nextwork.org

Build a Security Monitoring System

Chrispinus Jacob

Chrispinus Jacob
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate how to set up a monitoring and alert system using AWS CloudTrail, CloudWatch, and SNS. I'm doing this project to learn how AWS handles security, activity tracking, and notifications. The goal is to detect when someone accesses sensitive information stored in AWS Secrets Manager. When such access occurs, the system will trigger an alert and send an email notification. This helps me understand how to monitor and respond to critical security events in AWS.

Tools and concepts

Services I used were AWS Secrets Manager, CloudTrail, CloudWatch Logs, CloudWatch Alarms, and SNS. Key concepts I learnt include how to monitor access to sensitive resources using CloudTrail events, how to send those logs to CloudWatch for real-time analysis, how to create metric filters to detect specific actions, how to trigger alarms based on those metrics, and how to use SNS to send email notifications. I also gained a better understanding of security best practices like least privilege access and how to troubleshoot end-to-end monitoring workflows.

Project reflection

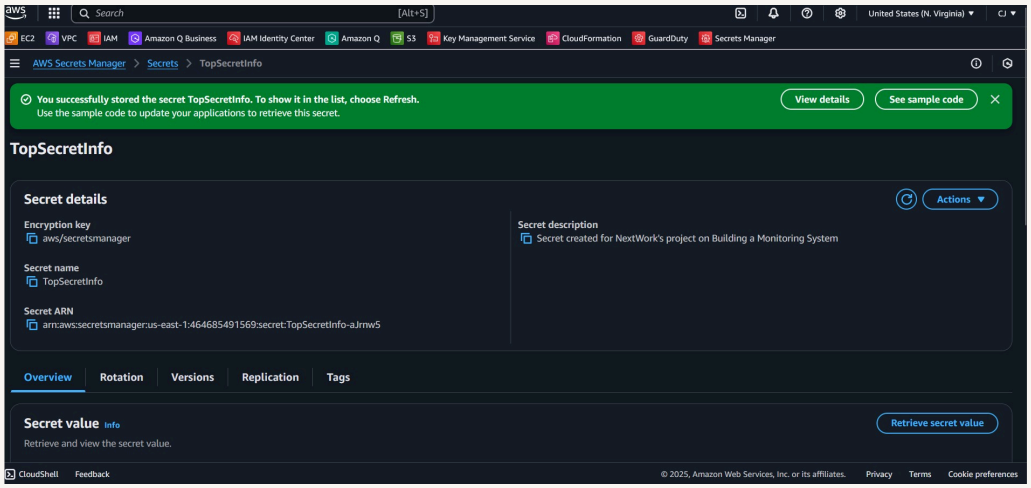
This project took me approximately a 3 hours to complete. The most challenging part was troubleshooting the CloudWatch alarm and ensuring the metric was correctly configured to trigger based on the sum of secret access events. It was most rewarding to see the entire system work end-to-end—getting an instant email alert when the secret was accessed, which confirmed that my monitoring and alerting setup was working as intended.

Chrispinus Jacob
NextWork Student

Create a Secret

Secrets Manager is an AWS service that helps you securely store and manage sensitive information such as database credentials, API keys, and other secrets. You could use Secrets Manager to protect access to critical data by controlling permissions and tracking usage through logging. In this step, I will create a secret in Secrets Manager because it will act as the sensitive resource we want to monitor. The rest of the project will focus on detecting access to this secret and triggering alerts when it happens.

To set up for my project, I created a secret called TopSecretInfo in AWS Secrets Manager that contains a fun but sensitive string. The secret holds a hot take from us: "I need 3 coffees a day to function." This string is stored securely, and throughout the rest of the project, I will focus on monitoring access to this secret to simulate protecting sensitive information in a real-world scenario.



Chrispinus Jacob
NextWork Student

Set Up CloudTrail

CloudTrail is an AWS service that records API calls and account activity across your AWS environment, providing a detailed history of actions taken through the console, SDKs, and CLI. I set up a trail to log all events in my account so I can monitor when someone accesses the secret I created, including details like who accessed it, when it was accessed, and from where. This is important for tracking and auditing sensitive operations.

CloudTrail events include types like management events, data events, and insightful events. In this project, I set up a trail to track management events because accessing a secret falls under that category. It is not considered a data event, which captures high-volume actions performed on resources. I chose management events because AWS allows us to track security-related operations like these for free, making it a cost-effective way to monitor sensitive activity.

