS topic associated with it. Use the AWS Management Console, AWS CLI, or service API to delete these resources separately.

Creating a trail for an organization

If you have created an organization in AWS Organizations, you can create a trail that will log all events for all AWS accounts in that organization. This is sometimes referred to as an *organization trail*. You can also choose to edit an existing trail in the management account and apply it to an organization, making it an organization trail. Organization trails log events for the management account and all member accounts in the organization. For more information about AWS Organizations, see [Organizations Terminology and Concepts](#).

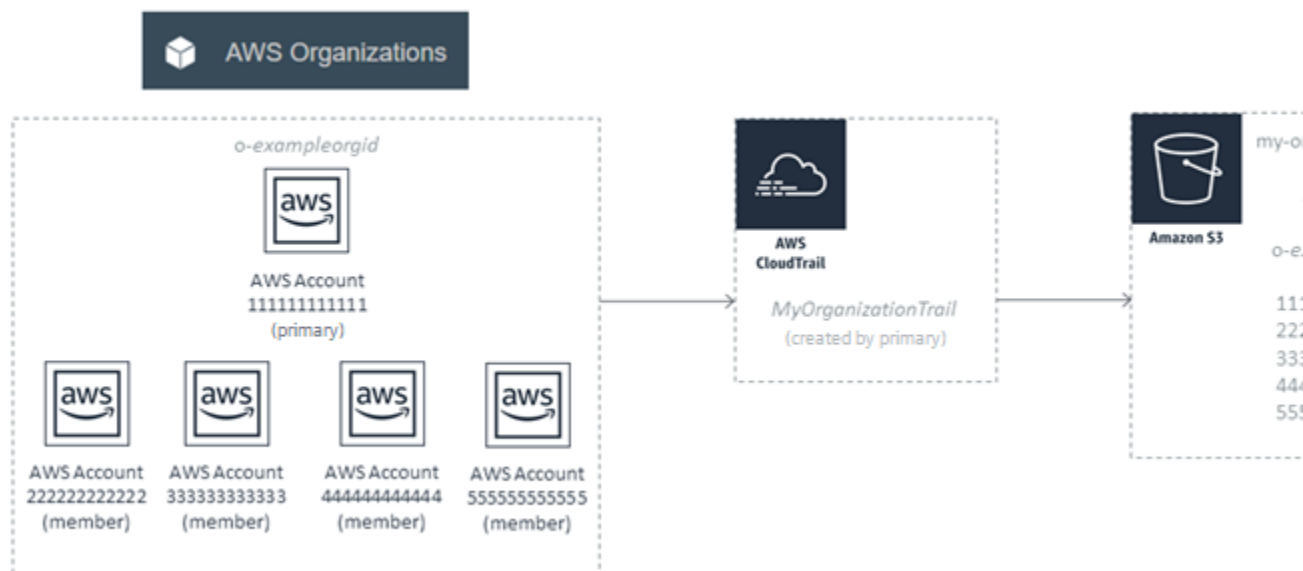
Note

You must be logged in with the management account for the organization to create an organization trail. You must also have sufficient permissions for the IAM user or role in the management account to successfully create an organization trail. If you do not have sufficient permissions, you cannot see the option to apply a trail to an organization.

When you create an organization trail, a trail with the name that you give it will be created in every AWS account that belongs to your organization. Users with CloudTrail permissions in member accounts will be able to see this trail when they log into the AWS CloudTrail console from their AWS accounts, or when they run AWS CLI commands such as `describe-trail`. However, users in member accounts will not have sufficient permissions to delete the organization trail, turn logging on or off, change what types of events are logged, or otherwise alter the organization trail in any way.

When you create an organization trail in the console, or when you enable CloudTrail as a trusted service in the Organizations, this creates a service-linked role to perform logging tasks in your organization's member accounts. This role is named **AWSServiceRoleForCloudTrail**, and is required for CloudTrail to successfully log events for an organization. If an AWS account is added to an organization, the organization trail and service-linked role will be added to that AWS account, and logging will begin for that account automatically in the organization trail. If an AWS account is removed from an organization, the organization trail and service-linked role will be deleted from the AWS account that is no longer part of the organization. However, log files for that removed account created prior to the account's removal will still remain in the Amazon S3 bucket where log files are stored for the trail.

In the following example, a user in the management account 111111111111 creates a trail named *MyOrganizationTrail* for the organization *o-exampleorgid*. The trail logs activity for all accounts in the organization in the same Amazon S3 bucket. All accounts in the organization can see *MyOrganizationTrail* in their list of trails, but member accounts will not be able to remove or modify the organization trail. Only the management account will be able to change or delete the trail for the organization, just as only the management account can remove a member account from an organization. Similarly, by default, only the management account has access to the Amazon S3 bucket *my-organization-bucket* for the trail and the logs contained within it. The high-level bucket structure for log files contains a folder named with the organization ID, with subfolders named with the account IDs for each account in the organization. Events for each member account are logged in the folder that corresponds to the member account ID. If member account 444444444444 is removed from the organization at some point in the future, *MyOrganizationTrail* and the service-linked role will no longer appear in AWS account 444444444444, and no further events will be logged for that account by the organization trail. However, the 444444444444 folder remains in the Amazon S3 bucket, with all logs created before the removal of the account from the organization.



In this example, the ARN of the trail created in the management account is `aws:cloudtrail:us-east-2:111111111111:trail/MyOrganizationTrail`. This ARN is the ARN for the trail in all member accounts as well.

Organization trails are similar to regular trails in many ways. You can create multiple trails for your organization, and choose whether to create an organization trail in all regions or a single region, and what kinds of events you want logged in your organization trail, just as in any other trail. However, there are some differences. For example, when you create a trail in the console and choose whether to log data events for Amazon S3 buckets or AWS Lambda functions, the only resources listed in the CloudTrail console are those for the management account, but you can add the ARNs for resources in member accounts. Data events for specified member account resources will be logged without having to manually configure cross-account access to those resources. For more information about logging management events, Insights events, and data events, see [Working with CloudTrail log files \(p. 133\)](#).

Note

In the console, you create a trail that logs all regions. This is a recommended best practice; logging activity in all regions helps you keep your AWS environment more secure. To create a single-region trail, [use the AWS CLI \(p. 90\)](#).

You can also configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs for an organization trail the same way you would for any other trail. For example, you can analyze the data in an organization trail using Amazon Athena. For more information, see [AWS service integrations with CloudTrail Logs \(p. 20\)](#).

Topics

- [Event history and organization trails \(p. 114\)](#)
- [Best practices for moving from member account trails to organization trails \(p. 114\)](#)
- [Prepare for creating a trail for your organization \(p. 114\)](#)
- [Creating a trail for your organization in the console \(p. 116\)](#)
- [Creating a trail for an organization with the AWS Command Line Interface \(p. 121\)](#)

Event history and organization trails

Organization trails make the most recent 90 days of management events visible in **Event history** the same way that individual accounts do. When you view events in **Event history** for an organization in AWS Organizations, you can view the events only for the AWS account with which you are signed in. For example, if you are signed in with the organization management account, **Event history** shows the last 90 days of management events for the management account. Organization member account events are not shown in **Event history** for the management account. To view member account events in **Event history**, sign in with the member account.

Best practices for moving from member account trails to organization trails

If you already have CloudTrail trails configured for individual member accounts, but want to move to an organization trail to log events in all accounts, you do not want to lose events by deleting individual member account trails before you create an organization trail. But when you have two trails, you incur higher costs because of the additional copy of events delivered to the organization trail.

To help manage costs, but avoid losing events before log delivery starts on the organization trail, consider keeping both your individual member account trails and your organization trail for up to one day. This ensures that the organization trail logs all events, but you incur duplicate event costs only for one day. After the first day, you can stop logging on (or delete) any individual member account trails.

Prepare for creating a trail for your organization

Before you create a trail for your organization, be sure that both your organization and your management account are set up correctly for trail creation.

- Your organization must have all features enabled before you can create a trail for it. For more information, see [Enabling All Features in Your Organization](#).
- The management account must have the **AWSServiceRoleForOrganizations** role. This role is created automatically by Organizations when you create your organization, and is required for CloudTrail to log events for an organization. For more information, see [Organizations and service-linked roles](#).
- The IAM user or role that will be used to create the organization trail in the management account must have sufficient permissions to create an organization trail. At a minimum, to create an organization trail, you must have the **AWSCloudTrail_FullAccess** policy or equivalent permissions applied. You must also have sufficient permissions in IAM and Organizations to create the service-linked role and enable trusted access. The following example policy shows the minimum required permissions.

Note

The **AWSCloudTrail_FullAccess** policy is not intended to be shared broadly across your AWS account. Instead, it should be restricted to AWS account administrators due to the highly sensitive information collected by CloudTrail. Users with this role have the ability to disable or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, access to this policy should be closely controlled and monitored.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "organizations:EnableAWSServiceAccess",
        "organizations:ListAccounts",
        "iam:CreateServiceLinkedRole",
        "organizations:DisableAWSServiceAccess",
        "organizations:DescribeOrganization",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    }
  ]
}
```

- To use the AWS CLI or the CloudTrail APIs to create an organization trail, you must enable trusted access for CloudTrail in Organizations, and you must manually create an Amazon S3 bucket with a policy that allows logging for an organization trail. For more information, see [Creating a trail for an organization with the AWS Command Line Interface \(p. 121\)](#).
- To use an existing IAM role to add monitoring of an organization trail to Amazon CloudWatch Logs, you must manually modify the IAM role to allow delivery of CloudWatch Logs for member accounts to the CloudWatch Logs group in the management account, as shown in the following example.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailCreateLogStream20141101",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
        "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
        "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/DefaultLogGroupTest:log-stream:o-exampleorgid_*"
      ]
    }
  ],
}
```



```
{
  "Sid": "AWSCloudTrailPutLogEvents20141101",
  "Effect": "Allow",
  "Action": [
    "logs:PutLogEvents"
  ],
  "Resource": [
    "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
    "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/DefaultLogGroupTest:log-stream:o-exampleorgid_*"
  ]
}
```

You can learn more about CloudTrail and Amazon CloudWatch Logs in [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#). In addition, consider the limits on CloudWatch Logs and the pricing considerations for the service before deciding to enable the experience for an organization trail. For more information, see [CloudWatch Logs Limits](#) and [Amazon CloudWatch Pricing](#).

- To log data events in your organization trail for specific resources in member accounts, have ready a list of Amazon Resource Names (ARNs) for each of those resources. Member account resources are not displayed in the CloudTrail console when you create a trail; you can browse for resources in the management account on which data event collection is supported, such as S3 buckets. Similarly, if you want to add specific member resources when creating or updating an organization trail at the command line, you need the ARNs for those resources.

Note

Additional charges apply for logging data events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

You should also consider reviewing how many trails already exist in the management account and in the member accounts before creating an organization trail. CloudTrail limits the number of trails that can be created in each region. You cannot exceed this limit in the region where you create the organization trail in the management account. However, the trail will be created in the member accounts even if member accounts have reached the limit of trails in a region. While the first trail of management events in any region is free, charges apply to additional trails. To reduce the potential cost of an organization trail, consider deleting any unneeded trails in the management and member accounts. For more information about CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Creating a trail for your organization in the console

To create an organization trail in the CloudTrail console, you must sign in to the console using an IAM user or role in the management account with [sufficient permissions \(p. 115\)](#). If you are not signed in with the management account, you will not see the option to apply a trail to an organization when you create or edit a trail in the CloudTrail console.

You can choose to configure an organization trail in various ways. For example, you can:

- By default, when you create a trail in the console, the trail logs all regions. Logging all regions in your account is a recommended best practice. To create a single-region trail, [use the AWS CLI \(p. 90\)](#). For more information, see [How CloudTrail works \(p. 1\)](#).
- Specify whether to apply the trail to your organization. The default is not to do so; you must choose this option to create an organization trail.
- Specify which Amazon S3 bucket to use to receive log files for the organization trail. You can choose an existing Amazon S3 bucket in the management account, or create one specifically for the organization trail.

- For management and data events, specify if you want to log **Read** events, **Write** events, or both. [CloudTrail Insights \(p. 154\)](#) events are logged only on management **Write** events. You can specify logging data events for resources in the management account by choosing them from the lists in the console, and in member accounts if you specify the ARNs of each resource for which you want to enable data event logging. For more information, see [Data events \(p. 140\)](#).

To create an organization 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/>.

You must be signed in as a user, role, or root account in the management account with [sufficient permissions \(p. 115\)](#) to create an organization trail.

2. Choose **Trails**, and then choose **Create trail**.
3. On the **Create Trail** page, for **Trail name**, type a name for your trail. For more information, see [CloudTrail trail naming requirements \(p. 130\)](#).
4. Select **Enable for all accounts in my organization**. You only see this option if you are signed in to the console with an IAM user or role in the management account. To successfully create an organization trail, be sure that the user or role has [sufficient permissions \(p. 115\)](#).
5. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies.

Note

If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket. The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

6. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files with SSE-KMS instead of SSE-S3. The default is **Enabled**. For more information about this encryption type, see [Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys \(SSE-S3\)](#).

If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#).

Note

You can also type the ARN of a key from another account. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). 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 [Configure AWS KMS key policies for CloudTrail \(p. 261\)](#).

7. In **Additional settings**, configure the following.
 - a. For **Log file validation**, choose **Enabled** 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 \(p. 196\)](#).
 - b. For **SNS notification delivery**, choose **Enabled** 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 more information, see [Configuring Amazon SNS notifications for CloudTrail \(p. 127\)](#).

If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions,

SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see [Amazon SNS topic policy for CloudTrail \(p. 246\)](#).

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](#).

8. Optionally, configure CloudTrail to send log files to CloudWatch Logs by choosing **Enabled in CloudWatch Logs**. For more information, see [Sending events to CloudWatch Logs \(p. 160\)](#).
 - a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.
 - b. If you choose **Existing**, choose a log group from the drop-down list.
 - c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see [Role policy document for CloudTrail to use CloudWatch Logs for monitoring \(p. 182\)](#).

Note

When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

9. For **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see [AWS Resource Groups](#) and [Why use tags for trails? \(p. 8\)](#).
10. On the **Choose log events** page, choose the event types that you want to log. For **Management events**, do the following.
 - a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see [Management events \(p. 136\)](#).
 - b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

- c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API

events. For more information about Amazon RDS Data API events, see [Logging Data API calls with AWS CloudTrail](#) in the *Amazon RDS User Guide for Aurora*.

11. For **Data events**, you can specify logging data events for Amazon S3 buckets, AWS Lambda functions, Amazon DynamoDB tables, or a combination of these resource types. By default, trails don't log data events. Additional charges apply for logging data events. For more information, see [Data events \(p. 140\)](#). For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Note

More data event types are available if you use advanced event selectors. If you have opted in to use advanced event selectors, follow steps in [Creating a trail in the console \(advanced event selectors\) \(p. 74\)](#) to configure data event logging on your trail.

For Amazon S3 buckets:

- a. For **Data event source**, choose **S3**.
- b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

Note

Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If the trail applies only to one Region, selecting the **Select all S3 buckets in your account** option enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

- c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.
- d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. To find specific buckets, type a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

To remove a bucket from logging, choose **X**.

12. To add another data type on which to log data events, choose **Add data event type**.

13. For Lambda functions:

- a. For **Data event source**, choose **Lambda**.
- b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

Note

If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for

all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

- c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

Note

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI \(p. 93\)](#).

14. For DynamoDB tables:

- a. For **Data event source**, choose **DynamoDB**.
- b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN is in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

15. When you are finished choosing event types to log, choose **Next**.
16. On the **Review and create** page, review your choices. Choose **Edit** in a section to change the trail settings shown in that section. When you are ready to create the trail, choose **Create trail**.
17. The new trail appears on the **Trails** page. An organization trail might take up to 24 hours to be created in all regions in all member accounts. The **Trails** page shows the trails in your account from all regions. In about 15 minutes, CloudTrail publishes log files that show the AWS API calls made in your organization. You can see the log files in the Amazon S3 bucket that you specified.

Note

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

Next steps

After you create your trail, you can return to the trail to make changes:

- Change the configuration of your trail by editing it. For more information, see [Updating a trail \(p. 79\)](#).
- Configure the Amazon S3 bucket to allow specific IAM users in member accounts to read the log files for the organization, if desired. For more information, see [Sharing CloudTrail log files between AWS accounts \(p. 186\)](#).
- Configure CloudTrail to send log files to CloudWatch Logs. For more information, see [Sending events to CloudWatch Logs \(p. 160\)](#) and [the CloudWatch Logs item \(p. 115\)](#) in [Prepare for creating a trail for your organization \(p. 114\)](#).
- Create a table and use it to run a query in Amazon Athena to analyze your AWS service activity. For more information, see [Creating a Table for CloudTrail Logs in the CloudTrail Console](#) in the [Amazon Athena User Guide](#).
- Add custom tags (key-value pairs) to the trail.

- To create another organization trail, return to the **Trails** page and choose **Create trail**.

Note

When you configure a trail, you can choose an Amazon S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

Creating a trail for an organization with the AWS Command Line Interface

You can create an organization trail by using the AWS CLI. The AWS CLI is regularly updated with additional functionality and commands. To help ensure success, be sure that you have installed or updated to a recent AWS CLI version before you begin.

Note

The examples in this section are specific to creating and updating organization trails. For examples of using the AWS CLI to manage trails, see [Managing trails with the AWS CLI \(p. 93\)](#). When creating or updating an organization trail with the AWS CLI, you must use an AWS CLI profile in the management account with sufficient permissions. You must configure the Amazon S3 bucket used for an organization trail with sufficient permissions.

Create or update an Amazon S3 bucket to use to store the log files for an organization trail

You must specify an Amazon S3 bucket to receive the log files for an organization trail. This bucket must have a policy that allows CloudTrail to put the log files for the organization into the bucket.

The following is an example policy for an Amazon S3 bucket named *my-organization-bucket*. This bucket is in an AWS account with the ID *111111111111*, which is the management account for an organization with the ID *o-exampleorgid* that allows logging for an organization trail. It also allows logging for the *111111111111* account in the event that the trail is changed from an organization trail to a trail for that account only.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailAclCheck20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "cloudtrail.amazonaws.com"
        ]
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::my-organization-bucket"
    },
    {
      "Sid": "AWSCloudTrailWrite20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "cloudtrail.amazonaws.com"
        ]
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::my-organization-bucket/AWSLogs/111111111111/*",
    }
  ]
}
```

```
        "Condition": {
          "StringEquals": {
            "s3:x-amz-acl": "bucket-owner-full-control"
          }
        },
      ],
      {
        "Sid": "AWSCloudTrailWrite20150319",
        "Effect": "Allow",
        "Principal": {
          "Service": [
            "cloudtrail.amazonaws.com"
          ]
        },
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::my-organization-bucket/AWSLogs/o-exampleorgid/*",
        "Condition": {
          "StringEquals": {
            "s3:x-amz-acl": "bucket-owner-full-control"
          }
        }
      }
    ]
  }
}
```

This example policy does not allow any users from member accounts to access the log files created for the organization. By default, organization log files are accessible only to the management account. For information about how to allow read access to the Amazon S3 bucket for IAM users in member accounts, see [Sharing CloudTrail log files between AWS accounts](#) (p. 186).

Enabling CloudTrail as a trusted service in AWS Organizations

Before you can create an organization trail, you must first enable all features in Organizations. For more information, see [Enabling All Features in Your Organization](#), or run the following command using a profile with sufficient permissions in the management account:

```
aws organizations enable-all-features
```

After you enable all features, you must configure Organizations to trust CloudTrail as a trusted service. .

To create the trusted service relationship between AWS Organizations and CloudTrail, open a terminal or command line and use a profile in the management account. Run the `aws organizations enable-aws-service-access` command, as demonstrated in the following example.

```
aws organizations enable-aws-service-access --service-principal cloudtrail.amazonaws.com
```

Using create-trail

Creating an organization trail that applies to all regions

To create an organization trail that applies to all regions, add the `--is-organization-trail` and `--is-multi-region-trail` options.

Note

When you create or update an organization trail with the AWS CLI, you must use an AWS CLI profile in the management account with sufficient permissions.

The following example creates an organization trail that delivers logs from all regions to an existing bucket named *my-bucket*:


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

To confirm that your trail exists in all regions, the `IsOrganizationTrail` and `IsMultiRegionTrail` parameters in the output are both set to `true`:

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

Note

Run the `start-logging` command to start logging for your trail. For more information, see [Stopping and starting logging for a trail \(p. 112\)](#).

Creating an organization trail as a single-region trail

The following command creates an organization trail that only logs events in a single AWS Region, also known as a single-region trail. The AWS Region where events are logged is the region specified in the configuration profile for the AWS CLI.

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

For more information, see [CloudTrail trail naming requirements \(p. 130\)](#).

Sample output:

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

By default, the `create-trail` command creates a single-region trail that does not enable log file validation.

Note

Run the `start-logging` command to start logging for your trail.

Running `update-trail` to update an organization trail

You can run the `update-trail` command to change the configuration settings for an organization trail, or to apply an existing trail for a single AWS account to an entire organization. Remember that you can run the `update-trail` command only from the region in which the trail was created.

Note

If you use the AWS CLI or one of the AWS SDKs to update a trail, be sure that the trail's bucket policy is up-to-date. For more information, see [Creating a trail for an organization with the AWS Command Line Interface \(p. 121\)](#).

When you create or update an organization trail with the AWS CLI, you must use an AWS CLI profile in the management account with sufficient permissions.

Applying an existing trail to an organization

To change an existing trail so that it also applies to an organization instead of a single AWS account, add the `--is-organization-trail` option, as shown in the following example.

```
aws cloudtrail update-trail --name my-trail --is-organization-trail
```

To confirm that the trail now applies to the organization, the `IsOrganizationTrail` parameter in the output has a value of `true`.

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

In the preceding example, the trail was configured to apply to all regions (`"IsMultiRegionTrail": true`). A trail that applied only to a single region would show `"IsMultiRegionTrail": false` in the output.

Converting an organization trail that applies to one region to apply to all regions

To change an existing organization trail so that it applies to all regions, add the `--is-multi-region-trail` option as shown in the following example.

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

To confirm that the trail now applies to all regions, the `IsMultiRegionTrail` parameter in the output has a value of `true`.

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

Getting and viewing your CloudTrail log files

After you create a trail and configure it to capture the log files you want, you 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. CloudTrail typically delivers logs within an average of about 15 minutes of an API call. This time is

not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information. Insights events are typically delivered to your bucket within 30 minutes of unusual activity. After you enable Insights events for the first time, allow up to 36 hours to see the first Insights events, if unusual activity is detected.

Topics

- [Finding your CloudTrail log files \(p. 125\)](#)
- [Downloading your CloudTrail log files \(p. 126\)](#)

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/region/YYYY/MM/DD/file_name.json.gz
```

Note

For organization trails, the log file object name includes the organization unit ID in the path, as follows:

```
bucket_name/prefix_name/AWSLogs/OU-ID/Account ID/  
CloudTrail/region/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 \(p. 272\)](#).

Downloading your CloudTrail log files

Log files are in JSON format. If you have a JSON viewer add-on installed, you can view the files directly in your browser. Double-click the log file name in the bucket to open a new browser window or tab. The JSON displays in a readable format.

For example, 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 bucket to open the log file in JSON format.

CloudTrail log files are Amazon S3 objects. You can use the Amazon S3 console, the AWS Command Line Interface (CLI), or the Amazon S3 API to retrieve log files.

For more information, see [Working with Amazon S3 Objects](#) in the *Amazon Simple Storage Service User Guide*.

The following procedure describes how to download a log file with the AWS Management Console.

To download and read a log file

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Choose the bucket and choose the log file that you want to download.
3. Choose **Download** or **Download as** and follow the prompts to save the file. This saves the file in compressed format.

Note

Some browsers, such as Chrome, automatically extract the log file for you. If your browser does this for you, skip to step 5.

4. Use a product such as [7-Zip](#) to extract the log file.
5. Open the log file in a text editor such as Notepad++.

For more information about the event fields that can appear in a log file entry, see [CloudTrail log event reference \(p. 270\)](#).

AWS partners with third-party specialists in logging and analysis to provide solutions that use CloudTrail output. For more information, see [AWS Partner Network - AWS CloudTrail Partners](#).

Note

You can also use the **Event history** feature to look up events for create, update, and delete API activity during the last 90 days.

For more information, see [Viewing events with CloudTrail Event history \(p. 44\)](#).

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 \(p. 127\)](#)

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-trail](#) 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*.

Note

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": [ "AWSLogs/123456789012/
CloudTrail/us-east-2/2013/12/13/123456789012_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-2/2016/08/11/123456789012_CloudTrail_us-east-2_20160811T2215Z_kpaMYavMQA9Ahp7L.json.gz",
    "AWSLogs/123456789012/CloudTrail/us-east-2/2016/08/11/123456789012_CloudTrail_us-east-2_20160811T2210Z_zqDkyQv3TK8ZdLr0.json.gz",
    "AWSLogs/123456789012/CloudTrail/us-east-2/2016/08/11/123456789012_CloudTrail_us-east-2_20160811T2205Z_jaMVRa6JfdLCJYHP.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 [Amazon SNS topic policy for CloudTrail](#) (p. 246).

Additional resources

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

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. To learn more, see [Identity and Access Management for AWS CloudTrail](#) (p. 225).

The following topics might also help you understand permissions, policies, and CloudTrail security:

- [Amazon S3 bucket naming requirements](#) (p. 131)
- [Amazon S3 bucket policy for CloudTrail](#) (p. 243)
- An example of a bucket policy for an organization trail in [Creating a trail for an organization with the AWS Command Line Interface](#) (p. 121).
- [Amazon SNS topic policy for CloudTrail](#) (p. 246)
- [Encrypting CloudTrail log files with AWS KMS-managed keys \(SSE-KMS\)](#) (p. 259)
- [Default KMS key policy created in CloudTrail console](#) (p. 266)
- [Granting permission to view AWS Config information on the CloudTrail console](#) (p. 242)
- [Sharing CloudTrail log files between AWS accounts](#) (p. 186)
- [Required permissions for creating an organization trail](#) (p. 115)
- [Using a previously-existing IAM role to add monitoring of an organization trail to Amazon CloudWatch Logs](#) (p. 115)

Tips for managing trails

- Beginning on April 12, 2019, trails will be viewable only in the AWS Regions where they log events. If you create a trail that logs events in all AWS Regions, it will appear in the console in all AWS Regions. If you create a trail that only logs events in a single AWS Region, you can view and manage it only in that AWS Region.
- To edit a trail in the list, choose the trail name.
- Configure at least one trail that applies to all regions so that you receive log files from all regions in your account.
- To log events from a specific region and deliver log files to an S3 bucket in the same region, you can update the trail to apply to a single region. This is useful if you want to keep your log files separate. For example, you may want users to manage their own logs in specific regions, or you may want to separate CloudWatch Logs alarms by region.
- To log events from multiple AWS accounts in one trail, consider creating an organization in AWS Organizations and then creating an organization trail.
- Creating multiple trails will incur additional costs. For more information about prices, see [AWS CloudTrail Pricing](#).

Topics

- [Managing CloudTrail costs \(p. 129\)](#)
- [CloudTrail trail naming requirements \(p. 130\)](#)
- [Amazon S3 bucket naming requirements \(p. 131\)](#)
- [AWS KMS alias naming requirements \(p. 131\)](#)

Managing CloudTrail costs

As a best practice, we recommend using AWS services and tools that can help you manage CloudTrail costs. You can also configure and manage CloudTrail trails in ways that capture the data you need while remaining cost-effective. For more information about CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Tools to help manage costs

AWS Budgets, a feature of AWS Billing and Cost Management, lets you set custom budgets that alert you when your costs or usage exceed (or are forecasted to exceed) your budgeted amount.

As you create multiple trails, creating a budget for CloudTrail by using AWS Budgets is a recommended best practice, and can help you track your CloudTrail spending. Cost-based budgets help promote awareness of how much you might be billed for your CloudTrail use. [Budget alerts](#) notify you when your bill reaches a threshold that you define. When you receive a budget alert, you can make changes before the end of the billing cycle to manage your costs.

After you [create a budget](#), you can use AWS Cost Explorer to see how your CloudTrail costs are influencing your overall AWS bill. In AWS Cost Explorer, after adding CloudTrail to the **Service** filter, you can compare your historical CloudTrail spending to that of your current month-to-date (MTD) spending, by both region and account. This feature helps you monitor and detect unexpected costs in your monthly CloudTrail spending. Additional features in Cost Explorer let you compare CloudTrail spending to monthly spending at the specific resource level, providing information about what might be driving cost increases or decreases in your bill.

To get started with AWS Budgets, open [AWS Billing and Cost Management](#), and then choose **Budgets** in the left navigation bar. We recommend configuring budget alerts as you create a budget to track

CloudTrail spending. For more information about how to use AWS Budgets, see [Managing Your Costs with Budgets](#) and [Best Practices for AWS Budgets](#).

Trail configuration

CloudTrail offers flexibility in how you configure trails in your account. Some decisions that you make during the setup process require that you understand the impacts to your CloudTrail bill. The following are examples of how trail configurations can influence your CloudTrail bill.

Multiple trail creation

The first delivery of each management event for an account is free. If you create more trails that deliver the same management events to other destinations, those subsequent deliveries incur CloudTrail costs. You can do this to allow different user groups (such as developers, security personnel, and IT auditors) to receive their own copies of log files. For data events, all deliveries incur CloudTrail costs, including the first.

As you create more trails, it is especially important to be familiar with your logs, and understand the types and volumes of events that are generated by resources in your account. This helps you anticipate the volume of events that are associated with an account, and plan for trail costs. For example, using AWS KMS-managed server-side encryption (SSE-KMS) on your S3 buckets can result in a large number of AWS KMS management events in CloudTrail. Larger volumes of events across multiple trails can also influence costs.

To help limit the number of events that are logged to your trail, you can filter out AWS KMS or Amazon RDS Data API events by choosing **Exclude AWS KMS events** or **Exclude Amazon RDS Data API events** on the **Create trail** or **Update trail** pages. The option to filter out events is only available if your trail is logging management events. For more information, see [Creating a trail \(p. 70\)](#) or [Updating a trail \(p. 79\)](#) in this guide.

AWS Organizations

When you set up an Organizations trail with CloudTrail, CloudTrail replicates the trail to each member account within your organization. The new trail is created *in addition to* any existing trails in member accounts. Be sure that the configuration of your management account trail matches how you want trails configured for all accounts within an organization, because the management account trail configuration propagates to all accounts.

Because Organizations creates a trail in each member account, an individual member account that creates an additional trail to collect the same management events as the Organizations trail is collecting a second copy of events. The account is charged for the second copy. Similarly, if an account has a multi-region trail, and creates a second trail in a single region to collect the same management events as the multi-region trail, the trail in the single region is delivering a second copy of events. The second copy incurs charges.

See also

- [AWS CloudTrail Pricing](#)
- [Managing Your Costs with Budgets](#)
- [Getting Started with Cost Explorer](#)
- [Prepare for creating a trail for your organization \(p. 114\)](#)

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).

Warning

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 User Guide*.

AWS KMS alias naming requirements

When you create an AWS KMS key, 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*.

Using AWS CloudTrail with interface VPC endpoints

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and AWS CloudTrail. You can use this connection to enable CloudTrail to communicate with your resources on your VPC without going through the public internet.

Amazon VPC is an AWS service that you can use to launch AWS resources in a virtual network that you define. With a VPC, you have control over your network settings, such the IP address range, subnets,

route tables, and network gateways. With VPC endpoints, the routing between the VPC and AWS services is handled by the AWS network, and you can use IAM policies to control access to service resources.

To connect your VPC to CloudTrail, you define an *interface VPC endpoint* for CloudTrail. An interface endpoint is an elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported AWS service. The endpoint provides reliable, scalable connectivity to CloudTrail without requiring an internet gateway, network address translation (NAT) instance, or VPN connection. For more information, see [What is Amazon VPC](#) in the *Amazon VPC User Guide*.

Interface VPC endpoints are powered by AWS PrivateLink, an AWS technology that enables private communication between AWS services using an elastic network interface with private IP addresses. For more information, see [AWS PrivateLink](#).

The following steps are for users of Amazon VPC. For more information, see [Getting Started](#) in the *Amazon VPC User Guide*.

Availability

CloudTrail currently supports VPC endpoints in the following AWS Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Africa (Cape Town)
- Asia Pacific (Mumbai)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Asia Pacific (Osaka)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Milan)
- Europe (Paris)
- Europe (Stockholm)
- Middle East (Bahrain)
- South America (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Create a VPC endpoint for CloudTrail

To start using CloudTrail with your VPC, create an interface VPC endpoint for CloudTrail. For more information, see [Creating an Interface Endpoint](#) in the *Amazon VPC User Guide*.

You don't need to change the settings for CloudTrail. CloudTrail calls other AWS services using either public endpoints or private interface VPC endpoints, whichever are in use.

Working with CloudTrail log files

You can perform more advanced tasks with your CloudTrail files.

- Create multiple trails per region.
- Monitor CloudTrail log files 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.

When an event occurs in your account, CloudTrail evaluates whether the event matches the settings for your trails. Only events that match your trail settings are delivered to your Amazon S3 bucket and Amazon CloudWatch Logs log group.

You can configure multiple trails differently so that the trails process and log only the events that you specify. For example, one trail can log read-only data and management events, so that all read-only events are delivered to one S3 bucket. Another trail can log only write-only data and management events, so that all write-only events are delivered to a separate S3 bucket.

You can also configure your trails to have one trail log and deliver all management events to one S3 bucket, and configure another trail to log and deliver all data events to another S3 bucket.

You can configure your trails to log the following:

- **Data events (p. 140):** These events provide visibility into the resource operations performed on or within a resource. These are also known as data plane operations.
- **Management events (p. 136):** Management events provide visibility into management operations that are performed on resources in your AWS account. These are also known as control plane operations. Management events can also include non-API events that occur in your account. For example, when a user logs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see [Non-API events captured by CloudTrail \(p. 294\)](#).

Note

Not all AWS services support CloudTrail events. For more information about supported services, see [CloudTrail supported services and integrations \(p. 20\)](#). For specific details about what APIs are logged for a specific service, see that service's documentation in [CloudTrail supported services and integrations \(p. 20\)](#).

- **Insights events (p. 154):** Insights events capture unusual activity that is detected in your account. If you have Insights events enabled, and CloudTrail detects unusual activity, Insights events are logged to the destination S3 bucket for your trail, but in a different folder. You can also see the type of Insights event and the incident time period when you view Insights events on the CloudTrail console. Unlike other types of events captured in a CloudTrail trail, Insights events are logged only when CloudTrail detects changes in your account's API usage that differ significantly from the account's typical usage patterns.

Insights events are generated only for **write** management APIs.

Topics

- [Create multiple trails \(p. 134\)](#)

- [Logging management events for trails \(p. 136\)](#)
- [Logging data events for trails \(p. 140\)](#)
- [Logging Insights events for trails \(p. 154\)](#)
- [Receiving CloudTrail log files from multiple regions \(p. 158\)](#)
- [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#)
- [Receiving CloudTrail log files from multiple accounts \(p. 183\)](#)
- [Sharing CloudTrail log files between AWS accounts \(p. 186\)](#)
- [Validating CloudTrail log file integrity \(p. 196\)](#)
- [Using the CloudTrail Processing Library \(p. 215\)](#)

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. You can configure trails to deliver log files to separate S3 buckets or shared S3 buckets.

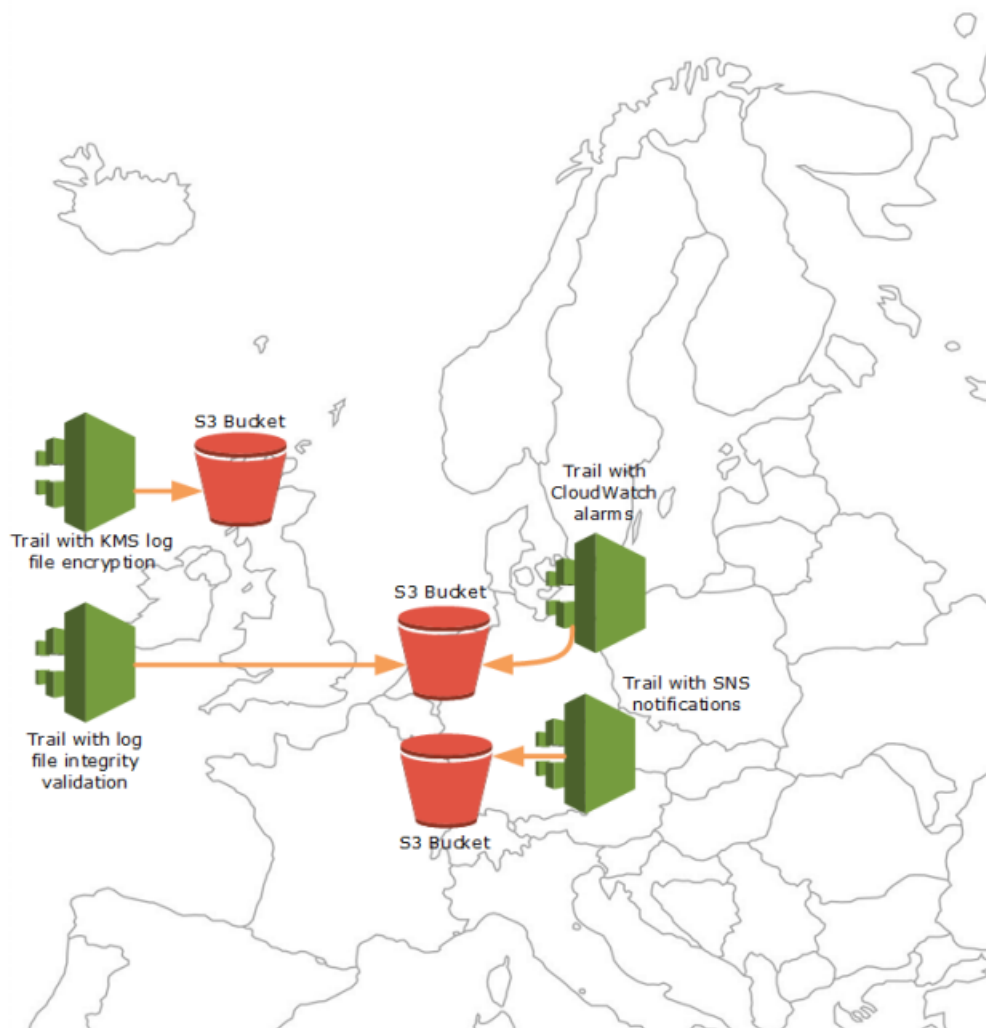
Note

Creating multiple trails will incur additional costs. For more information, see [AWS CloudTrail Pricing](#).

For example, you might have the following users:

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

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 [Examples: Creating and applying policies for actions on specific trails \(p. 234\)](#).

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

- [How does CloudTrail behave regionally and globally? \(p. 9\)](#)
- [CloudTrail FAQs](#)

Logging management events for trails

By default, trails log all management events and don't include data or Insights events. Additional charges apply for data or Insights events. For more information, see [AWS CloudTrail Pricing](#).

Contents

- [Management events \(p. 136\)](#)
 - [Logging management events with the AWS Management Console \(p. 136\)](#)
- [Read and write events \(p. 137\)](#)
- [Logging events with the AWS Command Line Interface \(p. 138\)](#)
- [Logging events with the AWS SDKs \(p. 140\)](#)
- [Sending events to Amazon CloudWatch Logs \(p. 140\)](#)

Management events

Management events provide visibility into management operations that are performed on resources in your AWS account. These are also known as control plane operations. Example management events include:

- Configuring security (for example, IAM `AttachRolePolicy` API operations)
- Registering devices (for example, Amazon EC2 `CreateDefaultVpc` API operations)
- Configuring rules for routing data (for example, Amazon EC2 `CreateSubnet` API operations)
- Setting up logging (for example, AWS CloudTrail `CreateTrail` API operations)

Management events can also include non-API events that occur in your account. For example, when a user logs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see [Non-API events captured by CloudTrail \(p. 294\)](#). For a list of supported management events that CloudTrail logs for AWS services, see [CloudTrail supported services and integrations \(p. 20\)](#).

By default, trails are configured to log management events. For a list of supported management events that CloudTrail logs for AWS services, see [CloudTrail supported services and integrations \(p. 20\)](#).

Note

The CloudTrail **Event history** feature supports only management events. Not all management events are supported in event history. You cannot exclude AWS KMS or Amazon RDS Data API events from **Event history**; settings that you apply to a trail do not apply to **Event history**. For more information, see [Viewing events with CloudTrail Event history \(p. 44\)](#).

Logging management events with the AWS Management Console

1. Open the **Trails** page of the CloudTrail console and choose the trail name.
2. For **Management events**, choose **Edit**.
 - Choose if you want your trail to log **Read** events, **Write** events, or both.
 - Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

- Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see [Logging Data API calls with AWS CloudTrail](#) in the *Amazon RDS User Guide for Aurora*.

Choose **Update trail** when you are finished.

Read and write events

When you configure your trail to log management events, you can specify whether you want read-only events, write-only events, or both.

- **Read**

Read-only events include API operations that read your resources, but don't make changes. For example, read-only events include the Amazon EC2 `DescribeSecurityGroups` and `DescribeSubnets` API operations. These operations return only information about your Amazon EC2 resources and don't change your configurations.

- **Write**

Write-only events include API operations that modify (or might modify) your resources. For example, the Amazon EC2 `RunInstances` and `TerminateInstances` API operations modify your instances.

Example: Logging read and write events for separate trails

The following example shows how you can configure trails to split log activity for an account into separate S3 buckets: one bucket receives read-only events and a second bucket receives write-only events.

1. You create a trail and choose an S3 bucket named `read-only-bucket` to receive log files. You then update the trail to specify that you want **Read** management events.
2. You create a second trail and choose an S3 bucket named `write-only-bucket` to receive log files. You then update the trail to specify that you want **Write** management events.
3. The Amazon EC2 `DescribeInstances` and `TerminateInstances` API operations occur in your account.
4. The `DescribeInstances` API operation is a read-only event and it matches the settings for the first trail. The trail logs and delivers the event to the `read-only-bucket`.
5. The `TerminateInstances` API operation is a write-only event and it matches the settings for the second trail. The trail logs and delivers the event to the `write-only-bucket`.

Logging events with the AWS Command Line Interface

You can configure your trails to log management events using the AWS CLI.

To view whether your trail is logging management events, run the `get-event-selectors` command.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

The following example returns the default settings for a trail. By default, trails log all management events, log events from all event sources, and don't log data events.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [],
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To configure your trail to log management events, run the `put-event-selectors` command. The following example shows how to configure your trail to include all management events for two S3 objects. You can specify from 1 to 5 event selectors for a trail. You can specify from 1 to 250 data resources for a trail.

Note

The maximum number of S3 data resources is 250, regardless of the number of event selectors.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors
'[{ "ReadWriteType": "All", "IncludeManagementEvents":true, "DataResources": [{ "Type":
  "AWS::S3::Object", "Values": ["arn:aws:s3:::mybucket/prefix", "arn:aws:s3:::mybucket2/
  prefix2"] }] }]'
```

