ATTACK DEFENSE

by PentesterAcademy

Name	AWS CloudTrail : Athena and CloudWatch Alerts
URL	
Type	AWS Cloud Security : Defense

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

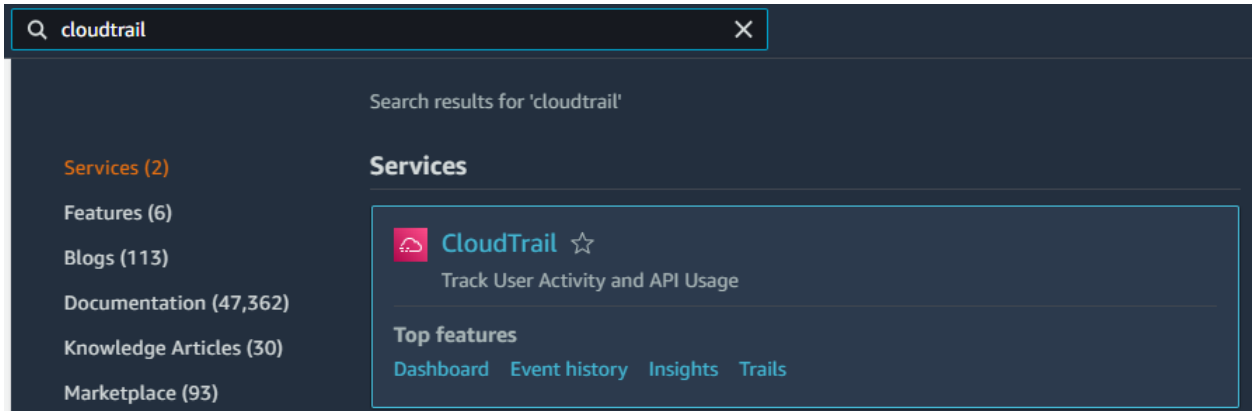
Solution:

Step 1: Click the lab link button to get access credentials.

Access Credentials to your AWS lab Account

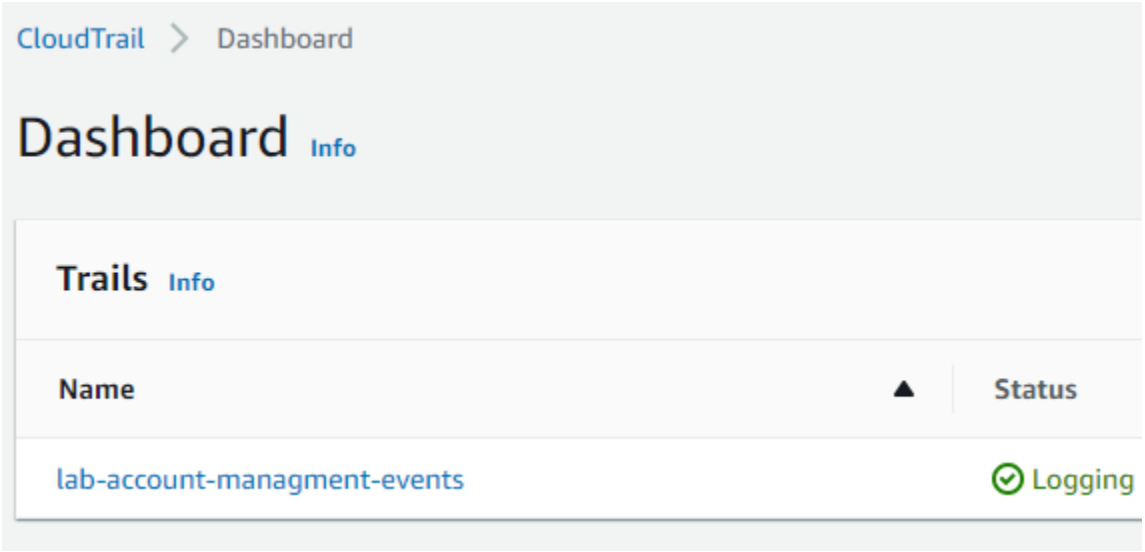
Login URL	https://664289593040.signin.aws.amazon.com/console
Region	US East (N. Virginia) us-east-1
Username	student
Password	Ad67hDB8ZtssUlZ7
Access Key ID	AKIAZVKV6Y3ICK5IJCEM
Secret Access Key	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

Step 2: Search for “CloudTrail” in the search bar and navigate to the CloudTrail dashboard.

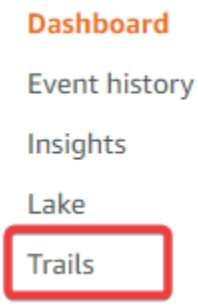


Dashboard will list all the available trails.

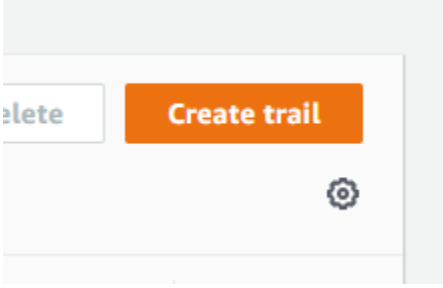
“lab-account-managment-events” trail was created by the management account for the organization and hence cannot edit or delete this trail through this account



Step 3: Click on “Trails” from the navigation pane.



Step 4: Click on the “Create trail” button.



Step 5: Set trail name as “students-events” and choose “Create new S3 bucket” and use the default bucket name.

Trail name

Enter a display name for your trail.

student-events

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 S

Choose an existi

Trail log bucket and folder

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

aws-cloudtrail-logs-664289593040-add4f277

Logs will be stored in aws-cloudtrail-logs-664289593040-add4f277/AWSLogs/664289593040

Disable Log file SSE-KMS encryption and Log file validation.

Log file SSE-KMS encryption Info

Enabled

Additional settings

Log file validation Info

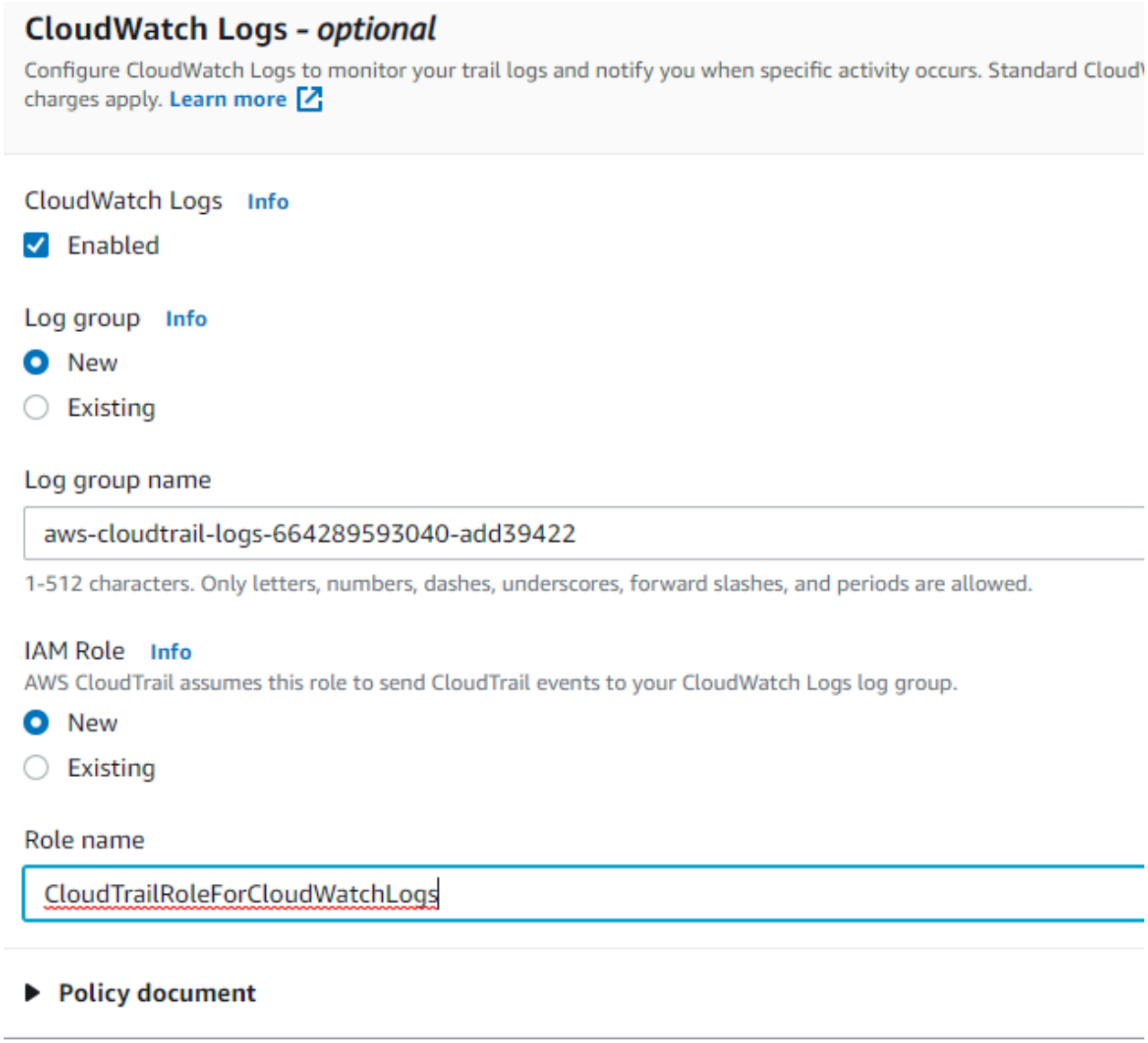
Enabled

SNS notification delivery Info

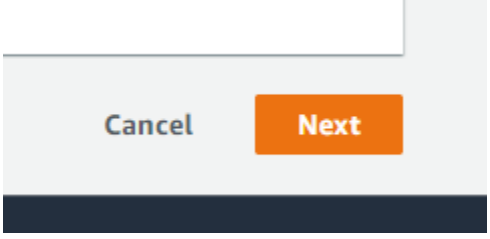
Enabled

Step 6: Enable CloudWatch logs and set choose a new log group and IAM role. Set the role name as “CloudTrailRoleForCloudWatchLogs” and use the default group name.

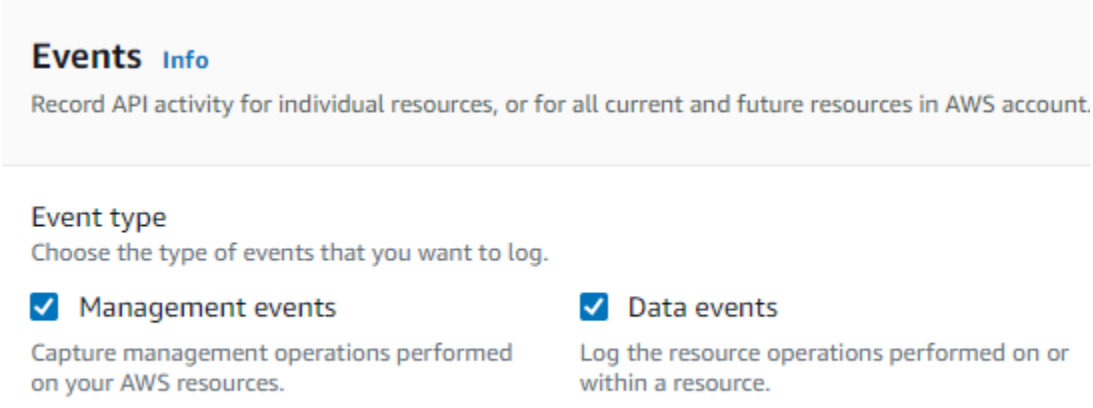
CloudTrail sends only the events that match your trail settings. For example, if you configure your trail to log data events only, your trail sends data events only to your CloudWatch Logs log group. CloudTrail supports sending data, Insights, and management events to CloudWatch Logs.



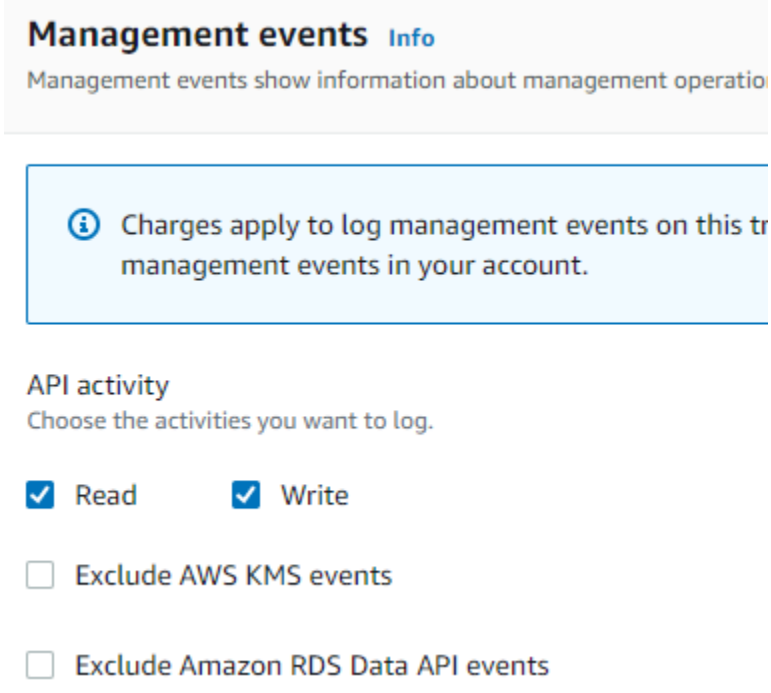
Click on the “Next” button.



