



# Follow the CloudTrail

*Learn what happens in your AWS account  
when you're not watching*

Keith Gregory  
AWS Practice Lead, Chariot Solutions

# It's the Access Log for AWS

All AWS tasks are performed via GET, PUT, or POST

CloudTrail captures most (but not all!) API calls, with request and response bodies

Two types of events:

Management events: anything that creates/updates/deletes service

Data events: S3 operations, Lambda invocations

CloudTrail Insights is a layer on top of these events

# Example Event

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "REDACTED",
    "arn": "arn:aws:iam::123456789012:user/kdgregory",
    "accountId": "123456789012",
    "accessKeyId": "REDACTED",
    "userName": "kdgregory"
  },
  "eventTime": "2020-11-09T13:12:00Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "DeleteQueue",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "1.2.3.4",
  "userAgent": "aws-sdk-go/1.35.19 (go1.14.5; linux; amd64) APN/1.0 HashiCorp/1.0 Terraform/0.12.29...",
  "requestParameters": {
    "queueUrl": "https://sqs.us-east-1.amazonaws.com/123456789012/ReportGeneration"
  },
  "responseElements": null,
  "requestID": "d274e87d-860c-54b1-b49d-b05ec46ceae6",
  "eventID": "0222214f-8887-460c-9958-47ca41a3177e",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

# Console View

The screenshot displays the AWS CloudTrail Management Console in a Mozilla Firefox browser. The page title is "CloudTrail Management Console — Mozilla Firefox". The address bar shows the URL: `https://console.aws.amazon.com/cloudtrail/home?region=us-east-1#/events?ReadOnly=false`. The AWS navigation bar at the top shows the user is logged in as "Keith D Gregory" in the "N. Virginia" region, with a "Support" link.

The main content area is titled "CloudTrail > Event history". It features a search bar with "Read-only" selected and a search term of "false". There are buttons for "Download events" and "Create Athena table". A filter menu is set to "30m".

The event history table contains the following data:

<input type="checkbox"/>	Event name	Event time	User name	Event source	Resource type	Resource name
<input type="checkbox"/>	DeleteQueue	November 09, 2020, 08:12:...	kdgregory	sqs.amazonaws.com	AWS::SQS::Queue	https://sqs.us-east-1.amazonaws.c...
<input type="checkbox"/>	DeleteQueue	November 09, 2020, 08:12:...	kdgregory	sqs.amazonaws.com	AWS::SQS::Queue	https://sqs.us-east-1.amazonaws.c...
<input type="checkbox"/>	DeleteRole	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Role	ApplicationRole
<input type="checkbox"/>	DeletePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy	arn:aws:iam::[redacted]:polic...
<input type="checkbox"/>	DeleteQueue	November 09, 2020, 08:12:...	kdgregory	sqs.amazonaws.com	AWS::SQS::Queue	https://sqs.us-east-1.amazonaws.c...
<input type="checkbox"/>	DeleteQueue	November 09, 2020, 08:12:...	kdgregory	sqs.amazonaws.com	AWS::SQS::Queue	https://sqs.us-east-1.amazonaws.c...
<input type="checkbox"/>	DeletePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy	arn:aws:iam::[redacted]:polic...
<input type="checkbox"/>	DeletePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy	arn:aws:iam::[redacted]:polic...
<input type="checkbox"/>	DeletePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy	arn:aws:iam::[redacted]:polic...
<input type="checkbox"/>	DeletePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy	arn:aws:iam::[redacted]:polic...
<input type="checkbox"/>	DeleteRolePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy, AWS::IAM::Role	application_role_queue_policy, Ap...
<input type="checkbox"/>	DeleteQueue	November 09, 2020, 08:12:...	kdgregory	sqs.amazonaws.com	AWS::SQS::Queue	https://sqs.us-east-1.amazonaws.c...
<input type="checkbox"/>	DeletePolicy	November 09, 2020, 08:12:...	kdgregory	iam.amazonaws.com	AWS::IAM::Policy	arn:aws:iam::717623742438:polic...
<input type="checkbox"/>	DeleteQueue	November 09, 2020, 08:12:...	kdgregory	sqs.amazonaws.com	AWS::SQS::Queue	https://sqs.us-east-1.amazonaws.c...

The footer of the console shows "Feedback", "English (US)", and copyright information: "© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved." along with links for "Privacy Policy" and "Terms of Use".

