Kyndryl AWS Bootcamp Training – 2024 Batch - 2

Task Assignment 2: Implementing Expiry Key Rotation over 90 Days Using AWS CloudTrail with SES Notifications

Presented by:

Name: KARTHIKEYAN J

Dept & Class: CSE A

Register Number: 111921CS01065

Documentation: Configuration steps of Expiry Key Rotation setup

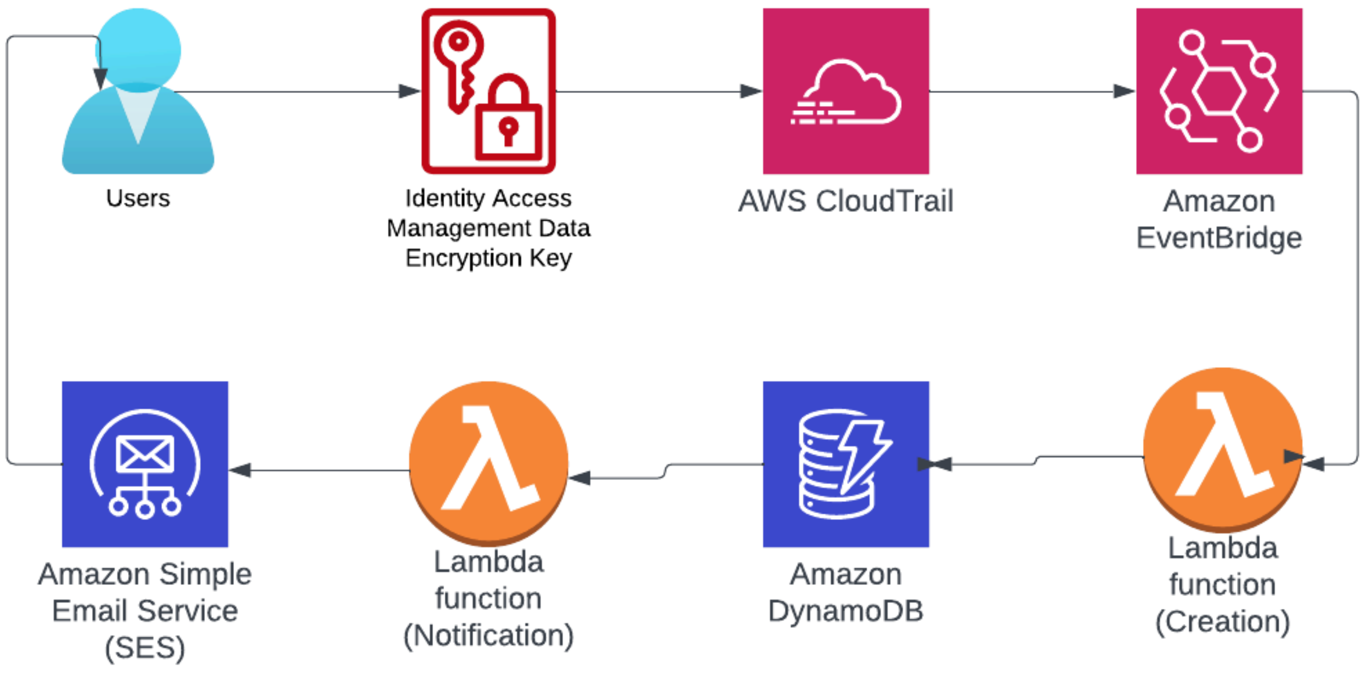
Overview:

This documentation outlines the steps to:

1. **AWS CloudTrail**: Monitors API calls for creating new access keys and triggers the workflow.
2. **Amazon EventBridge**: Filters the CreateKey event from CloudTrail and forwards it to a Lambda function.
3. **Lambda Functions**:
   - **Creation Lambda**: Stores details about the created keys (source IP address, creation time, CloudTrail event, and access key) in DynamoDB.
   - **Notification Lambda**: Handles email notifications through Amazon

     SES when keys expire.
4. **Amazon DynamoDB**: Stores the access key details along with Time- to-Live (TTL) for automated expiration.
5. **Amazon Simple Email Service (SES)**: Sends email notifications to the user when key expiration is near or triggered.

