Common Pitfalls and Best Practices in AWS CloudTrail

2 April 2025 - 4 min. read

AWS CloudTrail

Intro

How many times have you found yourself trying to track down actions performed in an AWS environment, attempting to reconstruct events, and figuring out who did what?

And how many times have you ended up abandoning the investigation without getting anywhere? Many of you have likely faced this situation, especially those working in AWS accounts shared by large teams or even in AWS accounts dedicated to multiple projects with multiple external consultants involved.

In some cases, when AWS environment governance is not well-structured, alternative methods are needed to determine the root cause of an issue. To do this, we'll use one of my favorite services: AWS CloudTrail.

In this blog post, we will analyze the typical AWS CloudTrail setup in an AWS account or organization, explore some common issues that may arise in daily use, and discuss best practices to facilitate investigation and overcome the limitations of a standard setup.

The Trail Is Missing

One of the most common issues is the absence of the Trail itself.

We often need to trace the ownership of a specific AWS resource to request more information about it. Unfortunately, there are times when AWS CloudTrail logs provide no evidence simply because the resource creation is too old and, by default, CloudTrail retains logs for only the last **90 days**.

Lesson learned: *When opening a new AWS account, always configure a **Trail** to maintain a long-term history of all activities within the account.*

Initial Setup

According to **AWS CloudTrail best practices**, we should enable a Trail and configure an S3 Bucket for long-term log storage.

At the same time, based on **Amazon S3 best practices**, we should implement lifecycle policies to adjust the storage class of stored objects based on how frequently they are accessed.

For bonus points, we can also create an Amazon Athena table using AWS default schema to simplify querying the logs.

“Ok, I'm running my query on Athena, but I don’t see any results beyond a certain date”

Sometimes, we need to investigate an action that occurred on a specific day. We run a query filtering for that single day and… we discover that the query returns no results.

"That’s weird… "– average DevOps.

An untrained eye might quickly conclude that the data doesn't exist. However, if we check the logs directly in the S3 bucket, we will see that the data is actually there… But, the files have been moved to the **Glacier Flexible Retrieval tier**.

Lesson learned: *By default, an Athena query ignores all objects stored in S3 Glacier Flexible Retrieval and S3 Glacier Deep Archive. While Athena can query these files, you must first enable this option for the specific table you want to query.*

“I'm trying to find out who overwrote my Route 53 record, but I can't find anything”

It’s common to search for actions performed on specific AWS services (e.g., IAM, Route 53, etc.) and come up empty-handed.

Why?

After all, we followed best practices and enabled CloudTrail.

The reason is that some actions are logged only in the North Virginia region (us-east-1) because they belong to global AWS services. This applies to Route 53 record modifications, IAM Role creations, CloudFront Distribution creations, and more.

Lesson learned: *Always enable multi-region logging when setting up AWS CloudTrail and remember that global AWS services log their activity in North Virginia (us-east-1).*

“I have tons of logs, and my queries are very slow. What can I do?”

The answer is partitioning.

CloudTrail organizes log storage in S3 using the following structure:

```
s3://<bucket-name>/<optional-prefix>/AWSLogs/<account-id>/CloudTrail/
<region>/<year>/<month>/<day>/<log-file>.json.gz
```

While this structure is well-organized, it’s not enough on its own. To run queries efficiently, we need to partition the data in Athena as well. There are two ways to do this:

1. Manually define partitions with `ALTER TABLE ADD PARTITION`

This allows us to query individual partitions, optimizing performance.

However, manually creating and managing partitions introduces overhead and complexity.

2. Use partition projection

Since CloudTrail log structures are predictable, Athena can automatically infer partition locations, eliminating the need to manually add new partitions.

Wrapping up

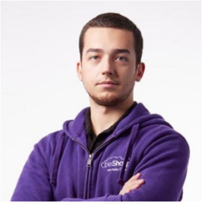
We’ve explored some lesser-known details and peculiarities of AWS CloudTrail, along with its related services, Amazon S3 and Amazon Athena. We’ve learned that S3 lifecycle policies can sometimes interfere with daily operations, requiring us to balance costs and data availability. We also saw some partitioning options that Amazon Athena offers and how query performance can be improved with minimal management effort.

Were you already aware of these aspects? Have you encountered other limitations or – hopefully – discovered even more best practices?

Let us know!

About Proud2beCloud

Proud2beCloud is a blog by beSharp, an Italian APN Premier Consulting Partner expert in designing, implementing, and managing complex Cloud infrastructures and advanced services on AWS. Before being writers, we are Cloud Experts working daily with AWS services since 2007. We are hungry readers, innovative builders, and gem-seekers. On Proud2beCloud, we regularly share our best AWS pro tips, configuration insights, in-depth news, tips&tricks, how-tos, and many other resources. Take part in the discussion!



Mehmed Dourmouch

DevOps Engineer. Very Dev, not so Ops. I like to break things and see what happens, I also automate everything. I often participate in cybersecurity CTFs and in my free time I produce cacophony with my guitar.