The following example returns the event selector configured for the trail.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [],
      "IncludeManagementEvents": true,
      "DataResources": [
        {
          "Values": [
            "arn:aws:s3:::mybucket/prefix",
            "arn:aws:s3:::mybucket2/prefix2",
          ],
          "Type": "AWS::S3::Object"
        }
      ],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To exclude AWS Key Management Service (AWS KMS) events from a trail's logs, run the `put-event-selectors` command and add the attribute `ExcludeManagementEventSources` with a value of `kms.amazonaws.com`. The following example creates an event selector for a trail named `TrailName` to include read-only and write-only management events, but exclude AWS KMS events. Because AWS KMS can generate a high volume of events, the user in this example might want to limit events to manage the cost of a trail.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors '[{"ReadWriteType": "All", "ExcludeManagementEventSources": ["kms.amazonaws.com"], "IncludeManagementEvents": true}]'
```

The example returns the event selector configured for the trail.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [ "kms.amazonaws.com" ],
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To exclude Amazon RDS Data API events from a trail's logs, run the `put-event-selectors` command and add the attribute `ExcludeManagementEventSources` with a value of `rdsvdata.amazonaws.com`. The following example creates an event selector for a trail named `TrailName` to include read-only and write-only management events, but exclude Amazon RDS Data API events. Because Amazon RDS Data API can generate a high volume of events, the user in this example might want to limit events to manage the cost of a trail.

```
{
  "EventSelectors": [
    {
      "ExcludeManagementEventSources": [ "rdsvdata.amazonaws.com" ],
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging AWS KMS or Amazon RDS Data API events to a trail again, pass an empty string as the value of `ExcludeManagementEventSources`, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors '[{"ReadWriteType": "All", "ExcludeManagementEventSources": [], "IncludeManagementEvents": true}]'
```

To log relevant AWS KMS events to a trail like `Disable`, `Delete` and `ScheduleKey`, but exclude high-volume AWS KMS events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, log write-only management events, and keep the default setting to log AWS KMS events, as shown in the following example.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors '[{"ReadWriteType": "WriteOnly", "ExcludeManagementEventSources": [], "IncludeManagementEvents": true}]'
```


Logging events with the AWS SDKs

Use the [GetEventSelectors](#) operation to see whether your trail is logging management events for a trail. You can configure your trails to log management events with the [PutEventSelectors](#) operation. For more information, see the [AWS CloudTrail API Reference](#).

Sending events to Amazon CloudWatch Logs

CloudTrail supports sending data and management events to CloudWatch Logs. When you configure your trail to send events to your CloudWatch Logs log group, CloudTrail sends only the events that you specify in your trail. For example, if you configure your trail to log management events only, your trail delivers management events only to your CloudWatch Logs log group. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).

Logging data events for trails

By default, trails do not log data events. Additional charges apply for data events. For more information, see [AWS CloudTrail Pricing](#).

Note

The events that are logged by your trails are available in Amazon CloudWatch Events. For example, if you configure a trail to log data events for S3 objects but not management events, your trail processes and logs only data events for the specified S3 objects. The data events for these S3 objects are available in Amazon CloudWatch Events. For more information, see [AWS API Call Events](#) in the *Amazon CloudWatch Events User Guide*.

Contents

- [Data events \(p. 140\)](#)
 - [Examples: Logging data events for Amazon S3 objects \(p. 146\)](#)
 - [Logging data events for S3 objects in other AWS accounts \(p. 147\)](#)
- [Read-only and write-only events \(p. 148\)](#)
- [Logging data events with the AWS Command Line Interface \(p. 149\)](#)
 - [Log events by using basic event selectors \(p. 149\)](#)
 - [Log events by using advanced event selectors \(p. 150\)](#)
 - [Log all Amazon S3 events for a bucket by using advanced event selectors \(p. 151\)](#)
 - [Log Amazon S3 on AWS Outposts events by using advanced event selectors \(p. 152\)](#)
- [Logging data events for AWS Config compliance \(p. 153\)](#)
- [Logging events with the AWS SDKs \(p. 153\)](#)
- [Sending events to Amazon CloudWatch Logs \(p. 153\)](#)

Data events

Data events provide visibility into the resource operations performed on or within a resource. These are also known as data plane operations. Data events are often high-volume activities.

Use the following data event resource types with basic event selectors:

- Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on buckets and objects in buckets

- AWS Lambda function execution activity (the `Invoke` API)
- Amazon DynamoDB object-level API activity on tables (for example, `PutItem`, `DeleteItem`, and `UpdateItem` API operations).

In addition to basic event selectors, use the following data types with advanced event selectors:

- Amazon S3 on Outposts object-level API activity
- Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, such as `eth_getBalance` or `eth_getBlockByNumber`
- Amazon S3 Object Lambda access points API activity, such as calls to `CompleteMultipartUpload` and `GetObject`
- [Amazon Elastic Block Store \(EBS\)](#) direct APIs, such as `PutSnapshotBlock`, `GetSnapshotBlock`, and `ListChangedBlocks` on Amazon EBS snapshots
- Amazon S3 API activity on access points
- Amazon DynamoDB API activity on streams

Data events are not logged by default when you create a trail. To record CloudTrail data events, you must explicitly add the supported resources or resource types for which you want to collect activity to a trail. For more information, see [Creating a trail \(p. 70\)](#).

On a single-region trail, you can log data events only for resources that you can access in that region. Though S3 buckets are global, AWS Lambda functions and DynamoDB tables are regional.

Additional charges apply for logging data events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

Steps for logging data events depend on whether you have advanced event selectors enabled on your trail. Use the procedure in this section that matches the kind of event selectors you have enabled on a trail.

Logging data events with basic event selectors in the AWS Management Console

1. Open the **Trails** page of the CloudTrail console and choose the trail name.

Note

While you can edit an existing trail to add logging data events, as a best practice, consider creating a separate trail specifically for logging data events.

2. For **Data events**, choose **Edit**.
3. For Amazon S3 buckets:
 - a. For **Data event source**, choose **S3**.
 - b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

Note

Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If you are creating a trail for a single Region (done by using the AWS CLI), selecting the **Select all S3 buckets in your account** option enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

- c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.

- d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. To find specific buckets, type a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

To remove a bucket from logging, choose **X**.

4. To add another data type on which to log data events, choose **Add data event type**.
5. For Lambda functions:
 - a. For **Data event source**, choose **Lambda**.
 - b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

Note

If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

- c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

Note

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI \(p. 93\)](#).

6. To add another data type on which to log data events, choose **Add data event type**.
7. For DynamoDB tables:
 - a. For **Data event source**, choose **DynamoDB**.
 - b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN is in the following format:

`arn:partition:dynamodb:region:account_ID:table/table_name`

To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

8. Choose **Update trail**.

Logging data events with advanced event selectors in the AWS Management Console

In the AWS Management Console, if you have advanced event selectors enabled, you can choose from predefined templates that log all data events on a selected resource (Amazon S3 buckets or access points, Lambda functions, S3 objects on AWS Outposts, Ethereum for Managed Blockchain nodes, or S3 Object Lambda access points). After you choose a log selector template, you can customize the template to include only the data events you most want to see. For more information and tips about using advanced event selectors, see [Log events by using advanced event selectors \(p. 150\)](#) in this topic.

1. On the **Dashboard** or **Trails** pages of the CloudTrail console, choose a trail name to open it.
2. On the trail's details page, in **Data events**, choose **Edit**.
3. If you are not already logging data events, choose the **Data events** check box.
4. For **Data event type**, choose the resource type on which you want to log data events.
5. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

Note

Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.

If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail.

If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions. Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

6. If you want to apply a predefined log selector template, and you do not want to add another data event resource type, choose **Save changes**. You do not need to follow the rest of this procedure. To apply a custom log selector template, go on to the next step.
7. To create a custom log selector template, in the **Log selector template** drop-down list, choose **Custom**.
8. (Optional) Enter a name for your custom log selector template.
9. In **Advanced event selectors**, build an expression for the specific S3 buckets, S3 access points, DynamoDB tables, S3 objects on AWS Outposts, S3 Object on Lambda access points, Managed Blockchain nodes, or Lambda functions on which you want to collect data events.
 - a. Choose from the following fields. For fields that accept an array (more than one value), CloudTrail adds an OR between values.
 - **readOnly** - readOnly can be set to **Equals** a value of true or false. Read-only data events are events that do not change the state of a resource, such as `Get*` or `Describe*` events. Write events add, change, or delete resources, attributes, or artifacts, such as `Put*`, `Delete*`, or `Write*` events. To log both read and write events, don't add a readOnly selector.

- **eventName** - eventName can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as PutBucket, GetItem, or GetSnapshotBlock. You can have multiple values for this field, separated by commas.
- **resources.type** - In the AWS Management Console, this field doesn't occur, because it's already populated by your choice of data event type from the **Data event type** drop-down list. In the AWS CLI and SDKs, resources.type can only use the **Equals** operator, and the value can be one of the following:
 - AWS::S3::Object
 - AWS::Lambda::Function
 - AWS::DynamoDB::Table
 - AWS::S3Outposts::Object
 - AWS::ManagedBlockchain::Node
 - AWS::S3ObjectLambda::AccessPoint
 - AWS::EC2::Snapshot
 - AWS::S3::AccessPoint
 - AWS::DynamoDB::Stream
- **resources.ARN** - You can use any operator with resources.ARN, but if you use **Equals** or **NotEquals**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of resources.type.

For example, when resources.type equals **AWS::S3::Object**, the ARN must be in one of the following formats. To log all data events for all objects in a specific S3 bucket, use the StartsWith operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

```
arn:partition:s3::bucket_name/  
arn:partition:s3::bucket_name/object_or_file_name/
```

When resources.type equals **AWS::Lambda::Function**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:lambda:region:account_ID:function:function_name
```

When resources.type equals **AWS::DynamoDB::Table**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

When resources.type equals **AWS::S3Outposts::Object**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:s3-outposts:region:account_ID:object_path
```

When resources.type equals **AWS::ManagedBlockchain::Node**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:managedblockchain:region:account_ID:nodes/node_ID
```

When resources.type equals **AWS::S3ObjectLambda::AccessPoint**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name
```

When `resources.type` equals **AWS::EC2::Snapshot**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:ec2:region::snapshot/snapshot_ID
```

When `resources.type` equals **AWS::S3::AccessPoint**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in one of the following formats. To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

```
arn:partition:s3:region:account_ID:accesspoint/access_point_name  
arn:partition:s3:region:account_ID:accesspoint/access_point_name/  
object/object_path
```

When `resources.type` equals **AWS::DynamoDB::Stream**, and the operator is set to **Equals** or **NotEquals**, the ARN must be in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name/stream/date_time
```

For more information about the ARN formats of data event resources, see [Actions, resources, and condition keys](#) in the *AWS Identity and Access Management User Guide*.

- b. For each field, choose **+ Conditions** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **NotEquals**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

To add the second S3 bucket, choose **+ Conditions**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

Note

You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.

If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still log all functions with a predefined selector template, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI](#) (p. 93).

- c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.
 - d. Save changes to your custom selector template by choosing **Next**. Do not choose another log selector template, or leave this page, or your custom selectors will be lost.
10. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 4 through this step to configure advanced event selectors for the data event type.

11. After you choose **Next**, in **Step 2: Choose log events**, review the log selector template options you've chosen. Choose **Edit** to go back and make changes.
12. After you've reviewed and verified your choices, choose **Update trail** if this is an existing trail, or **Create trail** if you are creating a new trail.

Examples: Logging data events for Amazon S3 objects

Logging data events for all S3 objects in an S3 bucket

The following example demonstrates how logging works when you configure logging of all data events for an S3 bucket named *bucket-1*. In this example, the CloudTrail user specified an empty prefix, and the option to log both **Read** and **Write** data events.

1. A user uploads an object to *bucket-1*.
2. The `PutObject` API operation is an Amazon S3 object-level API. It is recorded as a data event in CloudTrail. Because the CloudTrail user specified an S3 bucket with an empty prefix, events that occur on any object in that bucket are logged. The trail processes and logs the event.
3. Another user uploads an object to *bucket-2*.
4. The `PutObject` API operation occurred on an object in an S3 bucket that wasn't specified for the trail. The trail doesn't log the event.

Logging data events for specific S3 objects

The following example demonstrates how logging works when you configure a trail to log events for specific S3 objects. In this example, the CloudTrail user specified an S3 bucket named *bucket-3*, with the prefix *my-images*, and the option to log only **Write** data events.

1. A user deletes an object that begins with the *my-images* prefix in the bucket, such as `arn:aws:s3:::bucket-3/my-images/example.jpg`.
2. The `DeleteObject` API operation is an Amazon S3 object-level API. It is recorded as a **Write** data event in CloudTrail. The event occurred on an object that matches the S3 bucket and prefix specified in the trail. The trail processes and logs the event.
3. Another user deletes an object with a different prefix in the S3 bucket, such as `arn:aws:s3:::bucket-3/my-videos/example.avi`.
4. The event occurred on an object that doesn't match the prefix specified in your trail. The trail doesn't log the event.
5. A user calls the `GetObject` API operation for the object, `arn:aws:s3:::bucket-3/my-images/example.jpg`.
6. The event occurred on a bucket and prefix that are specified in the trail, but `GetObject` is a read-type Amazon S3 object-level API. It is recorded as a **Read** data event in CloudTrail, and the trail is not configured to log **Read** events. The trail doesn't log the event.

Note

If you are logging data events for specific Amazon S3 buckets, we recommend you do not use an Amazon S3 bucket for which you are logging data events to receive log files that you have specified in the data events section. Using the same Amazon S3 bucket causes your trail to log a data event each time log files are delivered to your Amazon S3 bucket. Log files are aggregated events delivered at intervals, so this is not a 1:1 ratio of event to log file; the event is logged in the next log file. For example, when the trail delivers logs, the `PutObject` event occurs on the S3 bucket. If the S3 bucket is also specified in the data events section, the trail processes and logs the `PutObject` event as a data event. That action is another `PutObject` event, and the trail processes and logs the event again. For more information, see [How CloudTrail works \(p. 1\)](#).

To avoid logging data events for the Amazon S3 bucket where you receive log files if you configure a trail to log all Amazon S3 data events in your AWS account, consider configuring delivery of log files to an Amazon S3 bucket that belongs to another AWS account. For more information, see [Receiving CloudTrail log files from multiple accounts \(p. 183\)](#).

Logging data events for S3 objects in other AWS accounts

When you configure your trail to log data events, you can also specify S3 objects that belong to other AWS accounts. When an event occurs on a specified object, CloudTrail evaluates whether the event matches any trails in each account. If the event matches the settings for a trail, the trail processes and logs the event for that account. Generally, both API callers and resource owners can receive events.

If you own an S3 object and you specify it in your trail, your trail logs events that occur on the object in your account. Because you own the object, your trail also logs events when other accounts call the object.

If you specify an S3 object in your trail, and another account owns the object, your trail only logs events that occur on that object in your account. Your trail doesn't log events that occur in other accounts.

Example: Logging data events for an Amazon S3 object for two AWS accounts

The following example shows how two AWS accounts configure CloudTrail to log events for the same S3 object.

1. In your account, you want your trail to log data events for all objects in your S3 bucket named `owner-bucket`. You configure the trail by specifying the S3 bucket with an empty object prefix.
2. Bob has a separate account that has been granted access to the S3 bucket. Bob also wants to log data events for all objects in the same S3 bucket. For his trail, he configures his trail and specifies the same S3 bucket with an empty object prefix.
3. Bob uploads an object to the S3 bucket with the `PutObject` API operation.
4. This event occurred in his account and it matches the settings for his trail. Bob's trail processes and logs the event.
5. Because you own the S3 bucket and the event matches the settings for your trail, your trail also processes and logs the same event. Because there are now two copies of the event (one logged in Bob's trail, and one logged in yours), CloudTrail charges for two copies of the data event.
6. You upload an object to the S3 bucket.
7. This event occurs in your account and it matches the settings for your trail. Your trail processes and logs the event.
8. Because the event didn't occur in Bob's account, and he doesn't own the S3 bucket, Bob's trail doesn't log the event. CloudTrail charges for only one copy of this data event.

Example: Logging data events for all buckets, including an S3 bucket used by two AWS accounts

The following example shows the logging behavior when **Select all S3 buckets in your account** is enabled for trails that collect data events in an AWS account.

1. In your account, you want your trail to log data events for all S3 buckets. You configure the trail by choosing **Read** events, **Write** events, or both for **All current and future S3 buckets** in **Data events**.
2. Bob has a separate account that has been granted access to an S3 bucket in your account. He wants to log data events for the bucket to which he has access. He configures his trail to get data events for all S3 buckets.
3. Bob uploads an object to the S3 bucket with the `PutObject` API operation.
4. This event occurred in his account and it matches the settings for his trail. Bob's trail processes and logs the event.

5. Because you own the S3 bucket and the event matches the settings for your trail, your trail also processes and logs the event. Because there are now two copies of the event (one logged in Bob's trail, and one logged in yours), CloudTrail charges each account for a copy of the data event.
6. You upload an object to the S3 bucket.
7. This event occurs in your account and it matches the settings for your trail. Your trail processes and logs the event.
8. Because the event didn't occur in Bob's account, and he doesn't own the S3 bucket, Bob's trail doesn't log the event. CloudTrail charges for only one copy of this data event in your account.
9. A third user, Mary, has access to the S3 bucket, and runs a `GetObject` operation on the bucket. She has a trail configured to log data events on all S3 buckets in her account. Because she is the API caller, CloudTrail logs a data event in her trail. Though Bob has access to the bucket, he is not the resource owner, so no event is logged in his trail this time. As the resource owner, you receive an event in your trail about the `GetObject` operation that Mary called. CloudTrail charges your account and Mary's account for each copy of the data event: one in Mary's trail, and one in yours.

Read-only and write-only events

When you configure your trail to log data and management events, you can specify whether you want read-only events, write-only events, or both.

- **Read**

Read events include API operations that read your resources, but don't make changes. For example, read-only events include the Amazon EC2 `DescribeSecurityGroups` and `DescribeSubnets` API operations. These operations return only information about your Amazon EC2 resources and don't change your configurations.

- **Write**

Write events include API operations that modify (or might modify) your resources. For example, the Amazon EC2 `RunInstances` and `TerminateInstances` API operations modify your instances.

Example: Logging read and write events for separate trails

The following example shows how you can configure trails to split log activity for an account into separate S3 buckets: one bucket receives read-only events and a second bucket receives write-only events.

1. You create a trail and choose an S3 bucket named `read-only-bucket` to receive log files. You then update the trail to specify that you want **Read** management events and data events.
2. You create a second trail and choose an S3 bucket named `write-only-bucket` to receive log files. You then update the trail to specify that you want **Write** management events and data events.
3. The Amazon EC2 `DescribeInstances` and `TerminateInstances` API operations occur in your account.
4. The `DescribeInstances` API operation is a read-only event and it matches the settings for the first trail. The trail logs and delivers the event to the `read-only-bucket`.
5. The `TerminateInstances` API operation is a write-only event and it matches the settings for the second trail. The trail logs and delivers the event to the `write-only-bucket`.

Logging data events with the AWS Command Line Interface

You can configure your trails to log management and data events using the AWS CLI. To see whether your trail is logging management and data events, run the `get-event-selectors` command.

Note

Be aware that if your account is logging more than one copy of management events, you incur charges. There is always a charge for logging data events. For more information, see [AWS CloudTrail Pricing](#).

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

The command returns the default settings for a trail.

Topics

- [Log events by using basic event selectors \(p. 149\)](#)
- [Log events by using advanced event selectors \(p. 150\)](#)
- [Log all Amazon S3 events for a bucket by using advanced event selectors \(p. 151\)](#)
- [Log Amazon S3 on AWS Outposts events by using advanced event selectors \(p. 152\)](#)

Log events by using basic event selectors

The following is an example result of the `get-event-selectors` command showing basic event selectors. By default, when you create a trail by using the AWS CLI, a trail logs all management events. By default, trails do not log data events.

```
{
  "EventSelectors": [
    {
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To configure your trail to log management and data events, run the `put-event-selectors` command.

The following example shows how to use basic event selectors to configure your trail to include all management and data events for two S3 objects. You can specify from 1 to 5 event selectors for a trail. You can specify from 1 to 250 data resources for a trail.

Note

The maximum number of S3 data resources is 250, if you choose to limit data events by using basic event selectors.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors
'[{ "ReadWriteType": "All", "IncludeManagementEvents":true, "DataResources": [{ "Type":
"AWS::S3::Object", "Values": ["arn:aws:s3:::mybucket/prefix", "arn:aws:s3:::mybucket2/
prefix2"] }] }]'
```

The command returns the event selectors that are configured for the trail.

```
{
```

```
"EventSelectors": [
  {
    "IncludeManagementEvents": true,
    "DataResources": [
      {
        "Values": [
          "arn:aws:s3:::mybucket/prefix",
          "arn:aws:s3:::mybucket2/prefix2",
        ],
        "Type": "AWS::S3::Object"
      }
    ],
    "ReadWriteType": "All"
  },
],
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Log events by using advanced event selectors

If you have opted to use advanced event selectors, the **get-event-selectors** command returns results similar to the following. By default, no advanced event selectors are configured for a trail.

```
{
  "AdvancedEventSelectors": [],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

The following example shows how to use advanced event selectors to log all management events (both `readOnly` and `writeOnly`), and include `PutObject` and `DeleteObject` events for the S3 objects in the same two S3 bucket prefixes. As shown here, you can use advanced event selectors to select not only the S3 prefix names by ARN, but the names of the specific events that you want to log. You can add up to 500 conditions to advanced event selectors per trail, including all selector values. You can specify from 1 to 250 data resources for a trail.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors
'[
  {
    "Name": "Log readOnly and writeOnly management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  },
  {
    "Name": "Log PutObject and DeleteObject events for two S3 prefixes",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "eventName", "Equals": ["PutObject", "DeleteObject"] },
      { "Field": "resources.ARN", "StartsWith": ["arn:aws:s3:::mybucket/
prefix", "arn:aws:s3:::mybucket2/prefix2"] }
    ]
  }
]
```

The result shows the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
```

```
{
  "Name": "Log readOnly and writeOnly management events",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Management" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
},
{
  "Name": "Log PutObject and DeleteObject events for two S3 prefixes",
  "FieldSelectors": [
    {
      "Field": "eventCategory",
      "Equals": [ "Data" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.type",
      "Equals": [ "AWS::S3::Object" ],
      "StartsWith": [],
      "EndsWith": [],
      "NotEquals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    },
    {
      "Field": "resources.ARN",
      "Equals": [],
      "StartsWith": [ "arn:aws:s3:::mybucket/prefix", "arn:aws:s3:::mybucket2/
prefix2" ],
      "EndsWith": [],
      "Equals": [],
      "NotStartsWith": [],
      "NotEndsWith": []
    }
  ]
}
],
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Log all Amazon S3 events for a bucket by using advanced event selectors

The following example shows how to configure your trail to include all data events for all Amazon S3 objects in a specific S3 bucket. The value for S3 events for the `resources.type` field is `AWS::S3::Object`. Because the ARN values for S3 objects and S3 buckets are slightly different, you must add the `StartsWith` operator for `resources.ARN` to capture all events.

```
aws cloudtrail put-event-selectors --trail-name TrailName --region region \
--advanced-event-selectors \
'[
  {
```

```
    "Name": "S3EventSelector",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "resources.ARN", "StartsWith":
[ "arn:partition:s3::bucket_name/" ] }
    ]
  }
]
```

The command returns the following example output.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "S3EventSelector",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [
            "Data"
          ]
        },
        {
          "Field": "resources.type",
          "Equals": [
            "AWS::S3::Object"
          ]
        },
        {
          "Field": "resources.ARN",
          "StartsWith": [
            "arn:partition:s3::bucket_name/"
          ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:region:account_ID:trail/TrailName"
}
```

Log Amazon S3 on AWS Outposts events by using advanced event selectors

The following example shows how to configure your trail to include all data events for all Amazon S3 on Outposts objects in your outpost. In this release, the supported value for S3 on Outposts events for the `resources.type` field is `AWS::S3Outposts::Object`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --region region \
--advanced-event-selectors \
'[
  {
    "Name": "OutpostsEventSelector",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3Outposts::Object"] }
    ]
  }
]
```

The command returns the following example output.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "OutpostsEventSelector",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [
            "Data"
          ]
        },
        {
          "Field": "resources.type",
          "Equals": [
            "AWS::S3Outposts::Object"
          ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:region:account_ID:trail/TrailName"
}
```

Logging data events for AWS Config compliance

If you are using AWS Config conformance packs to help your enterprise maintain compliance with formalized standards such as those required by Federal Risk and Authorization Management Program (FedRAMP) or National Institute of Standards and Technology (NIST), conformance packs for compliance frameworks generally require you to log data events for Amazon S3 buckets, at minimum. Conformance packs for compliance frameworks include a [managed rule](#) called `cloudtrail-s3-dataevents-enabled` that checks for S3 data event logging in your account. Many conformance packs that are not associated with compliance frameworks also require S3 data event logging. The following are examples of conformance packs that include this rule.

- [Operational Best Practices for AWS Well-Architected Framework Security Pillar](#)
- [Operational Best Practices for FDA Title 21 CFR Part 11](#)
- [Operational Best Practices for FFIEC](#)
- [Operational Best Practices for FedRAMP\(Moderate\)](#)
- [Operational Best Practices for HIPAA Security](#)
- [Operational Best Practices for K-ISMS](#)
- [Operational Best Practices for Logging](#)

For a full list of sample conformance packs available in AWS Config, see [Conformance pack sample templates](#) in the *AWS Config Developer Guide*.

Logging events with the AWS SDKs

Run the [GetEventSelectors](#) operation to see whether your trail is logging data events for a trail. You can configure your trails to log data events by running the [PutEventSelectors](#) operation. For more information, see the [AWS CloudTrail API Reference](#).

Sending events to Amazon CloudWatch Logs

CloudTrail supports sending data events to CloudWatch Logs. When you configure your trail to send events to your CloudWatch Logs log group, CloudTrail sends only the events that you specify in your trail. For example, if you configure your trail to log data events only, your trail delivers data events only

to your CloudWatch Logs log group. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs](#) (p. 159).

Logging Insights events for trails

AWS CloudTrail Insights helps AWS users identify and respond to unusual activity associated with `write` API calls by continuously analyzing CloudTrail management events.

Insights events are logged when CloudTrail detects unusual `write` management API activity in your account. If you have CloudTrail Insights enabled and CloudTrail detects unusual activity, Insights events are delivered to the destination S3 bucket for your trail. You can also see the type of insight and the incident time period when you view Insights events on the CloudTrail console. Unlike other types of events captured in a CloudTrail trail, Insights events are logged only when CloudTrail detects changes in your account's API usage that differ significantly from the account's typical usage patterns.