Architecture Setup



Users
Identity Access Management Data Encryption Key
AWS CloudTrail
Amazon EventBridge
Lambda function (Creation)
Amazon DynamoDB
Lambda function (Notification)
Amazon Simple Email Service (SES)

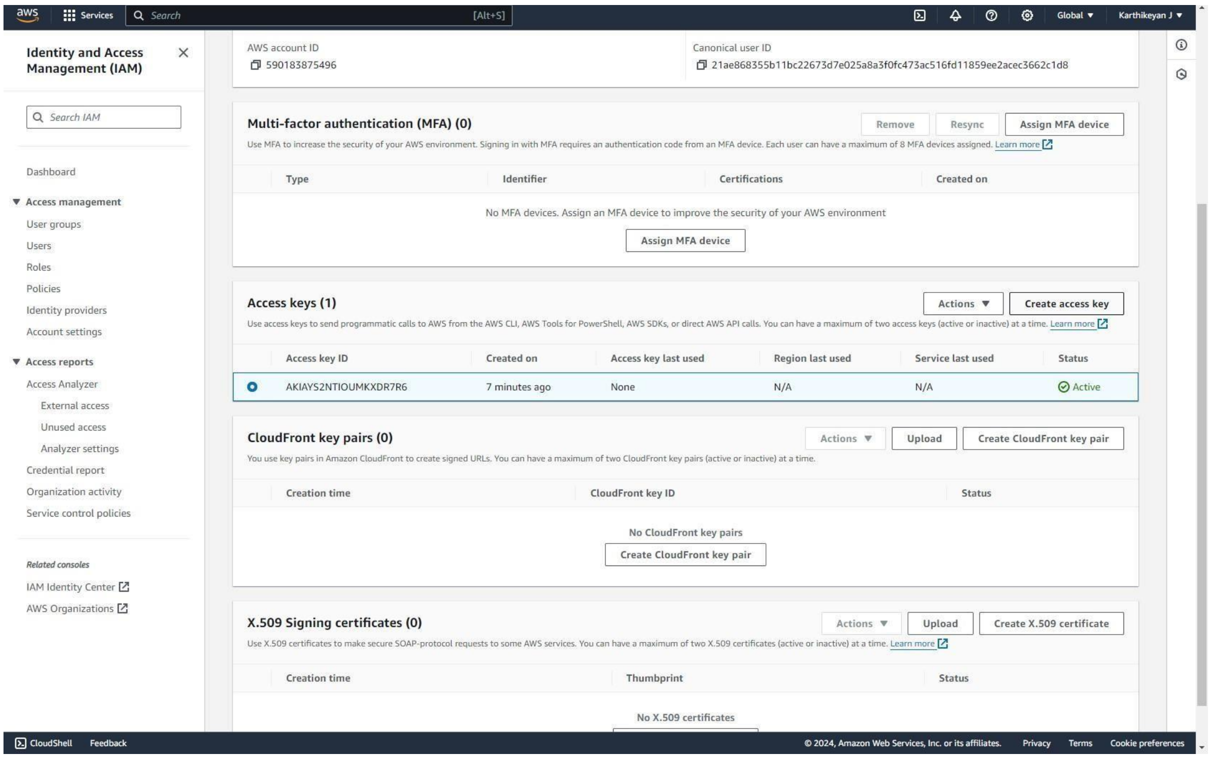
Architecture Explanation

- The architecture leverages AWS CloudTrail to monitor the creation of new IAM access keys. When a key is created, Amazon Eventbridge filters the `CreateAccessKey` event and triggers a Lambda function, which stores the access key details—such as the source IP address, creation time, and access key—in Amazon DynamoDB. A Time-to-Live (TTL) attribute is set in DynamoDB to automatically expire the key after 90 days. When the TTL is reached, DynamoDB Streams triggers another Lambda function, which sends an email notification to the user using Amazon SES. This automated process ensures secure key management and timely notifications for key expiration.

Step-by-Step Configuration

1. Create Access and Secret Keys

- Open the AWS Management Console and navigate to IAM (Identity and Access Management).
- Under Users, select an existing user or create a new one.
- Go to the Security credentials tab and click Create access key.
- Save the Access key ID and Secret access key securely, as this will not be available later.
- These credentials will be logged by CloudTrail, and the workflow will begin once they are created.