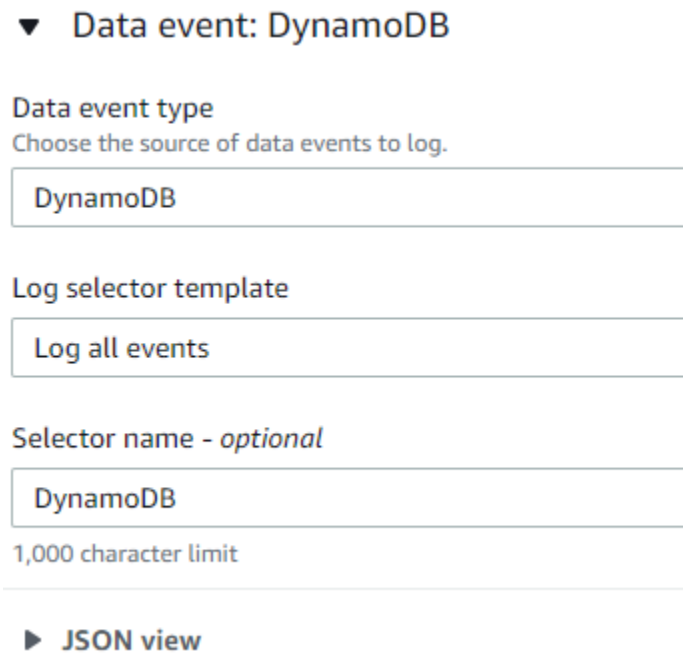
Step 7: Select Management events as well as Data events for the event type.



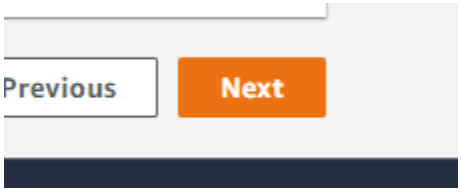
Enable read and write operation API activity logs.



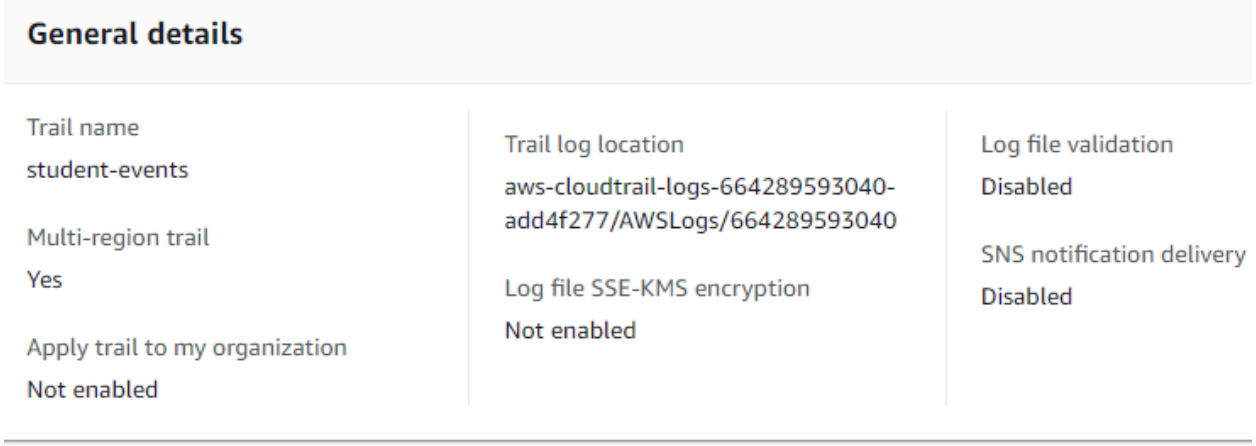
Select DynamoDB as data event type and set “Log all events” for log selector template.



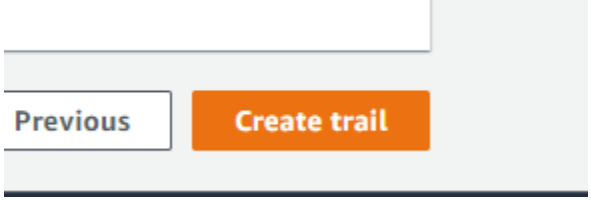
Click on the “Next” button.



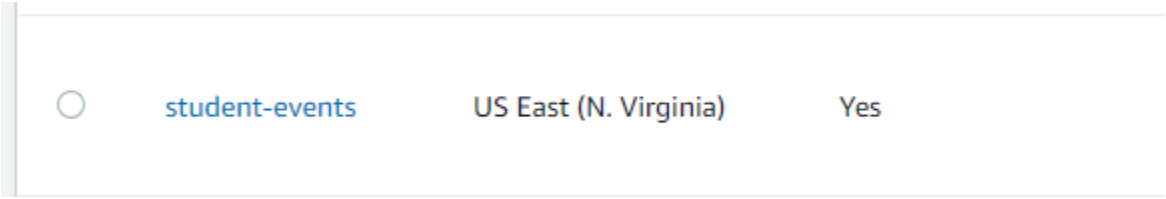
Review the trail configuration.



Click on the “Create trail” button.



Successfully created “student-events” trail. Click on “student-events”.



The details of the created trail will be available here.

General details

Trail logging
Logging

Trail name
student-events

Multi-region trail
Yes

Apply trail to my organization
Not enabled

Trail log location
aws-cloudtrail-logs-664289593040-add4f277/AWSLogs/664289593040

Last log file delivered
-

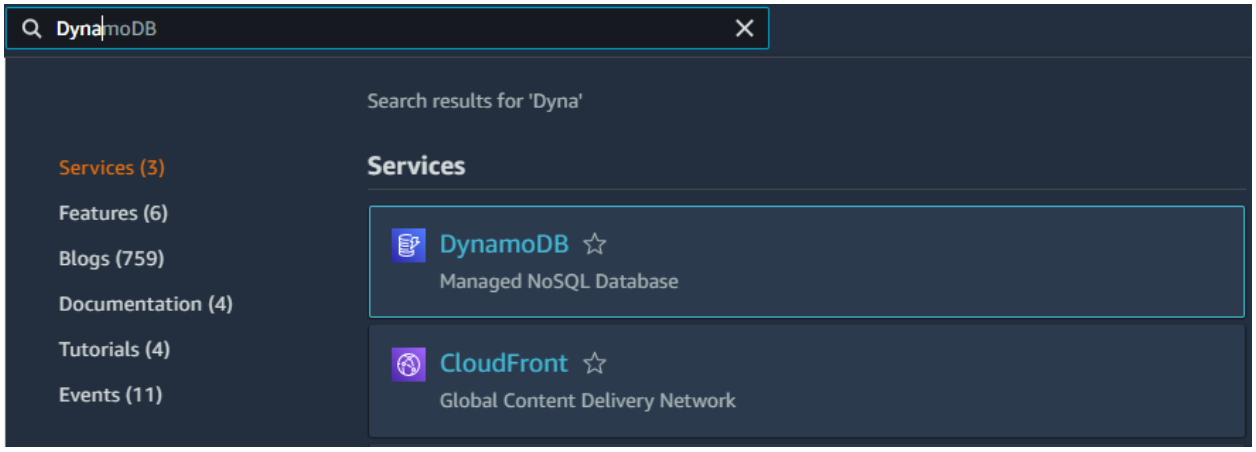
Log file SSE-KMS encryption
Not enabled

Check out the CloudWatch log group name. The created trail will 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.

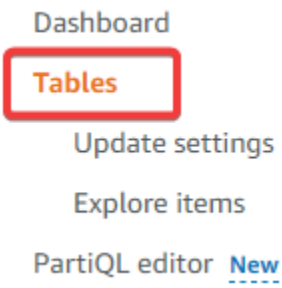
CloudWatch Logs

Log group
aws-cloudtrail-logs-664289593040-add39422

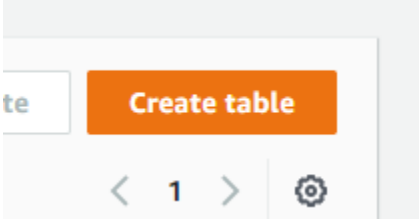
Step 8: Create or modify some resources to generate logs. Search for "DynamoDB" and navigate to the DynamoDB dashboard.



Step 9: Click on “Tables” from the navigation pane.



Click on the “Create table” button.



Step 10: Set table name as “Users” and partition key as “id”.

Table name

This will be used to identify your table.

Users

Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).

Partition key

The partition key is part of the table's primary key. It is a hash value that is used to retrieve items frc hosts for scalability and availability.

id | Number

1 to 255 characters and case sensitive.

Sort key - optional

Click on the “Create table” button.

Cancel | Create table

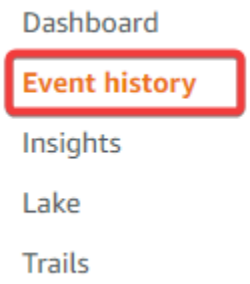
Successfully created the table named “Users”.

Tables (1) Info

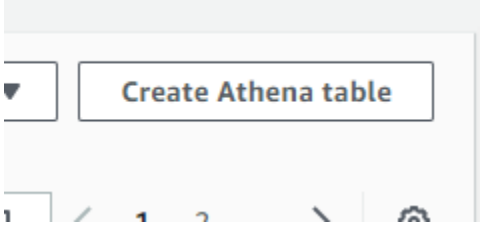
Find tables by table name

Name	Status	Partition key
Users	Active	id (N)

Step 11: Navigate back to the CloudTrail dashboard and click on “Event history” from the navigation pane.



Step 12: Click on the “Create Athena table” button.



Set Athena to query these log files directly from Amazon S3 by specifying the location of log files.

Step 13: Choose the same S3 bucket which contains CloudTrail log files.

CloudTrail saves logs as JSON text files in compressed gzip format (*.json.gzip). The location of the log files depends on how you set up trails, the AWS Region or Regions in which you are logging, and other factors.

Create a table in Amazon Athena

You can use Amazon Athena to analyze events that are stored in a trail's Amazon S3 bucket. Athena is an interactive c
you analyze data in S3 buckets by using standard SQL. Athena charges for running queries. Learn more

Storage location

aws-cloudtrail-logs-664289593040-add4f277

Choose an S3 bucket that contains CloudTrail log files

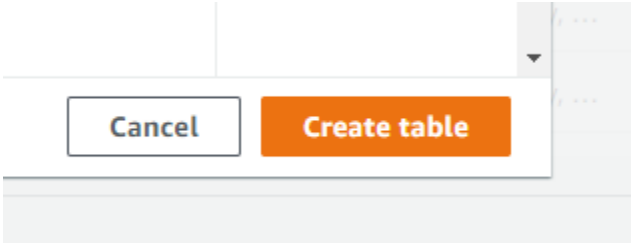
Athena table name

cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277

This name is auto-generated. You can rename it in Amazon Athena.

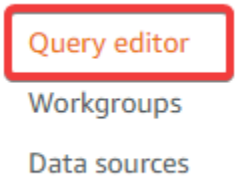
```
1 CREATE EXTERNAL TABLE
  cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277 (
2     eventVersion STRING,
```

Click on the “Create table” button.

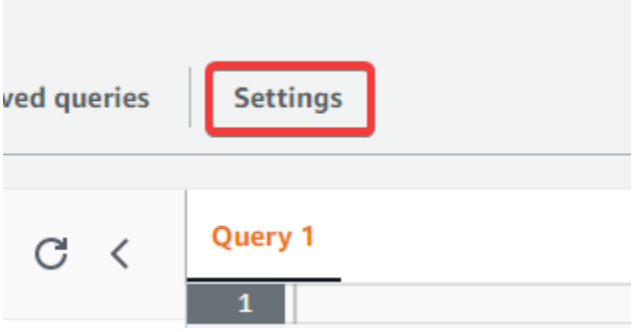


The table is created with a default name that includes the name of the Amazon S3 bucket. Navigate to the Athena dashboard in the new tab using the hyperlink.

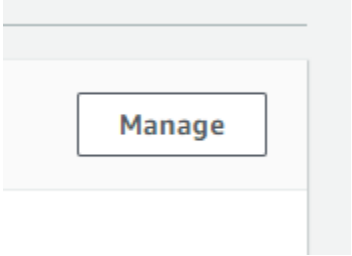
Step 14: Set the query result location. Click on “Query editor” from the navigation pane.



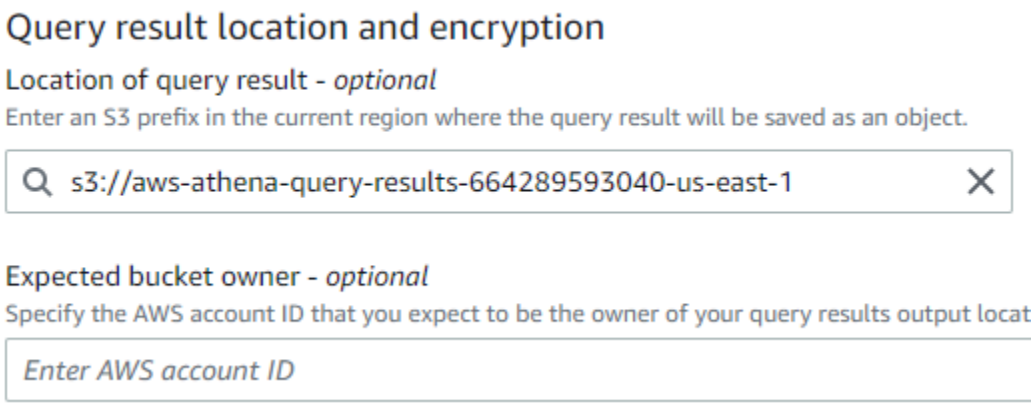
From the query editor, click on “Settings”.