CloudTrail Insights continuously monitors CloudTrail `write` management events, and uses mathematical models to determine the normal levels of API and service event activity for an account. CloudTrail Insights identifies behavior that is outside normal patterns, generates Insights events, and delivers those events to a `/CloudTrail-Insight` folder in the chosen destination S3 bucket for your trail. You can also access and view Insights events in the AWS Management Console for CloudTrail. For more information about how to access and view Insights events in the console and by using the AWS CLI, see [Viewing CloudTrail Insights events](#) (p. 55) in this guide.

By default, trails log all management events and don't include data events or Insights events. Additional charges apply for data and Insights events. For more information, see [AWS CloudTrail Pricing](#).

When an event occurs in your account, CloudTrail evaluates whether the event matches the settings for your trails. Only events that match your trail settings are delivered to your Amazon S3 bucket and Amazon CloudWatch Logs log group.

Contents

- [Understanding CloudTrail Insights](#) (p. 154)
 - [Filter column](#) (p. 155)
 - [Insights graph tab](#) (p. 155)
 - [Attributions tab](#) (p. 156)
 - [CloudTrail events tab](#) (p. 157)
 - [Insights event record tab](#) (p. 157)
- [Logging Insights events with the AWS Management Console](#) (p. 157)
- [Logging Insights events with the AWS Command Line Interface](#) (p. 157)
- [Logging events with the AWS SDKs](#) (p. 158)
- [Sending events to Amazon CloudWatch Logs](#) (p. 158)

Understanding CloudTrail Insights

CloudTrail Insights can help you detect unusual API activity in your AWS account by raising Insights events. CloudTrail Insights measures your normal patterns of API call volume, also called the *baseline*, and generates Insights events when the volume is outside normal patterns. Insights events are generated for `write` management APIs.

After you enable CloudTrail Insights for the first time on a trail, it can take up to 36 hours for CloudTrail to deliver the first Insights event, if unusual activity is detected. CloudTrail Insights analyzes `write` management events that occur in a single Region, not globally. A CloudTrail Insights event is generated in the same Region as its supporting management events are generated.

The following image shows an example of Insights events. You open details pages for an Insights event by choosing an Insights event name from the **Dashboard** or **Insights** pages.

If you disable CloudTrail Insights on a trail, or stop logging on a trail (which disables CloudTrail Insights), you may have Insights events stored in your destination S3 bucket, or shown on the **Insights** page of the console, that date from the earlier time that you had Insights enabled.

Filter column

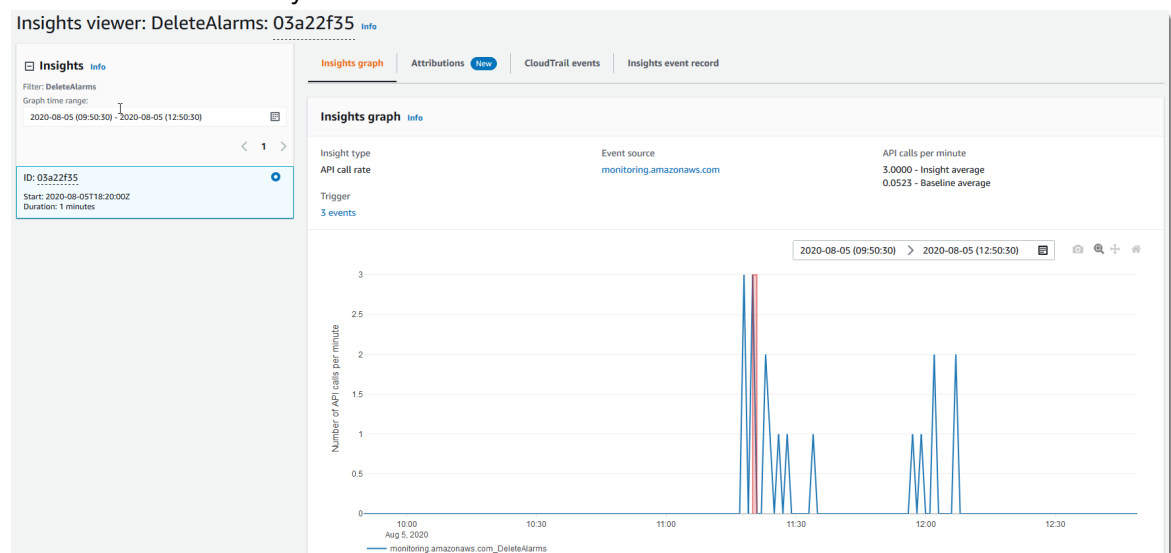
The left column lists Insights events that are related to the subject API, and that have the same Insights event type. The column lets you choose the Insights event about which you want more information. When you choose an event in this column, the event is highlighted in the graph on the **Insights graph** tab. By default, CloudTrail applies a filter that limits events shown on the **CloudTrail events** tab to those about the specific API that was called during the period of unusual activity that triggered the Insights event. To show all CloudTrail events called during the period of unusual activity, including events unrelated to the Insights event, turn off the filter.

Insights graph tab

On the **Insights graph** tab, the details page for an Insights event shows a graph of an API's call volume that occurred over a period of time before and after one or more Insights events are logged. In the graph, Insights events are highlighted with vertical bars, with the width of the bar showing the start and end time of the Insights event.

In this example, a vertical highlighting band shows unusual numbers of Amazon CloudWatch `DeleteAlarms` API calls in an account. In the highlighted area, because the number of `DeleteAlarms` calls rose above the account's baseline average of 0.05 calls per minute, CloudTrail logged an Insights event when it detected the unusual activity. The Insights event recorded that as many as 3 `DeleteAlarms` calls were made at about 11:20 a.m. This is about three more calls to that API per minute than is expected for the account. In this example, the graph's time span is three hours: 9:50 a.m. PDT on August 5, 2020 to 12:50 p.m. PDT on August 5, 2020. This event has a start time of 11:20 a.m. PDT on August 5, 2020, and an end time one minute later.

The baseline is calculated over the seven days preceding the start of an Insights event. Though the value of the baseline duration—the period that CloudTrail measures for normal activity on APIs—is approximately seven days, CloudTrail rounds the baseline duration to a whole integer day, so the exact baseline duration can vary.



Attributions tab

The **Attributions** tab shows the following information about an Insights event.

| Insights graph | Attributions New | CloudTrail events | Insights event record |
|--|--|-------------------|-----------------------|
| Top user identity ARNs during Insights event Info | | | |
| | User identity ARN | Insight average | Baseline average |
| 1 | arn:aws:sts::[REDACTED]:assumed-role/AWSServiceRoleForApplicationAutoScaling_DynamoDBTable/AutoScaling-ManagedAlarms | 3.0000 (100.000%) | 0.0523 (100.000%) |
| Average API calls during Insights event | | 3.0000 | 0.0523 |
| ▶ Top baseline user identity ARNs | | | |
| Top user agents during Insights event Info | | | |
| | User agent | Insight average | Baseline average |
| 1 | dynamodb.application-autoscaling.amazonaws.com | 3.0000 (100.000%) | 0.0523 (100.000%) |
| Average API calls during Insights event | | 3.0000 | 0.0523 |
| ▶ Top baseline user agents | | | |
| Top error codes during Insights event Info | | | |
| | Error code | Insight average | Baseline average |
| 1 | None | 3.0000 (100.000%) | 0.0523 (100.000%) |
| Average API calls during Insights event | | 3.0000 | 0.0523 |
| ▶ Top baseline error codes | | | |

- **Top user identity ARNs** - This table shows up to the top five AWS users or IAM roles (user identities) that contributed to API calls during the unusual activity and baseline periods, in descending order by the average number of API calls contributed. The percentage of the averages as a total of activity that contributed to the unusual activity is shown in parentheses. If more than five user identity ARNs contributed to the unusual activity, their activity is summed up in an **Other** row.
- **Top user agents** - This table shows up to the top five AWS tools by which the user identity contributed to API calls during the unusual activity and baseline periods, in descending order by the average number of API calls contributed. These tools include the AWS Management Console, AWS CLI, or the AWS SDKs. For example, a user agent named `ec2.amazonaws.com` indicates that the Amazon EC2 console was among the tools used to call the API. The percentage of the averages as a total of activity that contributed to the unusual activity is shown in parentheses. If more than five user agents contributed to the unusual activity, their activity is summed up in an **Other** row.
- **Top error codes** - This table shows up to the top five error codes that occurred on API calls during the unusual activity and baseline periods, in descending order from largest number of API calls to smallest. The percentage of the averages as a total of activity that contributed to the unusual activity is shown in parentheses. If more than five error codes occurred during the unusual or baseline activity, their activity is summed up in an **Other** row.

A value of `None` as one of the top five error code values means that a significant percentage of the calls that contributed to the Insights event did not result in errors. If the error code value is `None`, and there are no other error codes in the table, the values in the **Insight average** and **Baseline average** columns are the same as those for the Insights event overall. You can also see those values displayed in the **Insight average** and **Baseline average** legend on the **Insights graph** tab, under **API calls per minute**.

CloudTrail events tab

On the **CloudTrail events** tab, view related events that CloudTrail analyzed to determine that unusual activity occurred. By default, a filter is already applied for the Insights event name, which is also the name of the related API. To show all CloudTrail events logged during the period of unusual activity, turn off **Only show events for selected Insights event**. The **CloudTrail events** tab shows CloudTrail management events related to the subject API that occurred between the start and end time of the Insights event. These events help you perform deeper analysis to determine the probable cause of an Insights event, and reasons for unusual API activity.

Insights event record tab

Like any CloudTrail event, a CloudTrail Insights event is a record in JSON format. The **Insights event record** tab shows the JSON structure and content of the Insights start and end events, sometimes called the event *payload*. For more information about the fields and content of the Insights event record, see [Record fields for Insights events \(p. 279\)](#) and [CloudTrail Insights `insightDetails` element \(p. 288\)](#) in this guide.

Logging Insights events with the AWS Management Console

Enable CloudTrail Insights events on an existing trail. By default, Insights events are not enabled.

1. In the left navigation pane of the CloudTrail console, open the **Trails** page, and choose a trail name.
2. In **Insights events** choose **Edit**.

Note

Additional charges apply for logging Insights events. For CloudTrail pricing, see [AWS CloudTrail Pricing](#).

3. In **Event type**, select **Insights events**. You must be logging **Write** management events to log Insights events.
4. Choose **Update trail** to save your changes.

It can take up to 36 hours for CloudTrail to deliver the first Insights events, if unusual activity is detected.

Logging Insights events with the AWS Command Line Interface

You can configure your trails to log Insights events using the AWS CLI.

To view whether your trail is logging Insights events, run the `get-insight-selectors` command.

```
aws cloudtrail get-insight-selectors --trail-name TrailName
```

The following result shows the default settings for a trail. By default, trails don't log Insights events. The `InsightType` attribute value is empty, and no Insight event selectors are specified, because Insights event collection is not enabled.

```
{
  "InsightSelectors":
  [
```

```
{
  "InsightType": ""
},
"TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/TrailName"
}
```

To configure your trail to log Insights events, run the `put-insight-selectors` command. The following example shows how to configure your trail to include Insights events. In this release, the only Insights selector is `ApiCallRateInsight`.

```
aws cloudtrail put-insight-selectors --trail-name TrailName --insight-selectors
'[{ "InsightType": "ApiCallRateInsight" } ]'
```

The following result shows the Insights event selector that is configured for the trail.

```
{
  "InsightSelectors":
  [
    {
      "InsightType": "ApiCallRateInsight"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/TrailName"
}
```

Logging events with the AWS SDKs

Run the [GetInsightSelectors](#) operation to see whether your trail is logging Insights events for a trail. You can configure your trails to log Insights events with the [PutInsightSelectors](#) operation. For more information, see the [AWS CloudTrail API Reference](#).

Sending events to Amazon CloudWatch Logs

CloudTrail supports sending Insights events to CloudWatch Logs. When you configure your trail to send Insights events to your CloudWatch Logs log group, CloudTrail Insights sends only the events that you specify in your trail. For example, if you configure your trail to log management and Insights events, your trail delivers management and Insights events to your CloudWatch Logs log group. To configure CloudWatch Events with the CloudWatch console or API, choose the **AWS Insight via CloudTrail** event type on the **Create rule** page in the CloudWatch console. For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).

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 change an existing single-region trail to log all regions, CloudTrail logs events from all regions in your account. CloudTrail delivers log files to the same S3 bucket and CloudWatch Logs log group. As long as CloudTrail has permissions to write to an S3 bucket, the bucket for a multi-region trail does not have to be in the trail's home region.

In the console, by default, you create a trail that logs events in all AWS Regions. This is a recommended best practice. To log events in a single region (not recommended), [use the AWS CLI \(p. 90\)](#). To configure an existing single-region trail to log in all regions, you must use the AWS CLI.

To change an existing trail so that it applies to all Regions, add the `--is-multi-region-trail` option to the [update-trail](#) (p. 90) command.

```
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-east-2:123456789012:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
  "IsOrganizationTrail": false,
  "S3BucketName": "my-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? \(p. 9\)](#)
- [CloudTrail FAQs](#)

Monitoring CloudTrail Log Files with Amazon CloudWatch Logs

You can configure CloudTrail with CloudWatch Logs to monitor your trail logs and be notified when specific activity occurs.

1. Configure your trail to send log events to CloudWatch Logs.
2. Define CloudWatch Logs metric filters to evaluate log events for matches in terms, phrases, or values. For example, you can monitor for `ConsoleLogin` events.
3. Assign CloudWatch metrics to the metric filters.
4. Create CloudWatch alarms that are triggered according to thresholds and time periods that you specify. You can configure alarms to send notifications when alarms are triggered, so that you can take action.
5. You can also configure CloudWatch to automatically perform an action in response to an alarm.

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

For more information about the regions in which you can configure your trails to send logs to CloudWatch Logs, see [Amazon CloudWatch Logs Regions and Quotas](#) in the *AWS General Reference*.

The AWS GovCloud (US-West) region requires a separate account. For more information, see [AWS GovCloud \(US-West\)](#).

Topics

- [Sending events to CloudWatch Logs \(p. 160\)](#)

- [Creating CloudWatch alarms with an AWS CloudFormation template \(p. 164\)](#)
- [Creating CloudWatch alarms for CloudTrail events: examples \(p. 176\)](#)
- [Configuring notifications for CloudWatch Logs alarms \(p. 181\)](#)
- [Stopping CloudTrail from sending events to CloudWatch Logs \(p. 181\)](#)
- [CloudWatch log group and log stream naming for CloudTrail \(p. 182\)](#)
- [Role policy document for CloudTrail to use CloudWatch Logs for monitoring \(p. 182\)](#)

Sending events to CloudWatch Logs

When you configure your trail to send events to CloudWatch Logs, CloudTrail sends only the events that match your trail settings. For example, if you configure your trail to log data events only, your trail sends data events only to your CloudWatch Logs log group. CloudTrail supports sending data, Insights, and management events to CloudWatch Logs. For more information, see [Working with CloudTrail log files \(p. 133\)](#).

To send events to a CloudWatch Logs log group:

- Make sure you have sufficient permissions to create or specify an IAM role. For more information, see [Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console \(p. 242\)](#).
- Create a new trail or specify an existing one. For more information, see [Creating and updating a trail with the console \(p. 69\)](#).
- Create a log group or specify an existing one.
- Specify an IAM role. If you are modifying an existing IAM role for an organization trail, you must manually update the policy to allow logging for the organization trail. For more information, see [this policy example \(p. 163\)](#) and [Creating a trail for an organization \(p. 112\)](#).
- Attach a role policy or use the default.

Contents

- [Configuring CloudWatch Logs monitoring with the console \(p. 160\)](#)
 - [Creating a log group or specifying an existing log group \(p. 160\)](#)
 - [Specifying an IAM role \(p. 161\)](#)
 - [Viewing events in the CloudWatch console \(p. 161\)](#)
- [Configuring CloudWatch Logs monitoring with the AWS CLI \(p. 162\)](#)
 - [Creating a log group \(p. 162\)](#)
 - [Creating a role \(p. 162\)](#)
 - [Creating a policy document \(p. 163\)](#)
 - [Updating the trail \(p. 164\)](#)
- [Limitation \(p. 164\)](#)

Configuring CloudWatch Logs monitoring with the console

You can use the AWS Management Console to configure your trail to send 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 log group or specify an existing one.

To create or specify a log group

1. Make sure you are logged in with an administrative IAM user or role with sufficient permissions to configure CloudWatch Logs integration. For more information, see [Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console \(p. 242\)](#).
2. Open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
3. Choose the trail name. If you choose a trail that applies to all regions, you will be redirected to the region in which the trail was created. You can create a log group or choose an existing log group in the same region as the trail.

Note

A trail that applies to all regions sends log files from all regions to the CloudWatch Logs log group that you specify.

4. For **CloudWatch Logs**, choose **Configure**.
5. For **New or existing log group**, type the log group name, and then choose **Continue**. For more information, see [CloudWatch log group and log stream naming for CloudTrail \(p. 182\)](#).
6. For the IAM role, choose an existing role or create one. If you create an IAM role, type a role name.
7. Choose **Allow** to grant CloudTrail permissions to create a CloudWatch Logs log stream and deliver events.

Specifying an IAM role

You can specify a role for CloudTrail to assume to deliver events to the log stream.

To specify a role

1. By default, the `CloudTrail_CloudWatchLogs_Role` is specified for you. The default role policy has the required permissions to create a CloudWatch Logs log stream in a log group that you specify, and to deliver CloudTrail events to that log stream.

Note

If you want to use this role for a log group for an organization trail, you must manually modify the policy after you create the role. For more information, see [this policy example \(p. 163\)](#) and [Creating a trail for an organization \(p. 112\)](#).

- a. To verify the role, go to the AWS Identity and Access Management console at <https://console.aws.amazon.com/iam/>.
 - b. Choose **Roles** and then choose the **CloudTrail_CloudWatchLogs_Role**.
 - c. To see the contents of the role policy, choose **View Policy Document**.
2. You can specify another role, but you must attach the required role policy to the existing role if you want to use it to send events to CloudWatch Logs. For more information, see [Role policy document for CloudTrail to use CloudWatch Logs for monitoring \(p. 182\)](#).

Viewing events in the CloudWatch console

After you configure your trail to send events to your CloudWatch Logs log group, you can view the events in the CloudWatch console. CloudTrail typically delivers events to your log group within an average of about 15 minutes of an API call. This time is not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information.

To view events in the CloudWatch console

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.

2. Choose **Logs**.
3. Choose the log group that you specified for your trail.
4. Choose the log stream name.
5. To see the details of the event that your trail logged, choose an event.

Note

The **Time (UTC)** column in the CloudWatch console shows when the event was delivered to your log group. To see the actual time that the event was logged by CloudTrail, see the `eventTime` field.

Configuring CloudWatch Logs monitoring with the AWS CLI

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

Creating a log group

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

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

The following example creates a log group named `CloudTrail/logs`:

```
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

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 use gives `AssumeRole` permissions to CloudTrail. The `create-role` command creates the role with the required 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 `AssumeRole` 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.

Creating a policy document

Create the following role policy document for CloudTrail. This document grants CloudTrail the permissions required to create a CloudWatch Logs log stream in the log group 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:accountID_CloudTrail_region*"
      ]
    },
    {
      "Sid": "AWSCloudTrailPutLogEvents20141101",
      "Effect": "Allow",
      "Action": [
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:region:accountID:log-group:log_group_name:log-  
stream:accountID_CloudTrail_region*"
      ]
    }
  ]
}
```

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

If you're creating a policy that might be used for organization trails as well, you will need to configure it slightly differently. For example, the following policy grants CloudTrail the permissions required to create a CloudWatch Logs log stream in the log group you specify and to deliver CloudTrail events to that log stream for both trails in the AWS account 111111111111 and for organization trails created in the 111111111111 account that are applied to the AWS Organizations organization with the ID of *o-exampleorgid*:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailCreateLogStream20141101",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
```



```
        "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
        "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:o-exampleorgid_*",
    ]
},
{
    "Sid": "AWSCloudTrailPutLogEvents20141101",
    "Effect": "Allow",
    "Action": [
        "logs:PutLogEvents"
    ],
    "Resource": [
        "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
        "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:o-exampleorgid_*",
    ]
}
]
```

For more information about organization trails, see [Creating a trail for an organization \(p. 112\)](#).

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, see the [AWS CloudTrail Command Line Reference](#).

Limitation

CloudWatch Logs and CloudWatch Events each [allow a maximum event size of 256 KB](#). Although most service events have a maximum size of 256 KB, some services still have events that are larger. CloudTrail does not send these events to CloudWatch Logs or CloudWatch Events.

Starting with CloudTrail event version 1.05, events have a maximum size of 256 KB. This is to help prevent exploitation by malicious actors, and allow events to be consumed by other AWS services, such as CloudWatch Logs and CloudWatch Events.

Creating CloudWatch alarms with an AWS CloudFormation template

After you configure your trail to deliver log files to your CloudWatch log group, you can create CloudWatch metric filters and alarms to monitor the events in the log files. For example, you can specify an event such as the Amazon EC2 `RunInstances` operation, so that CloudWatch sends you notifications when that event occurs in your account. You can create your filters and alarms separately or use the AWS CloudFormation template to define them all at once.

You can use the example CloudFormation template as is, or as a reference to create your own template.

Topics

- [Example CloudFormation template \(p. 165\)](#)
- [Creating a CloudFormation stack with the template \(p. 165\)](#)
- [CloudFormation template contents \(p. 173\)](#)

Example CloudFormation template

The CloudFormation template has predefined CloudWatch metric filters and alarms, so that you receive email notifications when specific security-related API calls are made in your AWS account.

The template is available in a zip file at the following location:
[CloudWatch_Alarms_for_CloudTrail_API_Activity.zip](#).

The template defines metric filters that monitor create, delete, and update operations for the following resource types:

- Amazon EC2 instances
- IAM policies
- Internet gateways
- Network ACLs
- Security groups

When an API call occurs in your account, a metric filter monitors that API call. If the API call exceeds the thresholds that you specify, this triggers the alarm and CloudWatch sends you an email notification.

By default, most of the filters in the template trigger an alarm when a monitored event occurs within a five-minute period. You can modify these alarm thresholds for your own requirements. For example, you can monitor for three events in a ten-minute period. To make the changes, edit the template or, after uploading the template, specify the thresholds in the [CloudWatch console](#).

Note

Because CloudTrail typically delivers log files every five minutes, specify alarm periods of five minutes or more.

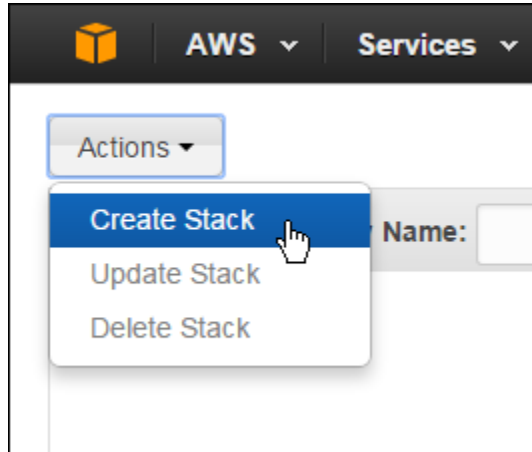
To see the metric filters and alarms in the template, and the API calls that trigger email notifications, see [CloudFormation template contents \(p. 173\)](#).

Creating a CloudFormation stack with the template

A CloudFormation stack is a collection of related resources that you provision and update as a single unit. The following procedure describes how to create the stack and validate the email address that receives notifications.

To create a CloudFormation stack with the template

1. Configure your trail to deliver log files to your CloudWatch Logs log group. See [Sending events to CloudWatch Logs \(p. 160\)](#).
2. Download the CloudFormation template: [CloudWatch_Alarms_for_CloudTrail_API_Activity.zip](#).
3. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
4. Choose **Create Stack**.



5. On the **Select Template** page, for **Name**, type a stack name. For example, CloudWatchAlarmsForCloudTrail.

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.

Name

6. For **Source**, choose **Upload a template to Amazon S3**.

Template

A template is a JSON-formatted text file that describes your stack's resources and their properties. The console 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

7. Choose **Choose File**, and then select the AWS CloudFormation template that you downloaded.
8. Choose **Next**.
9. On the **Specify Parameters** page, for **Email**, type the email address to receive notifications.
10. For **LogGroupName**, type the name of the log group that you specified when you configured your trail to deliver log files to CloudWatch Logs.

Specify Parameters

Specify values or use the default values for the parameters that are associated with your AWS CloudTrail log group.

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

11. Choose **Next**.
12. For **Options**, 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 keys per 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

13. Choose **Next**.
14. On the **Review** page, verify that your settings are correct.

Review

Template

| | |
|----------------------|---|
| Name | CloudWatchAlarmsForCloudTrail |
| Template URL | https://s3-us-west-2.amazonaws.com/cf-templates-1klf4fpqkb57z/CloudWatch_Alarms_for_CloudTrail_API_Activity.json |
| Description | AWS CloudTrail API Activity Alarm Template for CloudWatch Log |
| 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 |

15. Choose **Create**. The stack is created in a few minutes.

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

16. After the stack is created, you will receive an email at the address that you specified.

17. In the email, choose **Confirm subscription**. You receive email notifications when the alarms specified by the template are triggered.

You have chosen to subscribe to the topic:

**arn:aws:sns:us-west-2:111222333444:CloudWatchAlarmsForCloudTrail-AlarmNotification
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 emails, send email to [sns-opt-out](#)



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-east-1** region has entered the ALARM state, because "Threshold Crossed: 1 datapoint (7.0) was greater than 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=CloudTrail>

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 (7.0)
- 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 sec

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- ABC1DEFG
- 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 at <https://aws.amazon.com/support>

CloudFormation template contents

The following tables show the metric filters and alarms in the template, their purpose, and the API calls that trigger email notifications. Notifications are triggered when one or more of the API calls for a listed filter occur in your account.

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

Amazon S3 bucket events

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

Network events

| Metric Filter and Alarm | Monitor and Send Notifications for: | Notifications triggered by one or more of the following API operations: |
|---|--|--|
| SecurityGroupChangesMetricFilter SecurityGroupChangesAlarm | API calls that create, update, and delete security groups. | CreateSecurityGroup DeleteSecurityGroup AuthorizeSecurityGroupEgress RevokeSecurityGroupEgress AuthorizeSecurityGroupIngress RevokeSecurityGroupIngress |
| NetworkAclChangesMetricFilter NetworkAclChangesAlarm | API calls that create, update, and delete network ACLs. | CreateNetworkAcl DeleteNetworkAcl CreateNetworkAclEntry DeleteNetworkAclEntry ReplaceNetworkAclAssociation ReplaceNetworkAclEntry |

| Metric Filter and Alarm | Monitor and Send Notifications for: | Notifications triggered by one or more of the following API operations: |
|---|---|---|
| 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 the following API operations: |
|---|--|---|
| EC2InstanceChangesMetricFilter EC2InstanceChangesAlarm | The creation, termination, start, stop, and reboot of EC2 instances. | RebootInstances RunInstances StartInstances StopInstances TerminateInstances |
| EC2LargeInstanceChangesMetricFilter EC2LargeInstanceChangesAlarm | The creation, termination, start, stop, and reboot of 4x and 8x large EC2 instances. | At least one of the following API operations: RebootInstances RunInstances StartInstances |

| Metric Filter and Alarm | Monitor and Send Notifications for: | Notifications triggered by one or more of the following API operations: |
|-------------------------|-------------------------------------|---|
| | | StopInstances TerminateInstances and at least one of the following instance types: instancetype=*.4xlarge instancetype=*.8xlarge |

CloudTrail and IAM events

| Metric Filter and Alarm | Monitor and Send Notifications for: | Notifications triggered by one or more of the following API operations: |
|---|---|---|
| CloudTrailChangesMetricFilter CloudTrailChangesAlarm | Creating, deleting, and updating trails. The occurrence of starting and stopping logging for a trail. | CreateTrail DeleteTrail StartLogging StopLogging UpdateTrail |
| ConsoleSignInFailuresMetricFilter ConsoleSignInFailuresAlarm | Console login failures | eventName is ConsoleLogin and errorMessage is "Failed authentication" |
| AuthorizationFailuresMetricFilter AuthorizationFailuresAlarm | Authorization failures | Any API call that results in an error code: AccessDenied or *UnauthorizedOperation. |
| IAMPolicyChangesMetricFilter IAMPolicyChangesAlarm | Changes to IAM policies | AttachGroupPolicy DeleteGroupPolicy DetachGroupPolicy PutGroupPolicy CreatePolicy DeletePolicy CreatePolicyVersion DeletePolicyVersion AttachRolePolicy |

| Metric Filter and Alarm | Monitor and Send Notifications for: | Notifications triggered by one or more of the following API operations: |
|-------------------------|-------------------------------------|--|
| | | DeleteRolePolicy DetachRolePolicy PutRolePolicy AttachUserPolicy DeleteUserPolicy DetachUserPolicy PutUserPolicy |

Creating CloudWatch alarms for CloudTrail events: examples

This topic describes how to configure alarms for CloudTrail events, and includes examples.

Note

Instead of manually creating the following metric filters and alarms examples, you can use an AWS CloudFormation template to create them all at once. For more information, see [Creating CloudWatch alarms with an AWS CloudFormation template \(p. 164\)](#).

Topics

- [Prerequisites \(p. 176\)](#)
- [Create a metric filter and create an alarm \(p. 176\)](#)
- [Example security group configuration changes \(p. 177\)](#)
- [Example AWS Management Console sign-in failures \(p. 178\)](#)
- [Example: IAM policy changes \(p. 179\)](#)

Prerequisites

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

- Create a trail with the console or CLI.
- Create a log group, which you can do as part of creating a trail.
- 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 events to CloudWatch Logs \(p. 160\)](#). Examples in this section are performed in the Amazon CloudWatch Logs console. For more information about how to create metric filters and alarms, see [Creating metrics from log events using filters](#) and [Using Amazon CloudWatch alarms](#) in the *Amazon CloudWatch User Guide*.

Create a metric filter and create an alarm

To create an alarm, you must first create a metric filter, and then configure an alarm based on the filter. The procedures are shown for all examples. 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 Logs User Guide*.

Example security group configuration changes

Follow this procedure to create an Amazon CloudWatch alarm that is triggered when configuration changes occur on security groups.

Create a metric filter

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, choose **Logs**.
3. In the list of log groups, choose the log group that you created for CloudTrail log events.
4. Choose **Actions**, and then choose **Create metric filter**.
5. On the **Define pattern** page, in **Create filter pattern**, enter the following for **Filter pattern**.

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

6. In **Test pattern**, leave defaults. Choose **Next**.
7. On the **Assign metric** page, for **Filter name**, enter **SecurityGroupEvents**.
8. In **Metric details**, turn on **Create new**, and then enter **CloudTrailMetrics** for **Metric namespace**.
9. For **Metric name**, type **SecurityGroupEventCount**.
10. For **Metric value**, type **1**.
11. Leave **Default value** blank.
12. Choose **Next**.
13. On the **Review and create** page, review your choices. Choose **Create metric filter** to create the filter, or choose **Edit** to go back and change values.

Create an alarm

After you create the metric filter, the CloudWatch Logs log group details page for your CloudTrail trail log group opens. Follow this procedure to create an alarm.

1. On the **Metric filters** tab, find the metric filter you created in [the section called “Create a metric filter” \(p. 177\)](#). Fill the check box for the metric filter. In the **Metric filters** bar, choose **Create alarm**.
2. On the **Create Alarm** page, in **Specify metric and conditions**, enter the following.
 - a. For **Graph**, the line is set at **1** based on other settings you make when you create your alarm.
 - b. For **Metric name**, keep the current metric name, **SecurityGroupEventCount**.
 - c. For **Statistic**, keep the default, **Sum**.
 - d. For **Period**, keep the default, **5 minutes**.
 - e. In **Conditions**, for **Threshold type**, choose **Static**.
 - f. For **Whenever *metric_name* is**, choose **Greater/Equal**.
 - g. For **Threshold**, enter **1**.
 - h. In **Additional configuration**, leave defaults. Choose **Next**.
 - i.

3. On the **Configure actions** page, choose **In alarm**, which indicates that the action is taken when the threshold of 1 change event in 5 minutes is crossed, and **SecurityGroupEventCount** is in an alarm state.
 - a. For **Select an SNS topic**, choose **Create new**.
 - b. Enter **SecurityGroupChanges_CloudWatch_Alarms_Topic** as the name for the new Amazon SNS topic.
 - c. In **Email endpoints that will receive the notification**, enter email addresses of users whom you want to receive notifications if this alarm is raised. Separate email addresses with commas.

Email recipients specified here receive email asking them to confirm that they want to be subscribed to the Amazon SNS topic.
 - d. Choose **Create topic**.
4. For this example, skip the other action types. Choose **Next**.
5. On the **Add name and description** page, enter a friendly name for the alarm, and a description. For this example, enter **Security group configuration changes** for the name, and **Raises alarms if security group configuration changes occur** for the description. Choose **Next**.
6. On the **Preview and create** page, review your choices. Choose **Edit** to make changes, or choose **Create alarm** to create the alarm.

After you create the alarm, CloudWatch opens the **Alarms** page. The alarm's **Actions** column shows **Pending confirmation** until all email recipients on the SNS topic have confirmed that they want to subscribe to SNS notifications.

Example AWS Management Console sign-in failures

Follow this procedure to create an Amazon CloudWatch alarm that is triggered when there are three or more AWS Management Console 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, choose **Logs**.
3. In the list of log groups, choose the log group that you created for CloudTrail log events.
4. Choose **Actions**, and then choose **Create metric filter**.
5. On the **Define pattern** page, in **Create filter pattern**, enter the following for **Filter pattern**.

```
{ ($.eventName = ConsoleLogin) && ($.errorMessage = "Failed authentication") }
```

6. In **Test pattern**, leave defaults. Choose **Next**.
7. On the **Assign metric** page, for **Filter name**, enter **ConsoleSignInFailures**.
8. In **Metric details**, turn on **Create new**, and then enter **CloudTrailMetrics** for **Metric namespace**.
9. For **Metric name**, type **ConsoleSigninFailureCount**.
10. For **Metric value**, type **1**.
11. Leave **Default value** blank.
12. Choose **Next**.
13. On the **Review and create** page, review your choices. Choose **Create metric filter** to create the filter, or choose **Edit** to go back and change values.

Create an alarm

After you create the metric filter, the CloudWatch Logs log group details page for your CloudTrail trail log group opens. Follow this procedure to create an alarm.

1. On the **Metric filters** tab, find the metric filter you created in [the section called "Create a metric filter" \(p. 178\)](#). Fill the check box for the metric filter. In the **Metric filters** bar, choose **Create alarm**.
2. On the **Create Alarm** page, in **Specify metric and conditions**, enter the following.
 - a. For **Graph**, the line is set at **3** based on other settings you make when you create your alarm.
 - b. For **Metric name**, keep the current metric name, **ConsoleSignInFailureCount**.
 - c. For **Statistic**, keep the default, **Sum**.
 - d. For **Period**, keep the default, **5 minutes**.
 - e. In **Conditions**, for **Threshold type**, choose **Static**.
 - f. For **Whenever *metric_name* is**, choose **Greater/Equal**.
 - g. For **Threshold**, enter **3**.
 - h. In **Additional configuration**, leave defaults. Choose **Next**.
 - i.
3. On the **Configure actions** page, choose **In alarm**, which indicates that the action is taken when the threshold of 3 change events in 5 minutes is crossed, and **ConsoleSignInFailureCount** is in an alarm state.
 - a. For **Select an SNS topic**, choose **Create new**.
 - b. Enter **ConsoleSignInFailures_CloudWatch_Alarms_Topic** as the name for the new Amazon SNS topic.
 - c. In **Email endpoints that will receive the notification**, enter email addresses of users whom you want to receive notifications if this alarm is raised. Separate email addresses with commas.