AWS account ID
590183875496

Canonical user ID
21ae868355b11bc22673d7e025a8a3f0fc473ac516fd11859ee2acec3662c1d8

Multi-factor authentication (MFA) (0)

Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. Learn more

Type	Identifier	Certifications	Created on

No MFA devices. Assign an MFA device to improve the security of your AWS environment

Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

	Access key ID	Created on	Access key last used	Region last used	Service last used	Status
●	AKIAYS2NTIOUMKXDR7R6	7 minutes ago	None	N/A	N/A	✓ Active

CloudFront key pairs (0)

You use key pairs in Amazon CloudFront to create signed URLs. You can have a maximum of two CloudFront key pairs (active or inactive) at a time.

Creation time	CloudFront key ID	Status

No CloudFront key pairs

X.509 Signing certificates (0)

Use X.509 certificates to make secure SOAP-protocol requests to some AWS services. You can have a maximum of two X.509 certificates (active or inactive) at a time. Learn more

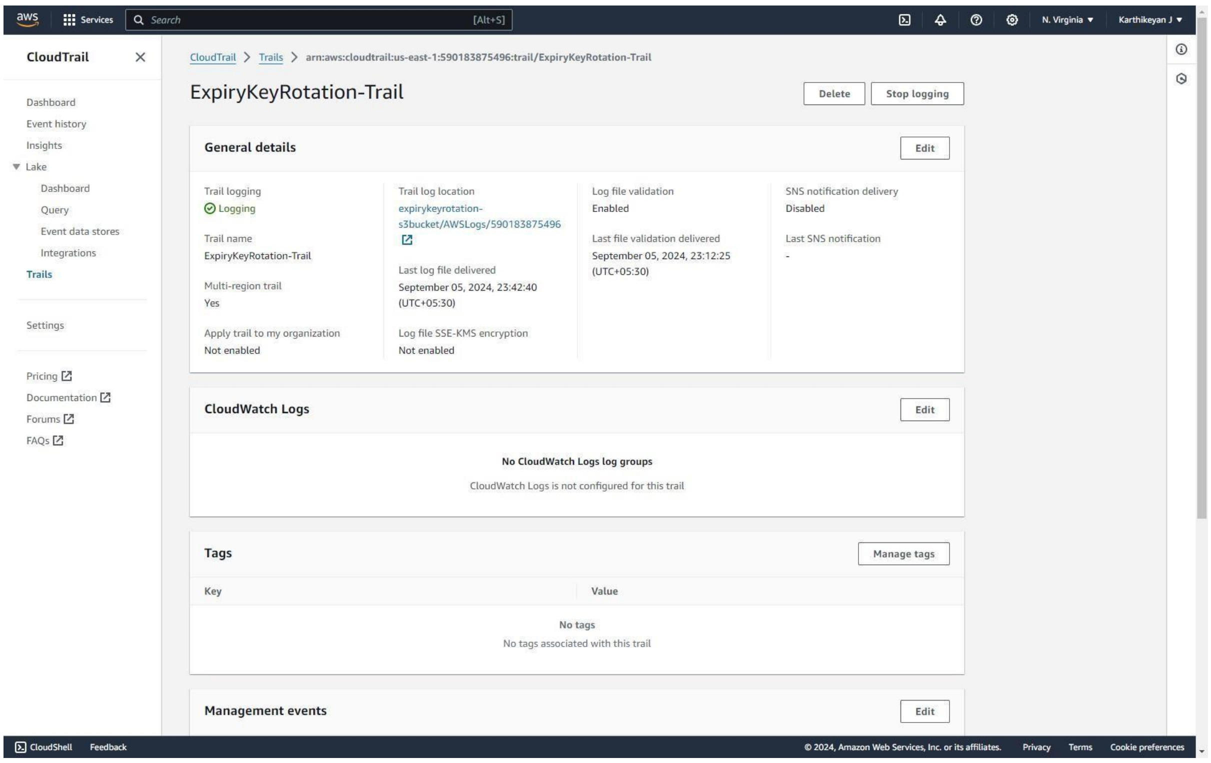
Creation time	Thumbprint	Status

No X.509 certificates

© 2024, Amazon Web Services, Inc. or its affiliates.

