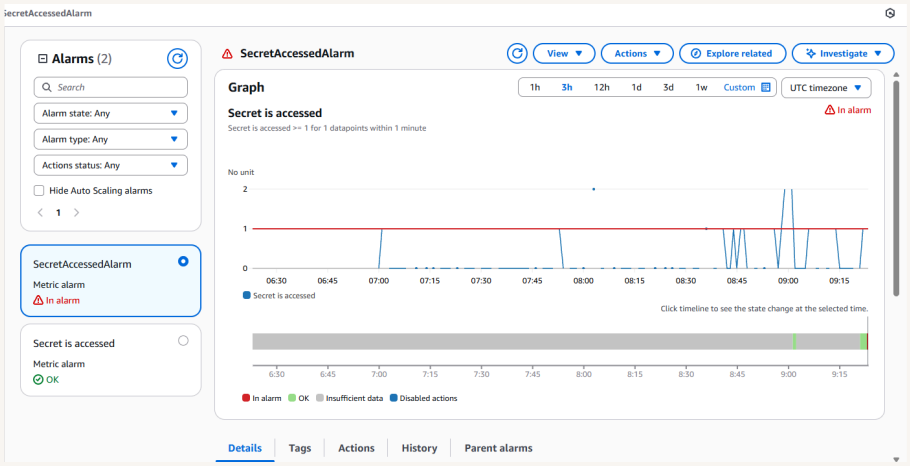
nextwork.org

Build a Security Monitoring System

L Louis Moyo



Louis Moyo
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate how to build a complete monitoring and alerting system in AWS using CloudTrail, CloudWatch, and SNS. I'm doing this project to learn how to securely store secrets, track who accesses them, and get notified instantly if something suspicious happens.

Tools and concepts

Services I used were AWS Secrets Manager, CloudTrail, CloudWatch Logs, CloudWatch Metric Filters, CloudWatch Alarms, and SNS. Key concepts I learnt include securely storing and accessing secrets, enabling CloudTrail to capture account activity, filtering logs in CloudWatch to detect specific events, setting up alarms based on those filters, and triggering real-time alerts through SNS notifications. I also understood how each service integrates together to create an end-to-end monitoring and alerting pipeline.

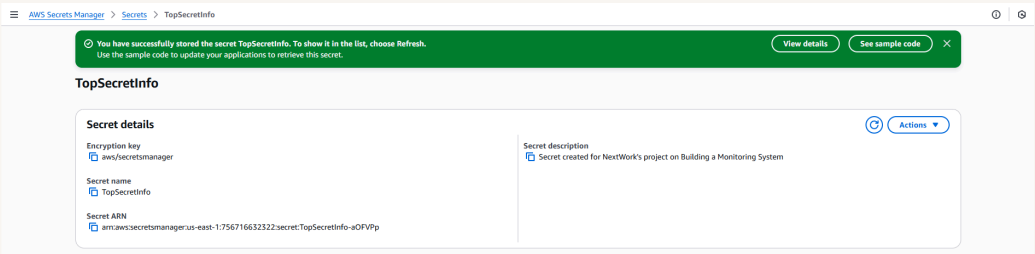
Project reflection

This project took me approximately 3 hours. The most challenging part was correctly creating the CloudWatch filter pattern so only relevant secret access events triggered the alarm. It was most rewarding to test the system end-to-end by accessing the secret, seeing the log captured in CloudTrail, watching the alarm fire in CloudWatch, and finally receiving the SNS email - confirming the system works exactly as intended.

Create a Secret

Secrets Manager is an AWS service that securely stores, manages, and rotates sensitive information such as passwords, API keys, and database credentials. You could use Secrets Manager to protect secrets from being hardcoded in applications, automatically rotate them for better security, and control access through fine-grained IAM policies while also tracking usage with CloudTrail.

To set up for my project, I created a secret called TopSecretInfo that contains a key-value pair where the key is “The Secret is” and the value is a random phrase I entered. This secret will be the item I monitor for access events throughout the rest of the project.



Set Up CloudTrail

CloudTrail is an AWS service that records all activity in your account, such as who performed an action, what they did, when it happened, and where it came from. I set up a trail to tell CloudTrail which activities to capture and to define where those logs should be stored, so I can later review and monitor access to my secret.