Email recipients specified here receive email asking them to confirm that they want to be subscribed to the Amazon SNS topic.
 - d. Choose **Create topic**.
4. For this example, skip the other action types. Choose **Next**.
5. On the **Add name and description** page, enter a friendly name for the alarm, and a description. For this example, enter **Console sign-in failures** for the name, and **Raises alarms if more than 3 console sign-in failures occur in 5 minutes** for the description. Choose **Next**.
6. On the **Preview and create** page, review your choices. Choose **Edit** to make changes, or choose **Create alarm** to create the alarm.

After you create the alarm, CloudWatch opens the **Alarms** page. The alarm's **Actions** column shows **Pending confirmation** until all email recipients on the SNS topic have confirmed that they want to subscribe to SNS notifications.

Example: IAM policy changes

Follow this procedure 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, choose **Logs**.

3. In the list of log groups, choose the log group that you created for CloudTrail log events.
4. Choose **Actions**, and then choose **Create metric filter**.
5. On the **Define pattern** page, in **Create filter pattern**, enter the following for **Filter pattern**.

```
{ ($.eventName=DeleteGroupPolicy)||($.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)}
```

6. In **Test pattern**, leave defaults. Choose **Next**.
7. On the **Assign metric** page, for **Filter name**, enter **IAMPolicyChanges**.
8. In **Metric details**, turn on **Create new**, and then enter **CloudTrailMetrics** for **Metric namespace**.
9. For **Metric name**, type **IAMPolicyEventCount**.
10. For **Metric value**, type **1**.
11. Leave **Default value** blank.
12. Choose **Next**.
13. On the **Review and create** page, review your choices. Choose **Create metric filter** to create the filter, or choose **Edit** to go back and change values.

Create an alarm

After you create the metric filter, the CloudWatch Logs log group details page for your CloudTrail trail log group opens. Follow this procedure to create an alarm.

1. On the **Metric filters** tab, find the metric filter you created in [the section called “Create a metric filter” \(p. 179\)](#). Fill the check box for the metric filter. In the **Metric filters** bar, choose **Create alarm**.
2. On the **Create Alarm** page, in **Specify metric and conditions**, enter the following.
 - a. For **Graph**, the line is set at **1** based on other settings you make when you create your alarm.
 - b. For **Metric name**, keep the current metric name, **IAMPolicyEventCount**.
 - c. For **Statistic**, keep the default, **Sum**.
 - d. For **Period**, keep the default, **5 minutes**.
 - e. In **Conditions**, for **Threshold type**, choose **Static**.
 - f. For **Whenever *metric_name* is**, choose **Greater/Equal**.
 - g. For **Threshold**, enter **1**.
 - h. In **Additional configuration**, leave defaults. Choose **Next**.
 - i.
3. On the **Configure actions** page, choose **In alarm**, which indicates that the action is taken when the threshold of 1 change event in 5 minutes is crossed, and **IAMPolicyEventCount** is in an alarm state.
 - a. For **Select an SNS topic**, choose **Create new**.
 - b. Enter **IAM_Policy_Changes_CloudWatch_Alarms_Topic** as the name for the new Amazon SNS topic.
 - c. In **Email endpoints that will receive the notification**, enter email addresses of users whom you want to receive notifications if this alarm is raised. Separate email addresses with commas.

Email recipients specified here receive email asking them to confirm that they want to be subscribed to the Amazon SNS topic.

- d. Choose **Create topic**.
4. For this example, skip the other action types. Choose **Next**.
5. On the **Add name and description** page, enter a friendly name for the alarm, and a description. For this example, enter **IAM Policy Changes** for the name, and **Raises alarms if IAM policy changes occur** for the description. Choose **Next**.
6. On the **Preview and create** page, review your choices. Choose **Edit** to make changes, or choose **Create alarm** to create the alarm.

After you create the alarm, CloudWatch opens the **Alarms** page. The alarm's **Actions** column shows **Pending confirmation** until all email recipients on the SNS topic have confirmed that they want to subscribe to SNS notifications.

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 AWS CloudTrail events to Amazon CloudWatch Logs by updating a trail to disable CloudWatch Logs settings.

Stop sending events to CloudWatch Logs (console)

To stop sending CloudTrail events to CloudWatch Logs

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**.
3. Choose the name of the trail for which you want to disable CloudWatch Logs integration.
4. In **CloudWatch Logs**, choose **Edit**.
5. On the **Update trail** page, in **CloudWatch Logs**, clear the **Enabled** check box.
6. Choose **Update trail** to save your changes.

Stop sending events to CloudWatch Logs (CLI)

You can remove the CloudWatch Logs log group as a delivery endpoint by running the [update-trail](#) (p. 90) command. The following command clears the log group and role from the trail configuration by replacing the values for the log group ARN and CloudWatch Logs role ARN with empty values.

```
aws cloudtrail update-trail --name trail_name --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**.

Follow these guidelines when naming a log group:

- Log group names must be unique within a region for an AWS account.
- Log group names can be between 1 and 512 characters long.
- Log group names consist of the following characters: a-z, A-Z, 0-9, '_' (underscore), '-' (hyphen), '/' (forward slash), '.' (period), and '#' (number sign).

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.

For more information about CloudWatch log groups, see [Working with log groups and log streams](#) in the *Amazon CloudWatch Logs User Guide* and [CreateLogGroup](#) in the *Amazon CloudWatch Logs API Reference*.

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 events to CloudWatch Logs](#) (p. 160). 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 example 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 in the US East (Ohio) region. (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-2: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-2:accountID:log-group:log_group_name:log-
stream:CloudTrail_log_stream_name_prefix*"
    ]
}
]
```

If you're creating a policy that might be used for organization trails as well, you will need to modify it from the default policy created for the role. For example, the following policy grants CloudTrail the permissions required to create a CloudWatch Logs log stream in the log group you specify as the value of `log_group_name`, and to deliver CloudTrail events to that log stream for both trails in the AWS account 111111111111 and for organization trails created in the 111111111111 account that are applied to the AWS Organizations organization with the ID of `o-exampleorgid`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailCreateLogStream20141101",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
        "arn:aws:logs:us-east-2:111111111111:log-
group:CloudTrail/log_group_name:log-stream:111111111111_CloudTrail_us-east-2*",
        "arn:aws:logs:us-east-2:111111111111:log-
group:CloudTrail/log_group_name:log-stream:o-exampleorgid_*"
      ]
    },
    {
      "Sid": "AWSCloudTrailPutLogEvents20141101",
      "Effect": "Allow",
      "Action": [
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:us-east-2:111111111111:log-
group:CloudTrail/log_group_name:log-stream:111111111111_CloudTrail_us-east-2*",
        "arn:aws:logs:us-east-2:111111111111:log-
group:CloudTrail/log_group_name:log-stream:o-exampleorgid_*"
      ]
    }
  ]
}
```

For more information about organization trails, see [Creating a trail for an organization \(p. 112\)](#).

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 \(p. 70\)](#).

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 \(p. 184\)](#).

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 \(p. 185\)](#).

Topics

- [Setting bucket policy for multiple accounts \(p. 184\)](#)
- [Turning on CloudTrail in additional accounts \(p. 185\)](#)

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.

Note

For security reasons, unauthorized users cannot create a trail that includes `AWSLogs/` as the `S3KeyPrefix` parameter.

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": "AWSCloudTrailAc1Check20131101",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::example-bucket"
    }
  ]
}
```

```
    "Resource": "arn:aws:s3::myBucketName"
  },
  {
    "Sid": "AWSCloudTrailWrite20131101",
    "Effect": "Allow",
    "Principal": {
      "Service": "cloudtrail.amazonaws.com"
    },
    "Action": "s3:PutObject",
    "Resource": [
      "arn:aws:s3::myBucketName/[optional] myLogFilePrefix/AWSLogs/1111111111/*",
      "arn:aws:s3::myBucketName/[optional] myLogFilePrefix/AWSLogs/2222222222/*"
    ],
    "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 [Amazon SNS topic policy for CloudTrail](#) (p. 246).

8. Choose **Turn On**.

CloudTrail starts publishing log files that show the AWS calls made in your accounts in this region. CloudTrail typically delivers logs within an average of about 15 minutes of an API call. This time is not guaranteed. Review the [AWS CloudTrail Service Level Agreement](#) for more information.

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-trail` command, specifying the following:

- `--name` specifies the name of the trail.
- `--s3-bucket-name` specifies the existing Amazon S3 bucket you 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).
- `--is-multi-region-trail` specifies that this trail will log events in all AWS Regions.

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-bucket-name` option. Amazon S3 bucket names are globally unique.

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

When you run the command, you will see output similar to the following:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "AWSCloudTrailExample",
  "TrailARN": "arn:aws:cloudtrail:us-east-2:222222222222:trail/my-trail",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
  "IsOrganizationTrail": false,
  "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.

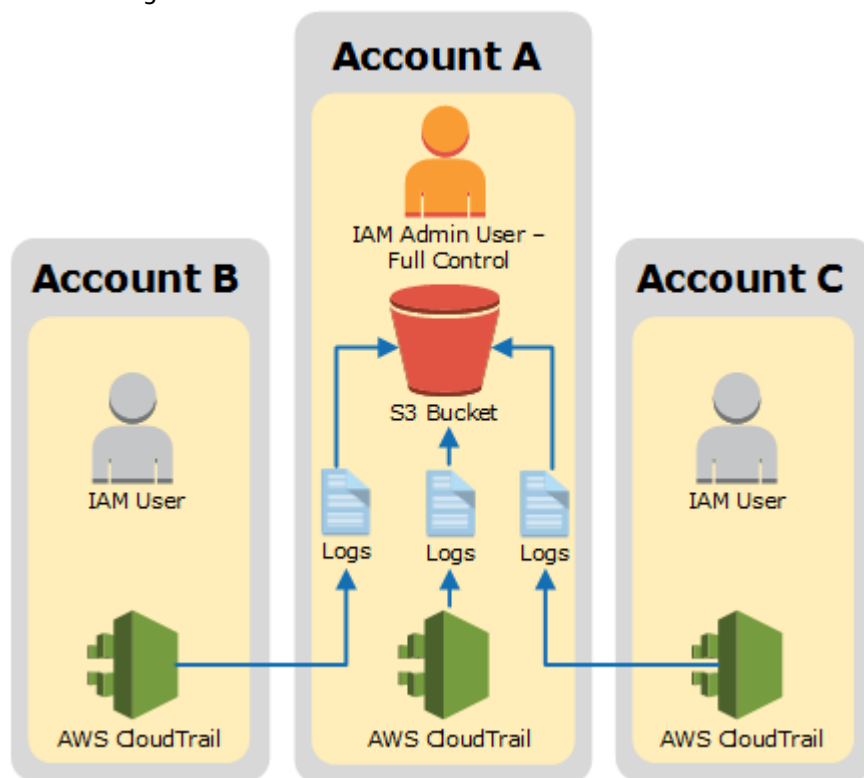
- Create an IAM role for each account that you want to share log files with.
- For each of these IAM roles, create an access policy that grants read-only access to the account you want to share the log files with.
- Have an IAM user in each account programmatically assume the appropriate role and retrieve 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*.

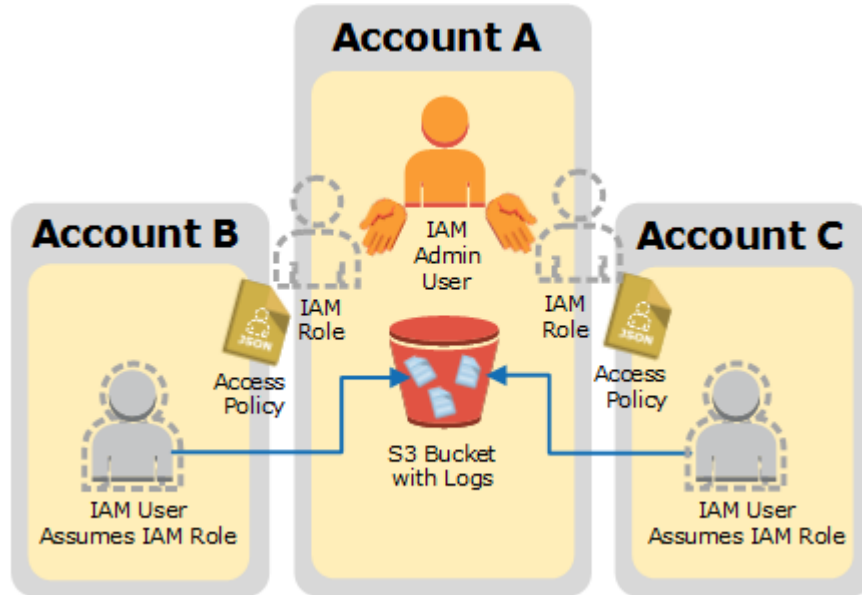
Scenario 1: Granting access to the account that generated the log files

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 \(p. 189\)](#)
- 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 \(p. 191\)](#)
- 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 \(p. 194\)](#).
- 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 \(p. 194\)](#).

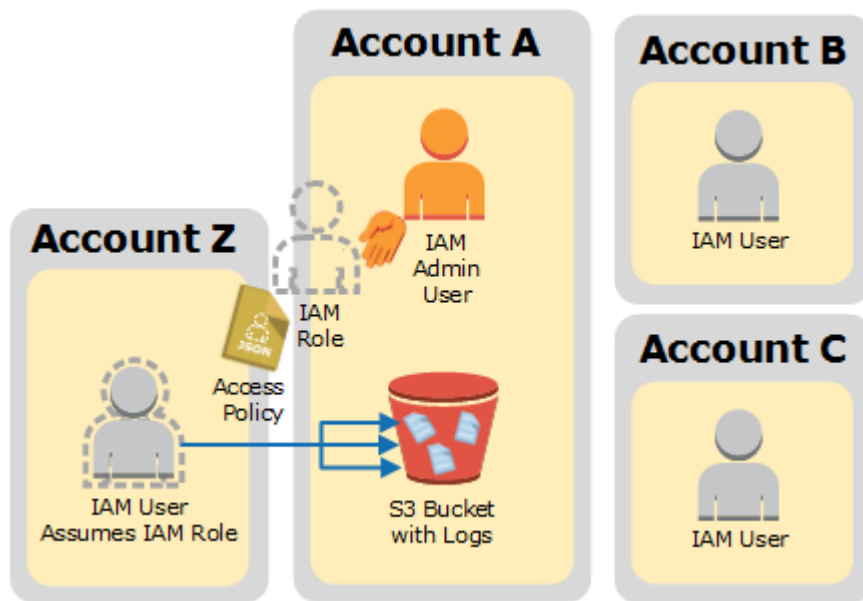
The IAM users in accounts B and C can then programmatically retrieve their own log files, but not the log files of any other account.

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 \(p. 189\)](#)
- 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 \(p. 192\)](#)
- 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 \(p. 194\)](#). 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 \(p. 194\)](#).

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 [IAM Roles](#) in the *IAM User Guide*. For more information about how to assume a role, see [Assuming a role \(p. 193\)](#).

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 \(p. 191\)](#). 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 \(p. 192\)](#).

For further details about creating and working with IAM policies, see [Access Management](#) in the *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, choose **Roles**.
4. Choose **Create New Role**.
5. Type a name for the new role, and then choose **Next Step**.
6. Choose **Role for Cross-Account Access**.
7. For Scenario 1, do the following to provide access between accounts you own:
 - a. Choose **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. Choose **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 [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party](#) in the *IAM User Guide*.
8. Choose **Next Step** to attach a policy that sets the permissions for this role.
 9. Under **Attach Policy**, choose 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 \(p. 191\)](#).
- 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 \(p. 192\)](#).

10 Choose **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 Choose **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 \(p. 185\)](#).

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 \(p. 193\)](#).

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. When you create the role, AWS automatically creates the trust relationship, which 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 \(p. 189\)](#).

For example, the trust relationship created by AWS specifies that Account Z is trusted to assume the role created by Account A. The following is an example trust policy:

```
{
  "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 following example access policy has a `Condition` element.

For more information, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party](#) in the *IAM User Guide*.

```
{
  "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 example. The wild card (*) at the end of the Resource value indicates that Account Z can access any log file in the 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 bucket. For more information, see [Assuming a role](#) (p. 193).

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. 191).
- 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. 192).

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. Choose the user whose permissions you want to modify.
3. Choose the **Permissions** tab.
4. Choose **Custom Policy**.
5. Choose **Use the policy editor to customize your own set of permissions**.
6. Type a name for the policy.
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 [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party](#) in the *IAM User Guide*. You can retrieve the ARN from the Account A by opening the IAM console.

To find the ARN Value in Account A with the IAM console

1. Choose **Roles**
2. Choose the role you want to examine.
3. Look for the **Role ARN** in the **Summary** section.

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.

```
def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the account.
    Uses the temporary credentials from the role to list the buckets that are owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                           grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        'sts', aws_access_key_id=user_key.id, aws_secret_access_key=user_key.secret)
    response = sts_client.assume_role(
        RoleArn=assume_role_arn, RoleSessionName=session_name)
    temp_credentials = response['Credentials']
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    # Create an S3 resource that can access the account with the temporary credentials.
    s3_resource = boto3.resource(
        's3',
        aws_access_key_id=temp_credentials['AccessKeyId'],
        aws_secret_access_key=temp_credentials['SecretAccessKey'],
        aws_session_token=temp_credentials['SessionToken'])

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        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. 189).

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.

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 \(p. 202\)](#).

Storing log and digest files

You can store the CloudTrail log files and digest files in Amazon S3 or S3 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 \(p. 197\)](#).

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 \(p. 197\)](#). For information on developing custom implementations of CloudTrail log file validation, see [Custom implementations of CloudTrail log file integrity validation \(p. 207\)](#).

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 console \(p. 69\)](#).

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 \(p. 202\)](#).

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 \(p. 202\)](#).

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 \(p. 207\)](#).

validate-logs

Syntax

The following is the syntax for `validate-logs`. Optional parameters are shown in brackets.

```
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 \(p. 202\)](#).

`--s3-bucket`

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.

`--trail-arn`

Specifies the Amazon Resource Name (ARN) of the trail to be validated. The format of a trail ARN follows.

```
arn:aws:cloudtrail:us-east-2:111111111111:trail/MyTrailName
```

Note

To obtain the trail ARN for a trail, you can use the `describe-trails` command before running `validate-logs`.

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-east-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. |
| Digest file | INVALID: public key not found for fingerprint <i>fingerprint</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>fingerprint</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-east-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-east-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-
east-2/2015/09/01/111111111111_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T201728Z.json.gz valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1925Z_WZWz1RymnjCRjxXc.json.gz
valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1915Z_POuvV87nu6pfAV2W.json.gz
valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1930Z_l2QgXhAKVm1QXiIA.json.gz
valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1920Z_eQJteBBrfpBCqOqw.json.gz
valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1950Z_9g5A6qlR2B5KaRdq.json.gz
valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1920Z_i4DNCC12BuXd6Ru7.json.gz
valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1915Z_Sg5caf2RH6Jdx0EJ.json.gz
valid
Digest file      s3://example-bucket/AWSLogs/111111111111/CloudTrail-Digest/us-
east-2/2015/09/01/111111111111_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T191728Z.json.gz valid
Log file         s3://example-bucket/AWSLogs/111111111111/CloudTrail/us-
east-2/2015/09/01/111111111111_CloudTrail_us-east-2_20150901T1910Z_YYSFiuFQk4nrtnEW.json.gz
valid
[...]
Log file         s3://example-bucket/AWSLogs/144218288521/CloudTrail/us-
east-2/2015/09/01/144218288521_CloudTrail_us-east-2_20150901T1055Z_0Sfy6m9f6iBzmoPF.json.gz
valid
Log file         s3://example-bucket/AWSLogs/144218288521/CloudTrail/us-
east-2/2015/09/01/144218288521_CloudTrail_us-east-2_20150901T1040Z_lLa3QzVLpOed7igR.json.gz
valid

Digest file      s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T101728Z.json.gz INVALID: signature verification failed

Digest file      s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T091728Z.json.gz valid
Log file         s3://example-bucket/AWSLogs/144218288521/CloudTrail/us-
east-2/2015/09/01/144218288521_CloudTrail_us-east-2_20150901T0830Z_eaFvO3dwHo4NCqqc.json.gz
valid
Digest file      s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T081728Z.json.gz valid
Digest file      s3://example-bucket/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T071728Z.json.gz valid
[...]
```

```
Log file      s3://example-bucket/AWSLogs/1111111111/CloudTrail/us-
east-2/2015/08/31/1111111111_CloudTrail_us-east-2_20150831T2245Z_mbJkEO5kNcDnVhGh.json.gz
valid
Log file      s3://example-bucket/AWSLogs/1111111111/CloudTrail/us-
east-2/2015/08/31/1111111111_CloudTrail_us-east-2_20150831T2225Z_IQ6kXy8sKU03RSPR.json.gz
valid
Log file      s3://example-bucket/AWSLogs/1111111111/CloudTrail/us-
east-2/2015/08/31/1111111111_CloudTrail_us-east-2_20150831T2230Z_eRPVRTxHQ5498ROA.json.gz
valid
Log file      s3://example-bucket/AWSLogs/1111111111/CloudTrail/us-
east-2/2015/08/31/1111111111_CloudTrail_us-east-2_20150831T2255Z_I1WawYZGvTWB5vYN.json.gz
valid
Digest file   s3://example-bucket/AWSLogs/1111111111/CloudTrail-Digest/us-
east-2/2015/08/31/1111111111_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-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-east-2:1111111111: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-east-2:1111111111: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-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-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/optional-prefix/AWSLogs/aws-account-id/CloudTrail-Digest/  
region/digest-end-year/digest-end-month/digest-end-date/  
aws-account-id_CloudTrail-Digest_region_trail-name_region_digest_end_timestamp.json.gz
```

Note

For organization trails, the bucket location also includes the organization unit ID, as follows:

```
s3://s3-bucket-name/optional-prefix/AWSLogs/OU-ID/aws-account-id/CloudTrail-Digest/  
region/digest-end-year/digest-end-month/digest-end-date/  
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-2/2015/08/17/111122223333_CloudTrail-Digest_us-east-2_your-trail-name_us-  
east-2_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-2/2015/08/17/111122223333_CloudTrail-Digest_us-east-2_your-trail-name_us-  
east-2_20150817T140131Z.json.gz",  
  "previousDigestHashValue":  
  "97fb791cf91ffc440d274f8190dbdd9aa09c34432aba82739df18b6d3c13df2d",  
  "previousDigestHashAlgorithm": "SHA-256",  
  "previousDigestSignature":  
  "50887ccffad4c002b97caa37cc9dc626e3c680207d41d27fa5835458e066e0d3652fc4dfc30937e4d5f4cc7f796e7a258fb50",  
  "logFiles": [  
    {  
      "s3Bucket": "S3-bucket-name",  
      "s3Object": "AWSLogs/111122223333/CloudTrail/us-  
east-2/2015/08/17/111122223333_CloudTrail_us-  
east-2_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-2/2015/08/20/111122223333_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150820T180131Z.json.gz",
  "digestPublicKeyFingerprint": "31e8b5433410dfb61a9dc45cc65b22ff",
  "digestSignatureAlgorithm": "SHA256withRSA",
  "newestEventTime": null,
  "oldestEventTime": null,
  "previousDigestS3Bucket": "example-bucket-name",
  "previousDigestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-
east-2/2015/08/20/111122223333_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150820T170131Z.json.gz",
  "previousDigestHashValue":
"ed96c4bac9eaa8fe9716ca0e515da51938be651b1db31d781956416a9d05cdfa",
  "previousDigestHashAlgorithm": "SHA-256",
  "previousDigestSignature":
"82705525fb0fe7f919f9434e5b7138cb41793c776c7414f3520c0242902daa8cc8286b29263d2627f2f259471c745b1654af7",
  "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:

```
3be472336fa2989ef34de1b3c1bf851f59eb030eaff3e2fb6600a082a23f4c6a82966565b994f9de4a5989d053d9d15d20fc5
28f1cc237f372264a51b611c01da429565def703539f4e71009051769469231bc22232fa260df02740047af532229885ea2b0
05d3ffcb5d2dd5dc28f8bb5b7993938e8a5f912a82b448a367eccb2ec0f198ba71e23eb0b97278cf65f3c8d1e652c6de33a22
```

- 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 \(p. 202\)](#).

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 sign 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

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": "MIIBCgKCAQEAAn1L2YZ9h7onug2ILi1MwyHiMRsTQjfwE
+pHVRlk1QjfWhirG+lpOa8NrwQ/r7Ah5bNL6HepznOU9XTDSfmmnP97mqyc7z/upfZdS/AHhYcGaz7n6Wc/
RRBU6VmiPCrAUojuSk6/GjvA8iOPFsYDuBtviXarvuLPlrT9kAd4Lb+rFfR5peEgBEkhlzc5HuWO7S0y
+KunqxX6jQBnXGmtxmPBPP0FylgWGNdFtkS/4YSKcgqW0YDcawP9GGDAeCIqPWIXDLG1jOjRRzWfCmD0iJUkz8vTsn4hq/5ZxRFE7
      "Fingerprint": "8eba5db5bea9b640d1c96a77256fe7f2"
    },
    {
      "ValidityStartTime": "1434589460.0",
      "ValidityEndTime": "1437181460.0",
      "Value": "MIIBCgKCAQEApfYL2FiZhpN74LNWVUzhr
+VheYhwhYm8w0n5Gf6i95ylW5kBAWKVEmnAQG7BvS5g9SMqFDQx52fw7NWV44IvfJ2xGXT
+wT+DgR6ZQ+6yxskQNqV5YcXj4Aa5Zz4jJfsYjDuO2MDTZNIzNvBNzaBJ+r2WIWAJ/
Xq54kyF63B6WE38vKuDE7nSd1FqQuEoNBFLPInvgggYe2Ym1Refe2z71wNcJ2kY
+q0h1BSHrSM8RWuJIw7MXwF9iQncg9jYzUlnJomozQzAG5wSRfbplcCYNY40xvGd/aAmO0m+Y
+XFMrKwtLCwseHPvj843qVno6x4BJN9bpWnoPo9sdsbGoiK3QIDAQAB",
      "Fingerprint": "8933b39ddc64d26d8e14ffbf6566fee4"
    },
    {
      "ValidityStartTime": "1434589370.0",
      "ValidityEndTime": "1437181370.0",
      "Value":
        "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqlzPJbvZJ42UdcmLfPUqXYNfOs6I8lCfao/
tOs8CmzPOEdtLWugB9xoIUz78qVHdKIqxbaG4jWHfJBioSSFBM0lt8cdVo4TnRa7oG9io5pysS6DJhBBaEXsicufsiFJR
+wrUNh8RSLxL4k6G1+BhLX20tJkZ/erT97tDGBujAelqseGg3vPZbTx9SMfOLN65PdLFudLP7GatOZ9p5jw/
rjpcIkfo9Bfc3heeBxWGKwBBOKnFAa9V57pOaosCvPKmHd9bg7jsQkI9Xp22IzGLSTFJZYVA3KiTAELDMu80iFXPHEq9hKNbt9e4UF
+1utKVEiLkR2disdCmPTKOVQIDAQAB",
      "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 |
|-----------|--|
| StartTime | Optionally specifies, in UTC, the start of the time range to look up public keys for CloudTrail digest files. If StartTime 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 EndTime 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 |
| ValidityStartTime | The starting time of validity of the public key. Type: DateTime |
| ValidityEndTime | 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-2/2015/08/12/111122223333_us-  
east-2_CloudTrail-Digest_us-east-2_20150812T040131Z.json.gz  
4ff08d7c6ecd6eb313257e839645d20363ee3784a2328a7d76b99b53cc9bcacd  
6e8540b83c3ac86a0312d971a225361d28ed0af20d70c211a2d405e32abf529a8145c2966e3bb47362383a52441545ed091fb81  
d4c7c09dd152b84e79099ce7a9ec35d2b264eb92eb6e090f1e5ec5d40ec8a0729c02ff57f9e30d5343a8591638f8b794972ce15  
98b0aee2c1c8af74ec620261529265e83a9834ebef6054979d3e9a6767dfa6fdb4ae153436c567d6ae208f988047ccfc8e5e41f
```

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 - http://
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, digestS3Object);

        // 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
            String dataToSign = String.format("%s%n%s/%s%n%s%n%s",
                digestFile.getString("digestEndTime"),
                digestFile.getString("digestS3Bucket"),
                digestFile.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("digestPublicKeyFingerprint"));

    // Transform the PKCS#1 formatted public key to x.509 format.
    RSAPublicKey rsaPublicKey = RSAPublicKey.getInstance(pkcs1PublicKeyBytes);
    AlgorithmIdentifier rsaEncryption = new
    AlgorithmIdentifier(PKCSObjectIdentifiers.rsaEncryption, null);
    SubjectPublicKeyInfo publicKeyInfo = new SubjectPublicKeyInfo(rsaEncryption,
    rsaPublicKey);

    // Create the PublicKey object needed for the signature validation
    PublicKey publicKey = KeyFactory.getInstance("RSA", "BC").generatePublic(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++) {

            JSONObject logFileMetadata =
            digestFile.getJSONArray("logFiles").getJSONObject(i);

            // Compute log file hash
            byte[] logFileContent = loadUncompressedLogFileInMemory(
                logFileMetadata.getString("s3Bucket"),
                logFileMetadata.getString("s3Object")
            );
            messageDigest.update(logFileContent);
            byte[] logFileHash = messageDigest.digest();
            messageDigest.reset();

            // Retrieve expected hash for the log file being processed
            byte[] expectedHash =
            Hex.decodeHex(logFileMetadata.getString("hashValue"));

            boolean signaturesMatch = Arrays.equals(expectedHash, logFileHash);
            if (!signaturesMatch) {
                System.err.println(String.format("Log file: %s/%s hash doesn't
match.\tExpected: %s Actual: %s",
                    logFileMetadata.getString("s3Bucket"),
                    logFileMetadata.getString("s3Object"),
                    Hex.encodeHexString(expectedHash),
                    Hex.encodeHexString(logFileHash)));
            } else {
                System.out.println(String.format("Log file: %s/%s hash match",
                    logFileMetadata.getString("s3Bucket"),
                    logFileMetadata.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. You provide configuration details about your CloudTrail SQS queue and write code to process events. The CloudTrail Processing Library does the rest. It polls your Amazon SQS queue, reads and parses queue messages, downloads CloudTrail log files, parses events in the log files, and passes the events to your code as Java objects.

The CloudTrail Processing Library is highly scalable and fault-tolerant. It handles parallel processing of log files so that you can process as many logs as needed. It handles network failures related to network timeouts and inaccessible resources.

The following topic shows you 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 \(p. 215\)](#)
- [Processing CloudTrail logs \(p. 215\)](#)
- [Advanced topics \(p. 219\)](#)
- [Additional resources \(p. 223\)](#)

Minimum requirements

To use the CloudTrail Processing Library, you must have the following:

- [AWS SDK for Java 1.11.830](#)
- [Java 1.8 \(Java SE 8\)](#)

Processing CloudTrail logs

To process CloudTrail logs in your Java application:

1. [Adding the CloudTrail Processing Library to your project \(p. 216\)](#)
2. [Configuring the CloudTrail Processing Library \(p. 217\)](#)
3. [Implementing the events processor \(p. 218\)](#)

4. [Instantiating and running the processing executor \(p. 219\)](#)

Adding the CloudTrail Processing Library to your project

To use the CloudTrail Processing Library, add it to your Java project's classpath.

Contents

- [Adding the library to an Apache Ant project \(p. 216\)](#)
- [Adding the library to an Apache Maven project \(p. 216\)](#)
- [Adding the library to an Eclipse project \(p. 216\)](#)
- [Adding the library to an IntelliJ project \(p. 217\)](#)

Adding the library to an Apache Ant project

To add the CloudTrail Processing Library to an Apache Ant project

1. Download or clone the CloudTrail Processing Library source code from GitHub:
 - <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 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-1.4.0.jar"/>
</classpath>
```

Adding the library to an Apache Maven project

The CloudTrail Processing Library is available for [Apache Maven](#). You can add it to your project by writing a single dependency in your project's pom.xml file.

To add the CloudTrail Processing Library to a Maven project

- 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>1.4.0</version>
</dependency>
```

Adding the 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:

- <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-1.4.0.jar to a directory in your project (typically lib).
4. Right-click your project's name in the Eclipse **Project Explorer**, choose **Build Path**, and then choose **Configure**
5. In the **Java Build Path** window, choose the **Libraries** tab.
6. Choose **Add JARs...** and navigate to the path where you copied aws-cloudtrail-processing-library-1.4.0.jar.
7. Choose **OK** to complete adding the .jar to your project.

Adding the library to an IntelliJ project

To add the CloudTrail Processing Library to an IntelliJ project

1. Download or clone the CloudTrail Processing Library source code from GitHub:
 - <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. From **File**, choose **Project Structure**.
4. Choose **Modules** and then choose **Dependencies**.
5. Choose **+ JARS or Directories** and then go to the path where you built the aws-cloudtrail-processing-library-1.4.0.jar.
6. Choose **Apply** and then choose **OK** to complete adding the .jar to your project.

Configuring 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. The following example file shows 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

# Whether to delete SQS message when the CloudTrail Processing Library is unable to process
# the notification.
deleteMessageUponFailure = false
```

The following parameters are required:

- `sqsUrl` – Provides the URL from which to pull your CloudTrail notifications. If you don't specify this value, the `AWSCloudTrailProcessingExecutor` throws an `IllegalStateException`.
- `accessKey` – A unique identifier for your account, such as `AKIAIOSFODNN7EXAMPLE`.
- `secretKey` – A unique identifier for your account, such as `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`.

The `accessKey` and `secretKey` parameters provide your AWS credentials to the library so the library can access AWS on your behalf.

Defaults for the other parameters are set by the library. For more information, 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, as shown in the following 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);
```

Implementing the events processor

To process CloudTrail logs, you must implement an `EventsProcessor` that receives the CloudTrail log data. The following 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 an `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 prints the event information for each event passed to `SampleEventsProcessor`. In your own implementation, you can process logs as you see fit. The `AWSCloudTrailProcessingExecutor` continues to send events to your `EventsProcessor` as long as it has events to send and is still running.

Instantiating and running the processing executor

After you write an `EventsProcessor` and 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 an `AWSCloudTrailProcessingExecutor`.