2. Enable CloudTrail for Monitoring API Calls

- Open the **AWS Management Console** and navigate to **CloudTrail**.
- Create a new CloudTrail or modify an existing one to capture **CreateKey** events:
    - Go to **Trails** and click **Create trail**.
    - In **Management events**, choose to log **Read/Write events** and select **Write-only** (to capture the creation of keys).
    - Enable logging for **IAM** in **Event type**.
- Set a destination (S3 or CloudWatch Logs) for storing your CloudTrail logs.



CloudTrail > Trails > arn:aws:cloudtrail:us-east-1:590183875496:trail/ExpiryKeyRotation-Trail

ExpiryKeyRotation-Trail

General details

Trail logging
Logging

Trail name
ExpiryKeyRotation-Trail

Multi-region trail
Yes

Apply trail to my organization
Not enabled

Trail log location
expirykeyrotation-s3bucket/AWSLogs/590183875496

Last log file delivered
September 05, 2024, 23:42:40 (UTC+05:30)

Log file SSE-KMS encryption
Not enabled

Log file validation
Enabled

Last file validation delivered
September 05, 2024, 23:12:25 (UTC+05:30)

SNS notification delivery
Disabled

Last SNS notification
-

CloudWatch Logs

No CloudWatch Logs log groups
CloudWatch Logs is not configured for this trail

Tags

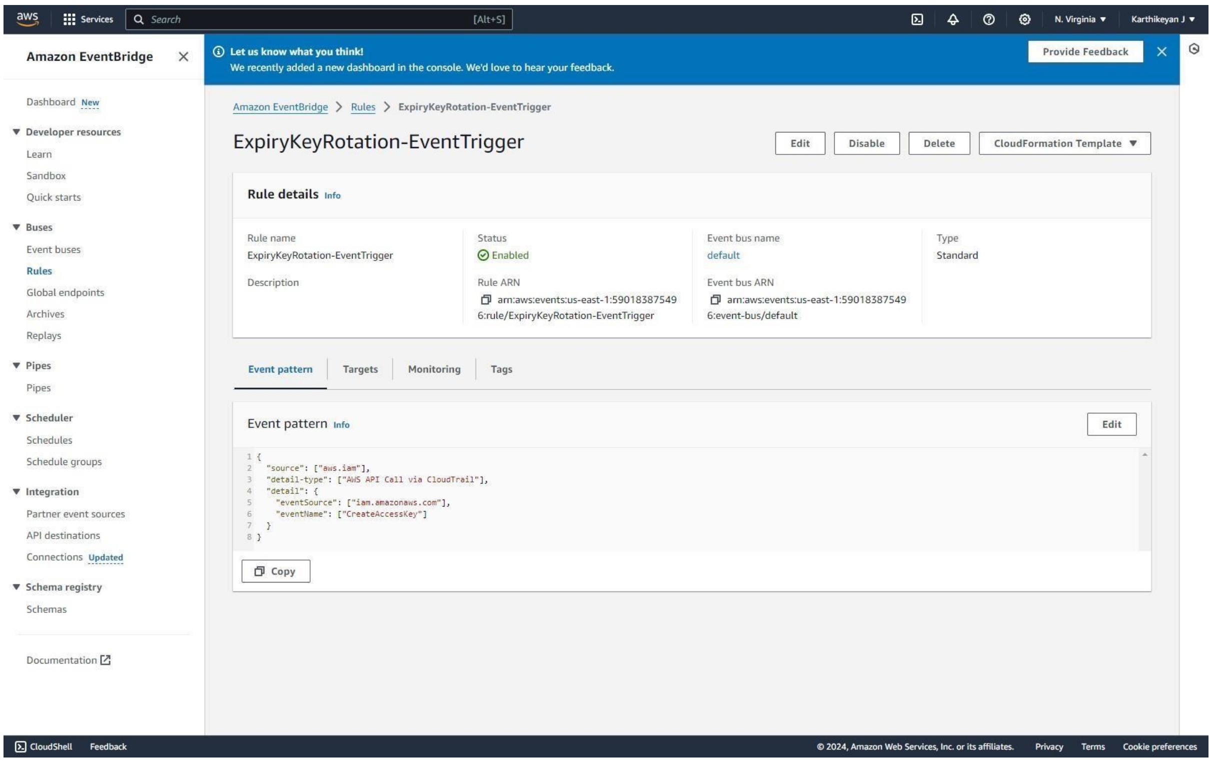
Key	Value

No tags
No tags associated with this trail

Management events

3. Set Up Amazon EventBridge for Filtering CreateKey Events

- In the **AWS Console**, navigate to **Amazon EventBridge**.
- Create a **new rule** to capture only the CreateKey events from CloudTrail:
  - Select **Event source** as **AWS services**.
  - In **Event pattern**, choose **IAM** as the service and **CreateAccessKey** as the event name.
- Set the **target** to the Lambda function you will use to process these events (the creation Lambda function).