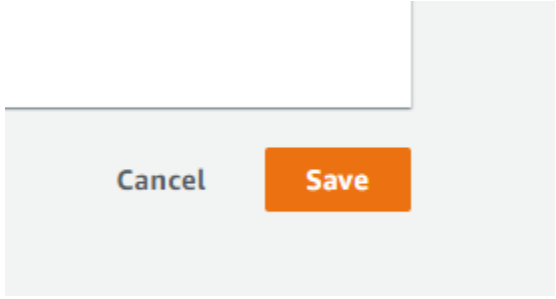
Click on the “Manage” button.



Step 15: Choose a bucket for the query results.

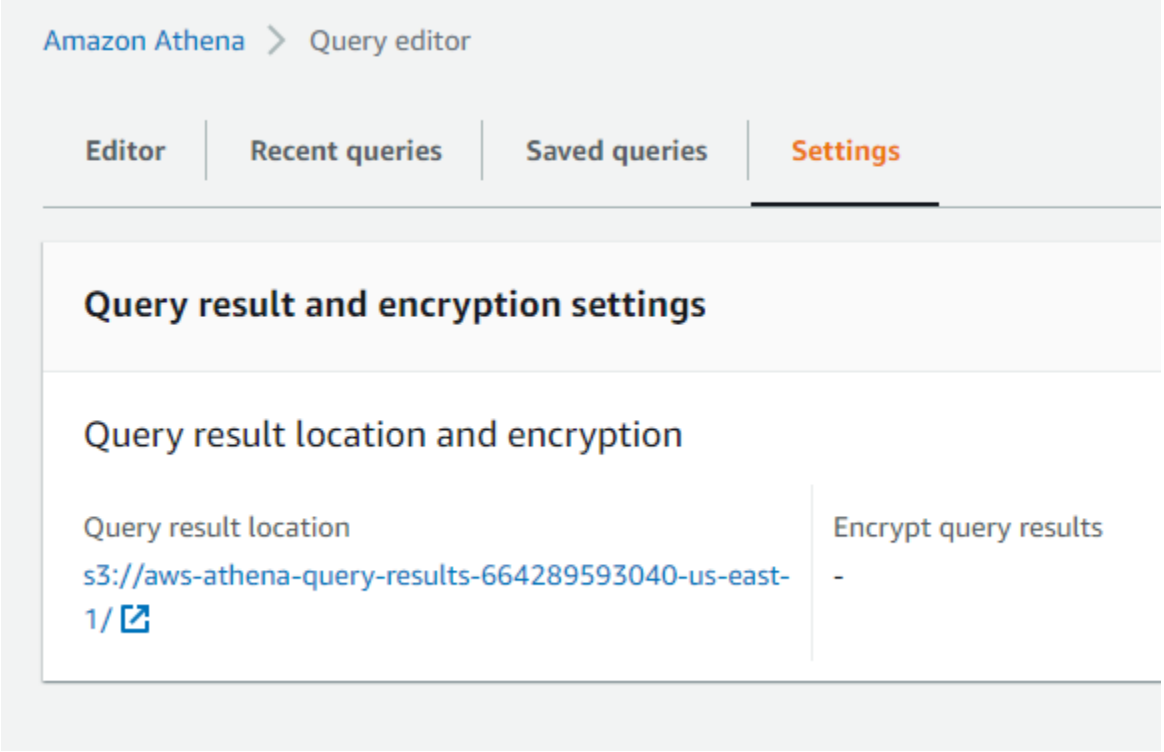


Click on the “Save” button.



Successfully set the query result location.

Amazon Athena automatically stores query results and metadata information for each query that runs in a query result location that you can specify in Amazon S3. If necessary, you can access the files in this location to work with them. You can also download query result files directly from the Athena console.



Step 16: Copy and paste the query and click on “RUN” to get the ‘UpdateTable’ and ‘CreateTable’ event details.

Query:

```
SELECT
 eventtime,
 eventsource,
 useridentity.arn,
 sourceipaddress
FROM cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277
WHERE eventname = 'UpdateTable'
OR  eventname = 'CreateTable'
```

Query 1

```
SELECT
 eventtime,
 eventsource,
 useridentity.arn,
 sourceipaddress
FROM cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277
WHERE eventname = 'UpdateTable'
OR  eventname = 'CreateTable'
```

Successfully got the query result showing the events.

Query results | Query stats

Completed — Time in queue: 122 ms — Run time: 631 ms — Data scanned: 107.71 KB

Results (3) — Copy — Download results

Search rows — 1

#	eventtime	eventsource	arn	sourceipaddress
1	2022-09-14T09:36:26Z	dynamodb.amazonaws.com	arn:aws:iam::664289593040:user/student	application-autoscaling.amazonaws.com
2	2022-09-14T09:36:25Z	dynamodb.amazonaws.com	arn:aws:iam::664289593040:user/student	123.178.204.43
3	2022-09-14T09:39:23Z	glue.amazonaws.com	arn:aws:iam::664289593040:user/student	glue.amazonaws.com

Step 17: Now list a few details from all the CloudTrail logs and create a new table from the result.

Query:

```
SELECT
 eventtype,
 eventtime,
 useridentity.arn,
 sourceipaddress
FROM cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277
```

aved queries | Settings

Query 1

```
SELECT
 eventtype,
 eventtime,
 useridentity.arn,
 sourceipaddress
FROM cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277
```

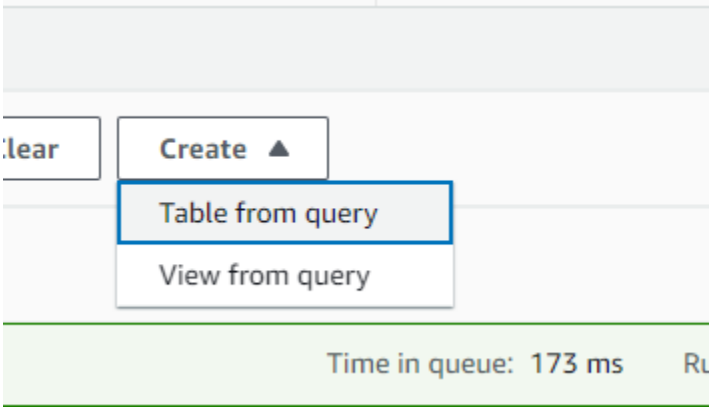
Successfully got the query result showing all the logs matching the query.

Results (100+)

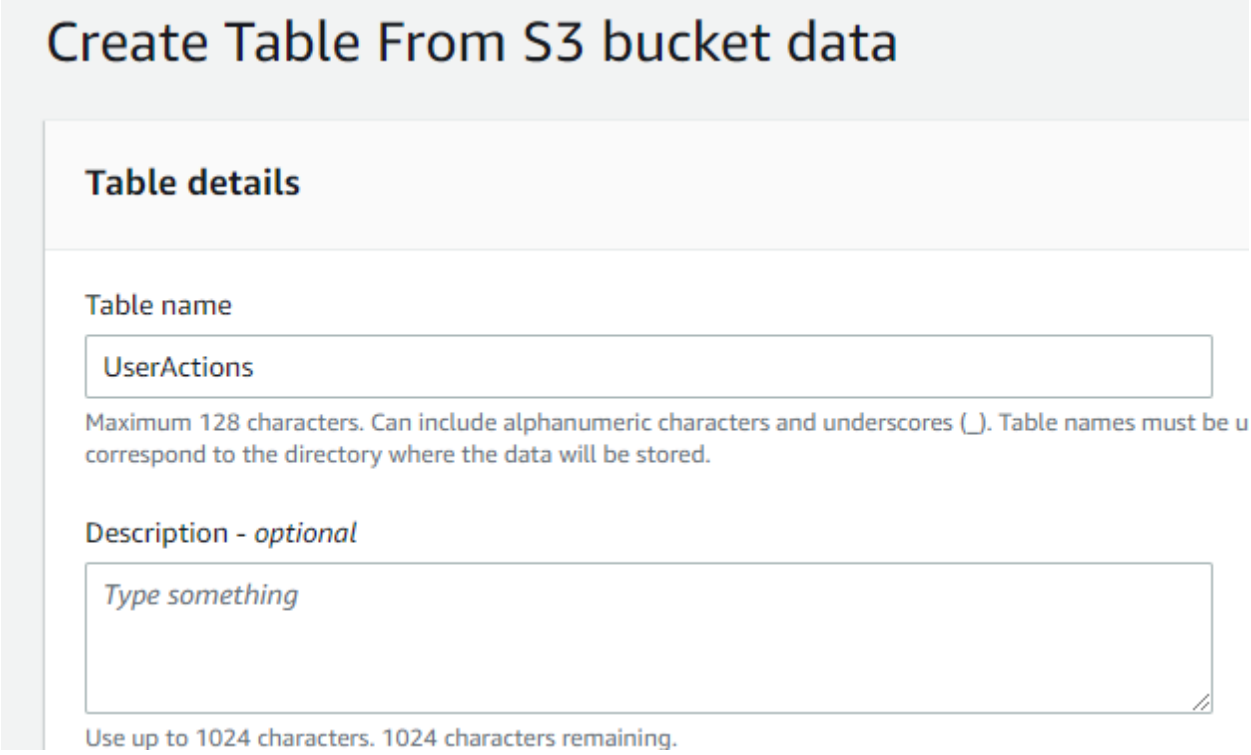
Search rows

#	eventtype	eventtime	arn
1	AwsApiCall	2022-09-14T10:39:37Z	arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSession
2	AwsApiCall	2022-09-14T11:24:40Z	arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSession
3	AwsApiCall	2022-09-14T09:37:39Z	arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSession
4	AwsApiCall	2022-09-14T09:38:40Z	arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSession
5	AwsApiCall	2022-09-14T09:40:39Z	arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSession
6	AwsApiCall	2022-09-14T09:41:40Z	arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSession

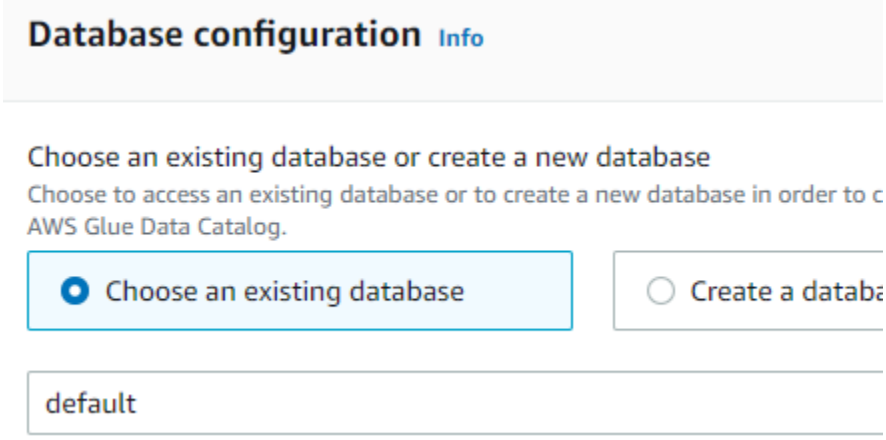
Step 18: Click on “Table from query” under the “Create” button.



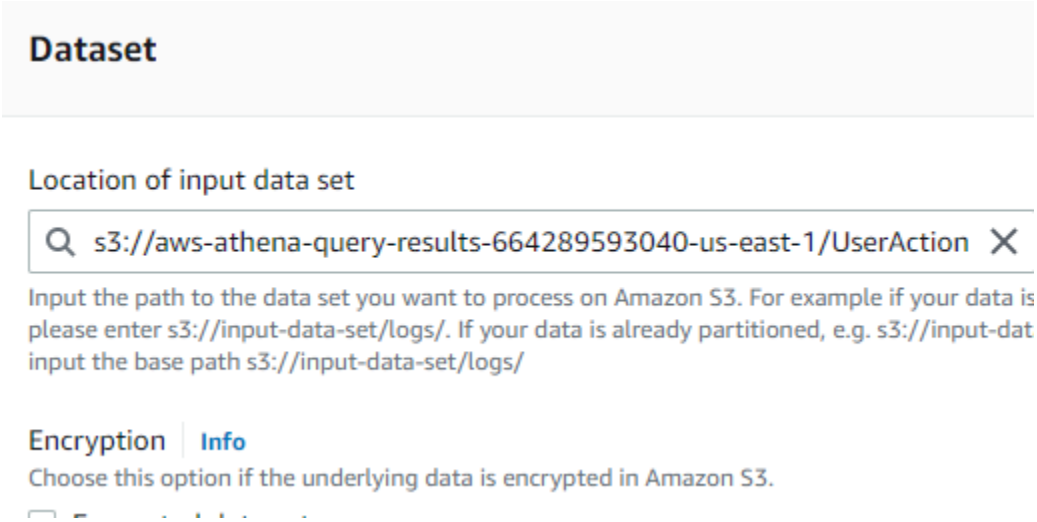
Step 19: Set table name as “UserActions”.