To use `AWSCloudTrailProcessingExecutor` to process `CloudTrail` events

1. Instantiate an `AWSCloudTrailProcessingExecutor.Builder` object. `Builder`'s constructor takes an `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 \(p. 220\)](#)
- [Processing data events \(p. 221\)](#)

- [Reporting progress \(p. 222\)](#)
- [Handling errors \(p. 222\)](#)

Filtering the events to process

By default, all logs in your Amazon SQS queue's S3 bucket and all events that they contain are 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 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 CloudTrail Processing Library calls the `filterSource()` method after the library polls for logs on the Amazon SQS queue. This occurs before the library starts event filtering or processing for the logs.

The following 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` is 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 is sent to your `EventsProcessor`. `EventFilter` declares a single callback method, `filterEvent()`, that receives a `CloudTrailEvent` object. To keep the event from being processed, return `false` from `filterEvent()`.

The CloudTrail Processing Library calls the `filterEvent()` method after the library polls for logs on the Amazon SQS queue and after source filtering. This occurs before the library starts event processing for the logs.

See the following 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` is used, which allows all events to be processed (it always returns true).

Processing data events

When CloudTrail processes data events, it preserves numbers in their original format, whether that is an integer (`int`) or a `float` (a number that contains a decimal). In events that have integers in the fields of a data event, CloudTrail historically processed these numbers as floats. Currently, CloudTrail processes numbers in these fields by keeping their original format.

As a best practice, to avoid breaking your automations, be flexible in any code or automation that you are using to process or filter CloudTrail data events, and allow both `int` and `float` formatted numbers. For best results, use version 1.4.0 or higher of the CloudTrail Processing Library.

The following example snippet shows a `float` formatted number, `2.0`, for the `desiredCount` parameter in the `ResponseParameters` block of a data event.

```
"eventName": "CreateService",
"awsRegion": "us-east-1",
"sourceIPAddress": "000.00.00.00",
"userAgent": "console.amazonaws.com",
"requestParameters": {
    "clientToken": "EXAMPLE",
    "cluster": "default",
    "desiredCount": 2.0
...
}
```

The following example snippet shows an `int` formatted number, `2`, for the `desiredCount` parameter in the `ResponseParameters` block of a data event.

```
"eventName": "CreateService",
"awsRegion": "us-east-1",
"sourceIPAddress": "000.00.00.00",
"userAgent": "console.amazonaws.com",
"requestParameters": {
    "clientToken": "EXAMPLE",
    "cluster": "default",
    "desiredCount": 2
...
}
```

```
        "desiredCount": 2  
    ...  
}
```

Reporting progress

Implement the `ProgressReporter` interface 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 that was performed. The `progressState` member holds a member of the `ProgressState` enumeration that identifies the current operation. This member can contain additional information in the `progressInfo` member. Additionally, any object that you return from `reportStart()` is passed to `reportEnd()`, so you can provide contextual information such as the time when the event began processing.

The following 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, is 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 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.

See the following 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 prints a standard error message, is used instead.

Note

If the `deleteMessageUponFailure` parameter is `true`, the CloudTrail Processing Library does not distinguish general exceptions from processing errors and may delete queue messages.

1. For example, you use the `SourceFilter` to filter messages by timestamp.
2. However, you don't have the required permissions to access the S3 bucket that receives the CloudTrail log files. Because you don't have the required permissions, an `AmazonServiceException` is thrown. The CloudTrail Processing Library wraps this in a `CallbackException`.
3. The `DefaultExceptionHandler` logs this as an error, but does not identify the root cause, which is that you don't have the required permissions. The CloudTrail Processing Library considers this a processing error and deletes the message, even if the message includes a valid CloudTrail log file.

If you want to filter messages with `SourceFilter`, verify that your `ExceptionHandler` can distinguish service exceptions from processing errors.

Additional resources

For more information about the CloudTrail Processing Library, see the following:

- [CloudTrail Processing Library](#) GitHub project, which includes [sample](#) code that demonstrates how to implement a CloudTrail Processing Library application.
- [CloudTrail Processing Library Java Package Documentation](#).

Security in AWS CloudTrail

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The [shared responsibility model](#) describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the [AWS compliance programs](#). To learn about the compliance programs that apply to AWS CloudTrail, see [AWS Services in Scope by Compliance Program](#).
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using CloudTrail. The following topics show you how to configure CloudTrail to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your CloudTrail resources.

Topics

- [Data protection in AWS CloudTrail \(p. 224\)](#)
- [Identity and Access Management for AWS CloudTrail \(p. 225\)](#)
- [Compliance validation for AWS CloudTrail \(p. 254\)](#)
- [Resilience in AWS CloudTrail \(p. 255\)](#)
- [Infrastructure security in AWS CloudTrail \(p. 255\)](#)
- [Security best practices in AWS CloudTrail \(p. 255\)](#)
- [Encrypting CloudTrail log files with AWS KMS-managed keys \(SSE-KMS\) \(p. 259\)](#)

Data protection in AWS CloudTrail

The AWS [shared responsibility model](#) applies to data protection in AWS CloudTrail. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the [Data Privacy FAQ](#). For information about data protection in Europe, see the [AWS Shared Responsibility Model and GDPR](#) blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.

- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with CloudTrail or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

By default, CloudTrail event log files are encrypted using Amazon S3 server-side encryption (SSE). You can also choose to encrypt your log files with an AWS Key Management Service (AWS KMS) key. You can store your log files in your bucket for as long as you want. You can also define Amazon S3 lifecycle rules to archive or delete log files automatically. If you want notifications about log file delivery and validation, you can set up Amazon SNS notifications.

The following security best practices also address data protection in CloudTrail:

- [Encrypting CloudTrail log files with AWS KMS–managed keys \(SSE-KMS\) \(p. 259\)](#)
- [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#)
- [Validating CloudTrail log file integrity \(p. 196\)](#)
- [Sharing CloudTrail log files between AWS accounts \(p. 186\)](#)

Because CloudTrail logs files are stored in a bucket or buckets in Amazon S3, you should also review the data protection information in the Amazon Simple Storage Service User Guide. For more information, see [Protecting Data in Amazon S3](#).

Identity and Access Management for AWS CloudTrail

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use CloudTrail resources. IAM is an AWS service that you can use with no additional charge.

Topics

- [Audience \(p. 226\)](#)
- [Authenticating with identities \(p. 226\)](#)
- [Managing access using policies \(p. 228\)](#)
- [How AWS CloudTrail works with IAM \(p. 229\)](#)
- [AWS CloudTrail identity-based policy examples \(p. 233\)](#)
- [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#)
- [Amazon SNS topic policy for CloudTrail \(p. 246\)](#)
- [Troubleshooting AWS CloudTrail identity and access \(p. 251\)](#)

- [Using service-linked roles for AWS CloudTrail \(p. 252\)](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in CloudTrail.

Service user – If you use the CloudTrail service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more CloudTrail features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in CloudTrail, see [Troubleshooting AWS CloudTrail identity and access \(p. 251\)](#).

Service administrator – If you're in charge of CloudTrail resources at your company, you probably have full access to CloudTrail. It's your job to determine which CloudTrail features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with CloudTrail, see [How AWS CloudTrail works with IAM \(p. 229\)](#).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to CloudTrail. To view example CloudTrail identity-based policies that you can use in IAM, see [AWS CloudTrail identity-based policy examples \(p. 233\)](#).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see [Signing in to the AWS Management Console as an IAM user or root user](#) in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the [AWS Management Console](#), use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see [Signature Version 4 signing process](#) in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see [Managing access keys for IAM users](#) in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [When to create an IAM user \(instead of a role\)](#) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by [switching roles](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Using IAM roles](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider](#). For more information about federated users, see [Federated users and roles](#) in the *IAM User Guide*.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
- **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see [Actions, Resources, and Condition Keys for AWS CloudTrail](#) in the *Service Authorization Reference*.
- **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear

in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see [When to create an IAM role \(instead of a user\)](#) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Creating IAM policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choosing between managed policies and inline policies](#) in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access control list \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [How SCPs work](#) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see [Session policies](#) in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see [Policy evaluation logic](#) in the *IAM User Guide*.

How AWS CloudTrail works with IAM

Before you use IAM to manage access to CloudTrail, you should understand what IAM features are available to use with CloudTrail. To get a high-level view of how CloudTrail and other AWS services work with IAM, see [AWS Services That Work with IAM](#) in the *IAM User Guide*.

CloudTrail works with IAM identity-based policies, but does not work with resource-based policies. For more information on the differences between identity-based policies and resource-based policies, see [Identity-Based Policies and Resource-Based Policies](#) in the *IAM User Guide*.

Topics

- [CloudTrail identity-based policies \(p. 230\)](#)

- [CloudTrail resource-based policies \(p. 232\)](#)
- [Access control lists \(p. 232\)](#)
- [Authorization based on CloudTrail tags \(p. 232\)](#)
- [CloudTrail IAM roles \(p. 232\)](#)

CloudTrail identity-based policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. CloudTrail supports specific actions and resources. There are no CloudTrail service-specific condition keys that can be used in the `Condition` element of policy statements. To learn about all of the elements that you use in a JSON policy, see [IAM JSON Policy Elements Reference](#) in the *IAM User Guide*.

Actions

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

Policy actions in CloudTrail use the following prefix before the action: `cloudtrail:`. For example, to grant someone permission to list tags for a trail with the `ListTags` API operation, you include the `cloudtrail:ListTags` action in their policy. Policy statements must include either an `Action` or `NotAction` element. CloudTrail defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows:

```
"Action": [
    "cloudtrail:AddTags",
    "cloudtrail:ListTags",
    "cloudtrail:RemoveTags"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Get`, include the following action:

```
"Action": "cloudtrail:Get*"
```

To see a list of CloudTrail actions, see [Actions Defined by AWS CloudTrail](#) in the *IAM User Guide*.

Resources

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its [Amazon Resource Name \(ARN\)](#). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

In CloudTrail, the primary resource is a trail. Each resource has a unique Amazon Resource Name (ARN) associated with it. In a policy, you use an ARN to identify the resource that the policy applies to. CloudTrail does not currently support other resource types, which are sometimes referred to as subresources.

The CloudTrail trail resource has the following ARN:

```
arn:${Partition}:cloudtrail:${Region}:${Account}:trail/{TrailName}
```

For more information about the format of ARNs, see [Amazon Resource Names \(ARNs\) and AWS Service Namespaces](#).

For example, for an AWS account with the ID `123456789012`, to specify a trail named `My-Trail` that exists in the US East (Ohio) Region in your statement, use the following ARN:

```
"Resource": "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-Trail"
```

To specify all trails that belong to a specific account in that AWS Region, use the wildcard (*):

```
"Resource": "arn:aws:cloudtrail:us-east-2:123456789012:trail/*"
```

Some CloudTrail actions, such as those for creating resources, cannot be performed on a specific resource. In those cases, you must use the wildcard (*).

```
"Resource": "*"
```

Many CloudTrail API actions involve multiple resources. For example, `CreateTrail` requires an Amazon S3 bucket for storing log files, so an IAM user must have permissions to write to the bucket. To specify multiple resources in a single statement, separate the ARNs with commas.

```
"Resource": [  
    "resource1",  
    "resource2"
```

To see a list of CloudTrail resource types and their ARNs, see [Resources Defined by AWS CloudTrail](#) in the *IAM User Guide*. To learn with which actions you can specify the ARN of each resource, see [Actions Defined by AWS CloudTrail](#).

Condition keys

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or *Condition block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use [condition operators](#), such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical AND operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical OR operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see [IAM policy elements: variables and tags](#) in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see [AWS global condition context keys](#) in the *IAM User Guide*.

CloudTrail does not define its own condition keys, but it supports using some global condition keys. To see all AWS global condition keys, see [AWS Global Condition Context Keys](#) in the *IAM User Guide*.

To see a list of condition keys supported for CloudTrail, see [Condition Keys for AWS CloudTrail](#) in the *IAM User Guide*. To learn with which actions and resources you can use a condition key, see [Actions Defined by AWS CloudTrail](#).

Examples

To view examples of CloudTrail identity-based policies, see [AWS CloudTrail identity-based policy examples \(p. 233\)](#).

CloudTrail resource-based policies

CloudTrail does not support resource-based policies.

Access control lists

Access control lists (ACLs) are lists of grantees that you can attach to resources. They grant accounts permissions to access the resource to which they are attached. While CloudTrail does not support ACLs, Amazon S3 does. For example, you can attach ACLs to an Amazon S3 bucket resource where you store log files for one or more trails. For more information about attaching ACLs to buckets, see [Managing Access with ACLs](#) in the Amazon Simple Storage Service User Guide.

Authorization based on CloudTrail tags

Although you can attach tags to CloudTrail resources, CloudTrail does not support controlling access based on tags.

You can attach tags to CloudTrail resources or pass tags in a request to CloudTrail. For more information about tagging CloudTrail resources, see [Creating a trail \(p. 70\)](#) and [Creating, updating, and managing trails with the AWS Command Line Interface \(p. 87\)](#).

CloudTrail IAM roles

An [IAM role](#) is an entity within your AWS account that has specific permissions.

Using temporary credentials with CloudTrail

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as [AssumeRole](#) or [GetFederationToken](#).

CloudTrail supports using temporary credentials.

Service-linked roles

[Service-linked roles](#) allow AWS services to access resources in other services to complete an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view but not edit the permissions for service-linked roles.

CloudTrail supports a service-linked role for integration with AWS Organizations. This role is required for the creation of an organization trail, a trail that logs events for all AWS accounts in an organization. For details about creating or managing CloudTrail service-linked roles, see [the section called “Using service-linked roles”](#) (p. 252).

Service roles

This feature allows a service to assume a [service role](#) on your behalf. This role allows the service to access resources in other services to complete an action on your behalf. Service roles appear in your IAM account and are owned by the account. This means that an IAM administrator can change the permissions for this role. However, doing so might break the functionality of the service.

CloudTrail supports service roles.

AWS CloudTrail identity-based policy examples

By default, IAM users and roles don't have permission to create or modify CloudTrail resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users, groups, or roles that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see [Creating Policies on the JSON Tab](#) in the *IAM User Guide*.

Topics

- [Policy best practices](#) (p. 233)
- [Example: Allowing and denying actions for a specified trail](#) (p. 234)
- [Examples: Creating and applying policies for actions on specific trails](#) (p. 234)
- [Granting permissions for using the CloudTrail console](#) (p. 236)
- [Allow users to view their own permissions](#) (p. 238)
- [Granting custom permissions for CloudTrail users](#) (p. 238)

Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete CloudTrail resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using CloudTrail quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see [Get started using permissions with AWS managed policies](#) in the *IAM User Guide*.
- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see [Grant least privilege](#) in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see [IAM JSON policy elements: Condition](#) in the *IAM User Guide*.

CloudTrail does not have service-specific context keys that can be used in the `Condition` element of policy statements.

Example: Allowing and denying actions for a specified trail

The following example demonstrates a policy that allows users with this policy to view the status and configuration of a trail and start and stop logging for a trail named *My-First-Trail*. This trail was created in the US East (Ohio) Region (its home region) in the AWS account with the ID *123456789012*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudtrail:StartLogging",
        "cloudtrail:StopLogging",
        "cloudtrail:GetTrail",
        "cloudtrail:GetTrailStatus",
        "cloudtrail:GetEventSelectors"
      ],
      "Resource": [
        "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-First-Trail"
      ]
    }
  ]
}
```

The following example demonstrates a policy that explicitly denies CloudTrail actions for any trail not named *My-First-Trail*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "cloudtrail:*"
      ],
      "NotResource": [
        "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-First-Trail"
      ]
    }
  ]
}
```

Examples: Creating and applying policies for actions on specific trails

You can use permissions and policies 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 manage.

You can create two policy statements and then attach them to the developer group you create in IAM. For more information about groups in IAM, see [IAM Groups](#) in the *IAM User Guide*.

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-2:123456789012:trail/Example-Trail`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1446057698000",
      "Effect": "Deny",
      "Action": [
        "cloudtrail:StartLogging",
        "cloudtrail:StopLogging"
      ],
      "Resource": [
        "arn:aws:cloudtrail:us-east-2:123456789012:trail/Example-Trail"
      ]
    }
  ]
}
```

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:GetTrail",
        "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-2:123456789012:trail/Example-Trail",
      "IsMultiRegionTrail": false,

```



```
        "S3BucketName": "myS3bucket ",  
        "HomeRegion": "us-east-2"  
    }  
]  
}
```

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 Example-Trail
```

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 Example-Trail
```

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 Example-Trail
```

The command returns an access denied exception:

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

Granting permissions for using the CloudTrail console

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 [Amazon S3 bucket policy for CloudTrail \(p. 243\)](#).

If you configure integration with Amazon CloudWatch Logs, CloudTrail also requires a role that it can assume to deliver events to an Amazon CloudWatch Logs log group. This will require additional permissions to create the role, as well as the role itself. For more information, see [Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console \(p. 242\)](#) and [Sending events to CloudWatch Logs \(p. 160\)](#).

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 select one of the following policies for CloudTrail:
 - **AWSCloudTrail_FullAccess**. This policy gives users in the group full access to CloudTrail actions. These users have permissions to manage (but not delete) the Amazon S3 bucket, the log group for CloudWatch Logs, and an Amazon SNS topic for a trail.

Note

The **AWSCloudTrail_FullAccess** policy is not intended to be shared broadly across your AWS account. Users with this role have the ability to disable or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, this policy should be applied only to account administrators, and use of this policy should be closely controlled and monitored.

- **AWSCloudTrailReadOnlyAccess**. This policy lets users in the group view the CloudTrail console, including recent events and event history. These users can also view existing trails and their buckets. Users can download a file of event history, but they cannot create or update trails.

Note

You can also create a custom policy that grants permissions to individual actions. For more information, see [Granting custom permissions for CloudTrail users \(p. 238\)](#).

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 their 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*.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

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. For specific examples, see [Example: Allowing and denying actions for a specified trail \(p. 234\)](#) and [Examples: Creating and applying policies for actions on specific trails \(p. 234\)](#).

Contents

- [Read-only access \(p. 239\)](#)
- [Full access \(p. 240\)](#)
- [Granting permission to view AWS Config information on the CloudTrail console \(p. 242\)](#)
- [Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console \(p. 242\)](#)
- [Additional information \(p. 242\)](#)

Read-only access

The following example shows a policy that grants read-only access to CloudTrail trails. This is equivalent to the managed policy **AWSCloudTrailReadOnlyAccess**. 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:GetTrail",
        "cloudtrail:GetTrailStatus",
        "cloudtrail:LookupEvents",
        "cloudtrail:ListPublicKeys",
        "cloudtrail:ListTags",
        "s3:ListAllMyBuckets",
        "kms:ListAliases",
        "lambda:ListFunctions"
      ],
      "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 usually 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. This is equivalent to the managed policy **AWSCloudTrail_FullAccess**. It grants users the permission to perform all CloudTrail actions. It also lets users log data events in Amazon S3 and AWS Lambda, 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.

Important

The **AWSCloudTrail_FullAccess** policy or equivalent permissions are not intended to be shared broadly across your AWS account. Users with this role or equivalent access have the ability to disable or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, this policy should be applied only to account administrators, and use of this policy should be closely controlled and monitored.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sns:AddPermission",
        "sns:CreateTopic",
        "sns:SetTopicAttributes",
        "sns:GetTopicAttributes"
      ],
      "Resource": [
        "arn:aws:sns:*:*:aws-cloudtrail-logs*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:ListTopics"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:PutBucketPolicy"
      ],
      "Resource": [
        "arn:aws:s3:::aws-cloudtrail-logs*"
      ]
    },
    {
      "Effect": "Allow",
```

```
        "Action": [
            "s3:ListAllMyBuckets",
            "s3:GetBucketLocation",
            "s3:GetBucketPolicy"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": "cloudtrail:*",
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogGroup"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:aws-cloudtrail-logs*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "iam:ListRoles",
            "iam:GetRolePolicy",
            "iam:GetUser"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "iam:PassRole"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "iam:PassedToService": "cloudtrail.amazonaws.com"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "kms:CreateKey",
            "kms:CreateAlias",
            "kms:ListKeys",
            "kms:ListAliases"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "lambda:ListFunctions"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "dynamodb:ListGlobalTables",
            "dynamodb:ListTables"
        ],
        "Resource": "*"
    }
```

```
}  
]  
}
```

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 \(p. 48\)](#).

Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console

You can view and configure delivery of events to CloudWatch Logs in the CloudTrail console if you have sufficient permissions. These are permissions that may be beyond those granted for CloudTrail administrators. Attach this policy to administrators who will configure and manage CloudTrail integration with CloudWatch Logs. The policy doesn't grant them permissions in CloudTrail or in CloudWatch Logs directly, but instead grants the permissions required to create and configure the role CloudTrail will assume to successfully deliver events to your CloudWatch Logs group.

```
{  
  "Version": "2012-10-17",  
  "Statement": [{  
    "Effect": "Allow",  
    "Action": [  
      "iam:CreateRole",  
      "iam:PutRolePolicy",  
      "iam:AttachRolePolicy",  
      "iam:ListRoles",  
      "iam:GetRolePolicy",  
      "iam:GetUser"  
    ],  
    "Resource": "*"  
  }]  
}
```

For more information, see [Monitoring CloudTrail Log Files with Amazon CloudWatch Logs \(p. 159\)](#).

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*.

Amazon S3 bucket policy for CloudTrail

By default, Amazon S3 buckets and objects are private. Only the resource owner (the AWS account that created the bucket) can access the bucket and objects it contains. The resource owner can grant access permissions to other resources and users by writing an access policy.

If you want to create or modify an Amazon S3 bucket to receive the log files for an organization trail, you must further modify the bucket policy. For more information, see [Creating a trail for an organization with the AWS Command Line Interface](#) (p. 121).

To deliver log files to an S3 bucket, CloudTrail must have the required permissions, and it cannot be configured as a [Requester Pays](#) bucket. CloudTrail automatically attaches the required permissions to a bucket when you create an Amazon S3 bucket as part of creating or updating a trail in the CloudTrail console.

CloudTrail adds the following fields in the policy for you:

- The allowed SIDs.
- The bucket name.
- The service principal name for CloudTrail.
- The name of the folder where the log files are stored, including the bucket name, a prefix (if you specified one), and your AWS account ID.

The following policy allows CloudTrail to write log files to the bucket from supported regions. For more information, see [CloudTrail supported regions](#) (p. 12).

S3 bucket policy

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

Contents

- [Specifying an existing bucket for CloudTrail log delivery](#) (p. 244)
- [Receiving log files from other accounts](#) (p. 244)
- [Create or update an Amazon S3 bucket to use to store the log files for an organization trail](#) (p. 244)
- [Troubleshooting the Amazon S3 bucket policy](#) (p. 245)
 - [Common Amazon S3 policy configuration errors](#) (p. 245)
 - [Changing a prefix for an existing bucket](#) (p. 246)

- [Additional resources \(p. 246\)](#)

Specifying an existing bucket for CloudTrail log delivery

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

Note

As a best practice, use a dedicated S3 bucket for CloudTrail logs.

To add the required CloudTrail policy to an Amazon S3 bucket

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Choose the bucket where you want CloudTrail to deliver your log files, and then choose **Properties**.
3. Choose **Permissions**.
4. Choose **Edit Bucket Policy**.
5. Copy the [S3 bucket policy \(p. 243\)](#) to the **Bucket Policy Editor** window. Replace the placeholders in italics with the names of your bucket, prefix, and account number. If you specified a prefix when you created your trail, include it here. The prefix is an optional addition to the S3 object key that creates a folder-like organization in your bucket.

Note

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

Receiving log files from other accounts

You can configure CloudTrail to deliver log files from multiple AWS accounts to a single S3 bucket. For more information, see [Receiving CloudTrail log files from multiple accounts \(p. 183\)](#).

Create or update an Amazon S3 bucket to use to store the log files for an organization trail

You must specify an Amazon S3 bucket to receive the log files for an organization trail. This bucket must have a policy that allows CloudTrail to put the log files for the organization into the bucket.

The following is an example policy for an Amazon S3 bucket named *my-organization-bucket*. This bucket is in an AWS account with the ID *111111111111*, which is the management account for an organization with the ID *o-exampleorgid* that allows logging for an organization trail. It also allows logging for the *111111111111* account in the event that the trail is changed from an organization trail to a trail for that account only.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailAclCheck20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "cloudtrail.amazonaws.com"
        ]
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::my-organization-bucket"
    }
  ]
}
```

```
{
  "Sid": "AWSCloudTrailWrite20150319",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "cloudtrail.amazonaws.com"
    ]
  },
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3::my-organization-bucket/AWSLogs/1111111111/*",
  "Condition": {
    "StringEquals": {
      "s3:x-amz-acl": "bucket-owner-full-control"
    }
  }
},
{
  "Sid": "AWSCloudTrailWrite20150319",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "cloudtrail.amazonaws.com"
    ]
  },
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3::my-organization-bucket/AWSLogs/o-exampleorgid/*",
  "Condition": {
    "StringEquals": {
      "s3:x-amz-acl": "bucket-owner-full-control"
    }
  }
}
]
```

This example policy does not allow any users from member accounts to access the log files created for the organization. By default, organization log files are accessible only to the management account. For information about how to allow read access to the Amazon S3 bucket for IAM users in member accounts, see [Sharing CloudTrail log files between AWS accounts](#) (p. 186).

Troubleshooting the Amazon S3 bucket policy

The following sections describe how to troubleshoot the S3 bucket policy.

Common Amazon S3 policy configuration errors

When you create a new bucket as part of creating or updating a trail, CloudTrail attaches the required permissions to your bucket. The bucket policy uses the service principal name, "cloudtrail.amazonaws.com", which allows CloudTrail to deliver logs for all regions.

If CloudTrail is not delivering logs for a region, it's possible that your bucket has an older policy that specifies CloudTrail account IDs for each region. This policy gives CloudTrail permission to deliver logs only for the regions specified.

As a best practice, update the policy to use a permission with the CloudTrail service principal. To do this, replace the account ID ARNs with the service principal name: "cloudtrail.amazonaws.com". This gives CloudTrail permission to deliver logs for current and new regions. The following is an example of a recommended policy configuration:

Example Example bucket policy with service principal name

```
{
```

```
"Version": "2012-10-17",
"Statement": [
  {
    "Sid": "AWSCloudTrailAclCheck20150319",
    "Effect": "Allow",
    "Principal": {"Service": "cloudtrail.amazonaws.com"},
    "Action": "s3:GetBucketAcl",
    "Resource": "arn:aws:s3:::bucket-1"
  },
  {
    "Sid": "AWSCloudTrailWrite20150319",
    "Effect": "Allow",
    "Principal": {"Service": "cloudtrail.amazonaws.com"},
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::bucket-1/my-prefix/AWSLogs/123456789012/*",
    "Condition": {"StringEquals": {"s3:x-amz-acl": "bucket-owner-full-control"}}
  }
]
```

Changing a prefix for an existing bucket

If you try to add, modify, or remove a log file prefix for an S3 bucket that receives logs from a trail, you may see the error: **There is a problem with the bucket policy**. A bucket policy with an incorrect prefix can prevent your trail from delivering logs to the bucket. To resolve this issue, use the Amazon S3 console to update the prefix in the bucket policy, and then use the CloudTrail console to specify the same prefix for the bucket in the trail.

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. Choose the bucket for which you want to modify the prefix, and then choose **Properties**.
3. Choose **Permissions**.
4. Choose **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 needed.

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

6. Choose **Save**.
7. Open the CloudTrail console at <https://console.aws.amazon.com/cloudtrail/>.
8. Choose your trail and for **Storage location**, click the pencil icon to edit the settings for your bucket.
9. For **S3 bucket**, choose the bucket with the prefix you are changing.
10. For **Log file prefix**, update the prefix to match the prefix that you entered in the bucket policy.
11. Choose **Save**.

Additional resources

For more information about S3 buckets and policies, see the [Amazon Simple Storage Service User Guide](#).

Amazon SNS topic policy for CloudTrail

To send notifications to an SNS topic, CloudTrail must have the required permissions. CloudTrail automatically attaches the required permissions to the topic when you create an Amazon SNS topic as part of creating or updating a trail in the CloudTrail console.

Important

As a security best practice, to restrict access to your SNS topic, we strongly recommend that after you create or update a trail to send SNS notifications, you manually edit the IAM policy that is attached to the SNS topic to add condition keys. For more information, see [the section called “Security best practice for SNS topic policy” \(p. 248\)](#) in this topic.

CloudTrail adds the following statement to the policy for you with the following fields:

- The allowed SIDs.
- The service principal name for CloudTrail.
- The SNS topic, including region, account ID, and topic name.

The following policy allows CloudTrail to send notifications about log file delivery from supported regions. For more information, see [CloudTrail supported regions \(p. 12\)](#). This is the default policy that is attached to a new or existing SNS topic policy when you create or update a trail, and choose to enable SNS notifications.

SNS topic policy

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

To use an AWS KMS-encrypted Amazon SNS topic to send notifications, you must also enable compatibility between the event source (CloudTrail) and the encrypted topic by adding the following statement to the policy of the AWS KMS key.

KMS key policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}
```

For more information, see [Enable Compatibility between Event Sources from AWS Services and Encrypted Topics](#).

Contents

- [Security best practice for SNS topic policy \(p. 248\)](#)
- [Specifying an existing topic for sending notifications \(p. 249\)](#)
- [Troubleshooting the SNS topic policy \(p. 250\)](#)
 - [Common SNS policy configuration errors \(p. 250\)](#)
- [Additional resources \(p. 251\)](#)

Security best practice for SNS topic policy

By default, the IAM policy statement that CloudTrail attaches to your Amazon SNS topic allows the CloudTrail service principal to publish to an SNS topic, identified by an ARN. To help prevent an attacker from gaining access to your SNS topic, and sending notifications on behalf of CloudTrail to topic recipients, manually edit your CloudTrail SNS topic policy to add at least one `aws:SourceArn` or `aws:SourceAccount` condition key to the policy statement attached by CloudTrail.

As a security best practice, we strongly recommend adding the `aws:SourceArn` condition key. The value of this key is the ARN of the trail; because it includes both the specific trail ID and the ID of the account that owns the trail, it restricts SNS topic access to only those accounts that have permission to manage the trail. Before you add condition keys to your SNS topic policy, get the SNS topic name from your trail's settings in the CloudTrail console.

To add the `aws:SourceArn` condition key to your SNS topic policy

1. Open the Amazon SNS console at <https://console.aws.amazon.com/sns/v3/home>.
2. In the navigation pane, choose **Topics**.
3. Choose the SNS topic that is shown in your trail settings, and then choose **Edit**.
4. Expand **Access policy**.
5. In the **Access policy** JSON editor, look for a block that resembles the following example.

```
{
  "Sid": "AWSCloudTrailSNSPolicy20150319",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "SNS:Publish",
  "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-logs-111122223333-61bbe496"
}
```

6. Add a new block for a condition, `aws:SourceArn`, as shown in the following example. The value of `aws:SourceArn` is the ARN of the trail about which you are sending notifications to SNS.

```
{
  "Sid": "AWSCloudTrailSNSPolicy20150319",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "SNS:Publish",
  "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-logs-111122223333-61bbe496",
  "Condition": {
    "StringEquals": {
      "aws:SourceArn": "arn:aws:cloudtrail:us-west-2:123456789012:trail/Tail13"
    }
  }
}
```

```
}  
}
```

7. When you are finished editing the SNS topic policy, choose **Save changes**.

To add the `aws:SourceAccount` condition key to your SNS topic policy

1. Open the Amazon SNS console at <https://console.aws.amazon.com/sns/v3/home>.
2. In the navigation pane, choose **Topics**.
3. Choose the SNS topic that is shown in your trail settings, and then choose **Edit**.
4. Expand **Access policy**.
5. In the **Access policy** JSON editor, look for a block that resembles the following example.

```
{  
  "Sid": "AWSCloudTrailSNSPolicy20150319",  
  "Effect": "Allow",  
  "Principal": {  
    "Service": "cloudtrail.amazonaws.com"  
  },  
  "Action": "SNS:Publish",  
  "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-logs-111122223333-61bbe496"  
}
```

6. Add a new block for a condition, `aws:SourceAccount`, as shown in the following example. The value of `aws:SourceAccount` is the ID of the account that owns the CloudTrail trail. This example restricts access to the SNS topic to only those users who can sign in to the AWS account 123456789012.

```
{  
  "Sid": "AWSCloudTrailSNSPolicy20150319",  
  "Effect": "Allow",  
  "Principal": {  
    "Service": "cloudtrail.amazonaws.com"  
  },  
  "Action": "SNS:Publish",  
  "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-logs-111122223333-61bbe496",  
  "Condition": {  
    "StringEquals": {  
      "aws:SourceAccount": "123456789012"  
    }  
  }  
}
```

7. When you are finished editing the SNS topic policy, choose **Save changes**.

Specifying an existing topic for sending notifications

You can manually add the permissions for an Amazon SNS topic to your topic policy in the Amazon SNS console and then specify the topic in the CloudTrail console.

To manually update an SNS topic policy

1. Open the Amazon SNS console at <https://console.aws.amazon.com/sns/v3/home>.
2. Choose **Topics** and then choose the topic.
3. Choose **Other topic actions** and then choose **Edit topic policy**.

4. Choose **Advanced view**, and add the statement from [SNS topic policy \(p. 247\)](#) with the appropriate values for the region, account ID, and topic name.
5. Choose **Update policy**.
6. If your topic is an encrypted topic, you must allow CloudTrail to have `kms:GenerateDataKey*` and the `kms:Decrypt` permissions. For more information, see [Encrypted SNS topic KMS key policy \(p. 247\)](#).
7. Return to the CloudTrail console and specify the topic for the trail.

Troubleshooting the SNS topic policy

The following sections describe how to troubleshoot the SNS topic policy.

Common SNS policy configuration errors

When you create a new topic as part of creating or updating a trail, CloudTrail attaches the required permissions to your topic. The topic policy uses the service principal name, `"cloudtrail.amazonaws.com"`, which allows CloudTrail to send notifications for all regions.

If CloudTrail is not sending notifications for a region, it's possible that your topic has an older policy that specifies CloudTrail account IDs for each region. This policy gives CloudTrail permission to send notifications only for the regions specified.

The following topic policy allows CloudTrail to send notifications for the specified nine regions only:

Example topic policy with account IDs

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AWSCloudTrailSNSPolicy20131101",
    "Effect": "Allow",
    "Principal": {"AWS": [
      "arn:aws:iam::903692715234:root",
      "arn:aws:iam::035351147821:root",
      "arn:aws:iam::859597730677:root",
      "arn:aws:iam::814480443879:root",
      "arn:aws:iam::216624486486:root",
      "arn:aws:iam::086441151436:root",
      "arn:aws:iam::388731089494:root",
      "arn:aws:iam::284668455005:root",
      "arn:aws:iam::113285607260:root"
    ]},
    "Action": "SNS:Publish",
    "Resource": "aws:arn:sns:us-east-1:123456789012:myTopic"
  }]
}
```

This policy uses a permission based on individual CloudTrail account IDs. To deliver logs for a new region, you must manually update the policy to include the CloudTrail account ID for that region. For example, because CloudTrail added support for the US East (Ohio) Region, you must update the policy to add the account ID ARN for that region: `"arn:aws:iam::475085895292:root"`.