# Creating a Trail

First trail is free!\*

\* management events only

## All events stored in S3

Limit access to this bucket: it may contain sensitive data

Use S3 Object Lock to prevent deletion (accidental or otherwise)

## Can also send to CloudWatch Logs

## Enable cross-region and cross-organization capture!

# How to Query Events

CloudTrail Console

Copy files from S3, use jq

Amazon Athena

CloudWatch Logs Insights

Elasticsearch / other search engines

---

# Problems

Why you don't want to use the Console, JQ, or Athena

# #1: Every event is different

Some fields are common

eventTime, eventSource, eventName, sourceIPAddress, ...

Most of the “interesting” information is event-specific

requestParameters: information the caller provides

responseElements: information about what AWS did

User identity changes depending on how API invoked

IAM user versus IAM role versus AWS internal



## #2: Who invoked that API?

Most API calls are made under assumed role

Or at least should be: Lambda execution role, EC2 instance role, ...

If you explicitly assume a role in your program, use a good session ID!

AWS services *usually* set `userIdentity.invokedBy`

Typically to the service name, sometimes to “AWS Internal”

Some events are tied to others

Example: `SharedSnapshotVolumeCreated`, from EC2 RunInstances

# #3: This data is *noisy*!

API calls are granular

Starting an EC2 instance from the Console involves a dozen calls

The Console does a lot that you don't know about

Internal AWS operations recorded alongside user actions

Load balancers regularly check their instances

Trusted Advisor, other monitoring services run regularly, cross regions

Lambda invocations may not be recorded, but their assume role and KMS decrypt operations are

## #4: There are a lot of files

Each file contains approximately 15 minutes of events

For one account, and one region

# #5: Not all API calls are tracked

Lambda invocations tracked as “data events”

Most services only track “infrequent” operations

Example: DynamoDB tracks `CreateTable` but not `Scan`

Some services don't expose a public API

Example: DeepLens

---

# Solutions

Search Engines to the rescue!

# CloudWatch Logs Insights

The screenshot displays the AWS CloudWatch Logs Insights console. The main area shows a query editor with the following query:

```
1 fields @timestamp, @message
2 | filter eventName == "CreateStream"
3 | sort @timestamp desc
4 | limit 20
```

Below the query editor is a histogram showing the distribution of records over time. The histogram title is "Showing 5 of 5 records matched" and "819 records (686.9 kB) scanned in 2.4s @ 336 records/s (282.3 kB/s)". The x-axis represents time from 10:15 to 11:30. There are two bars: one at 10:50 with a height of 3, and another at 10:55 with a height of 1.

Below the histogram is a table of log records:

#	@timestamp	@message
▶ 1	2020-11-22T10:56:56...	{"eventVersion": "1.05", "userIdentity": {"type": "IAMUser", "principalId": "AIDAIIFY23X54B6PSQZ3LY", "arn": "arn:aws:iam::...}}
▶ 2	2020-11-22T10:50:09...	{"eventVersion": "1.05", "userIdentity": {"type": "IAMUser", "principalId": "AIDAIIFY23X54B6PSQZ3LY", "arn": "arn:aws:iam::...}}
▶ 3	2020-11-22T10:49:57...	{"eventVersion": "1.05", "userIdentity": {"type": "IAMUser", "principalId": "AIDAIIFY23X54B6PSQZ3LY", "arn": "arn:aws:iam::...}}
▶ 4	2020-11-22T10:49:38...	{"eventVersion": "1.05", "userIdentity": {"type": "IAMUser", "principalId": "AIDAIIFY23X54B6PSQZ3LY", "arn": "arn:aws:iam::...}}
▶ 5	2020-11-22T10:49:38...	{"eventVersion": "1.05", "userIdentity": {"type": "IAMUser", "principalId": "AIDAIIFY23X54B6PSQZ3LY", "arn": "arn:aws:iam::...}}

On the right side, there is a "Discovered fields" panel with a search bar and a list of fields with their corresponding percentages:

- @IngestionTime 100%
- @logStream 100%
- @message 100%
- @timestamp 100%
- awsRegion 100%
- eventID 100%
- eventName 100%
- eventSource 100%
- eventTime 100%
- eventType 100%
- eventVersion 100%
- recipientAccountId 100%
- sourceIPAddress 100%
- userAgent 100%
- userIdentity.type 100%
- requestID 99%
- userIdentity.accessKeyId 99%
- userIdentity.accountId 99%
- userIdentity.arn 99%
- userIdentity.principalId 99%
- userIdentity.userName 99%
- requestParameters.streamName 90%
- userIdentity.sessionContext.attributes.crea... 8%
- userIdentity.sessionContext.attributes.mfa... 8%

The left sidebar contains navigation options: CloudWatch, Dashboards, Alarms, Billing, Logs, Insights, Metrics, Events, ServiceLens, Container Insights, Lambda Insights, Synthetics, Contributor Insights, and Settings. The bottom of the page includes a footer with "Feedback", "English (US)", and "© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use".