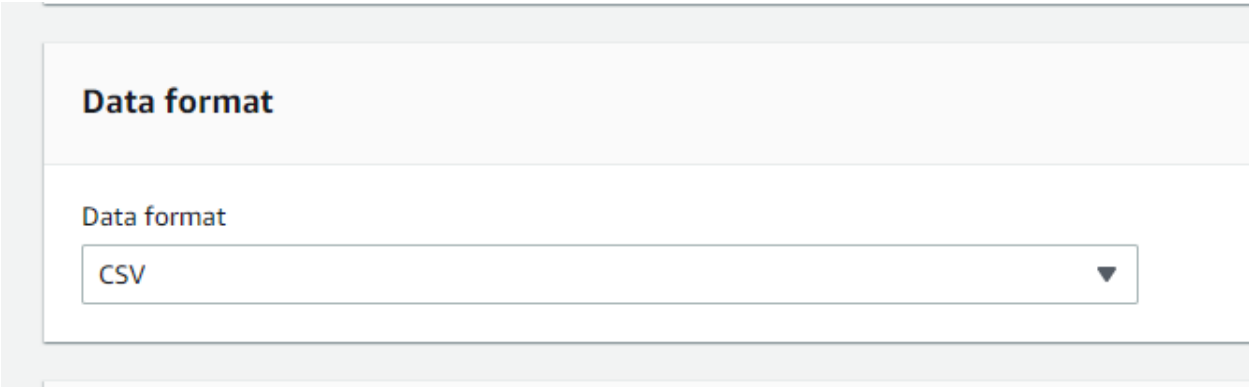
Step 20: Select “Choose an existing database” for Database configuration.



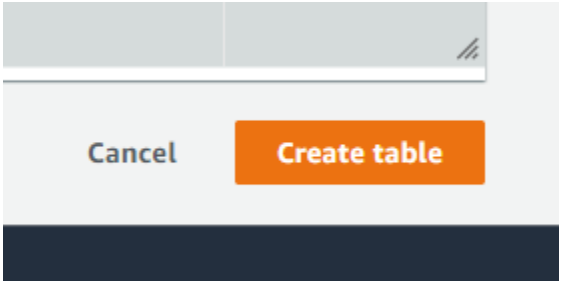
Step 21: Append "/UserAction" to the S3 URI to create a new directory and set it as the input data location.



Choose CSV as the format.



Click on the “Create table” button.



It will generate a query similar to the following. Click on the “RUN” button.

Query:

```
CREATE TABLE "default"."UserActions" WITH (
  format = 'TEXTFILE',
  external_location = 's3://aws-athena-query-results-664289593040-us-east-1/UserActions'
) AS
SELECT eventtype,
  eventtime,
  useridentity.arn,
  sourceipaddress
FROM cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277
```

Query 1 ✕ | Query 2 ✕

```
CREATE TABLE "default"."UserActions" WITH (
  format = 'TEXTFILE',
  external_location = 's3://aws-athena-query-results-664289593040-us-east-1/UserActions'
) AS
SELECT eventtype,
  eventtime,
  useridentity.arn,
  sourceipaddress
FROM cloudtrail_logs_aws_cloudtrail_logs_664289593040_add4f277
```

Successfully saved query results to the S3 bucket.

Query results | Query stats

Completed

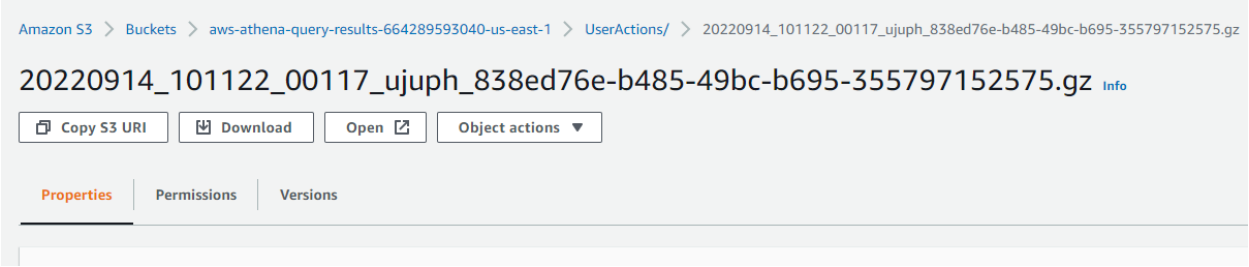
Query successful.

Step 22: Navigate to the S3 bucket location.

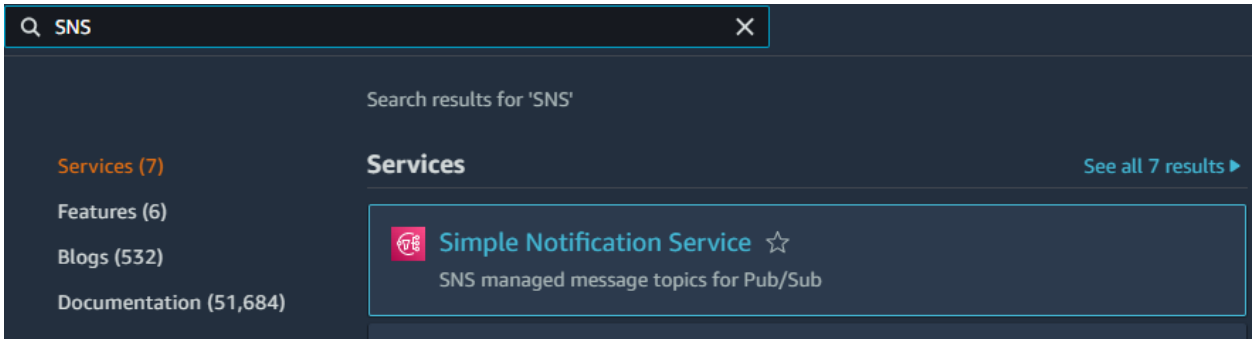
Find objects by prefix

Name
aws-cloudtrail-logs-664289593040-add4f277/
Unsaved/
UserActions/

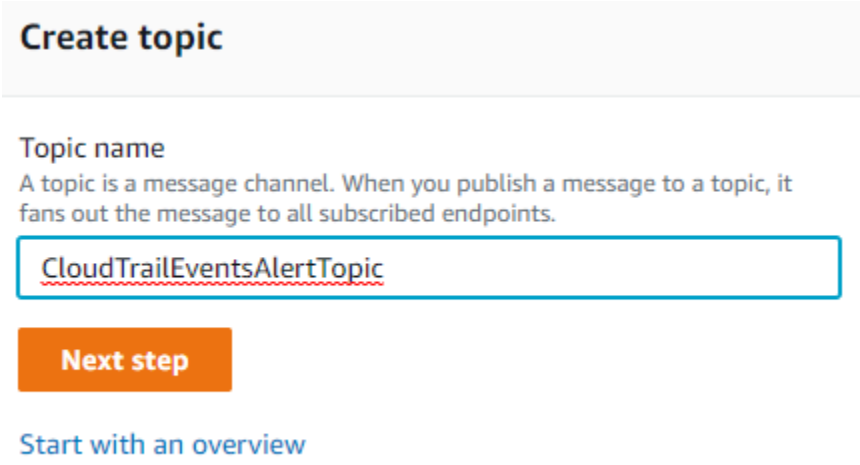
Click on any object and download or open it to view the log details in CSV format.



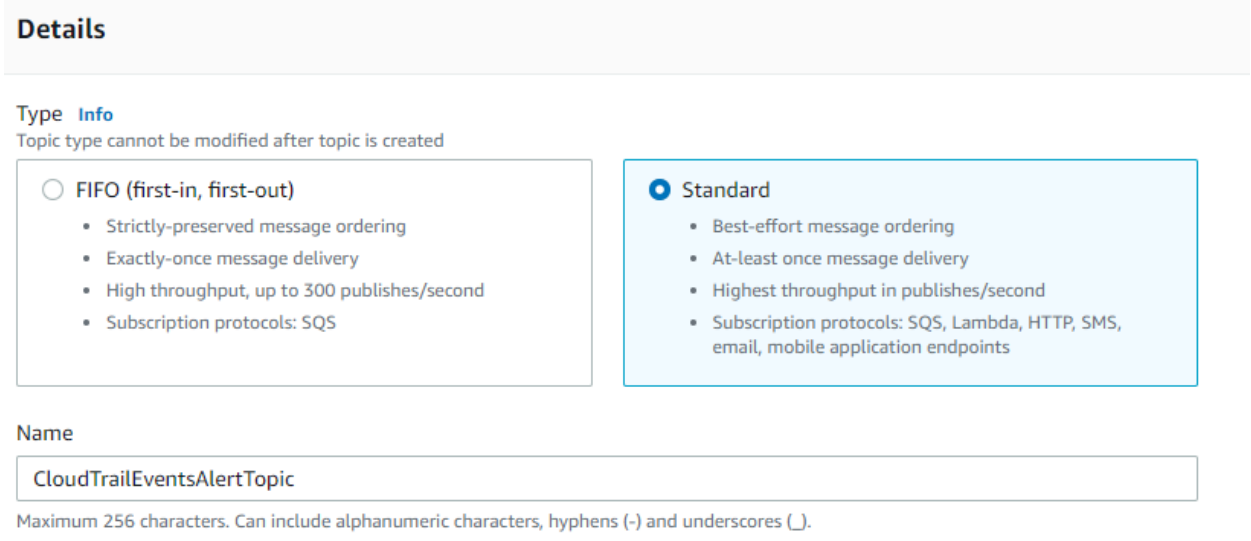
Step 23: Search for SNS in the search bar and navigate to the SNS dashboard.



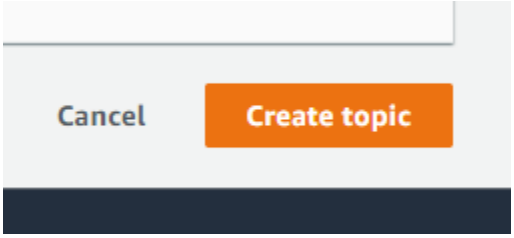
Step 24: Set topic name as “CloudTrailEventsAlertTopic”.



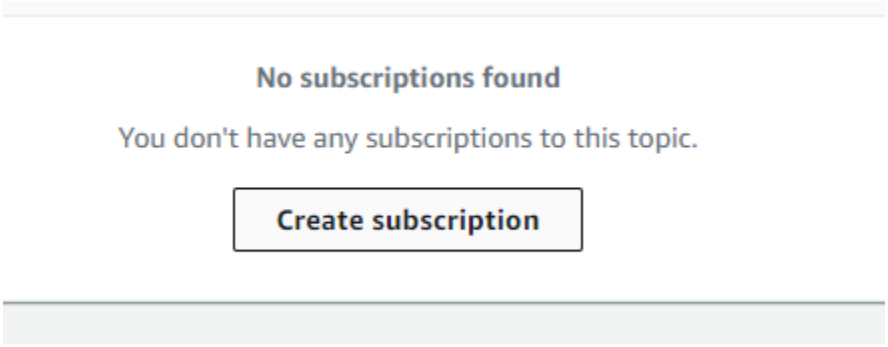
Choose type as “Standard”.



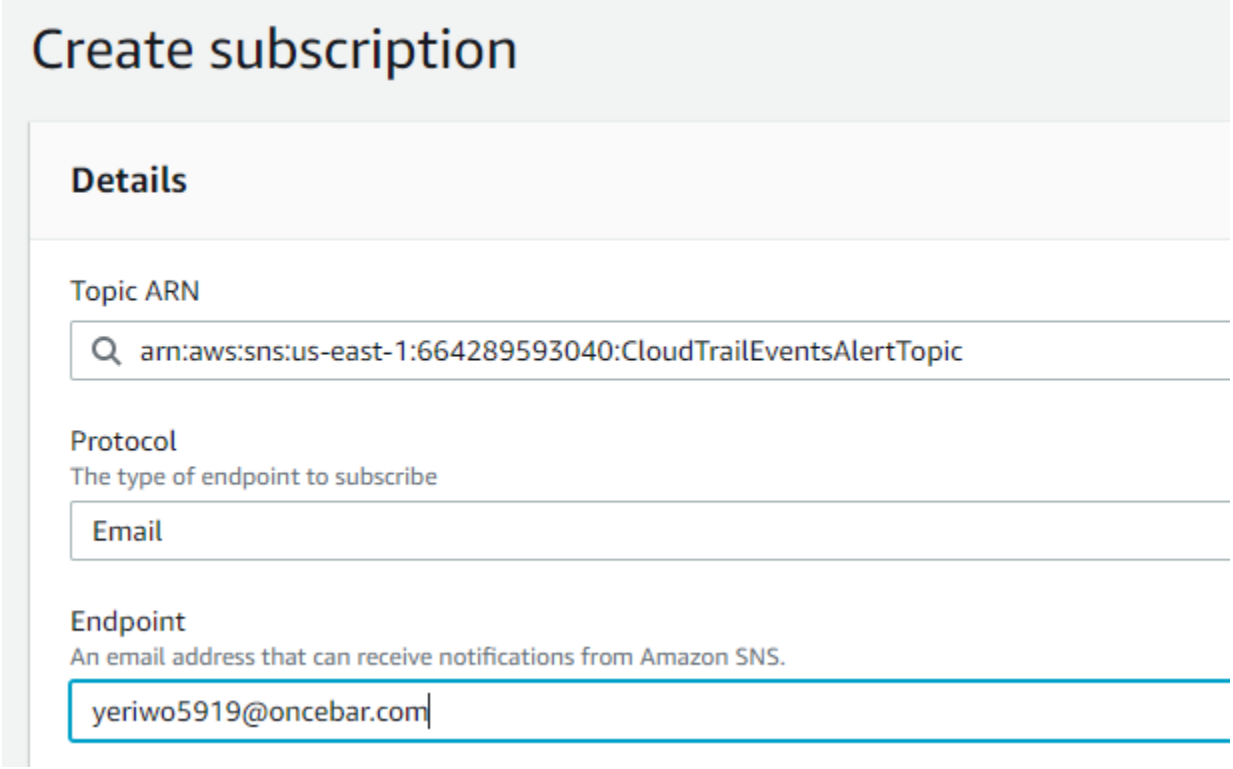
Click on the “Create topic” button.