As a best practice, update the policy to use a permission with the CloudTrail service principal. To do this, replace the account ID ARNs with the service principal name: `"cloudtrail.amazonaws.com"`.

This gives CloudTrail permission to send notifications for current and new regions. The following is an updated version of the previous policy:

Example topic policy with service principal name

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AWSCloudTrailSNSPolicy20131101",
    "Effect": "Allow",
    "Principal": {"Service": "cloudtrail.amazonaws.com"},
    "Action": "SNS:Publish",
    "Resource": "arn:aws:sns:us-west-2:123456789012:myTopic"
  }]
}
```

Verify that the policy has the correct values:

- In the **Resource** field, specify the account number of the topic owner. For topics that you create, specify your account number.
- Specify the appropriate values for the region and SNS topic name.

Additional resources

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

Troubleshooting AWS CloudTrail identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with CloudTrail and IAM.

Topics

- [I am not authorized to perform an action in CloudTrail \(p. 251\)](#)
- [I'm an Administrator and Want to Allow Others to Access CloudTrail \(p. 252\)](#)
- [I want to allow people outside of my AWS account to access my CloudTrail resources \(p. 252\)](#)

I am not authorized to perform an action in CloudTrail

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a trail but does not have either the appropriate CloudTrail managed policy (**AWSCloudTrail_FullAccess** or **AWSCloudTrailReadOnlyAccess**) or the equivalent permissions applied to his account.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
cloudtrail:GetTrailStatus on resource: My-Tail
```

In this case, Mateo asks his administrator to update his policies to allow him to access trail information and status in the console.

If you are signed in with an IAM user or role that has the **AWSCloudTrail_FullAccess** managed policy or its equivalent permissions, and you cannot configure AWS Config or Amazon CloudWatch Logs integration with a trail, you might be missing the required permissions for integration with those

services. For more information, see [Granting permission to view AWS Config information on the CloudTrail console \(p. 242\)](#) and [Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console \(p. 242\)](#).

I'm an Administrator and Want to Allow Others to Access CloudTrail

To allow others to access CloudTrail, you must create an IAM entity (user, group, or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in CloudTrail. For examples of how to do this, see [Granting custom permissions for CloudTrail users \(p. 238\)](#) and [Granting permissions for CloudTrail administration \(p. 236\)](#).

I want to allow people outside of my AWS account to access my CloudTrail resources

You can create a role and share CloudTrail information between multiple AWS accounts. For more information, see [Sharing CloudTrail log files between AWS accounts \(p. 186\)](#).

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether CloudTrail supports these features, see [How AWS CloudTrail works with IAM \(p. 229\)](#).
- To learn how to provide access to your resources across AWS accounts that you own, see [Providing access to an IAM user in another AWS account that you own](#) in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see [Providing access to AWS accounts owned by third parties](#) in the *IAM User Guide*.
- To learn how to provide access through identity federation, see [Providing access to externally authenticated users \(identity federation\)](#) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.

Using service-linked roles for AWS CloudTrail

AWS CloudTrail uses AWS Identity and Access Management (IAM) [service-linked roles](#). A service-linked role is a unique type of IAM role that is linked directly to CloudTrail. Service-linked roles are predefined by CloudTrail and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up CloudTrail easier because you don't have to manually add the necessary permissions. CloudTrail defines the permissions of its service-linked roles, and unless defined otherwise, only CloudTrail can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

For information about other services that support service-linked roles, see [AWS Services That Work with IAM](#) and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for CloudTrail

CloudTrail uses the service-linked role named **AWSServiceRoleForCloudTrail** – This service linked role is used for supporting the organization trail feature.

The AWSServiceRoleForCloudTrail service-linked role trusts the following services to assume the role:

- `cloudtrail.amazonaws.com`

This role is used to support the creation and management of organization trails in CloudTrail. For more information, see [Creating a trail for an organization \(p. 112\)](#).

The role permissions policy allows CloudTrail to complete the following actions on the specified resources:

- Action: `All` on all CloudTrail resources.
- Action: `organizations:DescribeAccount` on all Organizations resources.
- Action: `organizations:DescribeOrganizations` on all Organizations resources.
- Action: `organizations:ListAccounts` on all Organizations resources.
- Action: `organizations:ListAWSServiceAccessForOrganization` on all Organizations resources.

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see [Service-Linked Role Permissions](#) in the *IAM User Guide*.

Creating a service-linked role for CloudTrail

You don't need to manually create a service-linked role. When you create an organization trail in the AWS Management Console, the AWS CLI, or the AWS API, CloudTrail creates the service-linked role for you.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you create an organization trail, CloudTrail creates the service-linked role for you again.

Editing a service-linked role for CloudTrail

CloudTrail does not allow you to edit the AWSServiceRoleForCloudTrail service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see [Editing a Service-Linked Role](#) in the *IAM User Guide*.

Deleting a service-linked role for CloudTrail

You don't need to manually delete the AWSServiceRoleForCloudTrail role. If an AWS account is removed from an Organizations organization, the AWSServiceRoleForCloudTrail role is automatically removed from that AWS account. You cannot detach or remove policies from the AWSServiceRoleForCloudTrail service-linked role in an organization management account without removing the account from the organization.

You can also use the IAM console, the AWS CLI or the AWS API to manually delete the service-linked role. To do this, you must first manually clean up the resources for your service-linked role, and then you can manually delete it.

Note

If the CloudTrail service is using the role when you try to delete the resources, then deletion might fail. If that happens, wait for a few minutes and try the operation again.

To remove a resource being used by the AWSServiceRoleForCloudTrail role, you can do one of the following:

- Remove the AWS account from the organization in Organizations.
- Update the trail so that it is no longer an organization trail.
- Delete the trail.

For more information, see [Creating a trail for an organization \(p. 112\)](#), [Updating a trail \(p. 79\)](#), and [Deleting a trail \(p. 86\)](#).

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForCloudTrail service-linked role. For more information, see [Deleting a Service-Linked Role](#) in the *IAM User Guide*.

Supported regions for CloudTrail service-linked roles

CloudTrail supports using service-linked roles in all of the regions where CloudTrail and Organizations are both available. For more information, see [AWS Regions and Endpoints](#).

Compliance validation for AWS CloudTrail

Third-party auditors assess the security and compliance of AWS CloudTrail as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see [AWS Services in Scope by Compliance Program](#). For general information, see [AWS Compliance Programs](#).

You can download third-party audit reports using AWS Artifact. For more information, see [Downloading Reports in AWS Artifact](#).

Your compliance responsibility when using CloudTrail is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- [Security and Compliance Quick Start Guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- [Architecting for HIPAA Security and Compliance Whitepaper](#) – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- [AWS Compliance Resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [AWS Config](#) – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- [Amazon S3 Inventory](#) can help you audit and report on the replication and encryption status of the Amazon S3 buckets you use for storing log files and its objects for business, compliance, and regulatory needs.

Resilience in AWS CloudTrail

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures. If you specifically need to replicate your CloudTrail log files over greater geographic distances, you can use [Cross-Region Replication](#) for your trail Amazon S3 buckets, which enables automatic, asynchronous copying of objects across buckets in different AWS Regions.

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

In addition to the AWS global infrastructure, CloudTrail offers several features to help support your data resiliency and backup needs.

Trails that log events in all AWS Regions

When you apply a trail to all AWS Regions, CloudTrail creates trails with identical configurations in all other AWS Regions in your account. When AWS adds a new Region, that trail configuration is automatically created in the new Region.

Versioning, lifecycle configuration, and object lock protection for CloudTrail log data

Because CloudTrail uses Amazon S3 buckets to store log files, you can also use the features provided by Amazon S3 to help support your data resiliency and backup needs. For more information, see [Resilience in Amazon S3](#).

Infrastructure security in AWS CloudTrail

As a managed service, AWS CloudTrail is protected by the AWS global network security procedures that are described in the [Amazon Web Services: Overview of Security Processes](#) whitepaper.

You use AWS published API calls to access CloudTrail through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

The following security best practices also address infrastructure security in CloudTrail:

- [Consider Amazon VPC endpoints for trail access. \(p. 131\)](#)
- Consider Amazon VPC endpoints for Amazon S3 bucket access. For more information, see [Example Bucket Policies for VPC Endpoints for Amazon S3](#).
- Identify and audit all Amazon S3 buckets that contain CloudTrail log files. Consider using tags to help identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see [AWS Resource Groups](#).

Security best practices in AWS CloudTrail

AWS CloudTrail provides a number of security features to consider as you develop and implement your own security policies. The following best practices are general guidelines and don't represent a

complete security solution. Because these best practices might not be appropriate or sufficient for your environment, treat them as helpful considerations rather than prescriptions.

Topics

- [CloudTrail detective security best practices \(p. 256\)](#)
- [CloudTrail preventative security best practices \(p. 257\)](#)

CloudTrail detective security best practices

Create a trail

For an ongoing record of events in your AWS account, you must create a trail. Although CloudTrail provides 90 days of event history information for management events in the CloudTrail console without creating a trail, it is not a permanent record, and it does not provide information about all possible types of events. For an ongoing record, and for a record that contains all the event types you specify, you must create a trail, which delivers log files to an Amazon S3 bucket that you specify.

To help manage your CloudTrail data, consider creating one trail that logs management events in all AWS Regions, and then creating additional trails that log specific event types for resources, such as Amazon S3 bucket activity or AWS Lambda functions.

The following are some steps you can take:

- [Create a trail for your AWS account \(p. 70\)](#)
- [Create a trail for an organization \(p. 112\)](#)

Apply trails to all AWS Regions

In order to obtain a complete record of events taken by a user, role, or service in your AWS account, each trail should be configured to log events in all AWS Regions. By logging events in all AWS Regions, you ensure that all events that occur in your AWS account are logged, regardless of which AWS Region where they occurred. This includes logging [global service events \(p. 10\)](#), which are logged to an AWS Region specific to that service. When you create a trail that applies to all regions, CloudTrail records events in each region and delivers the CloudTrail event log files to an S3 bucket that you specify. If an AWS Region is added after you create a trail that applies to all regions, that new region is automatically included, and events in that region are logged. This is the default option when you create a trail in the CloudTrail console.

The following are some steps you can take:

- [Create a trail for your AWS account \(p. 70\)](#)
- [Update an existing trail \(p. 79\)](#) to log events in all AWS Regions
- Implement ongoing detective controls to help ensure all trails created are logging events in all AWS Regions by using the [multi-region-cloud-trail-enabled](#) rule in AWS Config.

Enable CloudTrail log file integrity

Validated log files are especially valuable 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. CloudTrail log file integrity validation uses industry standard algorithms: SHA-256 for hashing and SHA-256 with RSA for digital signing. This makes it computationally unfeasible to modify, delete or forge CloudTrail log files without detection. For more information, see [Enabling validation and validating files \(p. 196\)](#).

Integrate with Amazon CloudWatch Logs

CloudWatch Logs allows you to monitor and receive alerts for specific events captured by CloudTrail. The events sent to CloudWatch Logs are those configured to be logged by your trail, so make sure you have configured your trail or trails to log the event types (management events and/or data events) that you are interested in monitoring.

For example, you can monitor key security and network-related management events, such as [failed AWS Management Console sign-in events \(p. 178\)](#).

The following are some steps you can take:

- Review example [CloudWatch Logs integrations for CloudTrail \(p. 176\)](#).
- Configure your trail to [send events to CloudWatch Logs \(p. 159\)](#).
- Consider implementing ongoing detective controls to help ensure all trails are sending events to CloudWatch Logs for monitoring by using the [cloud-trail-cloud-watch-logs-enabled](#) rule in AWS Config.

CloudTrail preventative security best practices

The following best practices for CloudTrail can help prevent security incidents.

Log to a dedicated and centralized Amazon S3 bucket

CloudTrail log files are an audit log of actions taken by a user, role or an AWS service. The integrity, completeness and availability of these logs is crucial for forensic and auditing purposes. By logging to a dedicated and centralized Amazon S3 bucket, you can enforce strict security controls, access, and segregation of duties.

The following are some steps you can take:

- Create a separate AWS account as a log archive account. If you use AWS Organizations, enroll this account in the organization, and consider [creating an organization trail \(p. 112\)](#) to log data for all AWS accounts in your organization.
- If you do not use Organizations but want to log data for multiple AWS accounts, [create a trail \(p. 70\)](#) to log activity in this log archive account. Restrict access to this account to only trusted administrative users who should have access to account and auditing data.
- As part of creating a trail, whether it is an organization trail or a trail for a single AWS account, create a dedicated Amazon S3 bucket to store log files for this trail.
- If you want to log activity for more than one AWS account, [modify the bucket policy \(p. 184\)](#) to allow logging and storing log files for all AWS accounts that you want to log AWS account activity.
- If you are not using an organization trail, create trails in all of your AWS accounts, specifying the Amazon S3 bucket in the log archive account.

Use server-side encryption with AWS KMS managed keys

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 an [AWS KMS key](#), also known as a KMS key.

Note

If you use SSE-KMS and log file validation, and you have modified your Amazon S3 bucket policy to only allow SSE-KMS encrypted files, you will not be able to create trails that utilize that

bucket unless you modify your bucket policy to specifically allow AES256 encryption, as shown in the following example policy line.

```
"StringNotEquals": { "s3:x-amz-server-side-encryption": [ "aws:kms", "AES256" ] }
```

The following are some steps you can take:

- [Review the advantages of encrypting your log files with SSE-KMS \(p. 259\).](#)
- [Create a KMS key to use for encrypting log files \(p. 261\).](#)
- [Configure log file encryption for your trails. \(p. 267\)](#)
- Consider implementing ongoing detective controls to help ensure all trails are encrypting log files with SSE-KMS by using the [cloud-trail-encryption-enabled](#) rule in AWS Config.

Add a condition key to the default Amazon SNS topic policy

When you configure a trail to send notifications to Amazon SNS, CloudTrail adds a policy statement to your SNS topic access policy that allows CloudTrail to send content to an SNS topic. As a security best practice, we recommend adding an `aws:SourceArn` or `aws:SourceAccount` condition key to the CloudTrail policy statement. This helps prevent unauthorized account access to your SNS topic. For more information, see [Amazon SNS topic policy for CloudTrail](#).

Implement least privilege access to Amazon S3 buckets where you store log files

CloudTrail trails log events to an Amazon S3 bucket that you specify. These log files contain an audit log of actions taken by users, roles, and AWS services. The integrity and completeness of these log files are crucial for auditing and forensic purposes. In order to help ensure that integrity, you should adhere to the principle of least privilege when creating or modifying access to any Amazon S3 bucket used for storing CloudTrail log files.

The following are some steps you can take:

- Review the [Amazon S3 bucket policy \(p. 243\)](#) for any and all buckets where you store log files and adjust it if necessary to remove any unnecessary access. This bucket policy will be generated for you if you create a trail using the CloudTrail console, but can also be created and managed manually.
- If you are using the same Amazon S3 bucket to store log files for multiple AWS accounts, follow the guidance for [receiving log files for multiple accounts \(p. 183\)](#).
- If you are using an organization trail, make sure you follow the guidance for [organization trails \(p. 112\)](#), and review the example policy for an Amazon S3 bucket for an organization trail in [Creating a trail for an organization with the AWS Command Line Interface \(p. 121\)](#).
- Review the [Amazon S3 security documentation](#) and the [example walkthrough for securing a bucket](#).

Enable MFA Delete on the Amazon S3 bucket where you store log files

Configuring multi-factor authentication (MFA) ensures that any attempt to change the versioning state of your bucket or permanently delete an object version requires additional authentication. This helps prevent any operation that could compromise the integrity of your log files, even if a user acquires the password of an IAM user that has permissions to permanently delete Amazon S3 objects.

The following are some steps you can take:

- Review the [MFA Delete](#) guidance.
- [Add an Amazon S3 bucket policy to require MFA.](#)

Configure object lifecycle management on the Amazon S3 bucket where you store log files

The CloudTrail trail default is to store log files indefinitely in the Amazon S3 bucket configured for the trail. You can use the [Amazon S3 object lifecycle management rules](#) to define your own retention policy to better meet your business and auditing needs. For example, you might want to archive log files that are more than a year old to Amazon Glacier, or delete log files after a certain amount of time has passed.

Limit access to the `AWSCloudTrail_FullAccess` policy

Users with the [AWSCloudTrail_FullAccess](#) (p. 240) policy have the ability to disable or reconfigure the most sensitive and important auditing functions in their AWS accounts. This policy is not intended to be shared or applied broadly to users and roles in your AWS account. Limit application of this policy to as few individuals as possible, those you expect to act as AWS account administrators.

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.

Note

Enabling server-side encryption encrypts the log files but not the digest files with SSE-KMS. Digest files are encrypted with [Amazon S3–managed encryption keys \(SSE-S3\)](#).

If you are using an existing S3 bucket with an [S3 Bucket Key](#), CloudTrail must be allowed permission in the key policy to use the AWS KMS actions `GenerateDataKey` and `DescribeKey`. If `cloudtrail.amazonaws.com` is not granted those permissions in the key policy, you cannot create or update a trail.

To use SSE-KMS with CloudTrail, you create and manage a KMS key, also known as an [AWS KMS key](#). 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 KMS key encryption keys yourself.
- You can use a single KMS key 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, to read log files, the following permissions are required:
 - A user must have S3 read permissions for the bucket that contains the log files.
 - A user must also have a policy or role applied that allows decrypt permissions by the KMS key policy.
- Because S3 automatically decrypts the log files for requests from users authorized to use the KMS key, SSE-KMS encryption for CloudTrail log files is backward-compatible with applications that read CloudTrail log data.

Note

The KMS key that you choose must be created in the same AWS Region as the Amazon S3 bucket that receives your log files. For example, if the log files will be stored in a bucket in the US East (Ohio) Region, you must create or choose a KMS key that was created in that Region. To verify the Region for an Amazon S3 bucket, inspect its properties in the Amazon S3 console.

Enabling log file encryption

Note

If you create a KMS key in the CloudTrail console, CloudTrail adds the required KMS key 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 KMS key.
 - For information about creating a KMS key with the AWS Management Console, see [Creating Keys](#) in the *AWS Key Management Service Developer Guide*.
 - For information about creating a KMS key with the AWS CLI, see [create-key](#).

Note

The KMS key that you choose must be in the same region as the S3 bucket that receives your log files. To verify the region for an S3 bucket, 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 [Configure AWS KMS key policies for CloudTrail](#) (p. 261).

Warning

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 cannot read encrypted files until you grant them those permissions.

- For information about editing a policy with the IAM console, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.
 - For information about attaching a policy to a KMS key with the AWS CLI, see [put-key-policy](#).
3. Update your trail to use the KMS key whose policy you modified for CloudTrail.
 - To update your trail configuration by using the CloudTrail console, see [Updating a trail to use your KMS key](#) (p. 267).
 - To update your trail configuration by using the AWS CLI, see [Enabling and disabling CloudTrail log file encryption with the AWS CLI](#) (p. 268).

CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see [Using multi-Region keys](#) in the *AWS Key Management Service Developer Guide*.

The next section describes the policy sections that your KMS key policy requires for use with CloudTrail.

Granting permissions to create a KMS key

You can grant users permission to create an AWS KMS key with the `AWSKeyManagementServicePowerUser` policy.

To grant permission to create a KMS key

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 KMS key. If you want to create custom policies for your users, see [Creating Customer Managed Policies](#) in the *IAM User Guide*.

Configure AWS KMS key policies for CloudTrail

You can create an AWS KMS key in three ways:

- The CloudTrail console
- The IAM console
- The AWS CLI

Note

If you create a KMS key in the CloudTrail console, CloudTrail adds the required KMS key policy for you. You do not need to manually add the policy statements. See [Default KMS key policy created in CloudTrail console \(p. 266\)](#).

If you create a KMS key in the IAM console or the AWS CLI, you must 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 KMS key with the AWS CLI, see [create-key](#).
- To edit a KMS key 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 KMS key policy sections for use with CloudTrail

If you created a KMS key with the IAM console or the AWS CLI, then you must, at minimum, add three statements to your KMS key policy for it to work with CloudTrail. You do not need to manually add statements for a KMS key that is created using the CloudTrail console.

1. Enable CloudTrail log encrypt permissions. See [Granting encrypt permissions \(p. 262\)](#).
2. Enable CloudTrail log decrypt permissions. See [Granting decrypt permissions \(p. 263\)](#). If you are using an existing S3 bucket with an [S3 Bucket Key](#), `kms:Decrypt` permissions are required to create or update a trail with SSE-KMS encryption enabled.
3. Enable CloudTrail to describe KMS key properties. See [Enable CloudTrail to describe KMS key properties \(p. 265\)](#).

Note

When you add the new sections to your KMS key policy, do not change any existing sections in the policy.

Warning

If encryption is enabled on a trail and the KMS key is disabled or the KMS key policy is not correctly configured for CloudTrail, CloudTrail will not deliver logs until the KMS key issue is corrected.

Granting encrypt permissions

Example Allow CloudTrail to encrypt logs on behalf of specific accounts

CloudTrail needs explicit permission to use the KMS key to encrypt logs on behalf of specific accounts. To specify an account, add the following required statement to your KMS key 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 KMS key 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 KMS key to encrypt CloudTrail logs.

Scenario

- Your KMS key 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.

KMS key 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 KMS key 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 KMS key 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. If you are using an existing S3 bucket with an [S3 Bucket Key](#), `kms:Decrypt` permissions are required to create or update a trail with SSE-KMS encryption enabled.

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 KMS key policy, modifying the `Principal` section to add a line for every principal (role or user) that you want to be able to decrypt by using your KMS key.

```
{  
  "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 KMS key

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.

Example Scenario

- Your KMS key, 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.

KMS key 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"
  }
}
```

Allow users in other accounts to decrypt with your KMS key

You can allow users in other accounts to use your KMS key 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 KMS key 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.

KMS key 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-west-2:111111111111: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.

Example Scenario

- Your KMS key 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.

KMS key 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 KMS key policy for use with CloudTrail, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.

Enable CloudTrail to describe KMS key properties

CloudTrail requires the ability to describe the properties of the KMS key. To enable this functionality, add the following required statement as is to your KMS key 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 more information about editing KMS key policies, see [Editing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.

Default KMS key policy created in CloudTrail console

If you create an AWS KMS key in the CloudTrail console, the following policy is automatically created for you. The policy allows these permissions:

- Allows AWS account (root) permissions for the KMS key.
- Allows CloudTrail to encrypt log files under the KMS key and describe the KMS key.
- 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 KMS key.
- Enables cross-account log decryption for the account ID of the account that created the trail.

```
{
  "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 KMS key.

Updating a trail to use your KMS key

To update a trail to use the AWS KMS key that you modified for CloudTrail, complete the following steps in the CloudTrail console.

Note

Updating a trail with the following procedure encrypts the log files but not the digest files with SSE-KMS. Digest files are encrypted with [Amazon S3-managed encryption keys \(SSE-S3\)](#).

If you are using an existing S3 bucket with an [S3 Bucket Key](#), CloudTrail must be allowed permission in the key policy to use the AWS KMS actions `GenerateDataKey` and `DescribeKey`. If `cloudtrail.amazonaws.com` is not granted those permissions in the key policy, you cannot create or update a trail.

To update a trail using the AWS CLI, see [Enabling and disabling CloudTrail log file encryption with the AWS CLI](#) (p. 268).

To update a trail to use your KMS key

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. In **General details**, choose **Edit**.
4. For **Log file SSE-KMS encryption**, if it is not already enabled, choose **Enabled** to encrypt your log files with SSE-KMS instead of SSE-S3. The default is **Enabled**. For more information about this encryption type, see [Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys \(SSE-S3\)](#).

Choose **Existing** to update your trail with your AWS KMS key. Choose a KMS key that is in the same region as the S3 bucket that receives your log files. To verify the region for an S3 bucket, view its properties in the S3 console.

Note

You can also type the ARN of a key from another account. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#). 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 [Configure AWS KMS key policies for CloudTrail \(p. 261\)](#).

In **AWS KMS Alias**, specify the alias for which you changed the policy for use with CloudTrail, in the format `alias/MyAliasName`. For more information, see [Updating a trail to use your KMS key \(p. 267\)](#).

You can type the alias name, ARN, or the globally unique key ID. If the KMS key 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:region:123456789012:alias/MyAliasName`
 - **Key ARN:**
`arn:aws:kms:region:123456789012:key/12345678-1234-1234-1234-123456789012`
 - **Globally unique key ID:** `12345678-1234-1234-1234-123456789012`
5. Choose **Update trail**.

Note

If the KMS key that you chose is disabled or is pending deletion, you cannot save the trail with that KMS key. You can enable the KMS key or choose another one. For more information, see [Key state: Effect on your KMS key](#) in the *AWS Key Management Service Developer Guide*.

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 \(SSE-KMS\) \(p. 259\)](#).

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 [Configure AWS KMS key policies for CloudTrail \(p. 261\)](#).

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-2:123456789012:alias/MyAliasName`
- **Key ARN.** Example: `arn:aws:kms:us-east-2:123456789012: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-2:123456789012:trail/Default",
  "LogFileValidationEnabled": false,
  "KmsKeyId": "arn:aws:kms:us-east-2:123456789012: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-2:123456789012:trail/Default",
  "LogFileValidationEnabled": false,
  "S3BucketName": "my-bucket-name"
}
```

The absence of the `KmsKeyId` element indicates that log file encryption is no longer enabled.

CloudTrail log event reference

A CloudTrail log is a record in JSON format. The log 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 of an event where an IAM user named `Mary_Major` called the CloudTrail `StartLogging` API from the CloudTrail console to start the logging process.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAJDPLRKL7UEEXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/Mary_Major",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major",
    "sessionContext": {
      "sessionIssuer": {},
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-06-18T22:28:31Z"
      }
    }
  },
  "invokedBy": "signin.amazonaws.com",
  "eventTime": "2019-06-19T00:18:31Z",
  "eventSource": "cloudtrail.amazonaws.com",
  "eventName": "StartLogging",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.64",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "name": "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-First-Trail"
  },
  "responseElements": null,
  "requestID": "ddf5140f-EXAMPLE",
  "eventID": "7116c6a1-EXAMPLE",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
},
... additional entries ...
```

There are two events logged to show unusual activity in CloudTrail Insights: a start event and an end event. The following example shows a single log record of a starting Insights event that occurred when the Application Auto Scaling API `CompleteLifecycleAction` was called an unusual number of times. For Insights events, the value of `eventCategory` is `Insight`. An `insightDetails` block identifies the event state, source, name, Insights type, and context, including statistics and attributions. For more information about the `insightDetails` block, see [CloudTrail Insights `insightDetails` element](#) (p. 288).

```
{
  "eventVersion": "1.07",
  "eventTime": "2020-07-21T20:56:00Z",
```

```

"awsRegion": "us-east-1",
"eventID": "abcd00b0-ccfe-422d-961c-98a2198a408x",
"eventType": "AwsCloudTrailInsight",
"recipientAccountId": "838185438692",
"sharedEventID": "7bb000gg-22b3-4c03-94af-c74tj0c8m7c0",
"insightDetails": {
  "state": "Start",
  "eventSource": "autoscaling.amazonaws.com",
  "eventName": "CompleteLifecycleAction",
  "insightType": "ApiCallRateInsight",
  "insightContext": {
    "statistics": {
      "baseline": {
        "average": 0.0000882145
      },
      "insight": {
        "average": 0.6
      },
      "insightDuration": 5,
      "baselineDuration": 11336
    },
    "attributions": [
      {
        "attribute": "userIdentityArn",
        "insight": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole2",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole3",
            "average": 0.2
          }
        ],
        "baseline": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "userAgent",
        "insight": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.6
          }
        ],
        "baseline": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "errorCode",
        "insight": [
          {
            "value": "null",
            "average": 0.6
          }
        ]
      }
    ]
  }
}

```