Read vs Write Activity

Read API activity involves actions that retrieve data from AWS services, such as reading the value of a secret from Secrets Manager. Write API activity involves actions that create, modify, or delete resources, like adding a new secret or updating an existing one. For this project, we need to monitor both read and write API activity to detect when someone accesses sensitive information and also when any changes are made to the secret itself. This helps ensure full visibility and control over the secret's usage.

Verifying CloudTrail

I retrieved the secret in two ways: first through the AWS Management Console by navigating to Secrets Manager and viewing the secret value; second using the AWS CLI by running a command to get the secret value. These actions simulate real access to the secret, allowing me to confirm that CloudTrail logs both methods and records the necessary details for monitoring.

To analyze my CloudTrail events, I visited the CloudTrail console and opened the Event history section. I found events related to Secrets Manager, including `GetSecretValue` which showed when the secret was read. This tells me that CloudTrail is correctly logging read access to the secret, confirming that our monitoring setup is working as expected.

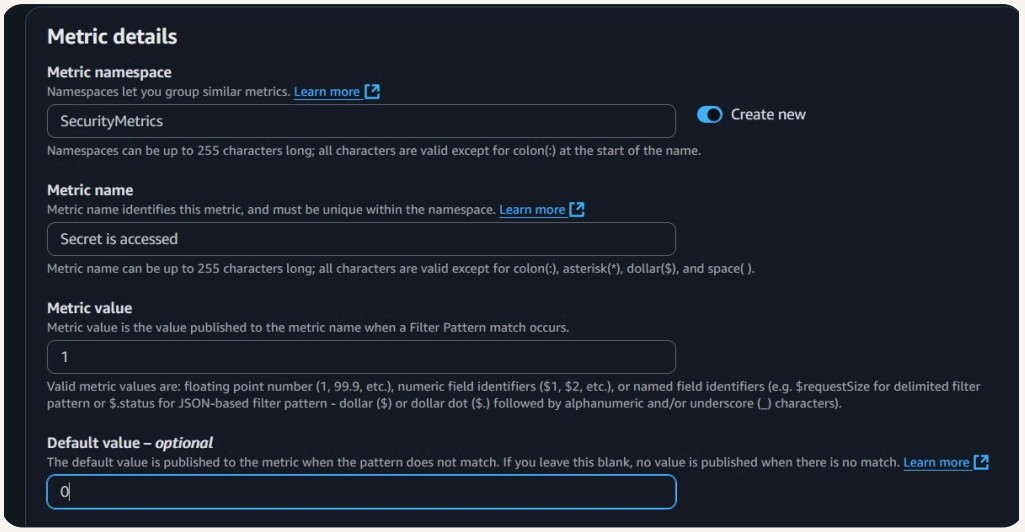
Chrispinus Jacob
NextWork Student

CloudWatch Metrics

CloudWatch Logs is an AWS service that collects and stores logs from different AWS resources, including CloudTrail. It's important for monitoring because it allows you to filter, search, and analyze logs in real time, create custom metrics based on log patterns, and set up alarms for specific activities. In this project, it helps us detect and respond to sensitive actions like secret access in Secrets Manager.

CloudTrail's Event History is useful for viewing recent account activity and investigating specific events over the last 90 days, while CloudWatch Logs are better for long-term storage, advanced filtering, and setting up automated alerts. In this project, Event History helps confirm if events are being tracked, but CloudWatch Logs is where we set up real-time monitoring and notifications for secret access.

A CloudWatch metric is a data point that tracks a specific event or performance indicator over time. When setting up a metric, the metric value represents what gets recorded when a matching log pattern is found—in this case, we use a metric value of 1 to indicate that the secret was accessed. The metric name is Secret is accessed, which helps identify what the metric is tracking. Default value is used when no matching event occurs during the time window, allowing CloudWatch to report a 0 so we can visualize periods with and without access.