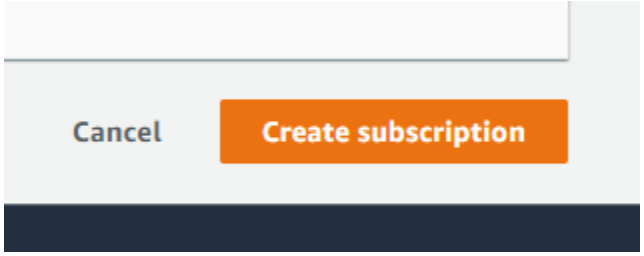
Step 25: Now create a subscription for the created topic. Click on the “Create subscription” button.



Set protocol as “Email” and enter your email address in the endpoint field.



Click on the “Create subscription” button.



A subscription confirmation email will be available at the provided email address. Click on the “Confirm subscription” link to confirm the email address.

AN

AWS Notifications
no-reply@sns.amazonaws.com

Date:
14-09-2022 15:49:34

Subject: AWS Notification - Subscription Confirmation

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:664289593040:CloudTrailEventsAlertTopic
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 email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to sns-opt-out

Successfully confirmed the subscription.

aws

Simple Notification Service

Subscription confirmed!

You have successfully subscribed.

Your subscription's id is:
`arn:aws:sns:us-east-1:664289593040:CloudTrailEventsAlertTopic:2def950f-37ca-4d5e-aded-ee8993a99caf`

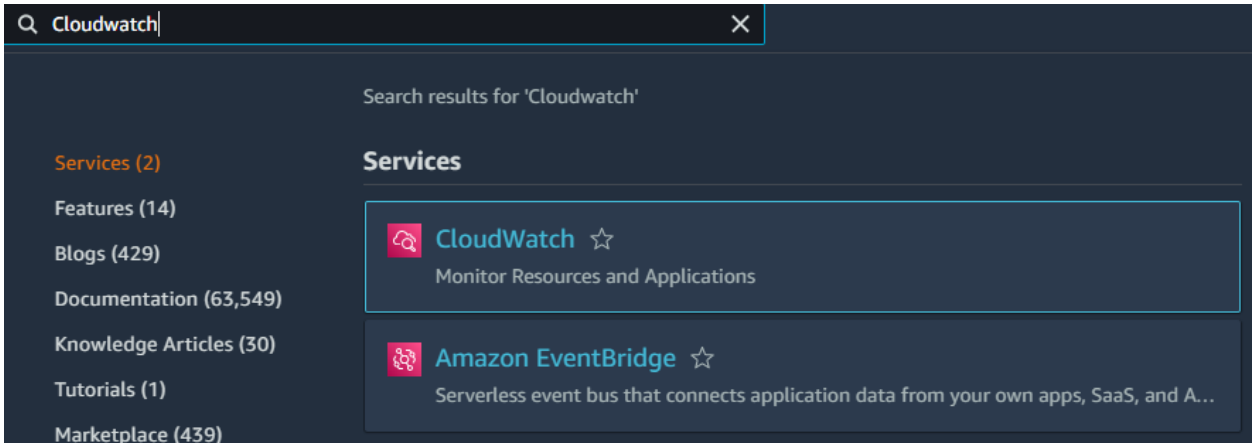
If it was not your intention to subscribe, click here to unsubscribe.

In this lab, we will configure and generate notifications using two methods.

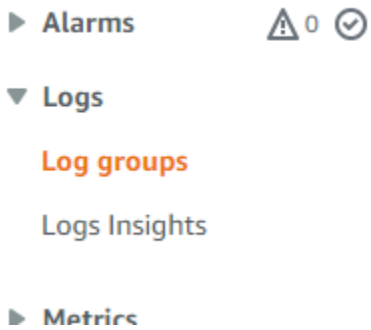
In the first method, we will define CloudWatch logs metric filters to evaluate log events that match the pattern and use the asterisk ("*") as a wildcard to match text such that it will match all the generated events. Then we will set the alarm threshold value as 1 and the alarm condition as “greater than or equal” so that it will create an alarm for every matched event.

In the second method, we will set up a lambda function triggered by Cloudwatch logs and parse the log using a python script and send a notification from the lambda function using the SNS service such that it will send a notification for all the events generated by CloudTrail.

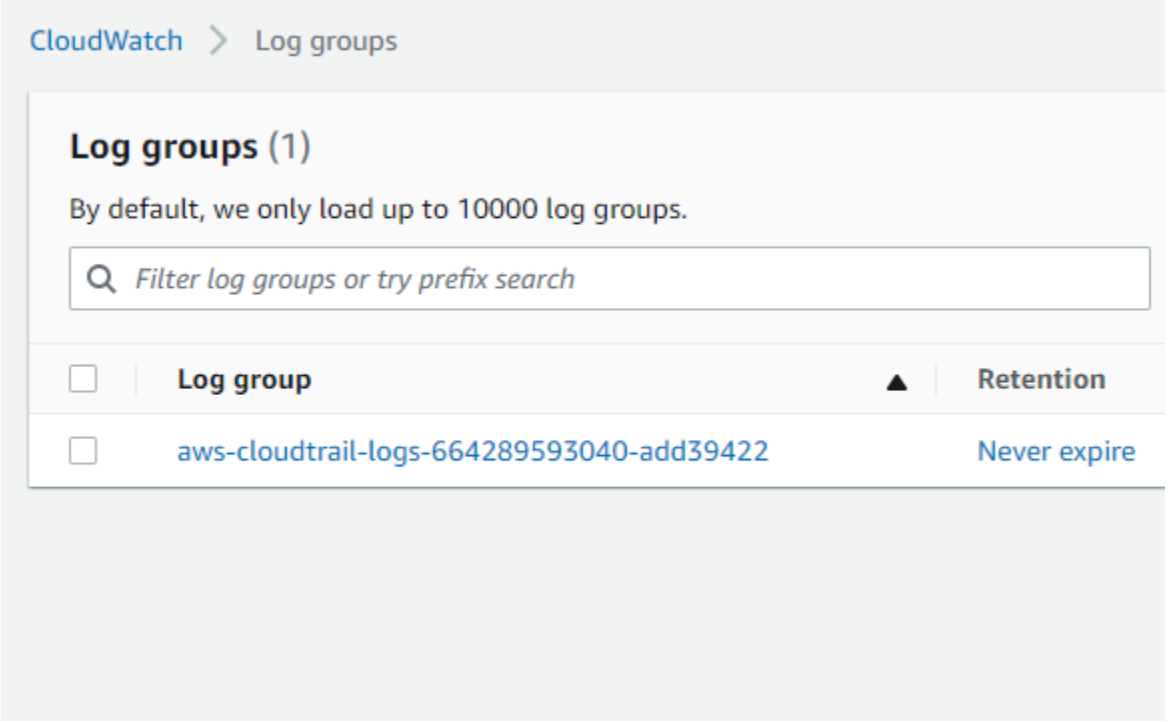
Step 26: Search for CloudWatch in the search bar and navigate to the CloudWatch dashboard.



Step 27: Click on “Log groups” from the navigation pane.

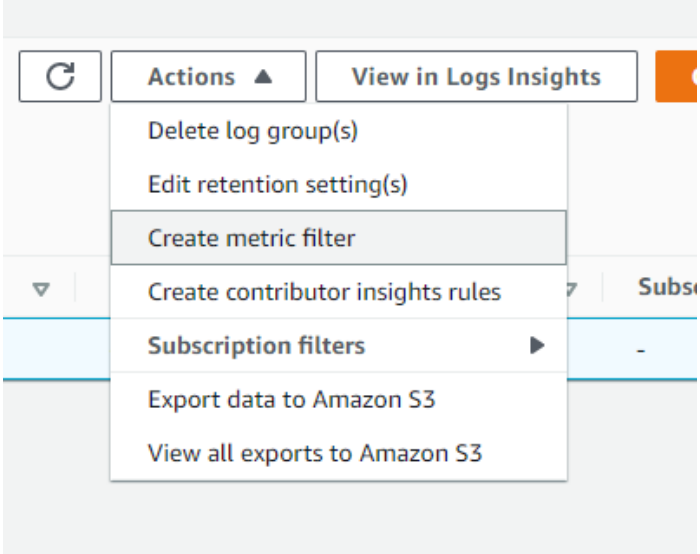


This will list the log group which is created while creating the CloudTrail trail. A log group is a group of log streams that share the same retention, monitoring, and access control settings.



Step 28: Select the log group and click on “Create metric filter” under “Actions”.

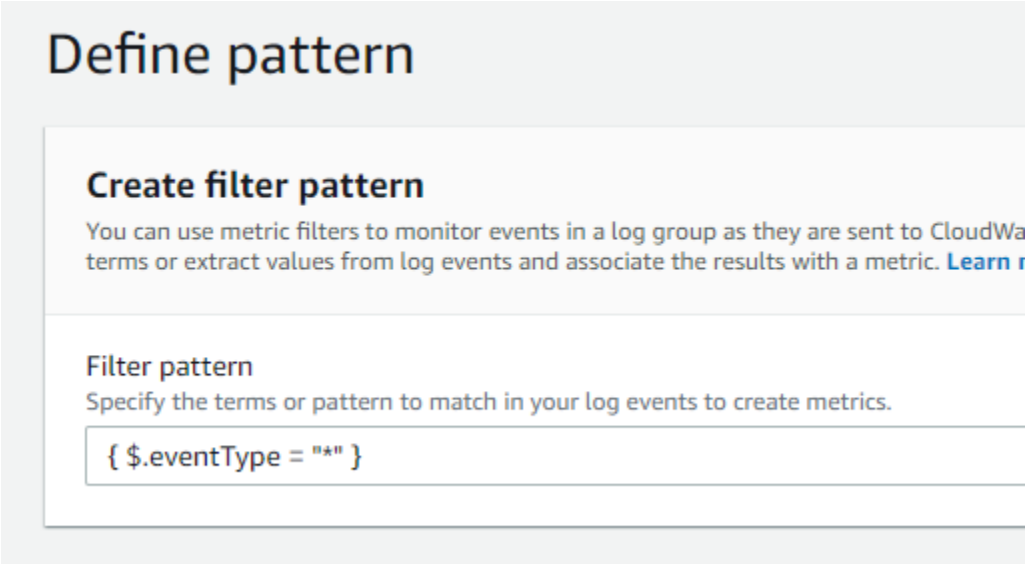
Metric filters define the terms and patterns to look for in log data as it is sent to CloudWatch Logs. CloudWatch Logs uses these metric filters to turn log data into numerical CloudWatch metrics that you can graph or set an alarm on.



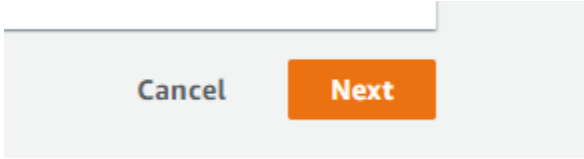
Step 29: Copy and paste the following as a filter pattern.

Filter patterns make up the syntax that metric filters use to match terms in log events. Terms can be words, exact phrases, or numeric values. Here will use the asterisk ("*") as a wildcard to match text such that it will match all the generated events.