# CloudWatch Logs Pros/Cons

Pro: set up with a few clicks

Pro: easy to enable log monitors, alerts

Con: every Insights search is a unique filter

You need to know what the event looks like

Con: you pay per search

It's only \$0.005 per GB of data scanned, but it adds up

Con: it's slow

# Elasticsearch

Discover - Kibana — Mozilla Firefox

Discover - Kibana

https://search-cloudtrail-events-ef6yajs2dttvmw72qakdl5i.us-east-1.es.amazonaws.com/\_plugin/kibana/app/kibana#/discover?\_g=(refreshInterval:(pause:t,value:0),time:(from:now-1h,mode:quick,to:n... 120%

5 hits

New Save Open Share Inspect Auto-refresh Last 1 hour

Search... (e.g. status:200 AND extension:PHP) Options Refresh

eventName.keyword: "CreateStream" Add a filter + Actions

cloudtrail-\*

Selected fields

- t awsRegion
- t eventName
- t eventSource
- t userIdentity.accountId
- t userIdentity.userName

Available fields

Popular

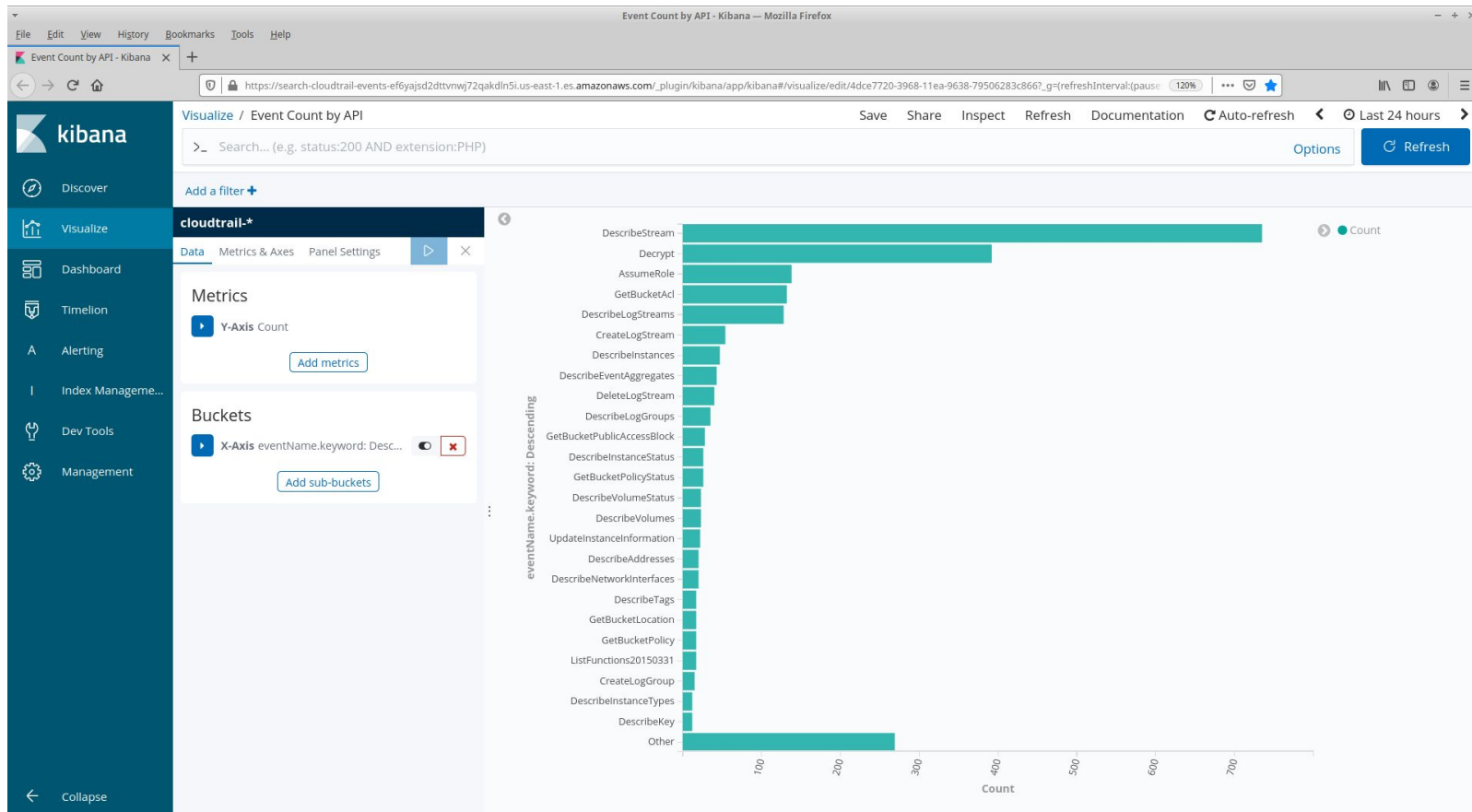
- t sourceIPAddress
- t \_id
- t \_index
- # \_score
- t \_type
- t eventId
- eventTime
- t eventType
- t eventVersion
- t recipientAccountId
- t requestID