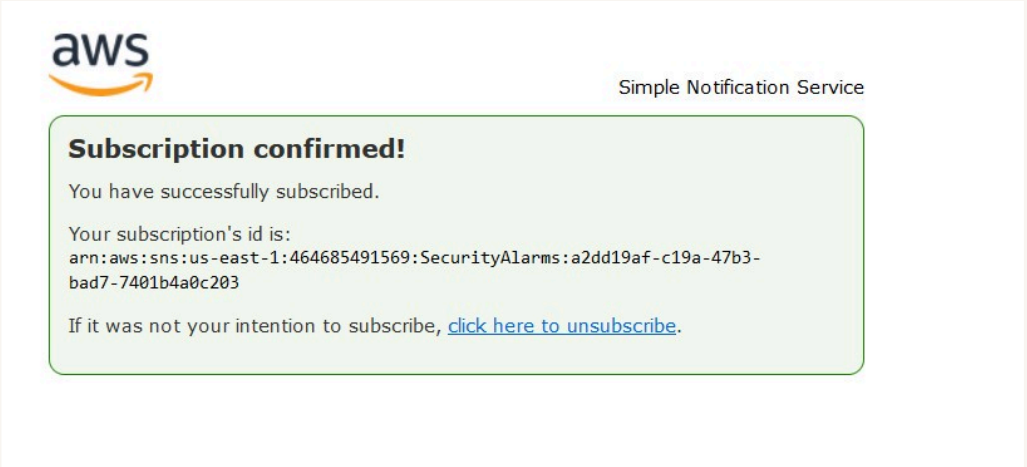

Metric details
Metric namespace
Namespaces let you group similar metrics. Learn more
SecurityMetrics
Create new
Namespaces can be up to 255 characters long; all characters are valid except for colon(:) at the start of the name.
Metric name
Metric name identifies this metric, and must be unique within 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

CloudWatch Alarm

A CloudWatch alarm is a tool that watches a specific metric and triggers an alert when the metric crosses a defined threshold. I set my CloudWatch alarm threshold to greater than or equal to 1 in a 5-minute period, so the alarm will trigger when the secret is accessed at least once during that time. This is a sensitive setting, which is ideal for high-security scenarios. In production, the threshold might be adjusted based on normal access patterns to reduce false alarms and only alert on unusual behavior.

I created an SNS topic along the way. An SNS topic is a communication channel that allows messages to be sent to one or more subscribers, such as email addresses or other AWS services. My SNS topic is set up to send an email notification whenever the CloudWatch alarm is triggered, so I can be instantly alerted if someone accesses the secret in Secrets Manager.

AWS requires email confirmation because it ensures that the person receiving the notifications has given permission to subscribe to the SNS topic. This helps prevent unwanted or spam messages from being sent to users who haven’t agreed to receive alerts, protecting privacy and reducing the risk of misuse.


aws
Simple Notification Service
Subscription confirmed!
You have successfully subscribed.
Your subscription's id is:
arn:aws:sns:us-east-1:464685491569:SecurityAlarms:a2dd19af-c19a-47b3-bad7-7401b4a0c203
If it was not your intention to subscribe, click here to unsubscribe.

Chrispinus Jacob
NextWork Student

Troubleshooting Notification Errors

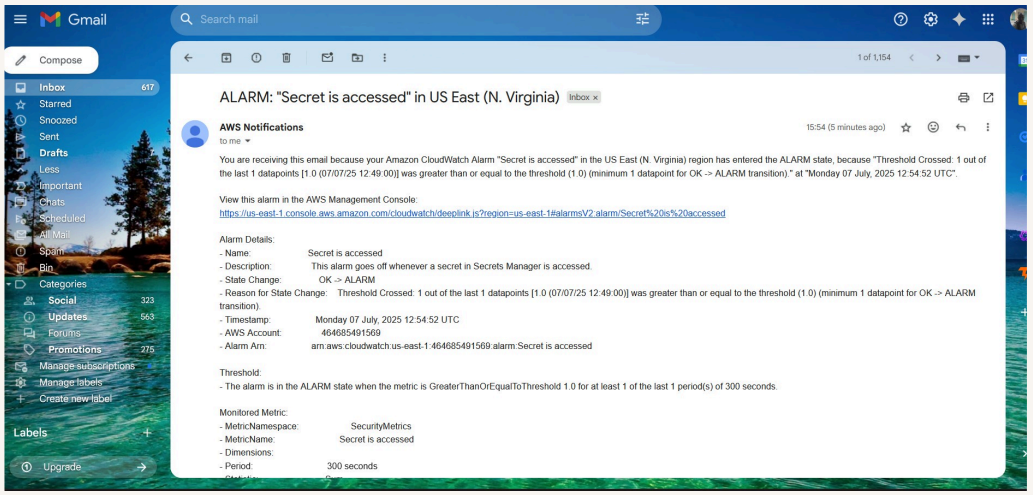
To test my monitoring system, I accessed the secret in Secrets Manager. The results were not successful—I didn't receive any email or notification about the secret being accessed. This means something in the monitoring pipeline isn't working correctly, and I'll need to troubleshoot each step to find out where the issue is. It could be a misconfigured metric filter, an inactive CloudWatch alarm, or a missing SNS subscription confirmation.

When troubleshooting the notification issues, I investigated each part of the monitoring setup. I checked if CloudTrail was logging the secret access events and whether those logs were being delivered to CloudWatch. I verified that the metric filter was correctly identifying the `GetSecretValue` events. I then ensured that the CloudWatch alarm was triggering based on the metric and that the SNS topic was correctly connected to the alarm. Finally, I confirmed that the SNS topic had a confirmed email subscription and that email notifications were being sent when the alarm state changed.