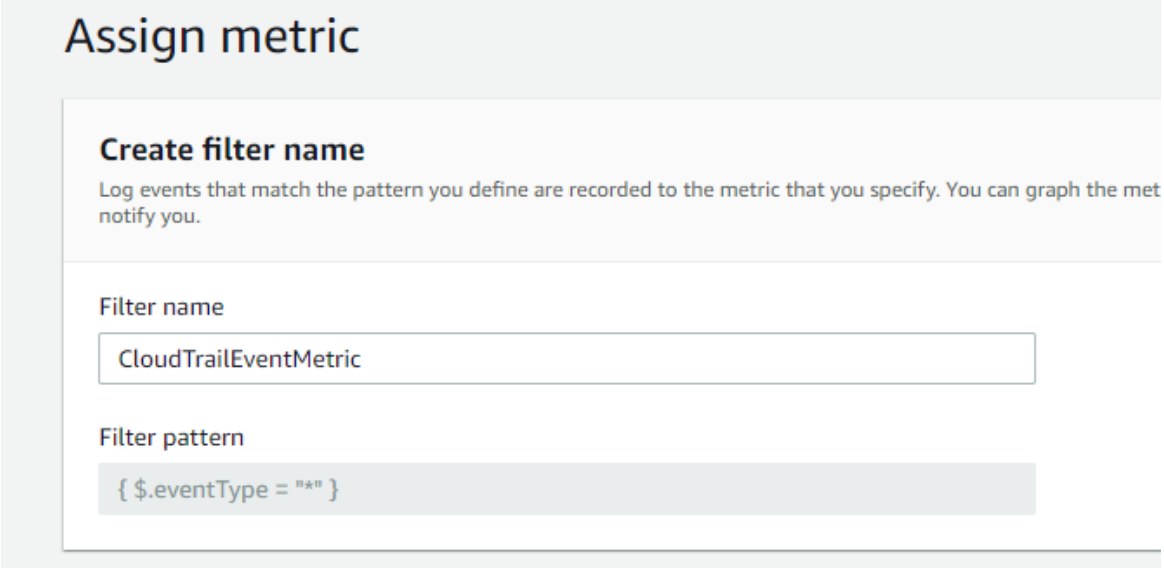
Pattern: { $.eventType = "*" }



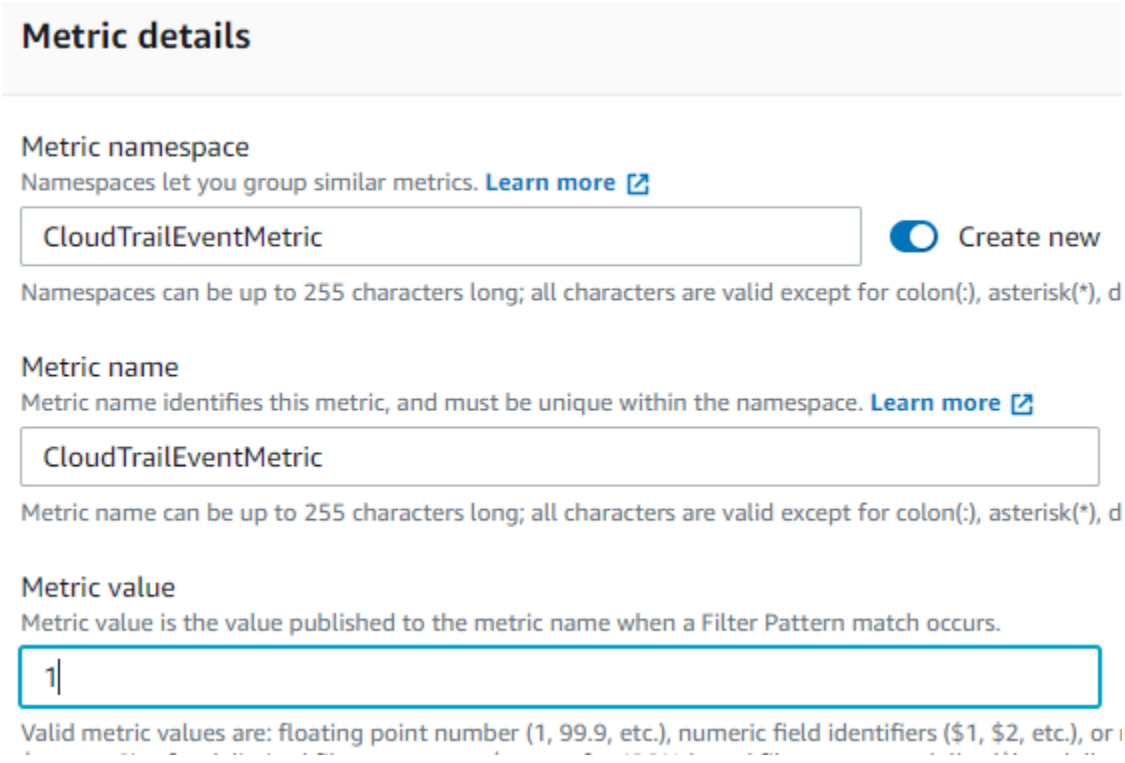
Click on the “Next” button.



Step 30: Set filter name as “CloudTrailEventMetric”.

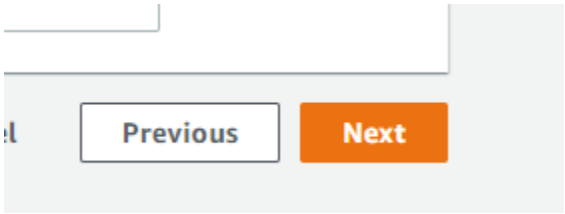


Step 31: Again set name and namespace as “CloudTrailEventMetric”.Set the Metric value as 1.

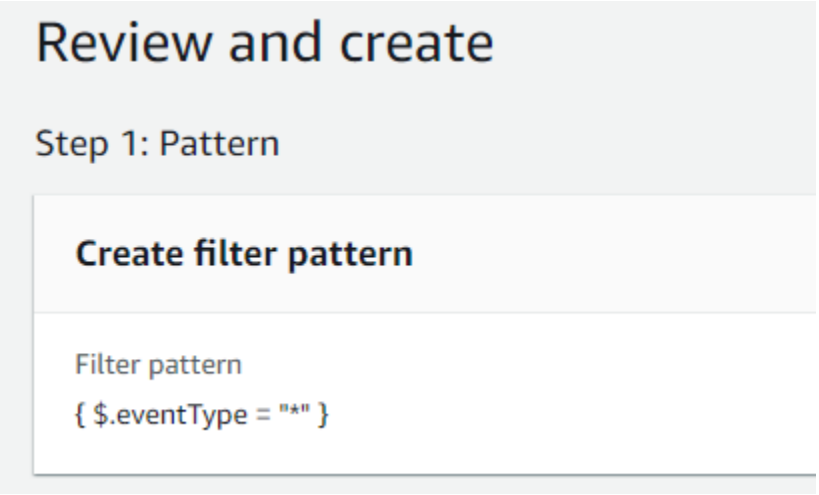


When your metric filter finds a match in your log events, it increments your metric's count by your metric's value. If your metric filter doesn't find a match, CloudWatch reports the metric's default value. For example, your log group publishes two records every minute, the metric value is 1, and the default value is 0. If your metric filter finds matches in both log records within the first minute, the metric value for that minute is 2.

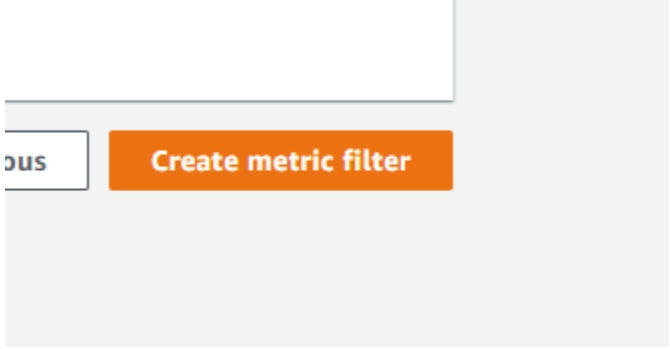
Click on the “Next” button.



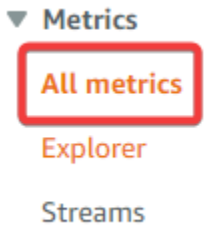
Review the metric filter configuration.



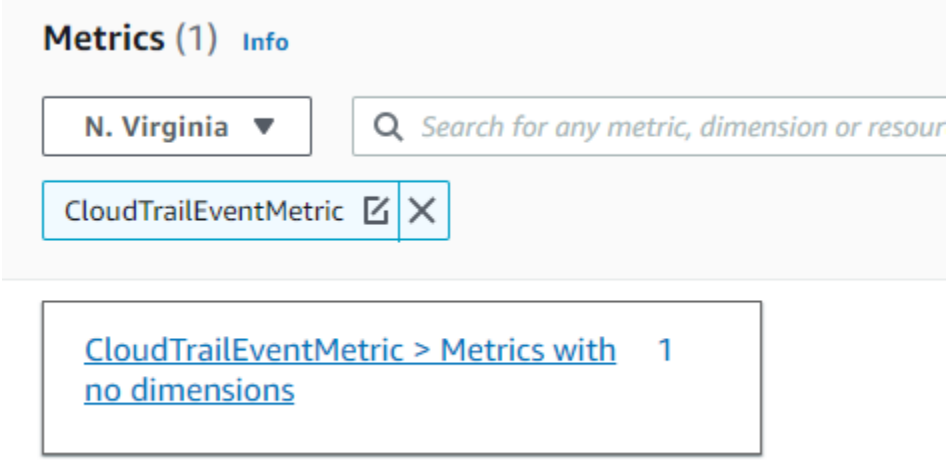
Click on the “Create metric filter” button.



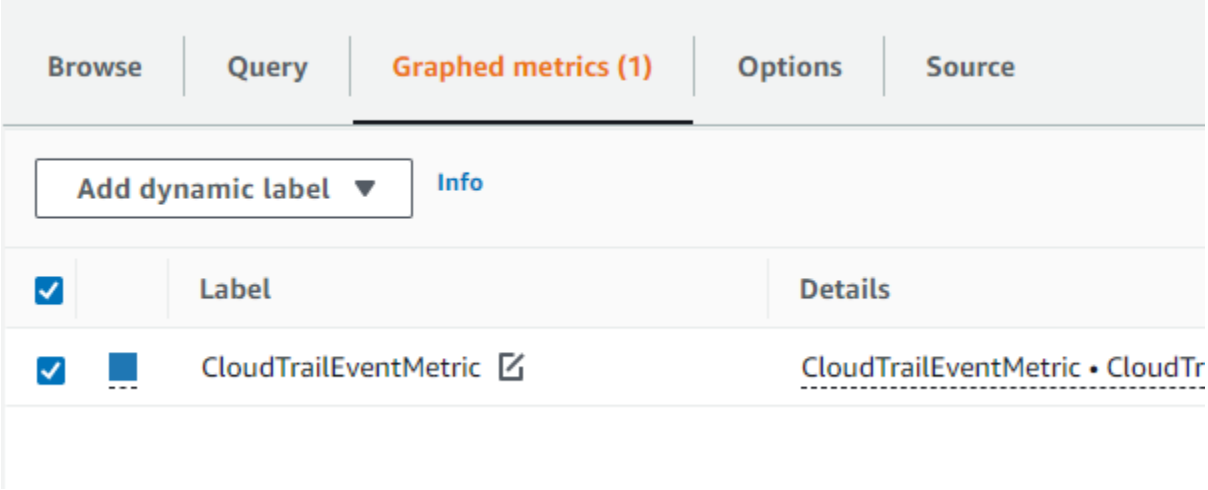
Step 32: Click on “All metrics” under the Metrics section from the navigation pane.



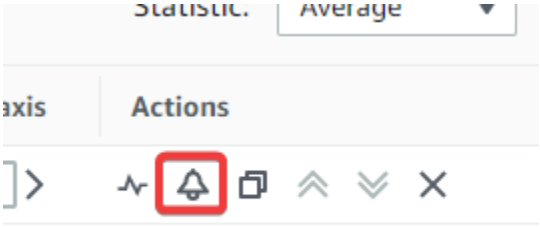
Search for “CloudTrailEventMetric” and select the metrics.



Navigate to “Graphed metrics” and select “CloudTrailEventMetric”.

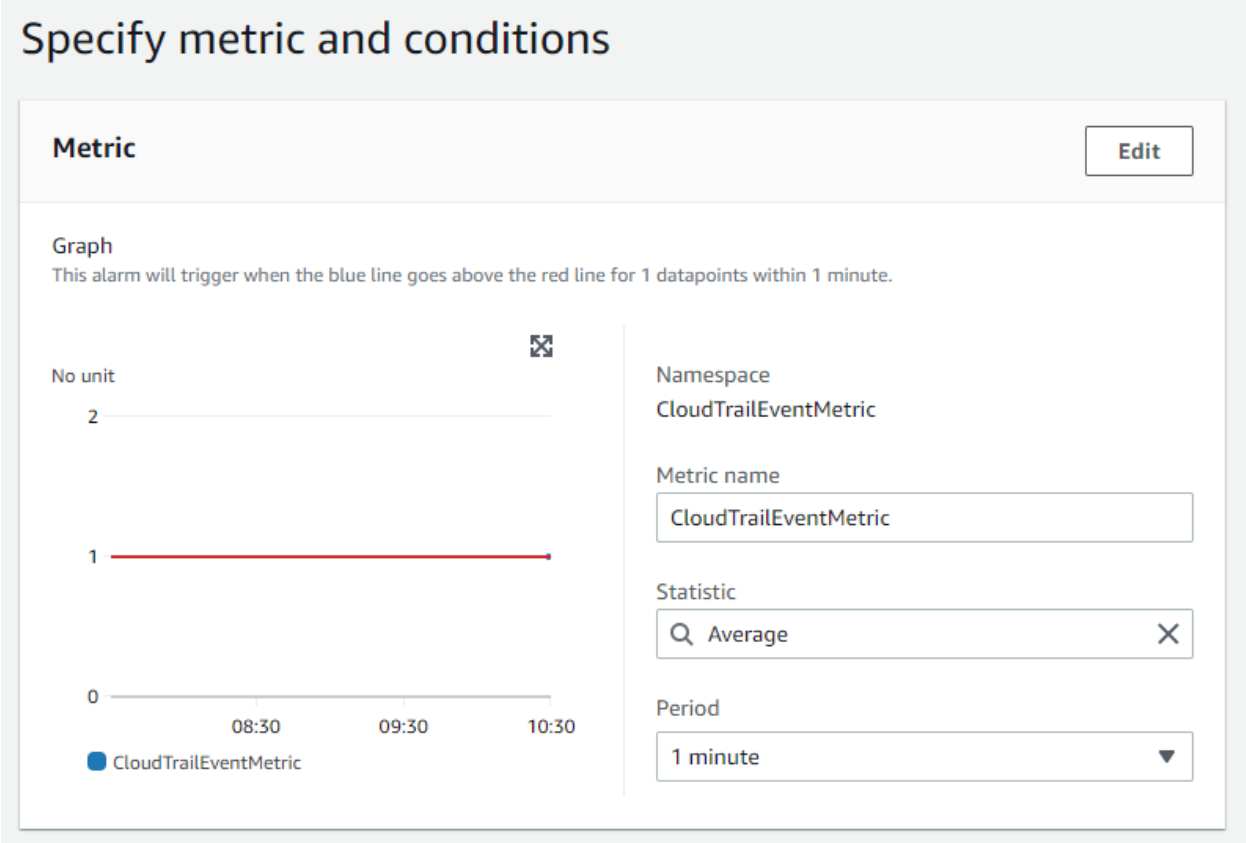


Click on the bell icon to create an alarm.