ExpiryKeyRotation-EventTrigger

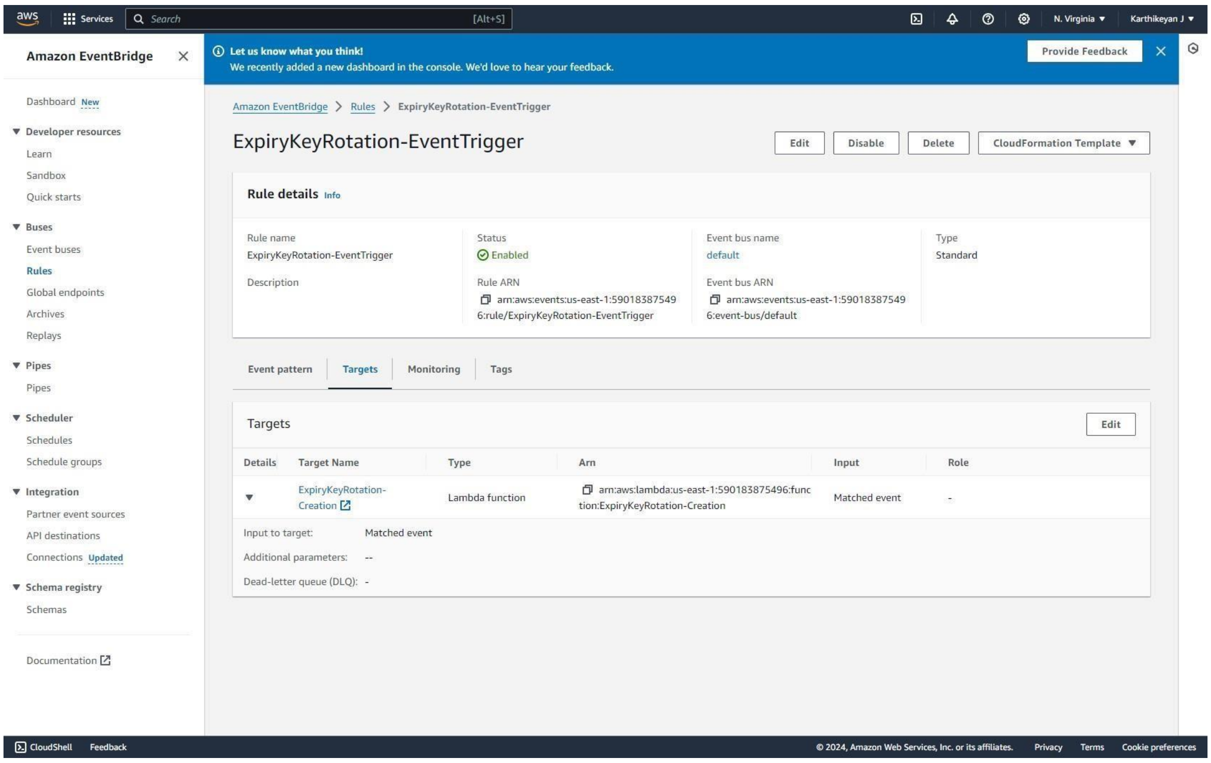
Rule details

Rule name: ExpiryKeyRotation-EventTrigger
Description:
Status: Enabled
Rule ARN: arn:aws:events:us-east-1:590183875496:rule/ExpiryKeyRotation-EventTrigger
Event bus name: default
Event bus ARN: arn:aws:events:us-east-1:590183875496:event-bus/default
Type: Standard

Event pattern

```
{
  "source": ["aws.iam"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail": {
    "eventSource": ["iam.amazonaws.com"],
    "eventName": ["CreateAccessKey"]
  }
}
```

© 2024, Amazon Web Services, Inc. or its affiliates.