CloudTrail events include types like Management events, Data events, Insights events, and Network activity events. Management events record control-plane API activities such as creating, updating, or deleting AWS resources (e.g., launching an EC2 instance, modifying a security group, or retrieving a secret from Secrets Manager). Data events record high-volume, resource-level API calls that act on data within a service, such as reading/writing objects in S3 or invoking a Lambda function. Insights events capture unusual patterns or anomalies in your management events, like a sudden spike in IAM role creations or failed login attempts. Network activity events track changes or interactions with your VPC and related network traffic. Together, these event types give you a full picture of who did what, when, and how inside your AWS account.

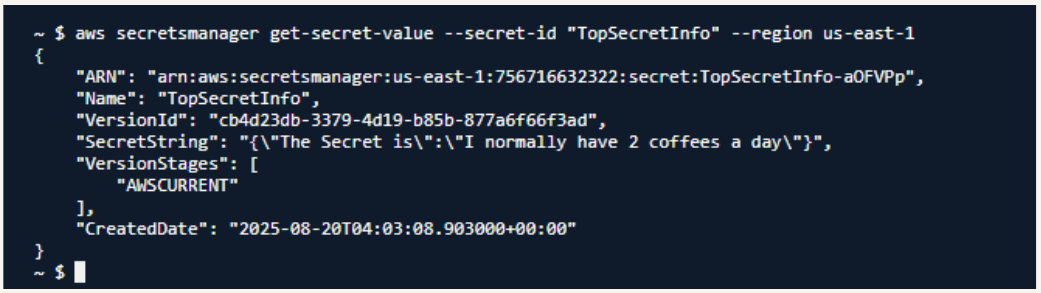
Read vs Write Activity

Read API activity involves viewing or listing resource details without changing or exposing the sensitive value - for example, listing secrets or checking their metadata. Write API activity involves making changes to a resource or retrieving sensitive values, such as creating, deleting, updating, or fetching the actual secret. For this project, I need Write API activity because I want to capture and monitor whenever the secret's value is accessed.

Verifying CloudTrail

I retrieved the secret in two ways: First through the AWS Management Console by viewing it in Secrets Manager, and second using the AWS CLI command get-secret-value in CloudShell.

To analyse my CloudTrail events, I visited the event history page in the CloudTrail console and filtered by the event source secretsmanager.amazonaws.com. I found GetSecretValue events, which shows that my secret's value was retrieved. This tells me CloudTrail successfully captured and logged my secret access activity.



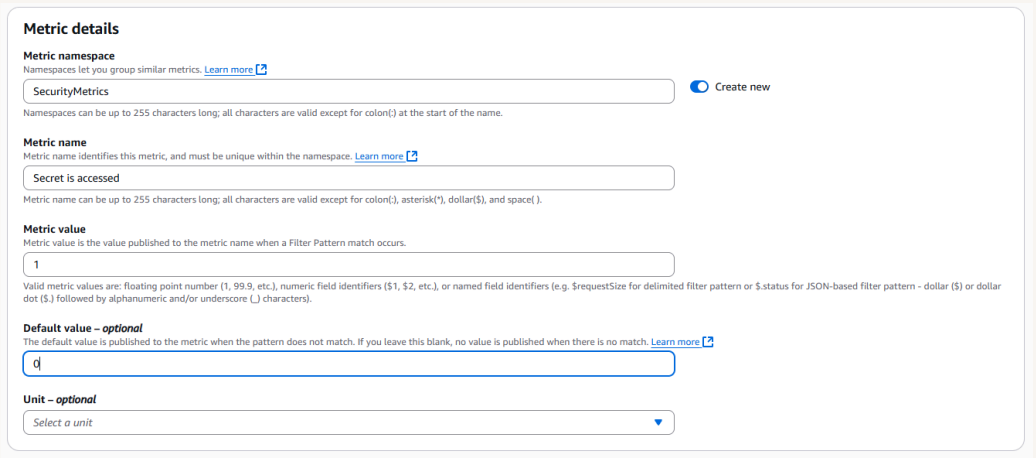
L
Louis Moyo
NextWork Student

CloudWatch Metrics

CloudWatch Logs is an AWS service that collects, stores, and manages log data from AWS resources and applications. It's important for monitoring because it lets you centralise logs, search for specific events, create alerts on suspicious activity (like secret access), and even trigger automated responses to improve security and troubleshooting.