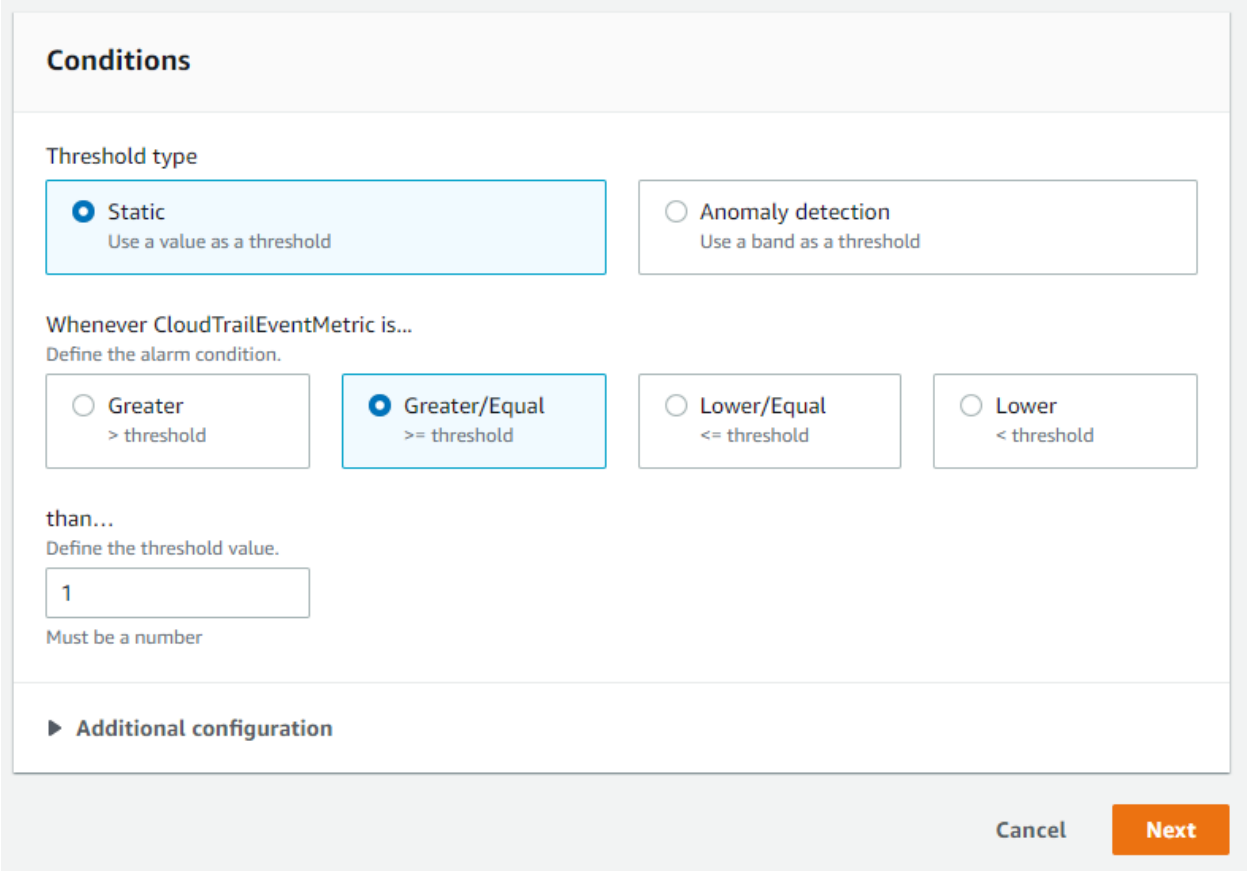


CloudWatch Alarms feature allows you to watch CloudWatch metrics and to receive notifications when the metrics fall outside of the levels (high or low thresholds) that you configure.

Step 33: Set metric name as “CloudTrailEventMetric” and statistic to “Average”. Set period as 1 minute.



Step 34: Set threshold type as “Static” and alarm condition as “Greater/Equal”. Set threshold value as 1.



A CloudWatch Alarm is always in one of three states: OK, ALARM, or INSUFFICIENT_DATA. When the metric is within the range that you have defined as acceptable, the Monitor is in the OK state. When it breaches a threshold it transitions to the ALARM state.

Step 35: Choose “In alarm” as an alarm state trigger. Set “Select an existing SNS topic” for the SNS topic and select the created topic name.

.

Alarm state trigger

Define the alarm state that will trigger this action.

In alarm
The metric or expression is outside of the defined threshold.

OK
The metric or expression is within the defined threshold.

Send a notification to the following SNS topic

Define the SNS (Simple Notification Service) topic that will receive the notification.

Select an existing SNS topic

Create new topic

Use topic ARN to notify other accounts

Send a notification to...

CloudTrailEventsAlertTopic

Only email lists for this account are available.

Email (endpoints)

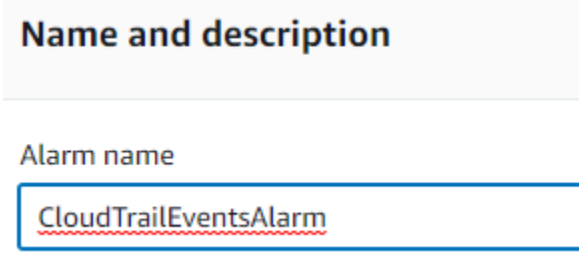
yeriwo5919@oncebar.com - View in SNS Console

Add notification

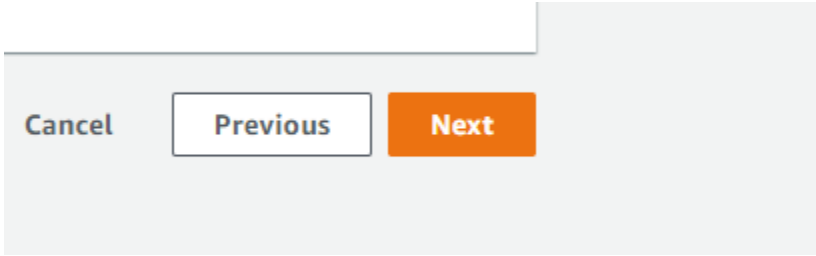
Click on the “Next” button.

Cancel Previous Next

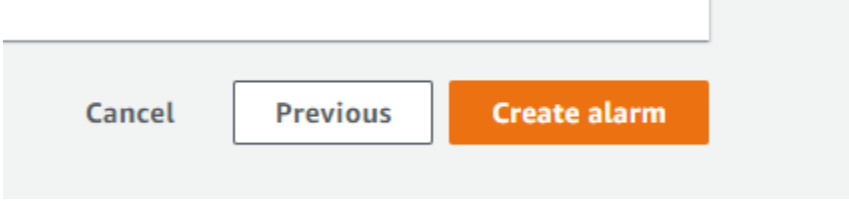
Set the alarm name as “CloudTrailEventsAlarm”.



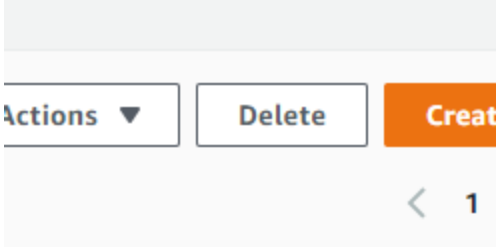
Click on the “Next” button.



Review the alarm configuration and click on the “Create alarm” button.



Step 36: Create or modify some resources to generate events. Navigate back to the DynamoDB dashboard and delete the created table.



Confirm the action by typing “delete” in the box. Then click on the “Delete table” button.

You are about to delete a table.

- Users

☑ Delete all CloudWatch alarms for this table.

☐ Create a backup of this table before deleting it.
If you do not select this check box, you will not be able to restore data being deleted.

To confirm the deletion of this table, type *delete* in the box.

delete

Cancel | Delete table

You will receive a notification generated by the CloudWatch alarm. Deleting a table action triggered the alarm because of the threshold value.

AN

AWS Notifications
no-reply@sns.amazonaws.com

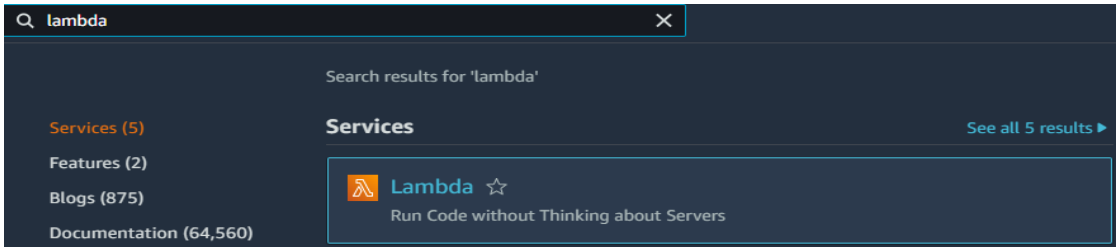
Date:
14-09-2022 16:04:58

Subject: ALARM: "CloudTrailEventsAlarm" in US East (N. Virginia)

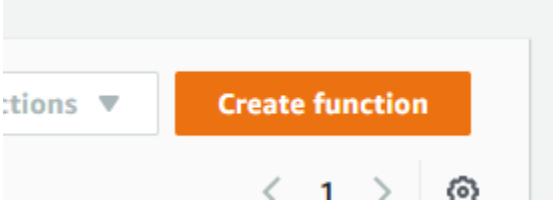
You are receiving this email because your Amazon CloudWatch Alarm "CloudTrailEventsAlarm" in the US East (N. Virginia) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [1.0 (14/09/22 10:33:00)] was greater than or equal to the threshold (1.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Wednesday 14 September, 2022 10:34:57 UTC".
View this alarm in the AWS Management Console:
https://us-east-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-east-1#alarmsV2:alarm/CloudTrailEventsAlarm
Alarm Details:
- Name: CloudTrailEventsAlarm
- Description:
- State Change: INSUFFICIENT_DATA -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [1.0 (14/09/22 10:33:00)] was greater than or equal to the threshold (1.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Wednesday 14 September, 2022 10:34:57 UTC
- AWS Account: 664289593040
- Alarm Arn: arn:aws:cloudwatch:us-east-1:664289593040:alarm:CloudTrailEventsAlarm
Threshold:
- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 1.0 for at least 1 of the last 1 period(s) of 60 seconds.
Monitored Metric:
- MetricNamespace: CloudTrailEventMetric
- MetricName: CloudTrailEventMetric
- Dimensions:

Now we will configure the alerts through the second method using a lambda function.

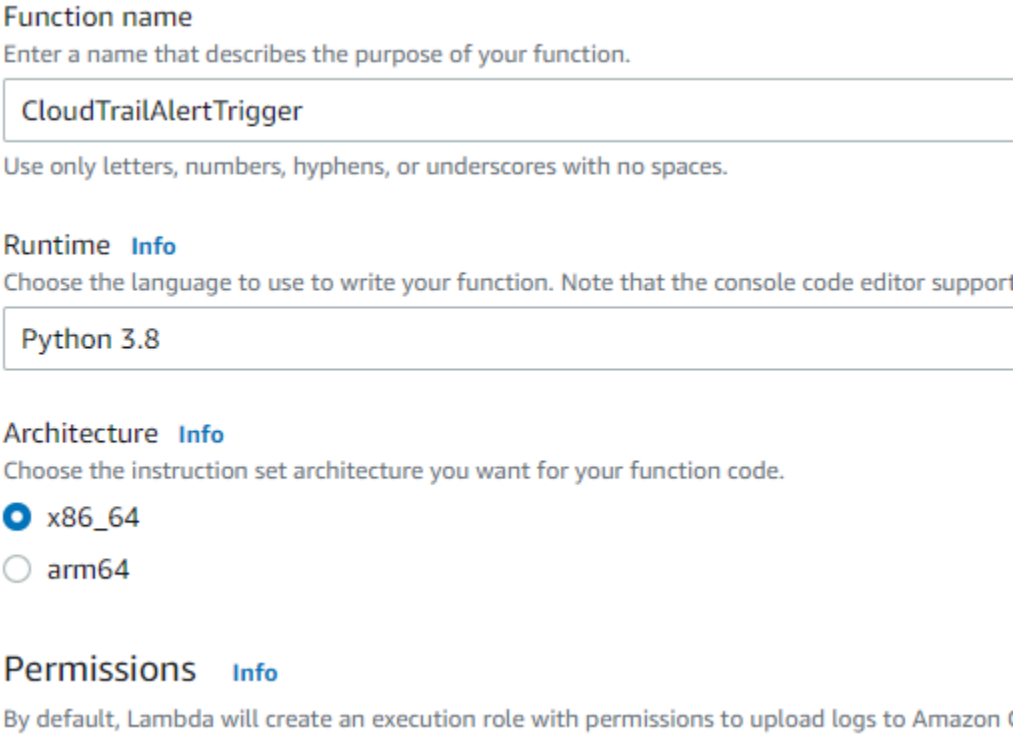
Step 37: Search for Lambda in the search bar and navigate to the Lambda dashboard.



Step 38: Click on the “Create function” button.