```
    }
  ],
  "baseline": [
    {
      "value": "null",
      "average": 0.0000882145
    }
  ]
}
},
"eventCategory": "Insight"
}
```

The following topics list the data fields that CloudTrail captures for each AWS API call and sign-in event.

Topics

- [CloudTrail record contents \(p. 272\)](#)
- [CloudTrail userIdentity element \(p. 281\)](#)
- [CloudTrail Insights insightDetails element \(p. 288\)](#)
- [Non-API events captured by CloudTrail \(p. 294\)](#)

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. When the value of **Optional** is **True**, the field is only present when it applies to the service, API, or event type. An **Optional** value of **False** means that the field is either always present, or that its presence does not depend on the service, API, or event type. An example is `responseElements`, which is present in events for actions that make changes (create, update, or delete actions).

eventTime

The date and time the request was made, in coordinated universal time (UTC). An event's time stamp comes from the local host that provides the service API endpoint on which the API call was made. For example, a **CreateBucket** API event that is run in the US West (Oregon) Region would get its time stamp from the time on an AWS host running the Amazon S3 endpoint, `s3.us-west-2.amazonaws.com`. In general, AWS services use Network Time Protocol (NTP) to synchronize their system clocks.

Since: 1.0

Optional: False

eventVersion

The version of the log event format. The current version is 1.08.

The `eventVersion` value is a major and minor version in the form *major_version.minor_version*. For example, you can have an `eventVersion` value of 1.07, where 1 is the major version, and 07 is the minor version.

CloudTrail increments the major version if a change is made to the event structure that is not backward-compatible. This includes removing a JSON field that already exists, or changing how the contents of a field are represented (for example, a date format). CloudTrail increments the minor version if a change adds new fields to the event structure. This can occur if new information

is available for some or all existing events, or if new information is available only for new event types. Applications can ignore new fields to stay compatible with new minor versions of the event structure.

If CloudTrail introduces new event types, but the structure of the event is otherwise unchanged, the event version does not change.

To be sure that your applications can parse the event structure, we recommend that you perform an equal-to comparison on the major version number. To be sure that fields that are expected by your application exist, we also recommend performing a greater-than-or-equal-to comparison on the minor version. There are no leading zeroes in the minor version. You can interpret both *major_version* and *minor_version* as numbers, and perform comparison operations.

Since: 1.0

Optional: False

userIdentity

Information about the user that made a request. For more information, see [CloudTrail userIdentity element \(p. 281\)](#).

Since: 1.0

Optional: False

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`.

Since: 1.0

Optional: False

eventName

The requested action, which is one of the actions in the API for that service.

Since: 1.0

Optional: False

awsRegion

The AWS region that the request was made to, such as `us-east-2`. See [CloudTrail supported regions \(p. 12\)](#).

Since: 1.0

Optional: False

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.

Since: 1.0

Optional: False

userAgent

The agent through which the request was made, such as the AWS Management Console, an AWS service, the AWS SDKs or the AWS CLI. This field has a maximum size of 1 KB; content exceeding that limit is truncated. The following are example values:

- `signin.amazonaws.com` – The request was made by an IAM user with the AWS Management Console.
- `console.amazonaws.com` – The request was made by a root user with the AWS Management Console.
- `lambda.amazonaws.com` – The request was made with AWS Lambda.
- `aws-sdk-java` – The request was made with the AWS SDK for Java.
- `aws-sdk-ruby` – The request was made with the AWS SDK for Ruby.
- `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.

Note

For events originated by AWS, this field is usually `AWS Internal/#`, where `#` is a number used for internal purposes.

Since: 1.0

Optional: False

errorCode

The AWS service error if the request returns an error. For an example that shows this field, see [Error code and message log example \(p. 18\)](#). This field has a maximum size of 1 KB; content exceeding that limit is truncated.

Since: 1.0

Optional: True

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 \(p. 18\)](#). This field has a maximum size of 1 KB; content exceeding that limit is truncated.

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`.

Since: 1.0

Optional: True

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. This field has a maximum size of 100 KB; content exceeding that limit is truncated.

Since: 1.0

Optional: False

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. This field has a maximum size of 100 KB; content exceeding that limit is truncated.

The `responseElements` value is useful to help you trace a request with AWS Support. Both `x-amz-request-id` and `x-amz-id-2` contain information that helps you trace a request with AWS Support. These values are the same as those that the service returns in the response to the request that initiates the events, so you can use them to match the event to the request.

Since: 1.0

Optional: False

additionalEventData

Additional data about the event that was not part of the request or response. This field has a maximum size of 28 KB; content exceeding that limit is truncated.

Support for this field begins with `eventVersion 1.00`.

Since: 1.0

Optional: True

requestID

The value that identifies the request. The service being called generates this value. This field has a maximum size of 1 KB; content exceeding that limit is truncated.

Support for this field begins with `eventVersion 1.01`.

Since: 1.01

Optional: False

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.

Since: 1.01

Optional: False

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 \(p. 294\)](#) – 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.
- `AwsConsoleAction` – An action was taken in the console that was not an API call.
- [AwsConsoleSignIn \(p. 295\)](#) – A user in your account (root, IAM, federated, SAML, or SwitchRole) signed in to the AWS Management Console.
- [AwsCloudTrailInsight \(p. 154\)](#) – If Insights events are enabled for the trail, CloudTrail generates Insights events when CloudTrail detects unusual operational activity such as spikes in resource provisioning or bursts of AWS Identity and Access Management (IAM) actions.

AwsCloudTrailInsight events do *not* use the following fields:

- `eventName`
- `eventSource`
- `sourceIPAddress`
- `userAgent`
- `userIdentity`

Since: 1.02

Optional: False

apiVersion

Identifies the API version associated with the `AwsApiCall` `eventType` value.

Since: 1.01

Optional: True

managementEvent

A Boolean value that identifies whether the event is a management event. `managementEvent` is shown in an event record if `eventVersion` is 1.06 or higher, and the event type is one of the following:

- `AwsApiCall`
- `AwsConsoleAction`
- `AwsConsoleSignIn`
- `AwsServiceEvent`

Since: 1.06

Optional: True

readOnly

Identifies whether this operation is a read-only operation. This can be one of the following values:

- `true` – The operation is read-only (for example, `DescribeTrails`).
- `false` – The operation is write-only (for example, `DeleteTrail`).

Since: 1.01

Optional: True

resources

A list of resources accessed in the event. The field can contain the following information:

- Resource ARNs
- Account ID of the resource owner
- Resource type identifier in the format: AWS::*aws-service-name*::*data-type-name*

For example, when an AssumeRole event is logged, the resources field can appear like the following:

- ARN: arn:aws:iam::123456789012:role/*myRole*
- Account ID: 123456789012
- Resource type identifier: AWS::IAM::Role

For example logs with the resources field, see [AWS STS API Event in CloudTrail Log File](#) in the *IAM User Guide* or [Logging AWS KMS API Calls](#) in the *AWS Key Management Service Developer Guide*.

Since: 1.01

Optional: True

recipientAccountId

Represents the account ID that received this event. The recipientAccountId may be different from the [CloudTrail userIdentity element \(p. 281\)](#) accountId. This can occur in cross-account resource access. For example, if a KMS key, also known as an [AWS KMS key](#), 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 KMS key.

Since: 1.02

Optional: True

serviceEventDetails

Identifies the service event, including what triggered the event and the result. For more information, see [AWS service events \(p. 294\)](#). This field has a maximum size of 100 KB; content exceeding that limit is truncated.

Since: 1.05

Optional: True

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 an [AWS KMS key](#) that belongs to another account, the account that used the KMS key and the account that owns the KMS key 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 [Example sharedEventID \(p. 280\)](#).

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.

Since: 1.03

Optional: True

vpcEndpointId

Identifies the VPC endpoint in which requests were made from a VPC to another AWS service, such as Amazon S3.

Since: 1.04

Optional: True

eventCategory

Shows the event category that is used in [LookupEvents](#) calls.

- For management events, the value is `Management`.
- For data events, the value is `Data`.
- For Insights events, the value is `Insight`.

Since: 1.07

Optional: False

addendum

If an event delivery was delayed, or additional information about an existing event becomes available after the event is logged, an addendum field shows information about why the event was delayed. If information was missing from an existing event, the addendum field includes the missing information and a reason for why it was missing. Contents include the following.

- **reason** - The reason that the event or some of its contents were missing. Values can be any of the following.
 - **DELIVERY_DELAY** – There was a delay delivering events. This could be caused by high network traffic, connectivity issues, or a CloudTrail service issue.
 - **UPDATED_DATA** – A field in the event record was missing or had an incorrect value.
 - **SERVICE_OUTAGE** – A service that logs events to CloudTrail had an outage, and couldn't log events to CloudTrail. This is exceptionally rare.
- **updatedFields** - The event record fields that are updated by the addendum. This is only provided if the reason is `UPDATED_DATA`.
- **originalRequestID** - The original unique ID of the request. This is only provided if the reason is `UPDATED_DATA`.
- **originalEventID** - The original event ID. This is only provided if the reason is `UPDATED_DATA`.

Since: 1.08

Optional: True

sessionCredentialFromConsole

Shows whether or not an event originated from an AWS Management Console session. This field is not shown unless the value is `true`, meaning that the client that was used to make the API call

was either a proxy or an external client. If a proxy client was used, the `tlsDetails` event field is not shown.

Since: 1.08

Optional: True

edgeDeviceDetails

Shows information about edge devices that are targets of a request. Currently, [S3 Outposts](#) device events include this field. This field has a maximum size of 28 KB; content exceeding that limit is truncated.

Since: 1.08

Optional: True

tlsDetails

Shows information about the Transport Layer Security (TLS) version, cipher suites, and the FQDN of the client-provided host name of a service API call. Contents include the following. CloudTrail still logs partial TLS details if expected information is missing or empty. For example, if the TLS version and cipher suite are present, but the `HOST` header is empty, available TLS details are still logged in the CloudTrail event.

If `sessionCredentialFromConsole` is present with a value of `true`, `tlsDetails` is present in an event record only if an external client was used to make the API call.

- **tlsVersion** - The TLS version of a request.
- **cipherSuite** - The cipher suite (combination of security algorithms used) of a request.
- **clientProvidedHostHeader** - The FQDN of the client that made the request.

Since: 1.08

Optional: True

Record fields for Insights events

The following are attributes shown in the JSON structure of an Insights event that differ from those in a management or data event.

sharedEventId

A `sharedEventId` for CloudTrail Insights events differs from the `sharedEventId` for the management and data types of CloudTrail events. In Insights events, a `sharedEventId` is a GUID that is generated by CloudTrail Insights to uniquely identify an Insights event. `sharedEventId` is common between the start and the end Insights events, and helps to connect both events to uniquely identify unusual activity. You can think of the `sharedEventId` as the overall Insights event ID.

Since: 1.07

Optional: False

insightDetails

Insights events only. Shows information about the underlying triggers of an Insights event, such as event source, user agent, statistics, API name, and whether the event is the start or end of the Insights event. For more information about the contents of the `insightDetails` block, see [CloudTrail Insights insightDetails element \(p. 288\)](#).

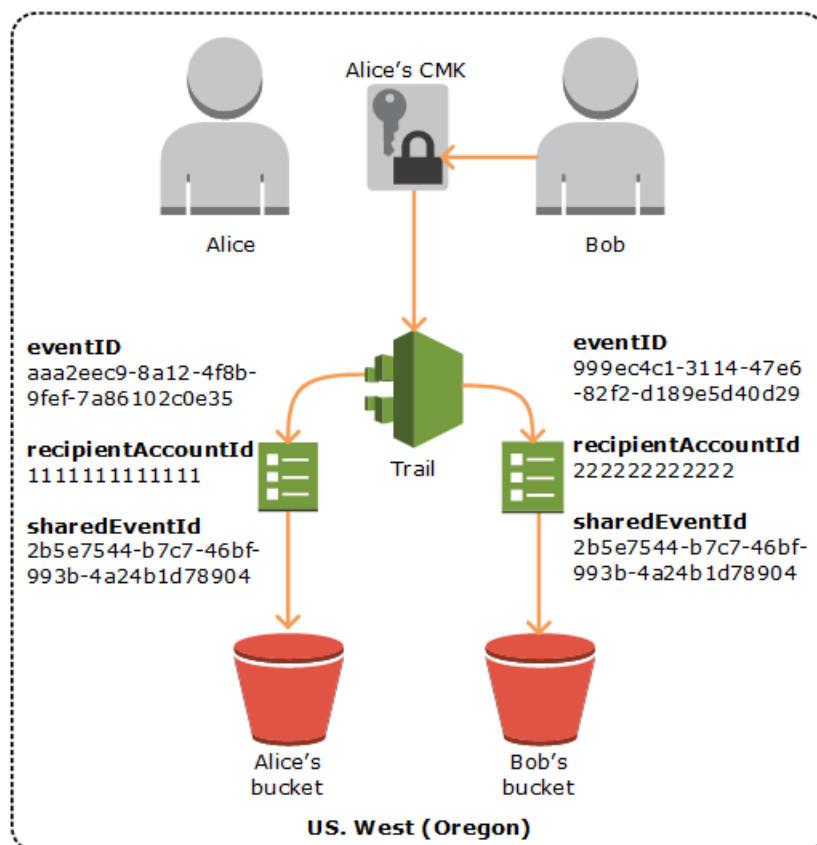
Since: 1.07

Optional: False

Example sharedEventID

The following is an example that describes how CloudTrail delivers two events for the same action:

1. Alice has AWS account (11111111111) and creates an AWS KMS key. She is the owner of this KMS key.
2. Bob has AWS account (22222222222). Alice gives Bob permission to use the KMS key.
3. Each account has a trail and a separate bucket.
4. Bob uses the KMS key to call the `Encrypt` API.
5. CloudTrail sends two separate events.
 - One event is sent to Bob. The event shows that he used the KMS key.
 - One event is sent to Alice. The event shows that Bob used the KMS key.
 - The events have the same `sharedEventID`, but the `eventID` and `recipientAccountID` are unique.



Shared event IDs in CloudTrail Insights

A `sharedEventID` for CloudTrail Insights events differs from the `sharedEventID` for the management and data types of CloudTrail events. In Insights events, a `sharedEventID` is a GUID that is generated

by CloudTrail Insights to uniquely identify a start and end pair of Insights events. `sharedEventID` is common between the start and the end Insights event, and helps to create a correlation between both events to uniquely identify unusual activity.

You can think of the `sharedEventID` as the overall Insights event ID.

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 \(p. 281\)](#)
- [Fields \(p. 282\)](#)
- [Values for AWS STS APIs with SAML and web identity federation \(p. 285\)](#)
- [AWS STS source identity \(p. 286\)](#)

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": "",
  "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": "",
  "sessionContext": {
    "attributes": {
      "mfaAuthenticated": "false",
      "creationDate": "20131102T010628Z"
    },
    "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 [Temporary Security Credentials](#) in the *IAM User Guide*.

- **Directory** – The request was made to a directory service, and the type is unknown. Directory services include the following: Amazon WorkDocs and Amazon QuickSight.
- **AWSAccount** – The request was made by another AWS account.
- **AWSService** – The request was made by an AWS account that belongs to an AWS service. For example, AWS Elastic Beanstalk assumes an IAM role in your account to call other AWS services on your behalf.
- **Unknown** – The request was made with an identity type that CloudTrail cannot determine.

Optional: False

AWSAccount and **AWSService** appear for `type` in your logs when there is cross-account access using an IAM role that you own.

Example: Cross-account access initiated by another AWS account

1. You own an IAM role in your account.
2. Another AWS account switches to that role to assume the role for your account.
3. Because you own the IAM role, you receive a log that shows the other account assumed the role. The type is **AWSAccount**. For an example log entry, see [AWS STS API Event in CloudTrail Log File](#).

Example: Cross-account access initiated by an AWS service

1. You own an IAM role in your account.
2. An AWS account owned by an AWS service assumes that role.
3. Because you own the IAM role, you receive a log that shows the AWS service assumed the role. The type is **AWSService**.

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`:

| type | userName | Description |
|---------------------|-------------------------------|---|
| Root (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. |
| Root (alias set) | The account alias | For more information about AWS account aliases, see Your AWS Account ID and Its Alias . |
| IAMUser | The user name of the IAM user | |
| AssumedRole | Not present | For <code>AssumedRole</code> type, you can find the <code>userName</code> field in <code>sessionContext</code> , as part of the sessionIssuer (p. 284) element. For an example entry, see Examples (p. 281). |
| FederatedUser | Not present | The <code>sessionContext</code> and <code>sessionIssuer</code> section contains information about the identity that issued the session for the federated user. |
| Directory | Can be present | For example, the value can be the account alias or email address of the associated AWS account ID . |
| AWSService | Not present | |
| AWSAccount | Not present | |
| Unknown | Can be present | For example, the value can be the account alias or email address of the associated AWS account ID . |

Optional: True

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.

Optional: True

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.

Optional: True

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.

Optional: True

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. For security reasons, accessKeyId might not be present, or might be displayed as an empty string.

Optional: True

sessionContext

If the request was made with temporary security credentials, sessionContext 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.
- **sourceIdentity** – See [AWS STS source identity \(p. 286\)](#) in this topic. The sourceIdentity field occurs in events when users assume an IAM role to perform an action. sourceIdentity identifies the original user identity making the request, whether that user's identity is an IAM user, an IAM role, a user authenticated by using SAML-based federation, or a user authenticated by using OpenID Connect (OIDC)-compliant web identity federation. For more information about how to configure AWS STS to collect source identity information, see [Monitor and control actions taken with assumed roles](#) in the *IAM User Guide*.

Optional: True

invokedBy

The name of the AWS service that made the request, such as Amazon EC2 Auto Scaling or AWS Elastic Beanstalk.

Optional: True

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 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**:

| sessionIssuer type | userName | Description |
|---------------------------|-------------------------------|---|
| Root (no alias set) | Not present | If you have not set up an alias for your account, the userName field does not appear. For more information about AWS account aliases, see Your AWS Account ID and Its Alias . Note that the userName field will never contain Root because Root is an identity type, 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 . |
| IAMUser | The user name of the IAM user | This also applies when a federated user is using a session issued by IAMUser . |
| Role | 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.

Optional: True

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*.

Optional: True

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](#).

AWS STS source identity

An IAM administrator can configure AWS Security Token Service to require that users specify their identity when they use temporary credentials to assume roles. The `sourceIdentity` field occurs in events when users assume an IAM role, or perform any actions with the assumed role.

The `sourceIdentity` field identifies the original user identity making the request, whether that user's identity is an IAM user, an IAM role, a user authenticated by using SAML-based federation, or a user authenticated by using OpenID Connect (OIDC)-compliant web identity federation. After the IAM administrator configures AWS STS, CloudTrail logs `sourceIdentity` information in the following events and locations within the event record:

- The AWS STS `AssumeRole`, `AssumeRoleWithSAML`, or `AssumeRoleWithWebIdentity` calls that a user identity makes when it assumes a role. `sourceIdentity` is found in the `requestParameters` block of the AWS STS calls.

- The AWS STS AssumeRole, AssumeRoleWithSAML, or AssumeRoleWithWebIdentity calls that a user identity makes if it uses a role to assume another role, known as [role chaining](#). sourceIdentity is found in the requestParameters block of the AWS STS calls.
- The AWS service API calls that the user identity makes while assuming a role and using the temporary credentials assigned by AWS STS. In service API events, sourceIdentity is found in the sessionContext block. For example, if a user identity creates a new S3 bucket, sourceIdentity occurs in the sessionContext block of the CreateBucket event.

For more information about how to configure AWS STS to collect source identity information, see [Monitor and control actions taken with assumed roles](#) in the *IAM User Guide*. For more information about AWS STS events that are logged to CloudTrail, see [Logging IAM and AWS STS API calls with AWS CloudTrail](#) in the *IAM User Guide*.

The following are example snippets of events that show the sourceIdentity field.

Example requestParameters section

In the following example event snippet, a user makes an AWS STS AssumeRole request, and sets a source identity, represented here by *source-identity-value-set*. The user assumes a role represented by the role ARN `arn:aws:iam::123456789012:role/Assumed_Role`. The sourceIdentity field is in the requestParameters block of the event.

```
"eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAJ45Q7YFFAREXAMPLE",
    "accountId": "123456789012"
  },
  "eventTime": "2020-04-02T18:20:53Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.64",
  "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
  "requestParameters": {
    "roleArn": "arn:aws:iam::123456789012:role/Assumed_Role",
    "roleSessionName": "Test1",
    "sourceIdentity": "source-identity-value-set",
  },
}
```

Example responseElements section

In the following example event snippet, a user makes an AWS STS AssumeRole request to assume a role named Developer_Role, and sets a source identity, Admin. The user assumes a role represented by the role ARN `arn:aws:iam::111122223333:role/Developer_Role`. The sourceIdentity field is shown in both the requestParameters and responseElements blocks of the event. The temporary credentials used to assume the role, the session token string, and the assumed role ID, session name, and session ARN are shown in the responseElements block, along with the source identity.

```
"requestParameters": {
  "roleArn": "arn:aws:iam::111122223333:role/Developer_Role",
  "roleSessionName": "Session_Name",
  "sourceIdentity": "Admin"
},
"responseElements": {
  "credentials": {
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "expiration": "Jan 22, 2021 12:46:28 AM",
    "sessionToken": "XXYYaz..."
  }
}
```

```
EXAMPLE_SESSION_TOKEN
XXyYazAz"

    },
    "assumedRoleUser": {
      "assumedRoleId": "AROACKCEVSQ6C2EXAMPLE:Session_Name",
      "arn": "arn:aws:sts::111122223333:assumed-role/Developer_Role/Session_Name"
    },
    "sourceIdentity": "Admin"
  }
  ...
}
```

Example sessionContext section

In the following example event snippet, a user is assuming a role named DevRole to call an AWS service API. The user sets a source identity, represented here by *source-identity-value-set*. The sourceIdentity field is in the sessionContext block, within the userIdentity block of the event.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE: Dev1",
    "arn": "arn: aws: sts: : 123456789012: assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn: aws: iam: : 123456789012: role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23: 46: 28Z"
      },
      "sourceIdentity": "source-identity-value-set"
    }
  }
}
```

CloudTrail Insights insightDetails element

AWS CloudTrail Insights event records include fields that are different from other CloudTrail events in their JSON structure, sometimes called *payload*. A CloudTrail Insights event record includes an insightDetails block that contains information about the underlying triggers of an Insights event, such as event source, user identities, user agents, historical averages or *baselines*, statistics, API name, and whether the event is the start or end of the Insights event. The insightDetails block contains the following information.

- **state** - Whether the event is the starting or ending Insights event. The value can be Start or End.

Since: 1.07

Optional: False

- **eventSource** - The AWS service endpoint that was the source of the unusual activity, such as ec2.amazonaws.com.

Since: 1.07

Optional: False

- **eventName** - The name of the Insights event, typically the name of the API that was the source of the unusual activity.

Since: 1.07

Optional: False

- **insightType** - The type of Insights event. Currently, this value is `ApiCallRateInsight`.

Since: 1.07

Optional: False

- **insightContext** -

Information about the AWS tools (called *user agents*), IAM users and roles (called *user identities*), and error codes associated with the events that were analyzed to generate the Insights event. This element also includes statistics showing how the unusual activity in an Insights event compares to *baseline*, or normal activity.

Since: 1.07

Optional: False

- **statistics** - Includes data about the *baseline*, or typical average rate of calls to the subject API by an account as measured during the baseline period, the average rate of calls that triggered the Insights event over the first minute of the Insights event, the duration, in minutes, of the Insights event, and the duration, in minutes, of the baseline measuring period.

Since: 1.07

Optional: False

- **baseline** - The average number of API calls per minute during the baseline duration on the Insights event's subject API for the account, calculated over the seven days preceding the start of the Insights event.

Since: 1.07

Optional: False

- **insight** -

For a starting Insights event, this value is the average number of API calls per minute during the start of the unusual activity. For an ending Insights event, this value is the average number of API calls per minute over the duration of the unusual activity.

Since: 1.07

Optional: False

- **insightDuration** - The duration, in minutes, of an Insights event (the time period from the start to the end of unusual activity on the subject API). `insightDuration` occurs in both starting and ending Insights events.

Since: 1.07

Optional: False

- **baselineDuration** - The duration, in minutes, of the baseline period (the time period that normal activity is measured on the subject API). `baselineDuration` is at minimum the seven

days (10080 minutes) preceding an Insights event. This field occurs in both starting and ending Insights events. The ending time of `baselineDuration` measurement is always the start of an Insights event.

Since: 1.07

Optional: False

- **attributions** - This block includes information about the user identities, user agents, and error codes correlated with unusual and baseline activity. A maximum of five user identities, five user agents, and five error codes are captured in an Insights event `attributions` block, sorted by an average of the count of activity, in descending order from highest to lowest.

Since: 1.07

Optional: True

- **attribute** - Contains the attribute type. Value can be `userIdentityArn`, `userAgent`, or `errorCode`.
- **userIdentityArn** - A block that shows up to the top five AWS users or IAM roles that contributed to API calls during the unusual activity and baseline periods. See also `userIdentity` in [CloudTrail record contents \(p. 272\)](#).

Since: 1.07

Optional: False

- **insight** - A block that shows up to the top five user identity ARNs that contributed to the API calls made during the unusual activity period, in descending order from largest number of API calls to smallest. It also shows the average number of API calls made by the user identities during the unusual activity period.

Since: 1.07

Optional: False

- **value** - The ARN of one of the top five user identities that contributed to the API calls made during the unusual activity period.

Since: 1.07

Optional: False

- **average** - The number of API calls per minute during the unusual activity period for the user identity in the `value` field.

Since: 1.07

Optional: False

- **baseline** - A block that shows up to the top five user identity ARNs that contributed the most to the API calls made during the normal activity period. It also shows the average number of API calls made by the user identities during the normal activity period.

Since: 1.07

Optional: False

- **value** - The ARN of one of the top five user identities that contributed to the API calls made during the normal activity period.

Since: 1.07

Optional: False

- **average** - The historic average of API calls per minute during the seven days preceding the Insights activity start time for the user identity in the `value` field.

Since: 1.07

Optional: False

- **userAgent** - A block that shows up to the top five AWS tools by which the user identity contributed to API calls during the unusual activity and baseline periods. These tools include the AWS Management Console, AWS CLI, or the AWS SDKs. See also `userAgent` in [CloudTrail record contents \(p. 272\)](#).

Since: 1.07

Optional: False

- **insight** - A block that shows up to the top five user agents that contributed to the API calls made during the unusual activity period, in descending order from largest number of API calls to smallest. It also shows the average number of API calls made by the user agents during the unusual activity period.

Since: 1.07

Optional: False

- **value** - One of the top five user agents that contributed to the API calls made during the unusual activity period.

Since: 1.07

Optional: False

- **average** - The number of API calls per minute during the unusual activity period for the user agent in the `value` field.

Since: 1.07

Optional: False

- **baseline** - A block that shows up to the top five user agents that contributed the most to the API calls made during the normal activity period. It also shows the average number of API calls made by the user agents during the normal activity period.

Since: 1.07

Optional: False

- **value** - One of the top five user agents that contributed to the API calls made during the normal activity period.

Since: 1.07

Optional: False

- **average** - The historic average of API calls per minute during the seven days preceding the Insights activity start time for the user agent in the `value` field.

Since: 1.07

Optional: False

- **errorCode** - A block that shows up to the top five error codes that occurred on API calls during the unusual activity and baseline periods, in descending order from largest number of API calls to smallest. See also `errorCode` in [CloudTrail record contents \(p. 272\)](#).

Since: 1.07

Optional: False

- **insight** - A block that shows up to the top five error codes that occurred on the API calls made during the unusual activity period, in descending order from largest number of associated API calls to smallest. It also shows the average number of API calls on which the errors occurred during the unusual activity period.

Since: 1.07

Optional: False

- **value** - One of the top five error codes that occurred on the API calls made during the unusual activity period, such as `AccessDeniedException`.

If none of the calls that triggered the Insights event resulted in errors, this value is `null`.

Since: 1.07

Optional: False

- **average** - The number of API calls per minute during the unusual activity period for the error code in the `value` field.

If the error code value is `null`, and there are no other error codes in the `insight` block, the value of the `average` is the same as that in the `statistics` block for the Insights event overall.

Since: 1.07

Optional: False

- **baseline** - A block that shows up to the top five error codes that occurred on the API calls made during the normal activity period. It also shows the average number of API calls made by the user agents during the normal activity period.

Since: 1.07

Optional: False

- **value** - One of the top five error codes that occurred on the API calls made during the normal activity period, such as `AccessDeniedException`.

Since: 1.07

Optional: False

- **average** - The historic average of API calls per minute during the seven days preceding the Insights activity start time for the error code in the `value` field.

Since: 1.07

Optional: False

Example insightDetails block

The following is an example of an Insights event `insightDetails` block for an Insights event that occurred when the Application Auto Scaling API `CompleteLifecycleAction` was called an unusual number of times. For an example of a full Insights event, see [CloudTrail log event reference \(p. 270\)](#).

This example is from a starting Insights event, indicated by "state": "Start". The top user identities that called the APIs associated with the Insights event, CodeDeployRole1, CodeDeployRole2, and CodeDeployRole3, are shown in the attributions block, along with their average API call rates for this Insights event, and the baseline for the CodeDeployRole1 role. The attributions block also shows that the user agent is codedeploy.amazonaws.com, meaning the top user identities used the AWS CodeDeploy console to run the API calls.

Because there are no error codes associated with the events that were analyzed to generate the Insights event (the value is null), the insight average for the error code is the same as the overall insight average for the entire Insights event, shown in the statistics block.

```
"insightDetails": {
  "state": "Start",
  "eventSource": "autoscaling.amazonaws.com",
  "eventName": "CompleteLifecycleAction",
  "insightType": "ApiCallRateInsight",
  "insightContext": {
    "statistics": {
      "baseline": {
        "average": 0.0000882145
      },
      "insight": {
        "average": 0.6
      },
      "insightDuration": 5,
      "baselineDuration": 11336
    },
    "attributions": [
      {
        "attribute": "userIdentityArn",
        "insight": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole2",
            "average": 0.2
          },
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole3",
            "average": 0.2
          }
        ],
        "baseline": [
          {
            "value": "arn:aws:sts::012345678901:assumed-role/CodeDeployRole1",
            "average": 0.0000882145
          }
        ]
      },
      {
        "attribute": "userAgent",
        "insight": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.6
          }
        ],
        "baseline": [
          {
            "value": "codedeploy.amazonaws.com",
            "average": 0.0000882145
          }
        ]
      }
    ]
  }
}
```

```
    ],  
    },  
    {  
      "attribute": "errorCode",  
      "insight": [  
        {  
          "value": "null",  
          "average": 0.6  
        }  
      ],  
      "baseline": [  
        {  
          "value": "null",  
          "average": 0.0000882145  
        }  
      ]  
    }  
  ]  
}  
}
```

Non-API events captured by CloudTrail

In addition to logging AWS API calls, CloudTrail 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 service events \(p. 294\)](#)
- [AWS Management Console sign-in events \(p. 295\)](#)

AWS service events

CloudTrail supports logging non-API service events. These events are created by AWS services but are not directly triggered by a request to a public AWS API. For these events, the `eventType` field is `AwsServiceEvent`.

The following is an example scenario of an AWS service event when a customer managed key is automatically rotated in AWS Key Management Service (AWS KMS). For more information about rotating KMS keys, see [Rotating KMS keys](#).

```
{  
  "eventVersion": "1.05",  
  "userIdentity": {  
    "accountId": "123456789012",  
    "invokedBy": "AWS Internal"  
  },  
  "eventTime": "2019-06-02T00:06:08Z",  
  "eventSource": "kms.amazonaws.com",  
  "eventName": "RotateKey",  
  "awsRegion": "us-east-2",  
  "sourceIPAddress": "AWS Internal",  
  "userAgent": "AWS Internal",  
  "requestParameters": null,  
  "responseElements": null,  
  "eventID": "234f004b-EXAMPLE",  
  "readOnly": false,  
  "resources": [  
    {  
      "arn": "arn:aws:kms:us-east-2:123456789012:key/12345678-1234-1234-1234-123456789012",  
      "type": "AWS::KMS::Key"  
    }  
  ]  
}
```

```
        "ARN": "arn:aws:kms:us-east-2:123456789012:key/7944f0ec-EXAMPLE",  
        "accountId": "123456789012",  
        "type": "AWS::KMS::Key"  
    },  
    ],  
    "eventType": "AwsServiceEvent",  
    "recipientAccountId": "123456789012",  
    "serviceEventDetails": {  
        "keyId": "7944f0ec-EXAMPLE"  
    }  
}
```

AWS Management Console sign-in events

CloudTrail logs attempts to sign into the AWS Management Console, the AWS Discussion Forums, and the AWS Support Center. All IAM user and root user sign-in events, as well as all federated user sign-in events, generate records in CloudTrail log files. AWS Management Console sign-in events are global service events.

Topics

- [Example records for IAM users \(p. 295\)](#)
- [Example event records for root users \(p. 297\)](#)

Example records for IAM users

The following examples show event records for several IAM user sign-in scenarios.

Topics

- [IAM user, successful sign-in \(p. 295\)](#)
- [IAM user, signing in with MFA \(p. 296\)](#)
- [IAM user, unsuccessful sign-in \(p. 296\)](#)
- [IAM user, sign-in process checks for MFA \(p. 297\)](#)

IAM user, successful sign-in

The following record shows that an IAM user named Anaya successfully signed into the AWS Management Console without using multi-factor authentication.

```
{  
  "Records": [  
    {  
      "eventVersion": "1.05",  
      "userIdentity": {  
        "type": "IAMUser",  
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",  
        "arn": "arn:aws:iam::111122223333:user/anaya",  
        "accountId": "111122223333",  
        "userName": "anaya"  
      },  
      "eventTime": "2018-08-29T16:24:34Z",  
      "eventSource": "signin.amazonaws.com",  
      "eventName": "ConsoleLogin",  
      "awsRegion": "us-east-2",  
      "sourceIPAddress": "192.0.2.0",  
      "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:62.0) Gecko/20100101  
Firefox/62.0",  
      "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"
  }
]
```

IAM user, signing in with MFA

The following record shows that an IAM user named Anaya logged into the AWS Management Console using multi-factor authentication (MFA).

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::111122223333:user/anaya",
    "accountId": "111122223333",
    "userName": "anaya"
  },
  "eventTime": "2018-07-24T18:34:07Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "Yes"
  },
  "eventID": "fed06f42-cb12-4764-8c69-121063dc79b9",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "111122223333"
}
```

IAM user, unsuccessful sign-in

The following record shows an IAM user's unsuccessful sign-in attempt.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "accountId": "111122223333",
    "accessKeyId": "",
    "userName": "anaya"
  },
  "eventTime": "2018-07-24T18:34:07Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "Yes"
  },
  "eventID": "fed06f42-cb12-4764-8c69-121063dc79b9",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "111122223333"
}
```

```
{
  "eventTime": "2018-07-24T18:32:11Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "Yes"
  },
  "eventID": "d38ce1b3-4575-4cb8-a632-611b8243bfc3",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "111122223333"
}
```

IAM user, sign-in process checks for MFA

The following shows that the sign-process checked whether multi-factor authentication (MFA) is required for an IAM user during sign-in.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "accountId": "111122223333",
    "accessKeyId": "",
    "userName": "anaya"
  },
  "eventTime": "2018-07-24T18:33:45Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "CheckMfa",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36",
  "requestParameters": null,
  "responseElements": {
    "CheckMfa": "Success"
  },
  "additionalEventData": {
    "MfaType": "U2F MFA"
  },
  "eventID": "f8ef8fc5-e3e8-4ee1-9d52-2f412ddf17e3",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "111122223333"
}
```

Example event records for root users

The following examples show event records for several root user sign-in scenarios.

Topics

- [Root user, successful sign-in \(p. 298\)](#)
- [Root user, unsuccessful sign-in \(p. 298\)](#)
- [Root user, MFA changed \(p. 299\)](#)
- [Root user, password changed \(p. 299\)](#)

Root user, successful sign-in

The following shows a successful sign-in event for a root user not using MFA.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "Root",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::111122223333:root",
    "accountId": "111122223333",
    "accessKeyId": ""
  },
  "eventTime": "2018-08-29T16:24:34Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:62.0) Gecko/20100101 Firefox/62.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "deb1e1f9-c99b-4612-8e9f-21f93b5d79c0",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "111122223333"
}
```

Root user, unsuccessful sign-in

The following shows an unsuccessful sign-in event for a root user not using MFA.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "Root",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::111122223333:root",
    "accountId": "111122223333",
    "accessKeyId": ""
  },
  "eventTime": "2018-08-25T18:10:29Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36",
  "errorMessage": "Failed authentication",
}
```

```
"requestParameters": null,
"responseElements": {
  "ConsoleLogin": "Failure"
},
"additionalEventData": {
  "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs%23&isauthcode=true",
  "MobileVersion": "No",
  "MFAUsed": "No"
},
"eventID": "a4fbbbe77-91a0-4238-804a-64314184edb6",
"eventType": "AwsConsoleSignIn",
"recipientAccountId": "11112223333"
}
```

Root user, MFA changed

The following shows an example event for a root user changing multi-factor authentication (MFA) settings.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "Root",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::11112223333:root",
    "accountId": "11112223333",
    "accessKeyId": "EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {},
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2020-10-13T21:05:40Z"
      }
    }
  },
  "eventTime": "2020-10-13T21:19:09Z",
  "eventSource": "iam.amazonaws.com",
  "eventName": "EnableMFADevice",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Coral/Netty4",
  "requestParameters": {
    "userName": "AWS ROOT USER",
    "serialNumber": "arn:aws:iam::11112223333:mfa/root-account-mfa-device"
  },
  "responseElements": null,
  "requestID": "EXAMPLE4-2cf7-4a44-af00-f61f0EXAMPLE",
  "eventID": "EXAMPLEeb-7dae-48cb-895f-20e86EXAMPLE",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "11112223333"
}
```

Root user, password changed

The following shows an example event for a root user changing their password.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
```