I initially didn't receive an email before because CloudWatch was configured to use the wrong threshold. Instead of calculating the sum of the number of times the secret was accessed within a set period, it was using the average. This caused the alarm not to trigger as expected when the secret was accessed. Changing the statistic from average to sum ensured that even a single access within the defined time period would be enough to trigger the alarm and send the email notification.

Success!

To validate that our monitoring system can successfully detect an alert when the secret is accessed, I checked the CloudWatch metrics and logs to ensure everything was connected properly. I then accessed my secret one more time to trigger the system. Within 2 to 5 minutes, I received an email notification confirming the access. Our alarm in CloudWatch moved into the "ALARM" state, which proves that the setup is working as expected.

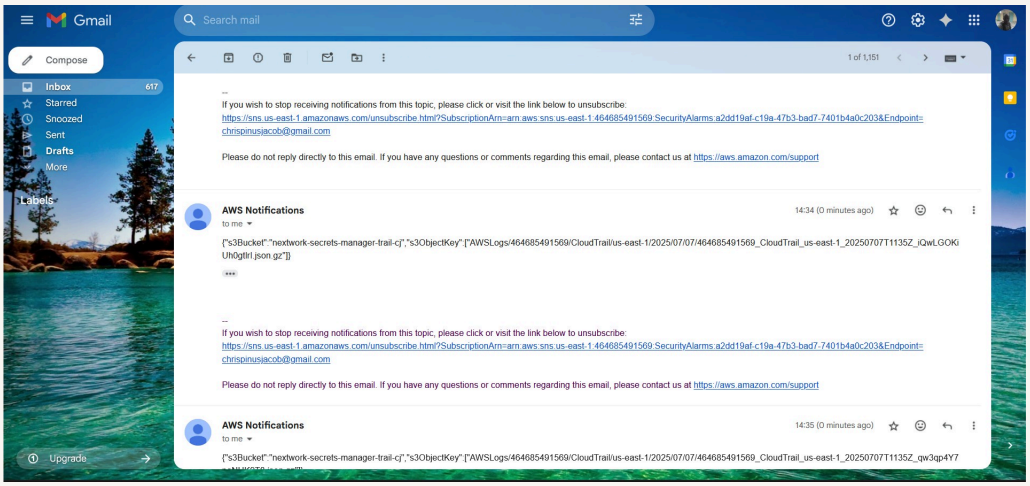


Chrispinus Jacob
NextWork Student

Comparing CloudWatch with CloudTrail Notifications

In a project extension, I enabled SNS notification delivery in CloudTrail because I wanted to test how CloudTrail can directly notify us when certain events occur, such as someone accessing our secret. This lets us evaluate the difference between CloudTrail sending real-time notifications versus using CloudWatch alarms for the same purpose, helping us understand which approach is faster, more flexible, or more suitable for different monitoring needs.

After enabling CloudTrail SNS notifications, my inbox started receiving emails almost instantly whenever the secret was accessed. In terms of the usefulness of these emails, I thought they were very detailed and timely, which is great for real-time alerting. However, they might require additional filtering or processing in a larger system to avoid being overwhelmed by noise if too many events are triggered.


Gmail
Search mail
Compose
Inbox 617
Starred
Snoozed
Sent
Drafts
More
Labels
Upgrade
1 of 1,151
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:464685491569:SecurityAlarms:a2dd19af-c19a-47b3-bad7-7401b4a0c203&Endpoint=chrispinusjacob@gmail.com
Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support
AWS Notifications
to me
14:34 (0 minutes ago)
{"s3Bucket":"nextwork-secrets-manager-trail-cj","s3ObjectKey":["AWSLogs/464685491569/CloudTrail/us-east-1/2025/07/07/464685491569_CloudTrail_us-east-1_20250707T1135Z_iQwLGOKiUh0gtIrl.json.gz"]}
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:464685491569:SecurityAlarms:a2dd19af-c19a-47b3-bad7-7401b4a0c203&Endpoint=chrispinusjacob@gmail.com
Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support
AWS Notifications
to me
14:35 (0 minutes ago)
{"s3Bucket":"nextwork-secrets-manager-trail-cj","s3ObjectKey":["AWSLogs/464685491569/CloudTrail/us-east-1/2025/07/07/464685491569_CloudTrail_us-east-1_20250707T1135Z_qw3qp4Y7

The place to learn & showcase your skills

Check out nextwork.org for more projects