November 22nd 2020, 12:37:26.442 - November 22nd 2020, 13:37:26.442 — Auto

Time	eventName	awsRegion	eventSource	userIdentity.userName	userIdentity.accountId
▶ November 22nd 2020, 13:14:37.000	CreateStream	us-east-1	kinesis.amazonaws.com	kgregory	[REDACTED]
▶ November 22nd 2020, 13:13:50.000	CreateStream	us-east-1	kinesis.amazonaws.com	kgregory	[REDACTED]
▶ November 22nd 2020, 13:13:39.000	CreateStream	us-east-1	kinesis.amazonaws.com	kgregory	[REDACTED]
▶ November 22nd 2020, 13:12:57.000	CreateStream	us-east-1	kinesis.amazonaws.com	kgregory	[REDACTED]
▶ November 22nd 2020, 13:12:41.000	CreateStream	us-east-1	kinesis.amazonaws.com	kgregory	[REDACTED]



# Elasticsearch Visualizations



# Elasticsearch Pros/Cons

## Pro: extremely flexible

Search by any value in the event (as long as it's indexed)

Easily narrow searches by time range; filter undesired values

## Pro: fast!

Most searches can be based on keywords

## Con: you have to configure event pipeline

Elasticsearch doesn't like that every event is different!

Logstash has a plugin to parse events

---

# Third-party Providers

The usual suspects: DataDog, Logz.io, Loggly, ...

Easy to get started; flexible and fast searches

Cost-effectiveness depends on log volume

---

# Use Cases

Including a few that aren't supported

---

# Who Started That EC2 Instance?

Event type: RunInstances

Filter by: `responseElements.instanceId`

Look at `userIdentity`

---

# Who Accessed That Bucket?

Must have data events enabled

Filter by `requestParameters.bucketName`

Optional: filter by `eventName`

Look at `userIdentity`, `sourceIPAddress`



# Is There Anything To Worry About?

Visualize events by region, source IP address

Filter out AWS-generated events

Switch to search mode to explore unexpected results



# ~~Is someone attacking me?~~

Events appear in the trail 10-20 minutes after they happen

Not all failed requests are captured





# ~~What permissions does my app need?~~

Applications often use API calls that aren't captured



# ~~What's doesn't my ABAC policy work?~~

User identity doesn't contain `aws:Principal tags`

---

# Wrapping Up



# You Can Trigger Lambdas

Via EventBridge (formerly CloudWatch Events)

Trigger format is ... challenging



# It's Not The Only Tool

Bucket access logs are cheaper alternative to data events

Makes more sense if you care about public bucket accesses

Cost Explorer often has better view of “what’s happening”

# Technology in the Service of Business.

Chariot Solutions is the Greater Philadelphia region's top IT consulting firm specializing in software development, systems integration, mobile application development and training.

Our team includes many of the top software architects in the area, with deep technical expertise, industry knowledge and a genuine passion for software development.

Visit us online at [chariotsolutions.com](http://chariotsolutions.com).