Let us know what you think!
We recently added a new dashboard in the console. We'd love to hear your feedback.

Amazon EventBridge > Rules > ExpiryKeyRotation-EventTrigger

ExpiryKeyRotation-EventTrigger

Rule details Info

Rule name: ExpiryKeyRotation-EventTrigger
Description:
Status: Enabled
Rule ARN: arn:aws:events:us-east-1:590183875496:rule/ExpiryKeyRotation-EventTrigger
Event bus name: default
Event bus ARN: arn:aws:events:us-east-1:590183875496:event-bus/default
Type: Standard

Targets

Details	Target Name	Type	Arn	Input	Role
▼	ExpiryKeyRotation-Creation	Lambda function	arn:aws:lambda:us-east-1:590183875496:function:ExpiryKeyRotation-Creation	Matched event	-

Input to target: Matched event
Additional parameters: --
Dead-letter queue (DLQ): -

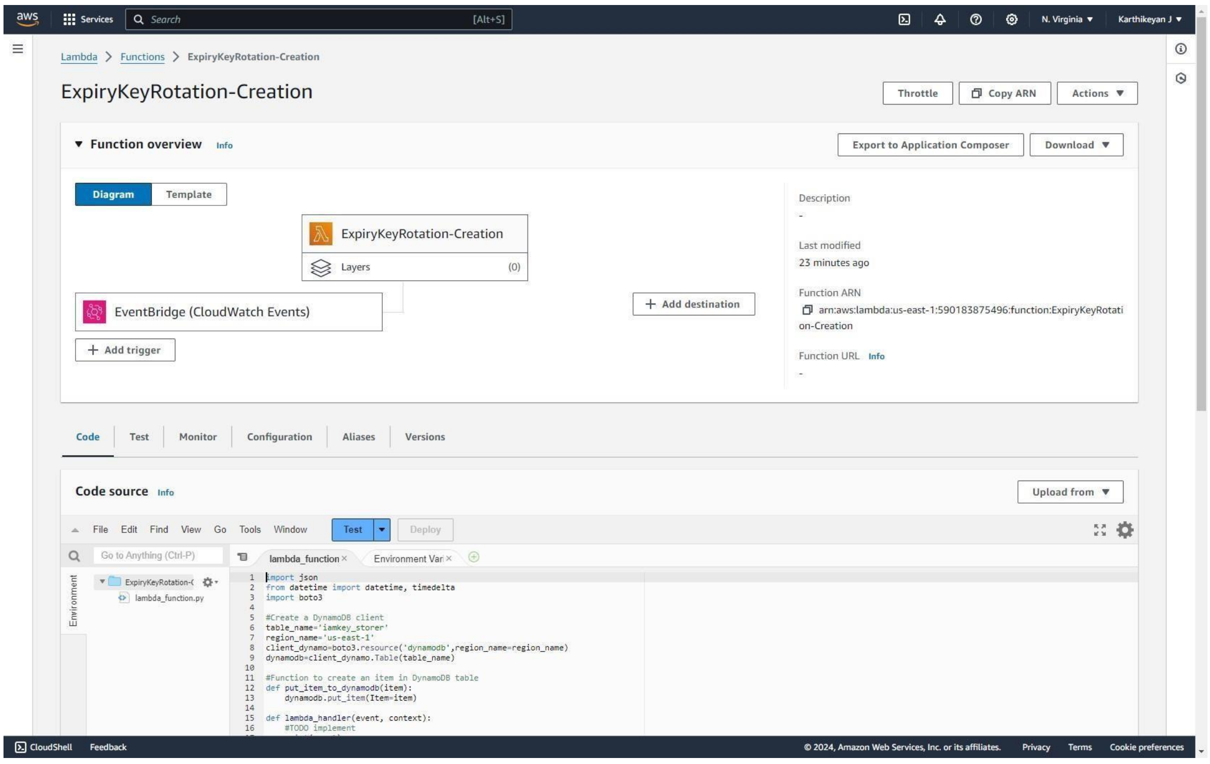
4. Create the Lambda Function to Process Key Creation Events

- In **AWS Lambda**, create a new function:
  - Select **Author from scratch** and give it a name like ExpiryKeyRotation-Creation.
  - Set the **runtime** to Python (**Node.js 20.x** or, depending on your code).
- Attach the necessary permissions to allow DynamoDB and CloudTrail interaction.