```
    "type": "Root",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::111122223333:root",
    "accountId": "111122223333",
    "accessKeyId": ""
  },
  "eventTime": "2020-10-13T21:47:13Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "PasswordUpdated",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101
Firefox/78.0",
  "requestParameters": null,
  "responseElements": {
    "PasswordUpdated": "Success"
  },
  "eventID": "EXAMPLEd-244c-4044-abbf-21c64EXAMPLE",
  "eventType": "AwsConsoleSignIn",
  "recipientAccountId": "111122223333"
}
```

Document history

The following table describes the important changes to the documentation for AWS CloudTrail. For notification about updates to this documentation, you can subscribe to an RSS feed.

- **API version:** 2013-11-01
- **Latest documentation update:** 2021-10-05

| update-history-change | update-history-description | update-history-date |
|--|--|---------------------|
| Added functionality (p. 301) | You can now log CloudTrail data events on DynamoDB streams by using advanced event selectors. For more information, see Logging data events for trails . | September 22, 2021 |
| Added functionality (p. 301) | You can now log data events on Amazon S3 access points. You can log Amazon S3 access point data events by using advanced event selectors. For more information, see Logging data events for trails . | August 24, 2021 |
| Changed functionality (p. 301) | When you configure a trail to send notifications to Amazon SNS, CloudTrail adds a policy statement to your SNS topic access policy that allows CloudTrail to send content to an SNS topic. As a security best practice, we recommend adding an <code>aws:SourceArn</code> or <code>aws:SourceAccount</code> condition key to the CloudTrail policy statement. For more information, see Amazon SNS topic policy for CloudTrail . | August 16, 2021 |
| Added service support (p. 301) | This release supports Amazon Route 53 Application Recovery Controller. See AWS CloudTrail Supported Services and Integrations . | July 27, 2021 |
| Added functionality (p. 301) | You can now log data events on Amazon EBS direct APIs run on EBS snapshots. You can log Amazon EBS direct API data events by using advanced event selectors. For more information, see Logging data events for trails . | July 27, 2021 |

| | | |
|--|--|----------------|
| Changed functionality (p. 301) | When CloudTrail processes data events, it preserves numbers in their original format, whether that is an integer (<code>int</code>) or a <code>float</code> . In events that have integers in the fields of a data event, CloudTrail historically processed these numbers as floats. Now, CloudTrail keeps the original format of integers in data events. For more information, see Using the CloudTrail Processing Library . | July 13, 2021 |
| Added functionality (p. 301) | You can now exclude Amazon RDS Data API management events from your trails. For more information, see Logging management events for trails . | July 1, 2021 |
| Added service support (p. 301) | This release supports AWS BugBust. See AWS CloudTrail Supported Services and Integrations . | June 24, 2021 |
| Added service support (p. 301) | This release supports Amazon Managed Grafana and Amazon Managed Service for Prometheus. See AWS CloudTrail Supported Services and Integrations . | June 2, 2021 |
| Added service support (p. 301) | This release supports AWS App Runner. See AWS CloudTrail Supported Services and Integrations . | May 18, 2021 |
| Added service support (p. 301) | This release supports AWS Systems Manager Incident Manager. See AWS CloudTrail Supported Services and Integrations . | May 10, 2021 |
| Updated documentation (p. 301) | This update describes data event logging requirements for AWS Config conformance packs, especially for compliance frameworks such as HIPAA or FedRAMP. For more information, see Logging data events for trails . | May 7, 2021 |
| Added service support (p. 301) | This release supports Service Quotas and Amazon EBS direct APIs. See AWS CloudTrail Supported Services and Integrations . | April 13, 2021 |

| | | |
|--|---|----------------|
| Added functionality (p. 301) | After an IAM administrator configures AWS STS , CloudTrail logs <code>sourceIdentity</code> information in events when users assume an IAM role, or perform any actions with the assumed role. For more information, see CloudTrail <code>userIdentity</code> Element . | April 13, 2021 |
| Updated documentation (p. 301) | This update documents limits, in kilobytes (KB), for content in some CloudTrail event record fields. For more information, see CloudTrail record contents . | April 8, 2021 |
| Added functionality (p. 301) | After an IAM administrator configures AWS STS , CloudTrail logs <code>sourceIdentity</code> information in events when users assume an IAM role, or perform any actions with the assumed role. For more information, see CloudTrail <code>userIdentity</code> Element . | April 6, 2021 |
| Added functionality (p. 301) | You can now log data events on Amazon DynamoDB tables. You can log DynamoDB data events by using either event selectors or advanced event selectors. For more information, see Logging data events for trails . | March 23, 2021 |
| Added service support (p. 301) | This release supports Amazon Managed Workflows for Apache Airflow. See AWS CloudTrail Supported Services and Integrations . | March 22, 2021 |
| Added functionality (p. 301) | You can now log data events on S3 Object Lambda access points if you have opted in to use advanced event selectors. For more information, see Logging data events for trails . | March 18, 2021 |
| Added service support (p. 301) | This release supports AWS Fault Injection Simulator. See AWS CloudTrail Supported Services and Integrations . | March 15, 2021 |

| | | |
|--|--|-------------------|
| Added functionality (p. 301) | You can now log data events on Ethereum nodes in Amazon Managed Blockchain if you have opted in to use advanced event selectors. For more information, see Logging data events for trails . | March 1, 2021 |
| Added service support (p. 301) | This release supports Amazon Managed Blockchain and the preview of Ethereum for Managed Blockchain. See AWS CloudTrail Supported Services and Integrations . | February 4, 2021 |
| Added service support (p. 301) | This release supports AWS Amplify. See AWS CloudTrail Supported Services and Integrations . | February 3, 2021 |
| Added service support (p. 301) | This release supports Amazon Lookout for Metrics. See AWS CloudTrail Supported Services and Integrations . | February 1, 2021 |
| Updated documentation (p. 301) | This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.4.0.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub. | January 12, 2021 |
| Added functionality (p. 301) | You can now log data events on Amazon S3 on AWS Outposts. For more information, see Logging data events for trails . | December 21, 2020 |
| Added service support (p. 301) | This release supports Amazon Lookout for Equipment, AWS Well-Architected Tool, and Amazon Location Service. See AWS CloudTrail Supported Services and Integrations . | December 16, 2020 |
| Added service support (p. 301) | This release supports AWS IoT Greengrass V2. See AWS CloudTrail Supported Services and Integrations . | December 15, 2020 |
| Added service support (p. 301) | This release supports Amazon EMR on EKS. See AWS CloudTrail Supported Services and Integrations . | December 10, 2020 |

| | | |
|--|---|--------------------|
| Added service support (p. 301) | This release supports AWS Audit Manager and Amazon HealthLake. See AWS CloudTrail Supported Services and Integrations . | December 8, 2020 |
| Added service support (p. 301) | This release supports Amazon Lookout for Vision. See AWS CloudTrail Supported Services and Integrations . | December 1, 2020 |
| Added functionality (p. 301) | The AWS CloudTrail event version is now 1.08. Version 1.08 introduces new fields for CloudTrail. For more information, see CloudTrail record contents . | November 24, 2020 |
| Added functionality (p. 301) | AWS CloudTrail introduces advanced event selectors for data events. Advanced event selectors allow finer-grained control over the data events that you log to your trail. You can include or exclude data events for specific AWS resources, and select specific APIs on those resources to log to your trail. For more information, see Logging data events for trails . | November 24, 2020 |
| Added service support (p. 301) | This release supports AWS Network Firewall. See AWS CloudTrail Supported Services and Integrations . | November 17, 2020 |
| Added service support (p. 301) | This release supports AWS Trusted Advisor. See AWS CloudTrail Supported Services and Integrations . | October 22, 2020 |
| Updated documentation (p. 301) | Added two new examples of event records for root user sign-in events. For more information, see AWS Console sign-in events . | October 13, 2020 |
| Changed functionality (p. 301) | Permissions in the <code>AWSCloudTrail_FullAccess</code> policy have been narrowed. This policy no longer allows you to delete Amazon SNS topics or Amazon S3 buckets, and the <code>getObject</code> action has been removed. For more information, see Granting custom permissions for CloudTrail users . | September 29, 2020 |

| | | |
|--|---|-----------------|
| Updated documentation (p. 301) | This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.3.0.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub. | August 28, 2020 |
| Added service support (p. 301) | This release supports AWS Outposts. See AWS CloudTrail Supported Services and Integrations . | August 28, 2020 |
| Added functionality (p. 301) | AWS CloudTrail Insights introduces attribution fields for CloudTrail Insights events. Attribution fields show the top user identities, user agents, and error codes that are associated with the anomalous activity that triggers Insights events. For comparison, attribution fields also show the top user identities, user agents, and error codes associated with normal, or baseline, activity. For more information, see Logging Insights events for trails . | August 13, 2020 |
| Added functionality (p. 301) | The AWS CloudTrail console has a new look that's designed to make it easier to use. The AWS CloudTrail User Guide has been updated with changes to procedures for how to perform tasks in the console, such as creating trails, updating trails, and downloading event history. | August 13, 2020 |
| Added service support (p. 301) | This release supports Amazon Interactive Video Service. See AWS CloudTrail Supported Services and Integrations . | July 15, 2020 |
| Added service support (p. 301) | This release supports Amazon Honeycode. See AWS CloudTrail Supported Services and Integrations . | June 24, 2020 |
| Added service support (p. 301) | This release supports Amazon Macie. See AWS CloudTrail Supported Services and Integrations . | May 19, 2020 |

| | | |
|---|--|-------------------|
| Added service support (p. 301) | This release supports Amazon Kendra. See AWS CloudTrail Supported Services and Integrations . | May 13, 2020 |
| Added service support (p. 301) | This release supports AWS IoT SiteWise. See AWS CloudTrail Supported Services and Integrations . | April 29, 2020 |
| Added region support (p. 301) | This release supports an additional region: Europe (Milan). See AWS CloudTrail Supported Regions . | April 28, 2020 |
| Added service and region support (p. 301) | This release supports Amazon AppFlow. See AWS CloudTrail Supported Services and Integrations . Support has also been added for the Africa (Cape Town) Region. See AWS CloudTrail Supported Regions . | April 22, 2020 |
| Added functionality (p. 301) | High-volume AWS KMS actions such as <code>Encrypt</code> , <code>Decrypt</code> , and <code>GenerateDataKey</code> are now logged as Read events. If you choose to log all AWS KMS events on your trail, and also choose to log Write management events, your trail logs relevant AWS KMS actions like <code>Disable</code> , <code>Delete</code> and <code>ScheduleKey</code> . | April 7, 2020 |
| Added service support (p. 301) | This release supports Amazon CodeGuru Reviewer. See AWS CloudTrail Supported Services and Integrations . | February 7, 2020 |
| Added service support (p. 301) | This release supports Amazon Managed Apache Cassandra Service. See AWS CloudTrail Supported Services and Integrations . | January 17, 2020 |
| Added service support (p. 301) | This release supports Amazon Connect. See AWS CloudTrail Supported Services and Integrations . | December 13, 2019 |

| | | |
|--|--|-------------------|
| Updated documentation (p. 301) | This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.2.0.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub. | November 21, 2019 |
| Added functionality (p. 301) | This release supports AWS CloudTrail Insights for helping you detect unusual activity in your account. See Logging Insights events for Trails . | November 20, 2019 |
| Added functionality (p. 301) | This release adds an option for filtering AWS Key Management Service events out of a trail. See Creating a Trail . | November 20, 2019 |
| Added service support (p. 301) | This release supports AWS CodeStar Notifications. See AWS CloudTrail Supported Services and Integrations . | November 7, 2019 |
| Added functionality (p. 301) | This release supports adding tags when you create a trail in CloudTrail, whether you use the CloudTrail console or API. This release adds two new APIs, <code>GetTrail</code> and <code>ListTrails</code> . | November 1, 2019 |
| Added service support (p. 301) | This release supports AWS App Mesh. See AWS CloudTrail Supported Services and Integrations . | October 17, 2019 |
| Added service support (p. 301) | This release supports Amazon Translate. See AWS CloudTrail Supported Services and Integrations . | October 17, 2019 |
| Documentation update (p. 301) | The Unsupported Services topic has been restored and updated to include only those AWS services that do not currently log events in CloudTrail. See CloudTrail Unsupported Services . | October 7, 2019 |

| | | |
|--|---|--------------------|
| Documentation update (p. 301) | The documentation has been updated with changes to the <code>AWSCloudTrailFullAccess</code> policy. A policy example that shows equivalent permissions to <code>AWSCloudTrailFullAccess</code> has been updated to restrict the resources on which the <code>iam:PassRole</code> action can act to those matching the following condition statement: <code>"iam:PassedToService": "cloudtrail.amazonaws.com"</code> . See AWS CloudTrail Identity-Based Policy Examples . | September 24, 2019 |
| Documentation update (p. 301) | The documentation has been updated with a new topic, Managing CloudTrail Costs , to help you get the log data you need out of CloudTrail while staying within a budget. | September 3, 2019 |
| Added service support (p. 301) | This release supports AWS Control Tower. See AWS CloudTrail Supported Services and Integrations . | August 13, 2019 |
| Added region support (p. 301) | This release supports an additional region: Middle East (Bahrain). See AWS CloudTrail Supported Regions . | July 29, 2019 |
| Documentation update (p. 301) | The documentation has been updated with information about security for CloudTrail. See Security in AWS CloudTrail . | July 3, 2019 |
| Added service support (p. 301) | This release supports AWS Ground Station. See AWS CloudTrail Supported Services and Integrations . | June 6, 2019 |
| Added service support (p. 301) | This release supports AWS IoT Things Graph. See AWS CloudTrail Supported Services and Integrations . | June 4, 2019 |
| Added service support (p. 301) | This release supports Amazon AppStream 2.0. See AWS CloudTrail Supported Services and Integrations . | April 25, 2019 |
| Added region support (p. 301) | This release supports an additional region: Asia Pacific (Hong Kong). See AWS CloudTrail Supported Regions . | April 24, 2019 |

| | | |
|--------------------------------|--|-------------------|
| Added service support (p. 301) | This release supports Amazon Kinesis Data Analytics. See AWS CloudTrail Supported Services and Integrations . | March 22, 2019 |
| Added service support (p. 301) | This release supports AWS Backup. See AWS CloudTrail Supported Services and Integrations . | February 4, 2019 |
| Added service support (p. 301) | This release supports Amazon WorkLink. See AWS CloudTrail Supported Services and Integrations . | January 23, 2019 |
| Added service support (p. 301) | This release supports AWS Cloud9. See AWS CloudTrail Supported Services and Integrations . | January 21, 2019 |
| Added service support (p. 301) | This release supports AWS Elemental MediaLive. See AWS CloudTrail Supported Services and Integrations . | January 19, 2019 |
| Added service support (p. 301) | This release supports Amazon Comprehend. See AWS CloudTrail Supported Services and Integrations . | January 18, 2019 |
| Added service support (p. 301) | This release supports AWS Elemental MediaPackage. See AWS CloudTrail Supported Services and Integrations . | December 21, 2018 |
| Added region support (p. 301) | This release supports an additional region: EU (Stockholm). See AWS CloudTrail Supported Regions . | December 11, 2018 |
| Documentation update (p. 301) | The documentation has been updated with information about supported and unsupported services. See AWS CloudTrail Supported Services and Integrations . | December 3, 2018 |
| Added service support (p. 301) | This release supports AWS Resource Access Manager (AWS RAM). See AWS CloudTrail Supported Services and Integrations . | November 20, 2018 |
| Updated functionality (p. 301) | This release supports creating a trail in CloudTrail that logs events for all AWS accounts in an organization in AWS Organizations. See Creating a Trail for an Organization . | November 19, 2018 |

| | | |
|---|--|-------------------|
| Added service support (p. 301) | This release supports Amazon Pinpoint SMS and Voice API. See AWS CloudTrail Supported Services and Integrations . | November 16, 2018 |
| Added service support (p. 301) | This release supports AWS IoT Greengrass. See AWS CloudTrail Supported Services and Integrations . | October 29, 2018 |
| Updated documentation (p. 301) | This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.1.3.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub. | October 18, 2018 |
| Added functionality (p. 301) | This release supports using additional filters in Event history . See Viewing CloudTrail Events in the CloudTrail Console . | October 18, 2018 |
| Added functionality (p. 301) | This release supports using Amazon Virtual Private Cloud (Amazon VPC) to establish a private connection between your VPC and AWS CloudTrail. See Using AWS CloudTrail with Interface VPC Endpoints . | August 9, 2018 |
| Added service support (p. 301) | This release supports Amazon Data Lifecycle Manager. See AWS CloudTrail Supported Services and Integrations . | July 24, 2018 |
| Added service support (p. 301) | This release supports Amazon MQ. See AWS CloudTrail Supported Services and Integrations . | July 19, 2018 |
| Added service support (p. 301) | This release supports AWS Mobile CLI. See AWS CloudTrail Supported Services and Integrations . | June 29, 2018 |
| AWS CloudTrail documentation history notification available through RSS feed (p. 301) | You can now receive notification about updates to the AWS CloudTrail documentation by subscribing to an RSS feed. | June 29, 2018 |

Earlier updates

The following table describes the documentation release history of AWS CloudTrail prior to June 29, 2018.

| Change | Description | Release Date |
|-----------------------|---|----------------|
| Added service support | This release supports Amazon RDS Performance Insights. For more information, see CloudTrail Supported Services and Integrations . | June 21, 2018 |
| Added functionality | This release supports logging all CloudTrail management events in Event history. For more information, see Viewing events with CloudTrail Event history (p. 44) . | June 14, 2018 |
| Added service support | This release supports AWS Billing and Cost Management. See CloudTrail supported services and integrations (p. 20) . | June 7, 2018 |
| Added service support | This release supports Amazon Elastic Container Service for Kubernetes (Amazon EKS). See CloudTrail supported services and integrations (p. 20) . | June 5, 2018 |
| Updated documentation | <p>This update supports the following patch release for the CloudTrail Processing Library:</p> <ul style="list-style-type: none"> Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.1.2.jar. <p>For more information, see Using the CloudTrail Processing Library (p. 215) and the CloudTrail Processing Library on GitHub.</p> | May 16, 2018 |
| Added service support | This release supports AWS Billing and Cost Management. See CloudTrail supported services and integrations (p. 20) . | June 7, 2018 |
| Added service support | This release supports Amazon Elastic Container Service for Kubernetes (Amazon EKS). See CloudTrail supported services and integrations (p. 20) . | June 5, 2018 |
| Updated documentation | <p>This update supports the following patch release for the CloudTrail Processing Library:</p> <ul style="list-style-type: none"> Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.1.2.jar. <p>For more information, see Using the CloudTrail Processing Library (p. 215) and the CloudTrail Processing Library on GitHub.</p> | May 16, 2018 |
| Added service support | This release supports AWS X-Ray. See CloudTrail supported services and integrations (p. 20) . | April 25, 2018 |

| Change | Description | Release Date |
|---------------------------------------|--|-------------------|
| Added service support | This release supports AWS IoT Analytics. See CloudTrail supported services and integrations (p. 20) . | April 23, 2018 |
| Added service support | This release supports Secrets Manager. See CloudTrail supported services and integrations (p. 20) . | April 10, 2018 |
| Added service support | This release supports Amazon Rekognition. See CloudTrail supported services and integrations (p. 20) . | April 6, 2018 |
| Added service support | This release supports AWS Private Certificate Authority (PCA). See CloudTrail supported services and integrations (p. 20) . | April 4, 2018 |
| Added functionality | This release supports making it easier to search CloudTrail log files with Amazon Athena. You can automatically create tables for querying logs directly from the CloudTrail console, and use those tables to run queries in Athena. For more information, see CloudTrail supported services and integrations (p. 20) and Creating a Table for CloudTrail Logs in the CloudTrail Console . | March 15, 2018 |
| Added service support | This release supports AWS AppSync. See CloudTrail supported services and integrations (p. 20) . | February 13, 2018 |
| Added region support | This release supports an additional region: Asia Pacific (Osaka) (ap-northeast-3). See CloudTrail supported regions (p. 12) . | February 12, 2018 |
| Added service support | This release supports AWS Shield. See CloudTrail supported services and integrations (p. 20) . | February 12, 2018 |
| Added service support | This release supports Amazon SageMaker. See CloudTrail supported services and integrations (p. 20) . | January 11, 2018 |
| Added service support | This release supports AWS Batch. See CloudTrail supported services and integrations (p. 20) . | January 10, 2018 |
| Added functionality | This release supports extending the amount of account activity that is available in CloudTrail event history to 90 days. You can also customize the display of columns to improve the view of your CloudTrail events. For more information, see Viewing events with CloudTrail Event history (p. 44) . | December 12, 2017 |
| Added service support | This release supports Amazon WorkMail. See CloudTrail supported services and integrations (p. 20) . | December 12, 2017 |
| Added service support | This release supports Alexa for Business, AWS Elemental MediaConvert, and AWS Elemental MediaStore. See CloudTrail supported services and integrations (p. 20) . | December 1, 2017 |
| Added functionality and documentation | This release supports logging data events for AWS Lambda functions. For more information, see Logging data events for trails (p. 140) . | November 30, 2017 |

| Change | Description | Release Date |
|---------------------------------------|---|--------------------|
| Added functionality and documentation | This release supports logging data events for AWS Lambda functions. For more information, see Logging data events for trails (p. 140) . | November 30, 2017 |
| Added functionality and documentation | This release supports the following updates to the CloudTrail Processing Library: <ul style="list-style-type: none"> • Add support for Boolean identification of management events. • Update the CloudTrail event version to 1.06. For more information, see Using the CloudTrail Processing Library (p. 215) and the CloudTrail Processing Library on GitHub . | November 30, 2017 |
| Added service support | This release supports AWS Glue. See CloudTrail supported services and integrations (p. 20) . | November 7, 2017 |
| New documentation | This release adds a new topic, Quotas in AWS CloudTrail (p. 32) . | October 19, 2017 |
| Updated documentation | This release updates the documentation of APIs supported in CloudTrail event history for Amazon Athena, AWS CodeBuild, Amazon Elastic Container Registry, and AWS Migration Hub. | October 13, 2017 |
| Added service support | This release supports Amazon Chime. See CloudTrail supported services and integrations (p. 20) . | September 27, 2017 |
| Added functionality and documentation | This release supports configuring data event logging for all Amazon S3 buckets in your AWS account. See Logging data events for trails (p. 140) . | September 20, 2017 |
| Added service support | This release supports Amazon Lex. See CloudTrail supported services and integrations (p. 20) . | August 15, 2017 |
| Added service support | This release supports AWS Migration Hub. See CloudTrail supported services and integrations (p. 20) . | August 14, 2017 |
| Added functionality and documentation | This release supports CloudTrail being enabled by default for all AWS accounts. The past seven days of account activity are available in CloudTrail event history, and the most recent events appear on the console dashboard. The feature formerly known as API activity history has been replaced by Event history . For more information, see How CloudTrail works (p. 1) . | August 14, 2017 |
| Added functionality and documentation | This release supports downloading events from the CloudTrail console on the API activity history page. You can download events in JSON or CSV format. For more information, see Downloading events (p. 48) . | July 27, 2017 |

| Change | Description | Release Date |
|---------------------------------------|---|---------------|
| Added functionality | <p>This release supports logging Amazon S3 object level API operations in two additional regions, Europe (London) and Canada (Central).</p> <p>For more information, see Working with CloudTrail log files (p. 133).</p> | July 19, 2017 |
| Added service support | <p>This release supports looking up APIs for Amazon CloudWatch Events in the CloudTrail API activity history feature.</p> | June 27, 2017 |
| Added functionality and documentation | <p>This release supports additional APIs in the CloudTrail API activity history feature for the following services:</p> <ul style="list-style-type: none">• AWS CloudHSM• Amazon Cognito• Amazon DynamoDB• Amazon EC2• Kinesis• AWS Storage Gateway | June 27, 2017 |
| Added service support | <p>This release supports AWS CodeStar. See CloudTrail supported services and integrations (p. 20).</p> | June 14, 2017 |
| Added functionality and documentation | <p>This release supports the following updates to the CloudTrail Processing Library:</p> <ul style="list-style-type: none">• Add support for different formats for SQS messages from the same SQS queue to identify CloudTrail log files. The following formats are supported:<ul style="list-style-type: none">• Notifications that CloudTrail sends to an SNS topic• Notifications that Amazon S3 sends to an SNS topic• Notifications that Amazon S3 sends directly to an SQS queue• Add support for the <code>deleteMessageUponFailure</code> property, which you can use to delete messages that can't be processed. <p>For more information, see Using the CloudTrail Processing Library (p. 215) and the CloudTrail Processing Library on GitHub.</p> | June 1, 2017 |
| Added service support | <p>This release supports Amazon Athena. See CloudTrail supported services and integrations (p. 20).</p> | May 19, 2017 |

| Change | Description | Release Date |
|---------------------------------------|--|-------------------|
| Added functionality | <p>This release supports sending data events to Amazon CloudWatch Logs.</p> <p>For more information about configuring your trail to log data events, see Data events (p. 140).</p> <p>For more information about sending events to CloudWatch Logs, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs (p. 159).</p> | May 9, 2017 |
| Added service support | <p>This release supports the AWS Marketplace Metering Service. See CloudTrail supported services and integrations (p. 20).</p> | May 2, 2017 |
| Added service support | <p>This release supports Amazon QuickSight. See CloudTrail supported services and integrations (p. 20).</p> | April 28, 2017 |
| Added functionality and documentation | <p>This release supports an updated console experience for creating new trails. You can now configure a new trail to log management and data events. For more information, see Creating a trail (p. 70).</p> | April 11, 2017 |
| Added documentation | <p>If CloudTrail is not delivering logs to your S3 bucket or sending SNS notifications from some regions in your account, you may need to update the policies.</p> <p>To learn more about updating your S3 bucket policy, see Common Amazon S3 policy configuration errors (p. 245).</p> <p>To learn more about updating your SNS topic policy, see Common SNS policy configuration errors (p. 250).</p> | March 31, 2017 |
| Added service support | <p>This release supports AWS Organizations. See CloudTrail supported services and integrations (p. 20).</p> | February 27, 2017 |
| Added functionality and documentation | <p>This release supports an updated console experience for configuring trails for logging management and data events. For more information, see Working with CloudTrail log files (p. 133).</p> | February 10, 2017 |
| Added service support | <p>This release supports Amazon Cloud Directory. See CloudTrail supported services and integrations (p. 20).</p> | January 26, 2017 |
| Added functionality and documentation | <p>This release supports looking up APIs for AWS CodeCommit, Amazon GameLift, and AWS Managed Services in the CloudTrail API activity history.</p> | January 26, 2017 |

| Change | Description | Release Date |
|---------------------------------------|--|-------------------|
| Added functionality | <p>This release supports integration with the AWS Personal Health Dashboard. You can use the AWS Personal Health Dashboard to identify if your trails are unable to deliver logs to an SNS topic or S3 bucket. This can occur when there is an issue with the policy for the S3 bucket or SNS topic. AWS Personal Health Dashboard notifies you about the affected trails and recommends ways to fix the policy.</p> <p>For more information, see the AWS Health User Guide.</p> | January 24, 2017 |
| Added functionality and documentation | <p>This release supports filtering by event source in the CloudTrail console. Event source shows the AWS service to which the request was made.</p> <p>For more information, see Viewing CloudTrail events in the CloudTrail console (p. 45).</p> | January 12, 2017 |
| Added service support | This release supports AWS CodeCommit. See CloudTrail supported services and integrations (p. 20). | January 11, 2017 |
| Added service support | This release supports Amazon Lightsail. See CloudTrail supported services and integrations (p. 20). | December 23, 2016 |
| Added service support | This release supports AWS Managed Services. See CloudTrail supported services and integrations (p. 20). | December 21, 2016 |
| Added region support | This release supports the Europe (London) Region. See CloudTrail supported regions (p. 12). | December 13, 2016 |
| Added region support | This release supports the Canada (Central) Region. See CloudTrail supported regions (p. 12). | December 8, 2016 |
| Added service support | <p>This release supports AWS CodeBuild. See CloudTrail supported services and integrations (p. 20).</p> <p>This release supports AWS Health. See CloudTrail supported services and integrations (p. 20).</p> <p>This release supports AWS Step Functions. See CloudTrail supported services and integrations (p. 20).</p> | December 1, 2016 |
| Added service support | This release supports Amazon Polly. See CloudTrail supported services and integrations (p. 20). | November 30, 2016 |
| Added service support | This release supports AWS OpsWorks for Chef Automate. See CloudTrail supported services and integrations (p. 20). | November 23, 2016 |

| Change | Description | Release Date |
|---------------------------------------|---|--------------------|
| Added functionality and documentation | <p>This release supports configuring your trail to log read-only, write-only, or all events.</p> <p>CloudTrail supports logging Amazon S3 object level API operations such as <code>GetObject</code>, <code>PutObject</code>, and <code>DeleteObject</code>. You can configure your trails to log object level API operations.</p> <p>For more information, see Working with CloudTrail log files (p. 133).</p> | November 21, 2016 |
| Added functionality and documentation | <p>This release supports additional values for the <code>userIdentity</code> element: <code>AWSAccount</code> and <code>AWSService</code>. For more information, see the Fields (p. 282) for <code>userIdentity</code>.</p> | November 16, 2016 |
| Added service support | <p>This release supports AWS Server Migration Service. See CloudTrail supported services and integrations (p. 20).</p> | November 14, 2016 |
| Added service support | <p>This release supports Application Auto Scaling. See CloudTrail supported services and integrations (p. 20).</p> | October 31, 2016 |
| Added region support | <p>This release supports the US East (Ohio) Region. See CloudTrail supported regions (p. 12).</p> | October 17, 2016 |
| Added functionality and documentation | <p>This release supports logging non-API AWS service events. For more information, see AWS service events (p. 294).</p> | September 23, 2016 |
| Added functionality and documentation | <p>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. 48).</p> | July 7, 2016 |
| Added service support | <p>This release supports AWS Service Catalog. See CloudTrail supported services and integrations (p. 20).</p> | July 6, 2016 |
| Added service support | <p>This release supports Amazon Elastic File System (Amazon EFS). See CloudTrail supported services and integrations (p. 20).</p> | June 28, 2016 |
| Added region support | <p>This release supports one additional region: <code>ap-south-1</code> (Asia Pacific (Mumbai)). See CloudTrail supported regions (p. 12).</p> | June 27, 2016 |
| Added service support | <p>This release supports AWS Application Discovery Service. See CloudTrail supported services and integrations (p. 20).</p> | May 12, 2016 |
| Added service support | <p>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. 159).</p> | May 6, 2016 |
| Added service support | <p>This release supports AWS WAF. See CloudTrail supported services and integrations (p. 20).</p> | April 28, 2016 |

| Change | Description | Release Date |
|---------------------------------------|--|-------------------|
| Added service support | This release supports AWS Support. See CloudTrail supported services and integrations (p. 20) . | April 21, 2016 |
| Added service support | This release supports Amazon Inspector. See CloudTrail supported services and integrations (p. 20) . | April 20, 2016 |
| Added service support | This release supports AWS IoT. See CloudTrail supported services and integrations (p. 20) . | April 11, 2016 |
| 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. 285) . | March 28, 2016 |
| Added service support | This release supports AWS Certificate Manager. See CloudTrail supported services and integrations (p. 20) . | March 25, 2016 |
| Added service support | This release supports Amazon Kinesis Data Firehose. See CloudTrail supported services and integrations (p. 20) . | March 17, 2016 |
| Added service support | This release supports Amazon CloudWatch Logs. See CloudTrail supported services and integrations (p. 20) . | March 10, 2016 |
| Added service support | This release supports Amazon Cognito. See CloudTrail supported services and integrations (p. 20) . | February 18, 2016 |
| Added service support | This release supports AWS Database Migration Service. See CloudTrail supported services and integrations (p. 20) . | February 4, 2016 |
| Added service support | This release supports Amazon GameLift (GameLift). See CloudTrail supported services and integrations (p. 20) . | January 27, 2016 |
| Added service support | This release supports Amazon CloudWatch Events. See CloudTrail supported services and integrations (p. 20) . | January 16, 2016 |
| Added region support | This release supports one additional region: ap-northeast-2 (Asia Pacific (Seoul)). See CloudTrail supported regions (p. 12) . | January 6, 2016 |
| Added service support | This release supports Amazon Elastic Container Registry (Amazon ECR). See CloudTrail supported services and integrations (p. 20) . | 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. 9) . | December 17, 2015 |
| Added service support | This release supports Amazon Machine Learning. See CloudTrail supported services and integrations (p. 20) . | December 10, 2015 |

| Change | Description | Release Date |
|---------------------------------------|--|-------------------|
| 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. 259) , Validating CloudTrail log file integrity (p. 196) , and Updating a trail (p. 79) . | October 1, 2015 |
| Added service support | This release supports Amazon OpenSearch Service. See CloudTrail supported services and integrations (p. 20) . | October 1, 2015 |
| Added service support | This release supports Amazon S3 bucket level events. See CloudTrail supported services and integrations (p. 20) . | September 1, 2015 |
| Added service support | This release supports AWS Device Farm. See CloudTrail supported services and integrations (p. 20) . | July 13, 2015 |
| Added service support | This release supports Amazon API Gateway. See CloudTrail supported services and integrations (p. 20) . | July 9, 2015 |
| Added service support | This release supports CodePipeline. See CloudTrail supported services and integrations (p. 20) . | July 9, 2015 |
| Added service support | This release supports Amazon DynamoDB. See CloudTrail supported services and integrations (p. 20) . | 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. 159) . | May 19, 2015 |
| Added service support | This release supports AWS Directory Service. See CloudTrail supported services and integrations (p. 20) . | May 14, 2015 |
| Added service support | This release supports Amazon Simple Email Service (Amazon SES). See CloudTrail supported services and integrations (p. 20) . | May 7, 2015 |
| Added service support | This release supports Amazon Elastic Container Service. See CloudTrail supported services and integrations (p. 20) . | April 9, 2015 |
| Added service support | This release supports AWS Lambda. See CloudTrail supported services and integrations (p. 20) . | April 9, 2015 |
| Added service support | This release supports Amazon WorkSpaces. See CloudTrail supported services and integrations (p. 20) . | April 9, 2015 |
| | 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 Event history (p. 44) . | March 12, 2015 |

| Change | Description | Release Date |
|---|---|-------------------|
| Added service support and new documentation | This release supports Amazon CloudWatch Logs in the Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), and Europe (Frankfurt) regions. Additional CloudWatch alarm examples have been added to Creating CloudWatch Alarms for CloudTrail Events , and a new page has been added: Using an AWS CloudFormation Template to Create CloudWatch Alarms . | March 5, 2015 |
| Added API support | This release supports Amazon EC2 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 CloudTrail supported services and integrations (p. 20) . | February 11, 2015 |
| Added service support | This release supports AWS Config. See CloudTrail supported services and integrations (p. 20) . | February 10, 2015 |
| Added service support | This release supports AWS CloudHSM. See CloudTrail supported services and integrations (p. 20) . | January 8, 2015 |
| Added service support | This release supports AWS CodeDeploy. See CloudTrail supported services and integrations (p. 20) . | December 17, 2014 |
| Added service support | This release supports AWS Storage Gateway. See CloudTrail supported services and integrations (p. 20) . | December 16, 2014 |
| Added region support | This release supports one additional region: us-gov-west-1 (AWS GovCloud (US-West)). See CloudTrail supported regions (p. 12) . | December 16, 2014 |
| Added service support | This release supports Amazon S3 Glacier. See CloudTrail supported services and integrations (p. 20) . | December 11, 2014 |
| Added service support | This release supports AWS Data Pipeline. See CloudTrail supported services and integrations (p. 20) . | December 2, 2014 |
| Added service support | This release supports AWS Key Management Service. See CloudTrail supported services and integrations (p. 20) . | November 12, 2014 |
| New documentation | A new section, Monitoring CloudTrail Log Files with Amazon CloudWatch Logs (p. 159) , has been added to the guide. It describes how to use Amazon CloudWatch Logs to monitor CloudTrail log events. | November 10, 2014 |

| Change | Description | Release Date |
|----------------------------|---|--------------------|
| New documentation | A new section, Using the CloudTrail Processing Library (p. 215) , 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 CloudTrail supported services and integrations (p. 20) . | October 27, 2014 |
| Added region support | This release supports one additional region: eu-central-1 (Europe (Frankfurt)). See CloudTrail supported regions (p. 12) . | October 23, 2014 |
| Added service support | This release supports Amazon CloudSearch. See CloudTrail supported services and integrations (p. 20) . | October 16, 2014 |
| Added service support | This release supports Amazon Simple Notification Service. See CloudTrail supported services and integrations (p. 20) . | October 09, 2014 |
| Added service support | This release supports Amazon ElastiCache. See CloudTrail supported services and integrations (p. 20) . | September 15, 2014 |
| Added service support | This release supports Amazon WorkDocs. See CloudTrail supported services and integrations (p. 20) . | August 27, 2014 |
| Added new content | This release includes a topic that discusses logging sign-in events. See AWS Management Console sign-in events (p. 295) . | July 24, 2014 |
| 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. 272) . | July 18, 2014 |
| Added service support | This release supports Auto Scaling (see CloudTrail supported services and integrations (p. 20)). | July 17, 2014 |
| Added region support | This release supports three additional regions: ap-southeast-1 (Asia Pacific (Singapore)), ap-northeast-1 (Asia Pacific (Tokyo)), sa-east-1 (South America (São Paulo)). See CloudTrail supported regions (p. 12) . | June 30, 2014 |
| Additional service support | This release supports Amazon Redshift. See CloudTrail supported services and integrations (p. 20) . | June 10, 2014 |
| Added service support | This release supports AWS OpsWorks. See CloudTrail supported services and integrations (p. 20) . | June 5, 2014 |
| Added service support | This release supports Amazon CloudFront. See CloudTrail supported services and integrations (p. 20) . | May 28, 2014 |
| Added region support | This release supports three additional regions: us-west-1 (US West (N. California)), eu-west-1 (Europe (Ireland)), ap-southeast-2 (Asia Pacific (Sydney)). See CloudTrail supported regions (p. 12) . | May 13, 2014 |

| Change | Description | Release Date |
|----------------------------|--|-------------------|
| Added service support | This release supports Amazon Simple Workflow Service. See CloudTrail supported services and integrations (p. 20) . | 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. 186) . | May 2, 2014 |
| Added service support | This release supports Amazon CloudWatch. See CloudTrail supported services and integrations (p. 20) . | April 28, 2014 |
| Added service support | This release supports Amazon Kinesis. See CloudTrail supported services and integrations (p. 20) . | April 22, 2014 |
| Added service support | This release supports AWS Direct Connect. See CloudTrail supported services and integrations (p. 20) . | April 11, 2014 |
| Added service support | This release supports Amazon EMR. See CloudTrail supported services and integrations (p. 20) . | April 4, 2014 |
| Added service support | This release supports Elastic Beanstalk. See CloudTrail supported services and integrations (p. 20) . | April 2, 2014 |
| Additional service support | This release supports AWS CloudFormation. See CloudTrail supported services and integrations (p. 20) . | March 7, 2014 |
| New guide | This release introduces AWS CloudTrail. | November 13, 2013 |

AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.