CloudTrail's Event History is useful for quickly viewing recent management events and auditing activity over the past 90 days, while CloudWatch Logs are better for long-term storage, detailed searching, creating metrics, and setting up alerts that notify you in real time when specific events occur.

A CloudWatch metric is a numeric measurement that tracks a specific event or resource activity over time. When setting up a metric, the metric value represents the count or statistic you want CloudWatch to record (for example, “1” each time a secret is accessed). Default value is used when no event data is present, ensuring the metric still has a baseline (like 0) instead of showing up as missing data.

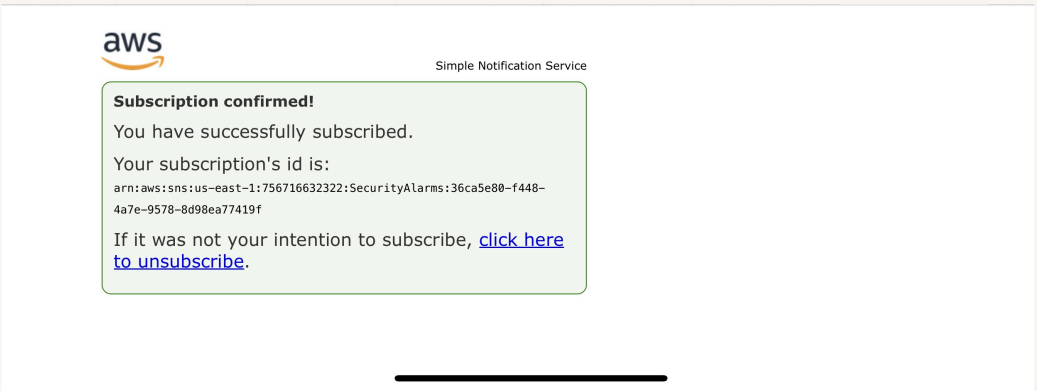

Metric details
Metric namespace
Namespaces let you group similar metrics. Learn more
SecurityMetrics
Create new
Namespaces can be up to 255 characters long; all characters are valid except for colon(:) at the start of the name.
Metric name
Metric name identifies this metric, and must be unique within the namespace. Learn more
Secret is accessed
Metric name can be up to 255 characters long; all characters are valid except for colon(:), asterisk(*), dollar($), and space( ).
Metric value
Metric value is the value published to the metric name when a Filter Pattern match occurs.
1
Valid metric values are: floating point number (1, 99.9, etc.), numeric field identifiers ($1, $2, etc.), or named field identifiers (e.g. $requestSize for delimited filter pattern or $.status for JSON-based filter pattern - dollar ($) or dollar dot ($.) followed by alphanumeric and/or underscore (_) characters).
Default value – optional
The default value is published to the metric when the pattern does not match. If you leave this blank, no value is published when there is no match. Learn more
0
Unit – optional
Select a unit

CloudWatch Alarm

A CloudWatch alarm is a monitoring tool that watches a specific metric and triggers an action when the metric crosses a defined threshold. I set my CloudWatch alarm threshold to greater than or equal to 1 for the SecretIsAccessed metric, so the alarm will trigger whenever the secret is accessed at least once within a 5-minute period. This ensures I get an immediate alert if my secret is retrieved, which is critical for detecting potential security issues.

I created an SNS topic along the way. An SNS topic is a messaging channel that lets AWS send notifications to multiple subscribers, such as emails, SMS, or apps. My SNS topic is set up to send an email alert to me whenever the CloudWatch alarm detects that my secret has been accessed.

AWS requires email confirmation because it needs to verify that the recipient actually owns and can access the email address being subscribed. This helps prevent unwanted spam, accidental subscriptions, or malicious users subscribing others without their consent.


aws
Simple Notification Service
Subscription confirmed!
You have successfully subscribed.
Your subscription's id is:
arn:aws:sns:us-east-1:756716632322:SecurityAlarms:36ca5e80-f448-4a7e-9578-8d98ea77419f
If it was not your intention to subscribe, click here to unsubscribe.