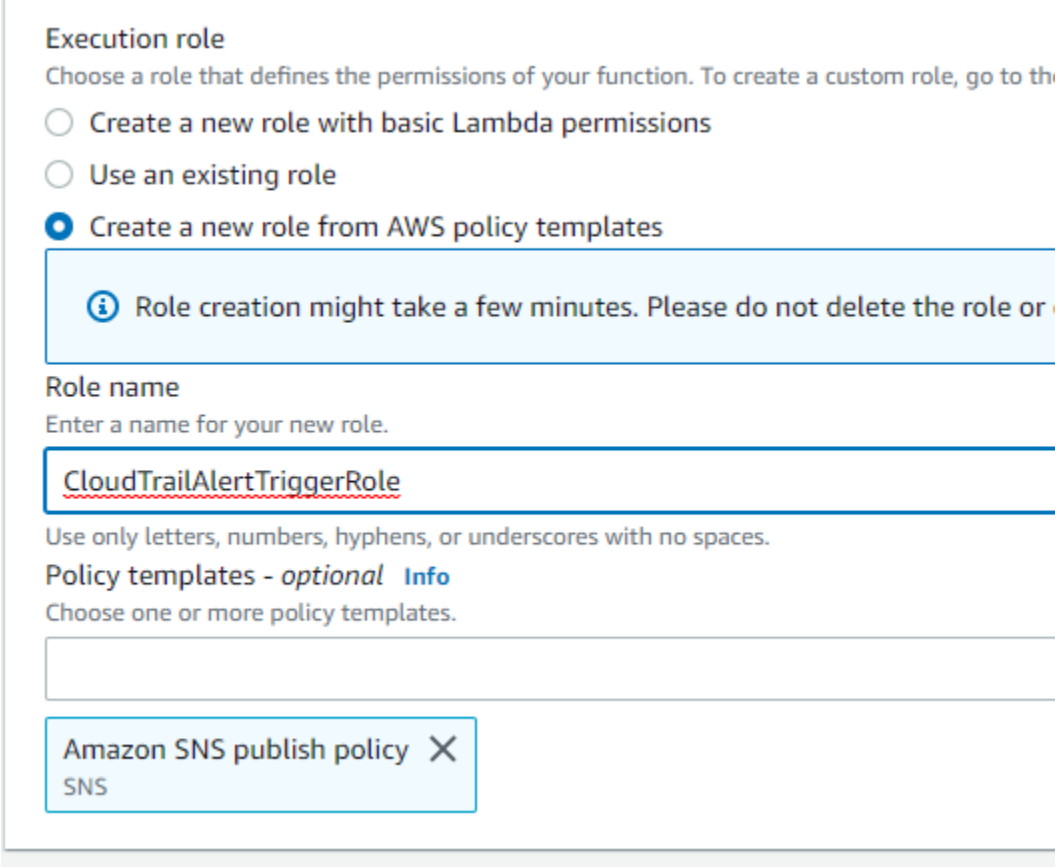


Step 39: Set the function name as “CloudTrailAlertTrigger” and runtime as “Python 3.8”.

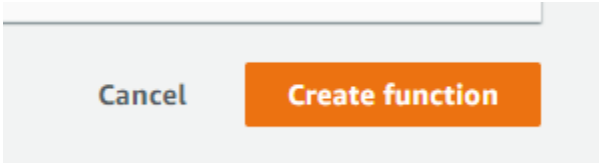


Step 40: Choose the execution role as “Create a new role from AWS policy templates”. Set role name as “CloudTrailAlertTriggerRole”. Select “Amazon SNS publish policy” from policy templates.

Every Lambda function has an IAM role called an execution role. In this role, you can attach a policy that defines the permissions that your function needs to access other AWS services and resources. At a minimum, your function needs access to Amazon CloudWatch Logs for log streaming. Here we will add “Amazon SNS publish policy” for publishing notifications from the lambda function.



Click on “Create function”



Step 41: Copy and replace the python code in the lambda_function.py file. Replace the SNS topic ARN with the created ARN and click on deploy.

This code will parse AWS logs and send SNS notifications for every log generated in the CloudWatch logs group.

Code:

```
import json
import boto3
import gzip
import base64
import os

sns_client = boto3.client('sns')

def lambda_handler(event, context):
    decoded_event = json.loads(gzip.decompress(base64.b64decode(event['awslogs']['data'])))
    body = '''
    LogGroup: {loggroup}
    Logstream: {logstream}
    Filter Match: {filtermatch}
    '''.format(
        loggroup=decoded_event['logGroup'],
        logstream=decoded_event['logStream'],
        filtermatch=decoded_event['logEvents'][0]['message'],
    )

    def send_message(body):
        sns = sns_client.publish(
            TopicArn = 'arn:aws:sns:us-east-1:809795150143:CloudTrailAlertsTopic',
            Message = body,
        )

    send_message(body)
```

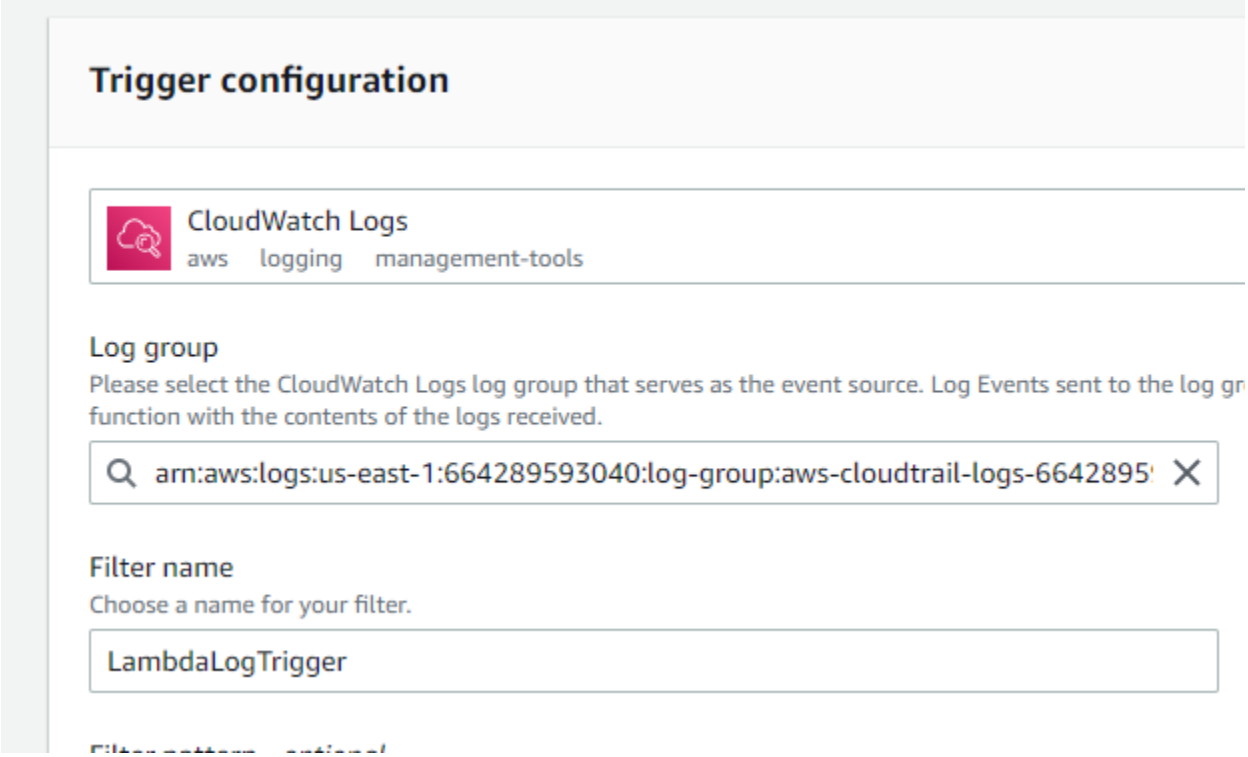
Go Tools Window Test Deploy

lambda_function

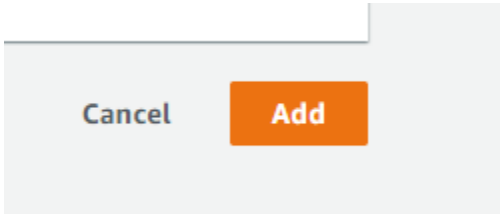
```python
import json
import boto3
import gzip
import base64
import os

sns_client = boto3.client('sns')

def lambda_handler(event, context):
    decoded_event = json.loads(gzip.decompress(base64.b64decode(event['awslogs']['data'])))
    body = '''
    LogGroup: {loggroup}
    Logstream: {logstream}
    Filter Match: {filtermatch}
    '''.format(
        loggroup=decoded_event['logGroup'],
        logstream=decoded_event['logStream'],
        filtermatch=decoded_event['logEvents'][0]['message'],
    )

    def send_message(body):
        sns = sns_client.publish(
            TopicArn = 'arn:aws:sns:us-east-1:664289593040:CloudTrailEventsAlertTopic',
            Message = body,
        )

    send_message(body)
```

Click on the “Add trigger” button.

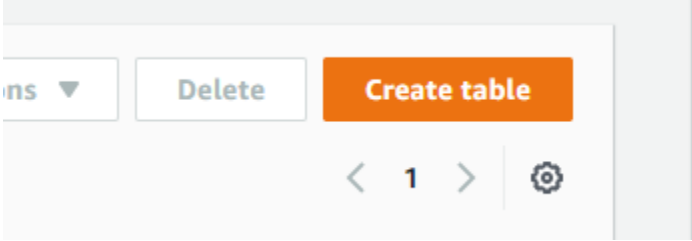
Step 42: Set “CloudWatch Logs” as the trigger and select the log group created by CloudTrail. Set the filter name as “LambdaLogTrigger”.



Click on the “Add” button.



Step 43: Navigate back to the DynamoDB dashboard and create a table again to make a log entry.



Set table name as “Users” and partition key as “id” with the data type as “Number”.

Table name

This will be used to identify your table.

Users

Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).

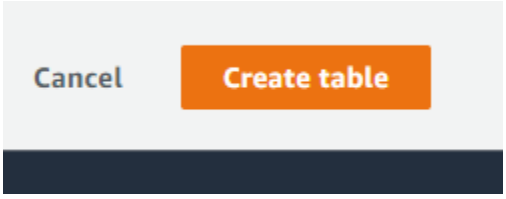
Partition key

The partition key is part of the table's primary key. It is a hash value that is used to retrieve items hosts for scalability and availability.

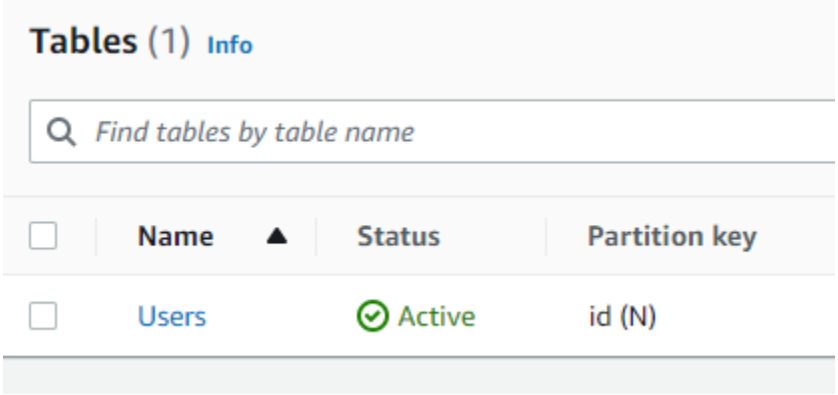
id | Number

1 to 255 characters and case sensitive.

Click on the “Create table” button.



Successfully created a table.



Step 44: Navigate to the inbox of the provided email. Check out the email with the same format provided in the lambda function.

This email is triggered by the lambda function when a log is added to the CloudWatch log group corresponding to the “CreateTable” event.

AN

AWS Notifications
no-reply@sns.amazonaws.com

Date:
14-09-2022 16:27:07

Subject: AWS Notification Message

LogGroup: aws-cloudtrail-logs-664289593040-add39422
Logstream: 664289593040_CloudTrail_us-east-1_4
Filter Match: {"eventVersion":"1.08","userIdentity":
{"type":"AssumedRole","principalId":"AROAZVKV6Y3ILQL23KOTR:AdminSessionRole","arn":"arn:aws:sts::664289593040:assumed-role/TheOracle/AdminSessionRole","accountId":"664289593040","accessKeyId":"ASIAZVKV6Y3IM6EU4MHA","sessionContext":
{"sessionIssuer":
{"type":"Role","principalId":"AROAZVKV6Y3ILQL23KOTR","arn":"arn:aws:iam::664289593040:role/student","accountId":"664289593040"
,"userName":"student"},"attributes":{"creationDate":"2022-09-14T10:54:30Z","mfaAuthenticated":"false"}}},"eventTime":"2022-09-14T10:54:38Z","eventSource":"dynamodb.amazonaws.com","eventName":"CreateTable","awsRegion":"us-west-2","sourceIPAddress":"3.235.228.238","userAgent":"Boto3/1.20.32 Python/3.8.13 Linux/4.14.255-276-224.499.amzn2.x86_64 exec-env/AWS_Lambda_python3.8
Botocore/1.23.32","requestParameters":null,"responseElements":null,"requestID":"CBGU5EC2S03SMK2ICTG2QGBD2FVV4KQNSO5AEMVJF66Q9ASUAAJG","eventID":"756f97a9-caaa-42b2-9f10-16c7de1997ee"
,"readOnly":true,"eventType":"AwsApiCall","apiVersion":"2012-08-10","managementEvent":true,"recipientAccountId":"664289593040","eventCategory":"Management","tlsDetails":
{"tlsVersion":"TLSv1.2","cipherSuite":"ECDHE-RSA-AES128-GCM-SHA256","clientProvidedHostHeader":"dynamodb.us-west-2.amazonaws.com"}}
--

References:

1. AWS CloudTrail (https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html)
2. AWS Athena (https://docs.aws.amazon.com/athena/latest/ug/what-is.html)
3. AWS Athena and CloudTrail Logs (https://docs.aws.amazon.com/athena/latest/ug/cloudtrail-logs.html)
4. CloudWatch Logs (https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/WhatIsCloudWatchLogs.html)