Troubleshooting Notification Errors

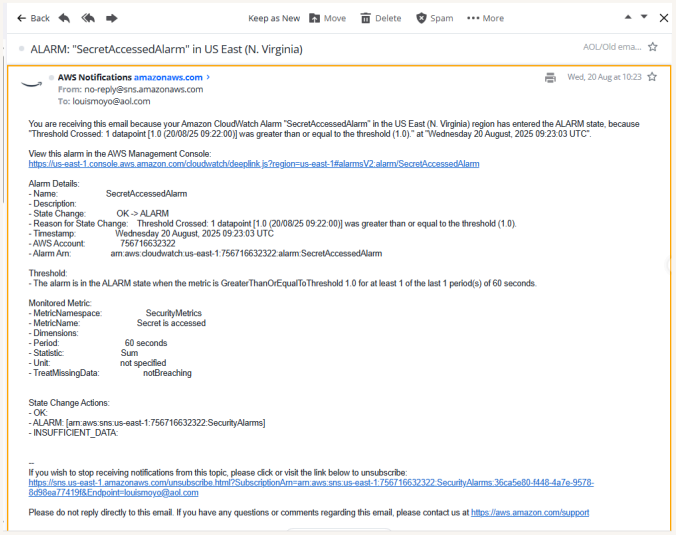
To test my monitoring system, I retrieved the secret value again to trigger the CloudWatch alarm and SNS notification. The results were that I didn't receive an email alert as expected, which means I need to troubleshoot the alarm or SNS configuration to ensure notifications are delivered properly.

When troubleshooting the notification issues I: Checked CloudTrail to confirm that the GetSecretValue event was being logged. Verified log delivery from CloudTrail to CloudWatch Logs to ensure the events were flowing through. Reviewed the metric filter to confirm it was correctly detecting secret access events. Inspected the CloudWatch alarm configuration and history to see if it actually entered the ALARM state and triggered actions. Tested the SNS subscription by confirming the email status was "Confirmed" and publishing a manual test message to ensure email delivery worked.

I initially didn't receive an email before because my CloudWatch alarm was using the Average statistic instead of Sum, so a single secret access event didn't meet the threshold. The key solution was changing the statistic to Sum, which correctly counts each event and ensures the alarm triggers when the secret is accessed.

Success!

To validate that my monitoring system works, I checked the CloudWatch metric filters I created for Secrets Manager access events and confirmed that they were generating metrics whenever a GetSecretValue API call occurred. I then triggered a test by accessing a secret directly, which caused the custom metric to increment. I verified in CloudWatch that the alarm condition was met and that the alarm transitioned into the In alarm state. I received the alarm notification through the configured SNS topic, confirming that both the metric, the threshold, and the notification pipeline were working as expected. This end-to-end test showed that my monitoring system correctly detects secret access and alerts in real time.

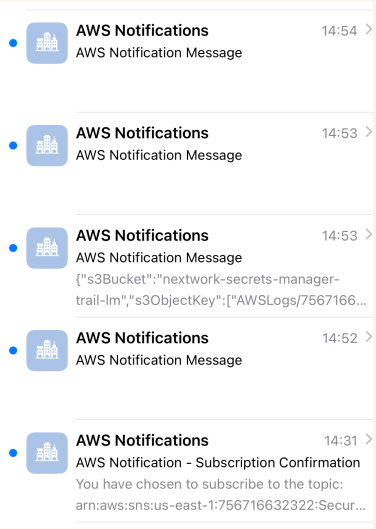


L
Louis Moyo
NextWork Student

Comparing CloudWatch with CloudTrail Notifications

In a project extension, I updated my CloudTrail configurations to enable direct SNS notifications because I wanted to compare receiving alerts from CloudTrail log deliveries versus CloudWatch alarms, and understand the trade-offs in how different AWS services handle monitoring and notifications.

After enabling CloudTrail SNS notifications, my inbox started receiving emails whenever new CloudTrail log files were delivered to S3. In terms of the usefulness of these emails, I thought they were less targeted than CloudWatch alarms since they notify about all log deliveries rather than specifically when my secret is accessed.



The place to learn & showcase your skills

Check out nextwork.org for more projects