- Write code to:

1. Extract event data (such as sourceIPAddress, cloudtrail_key, and access_key).
2. Store this data in a DynamoDB table (with attributes for Time-to-Live (TTL)).
3. Example structure of the item stored in DynamoDB:

```
{
  "cloudtrail_key": "abc123",
  "access_key": "AKIA...",
  "created_on": "2024-09-06T12:00:00Z",
  "sourceIPAddress": "192.168.1.1",
  "ttl": "1691434000"  // TTL in UNIX timestamp (90 days later)
}
```



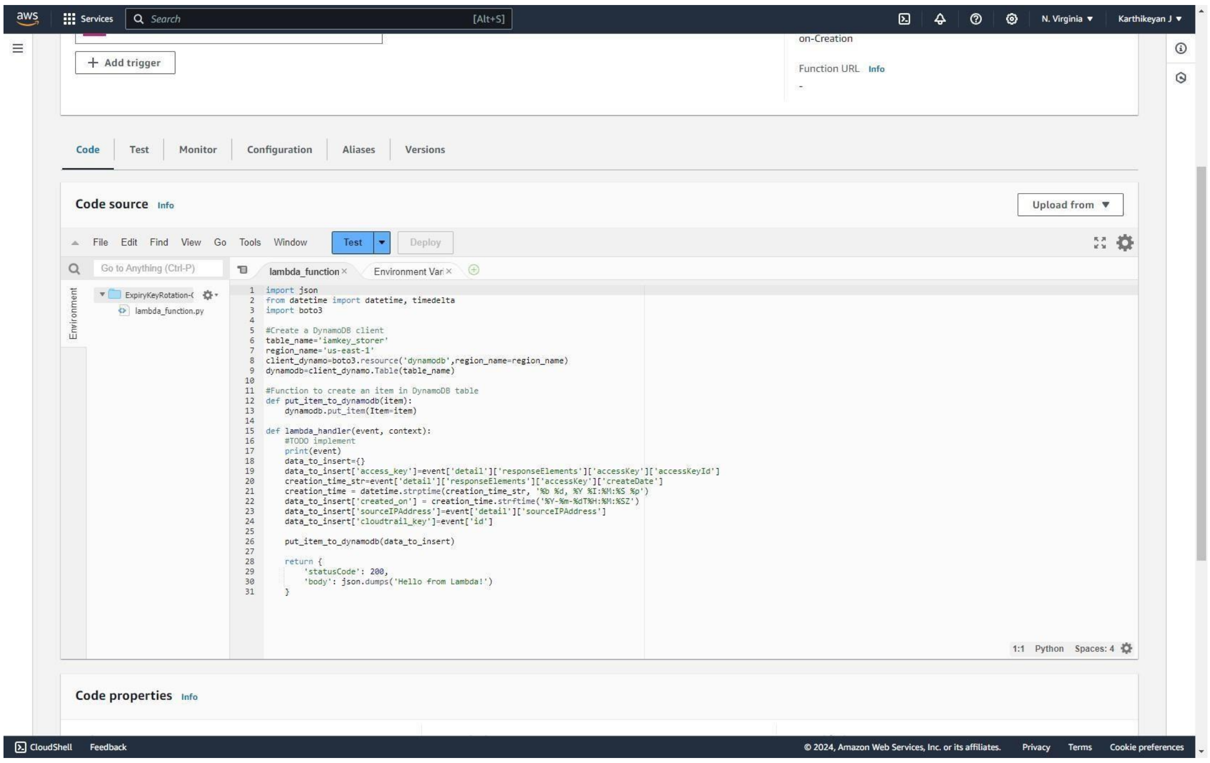
Description
-

Last modified
23 minutes ago

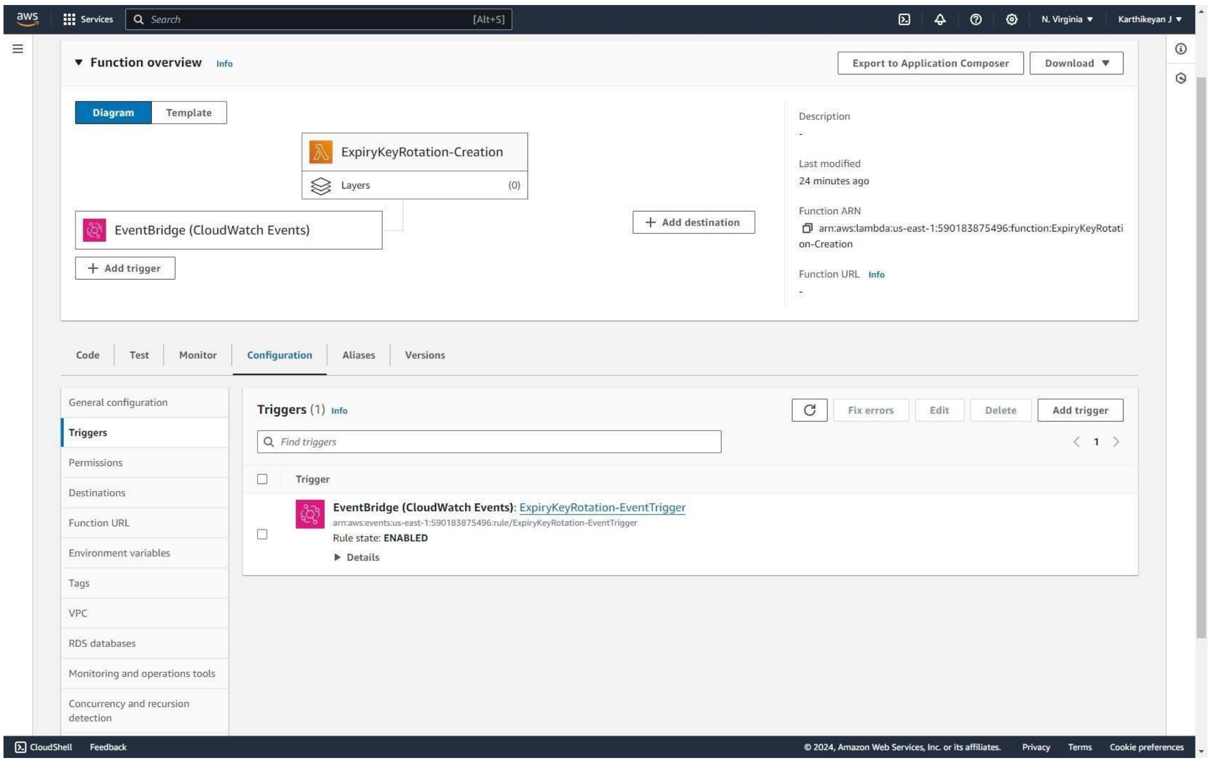
Function ARN
arn:aws:lambda:us-east-1:590183875496:function:ExpiryKeyRotation-Creation

Function URL Info
-

```python
import json
from datetime import datetime, timedelta
import boto3

#Create a DynamoDB client
table_name='iamkey_storer'
region_name='us-east-1'
client_dynamo=boto3.resource('dynamodb',region_name=region_name)
dynamodb=client_dynamo.Table(table_name)

#Function to create an item in DynamoDB table
def put_item_to_dynamodb(item):
    dynamodb.put_item(Item=item)

def lambda_handler(event, context):
    #TODO implement
```



```python
import json
from datetime import datetime, timedelta
import boto3

#Create a DynamoDB client
table_name='iamkey_storer'
region_name='us-east-1'
client_dynamo=boto3.resource('dynamodb',region_name=region_name)
dynamodb=client_dynamo.Table(table_name)

#Function to create an item in DynamoDB table
def put_item_to_dynamodb(item):
    dynamodb.put_item(Item=item)

def lambda_handler(event, context):
    #TODO implement
    print(event)
    data_to_insert={}
    data_to_insert['access_key']=event['detail']['responseElements']['accessKey']['accessKeyId']
    creation_time_str=event['detail']['responseElements']['accessKey']['createDate']
    creation_time = datetime.strptime(creation_time_str, '%b %d, %Y %I:%M:%S %p')
    data_to_insert['created_on'] = creation_time.strftime('%Y-%m-%dT%H:%M:%SZ')
    data_to_insert['sourceIPAddress']=event['detail']['sourceIPAddress']
    data_to_insert['cloudtrail_key']=event['id']

    put_item_to_dynamodb(data_to_insert)

    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```



Description
-

Last modified
24 minutes ago

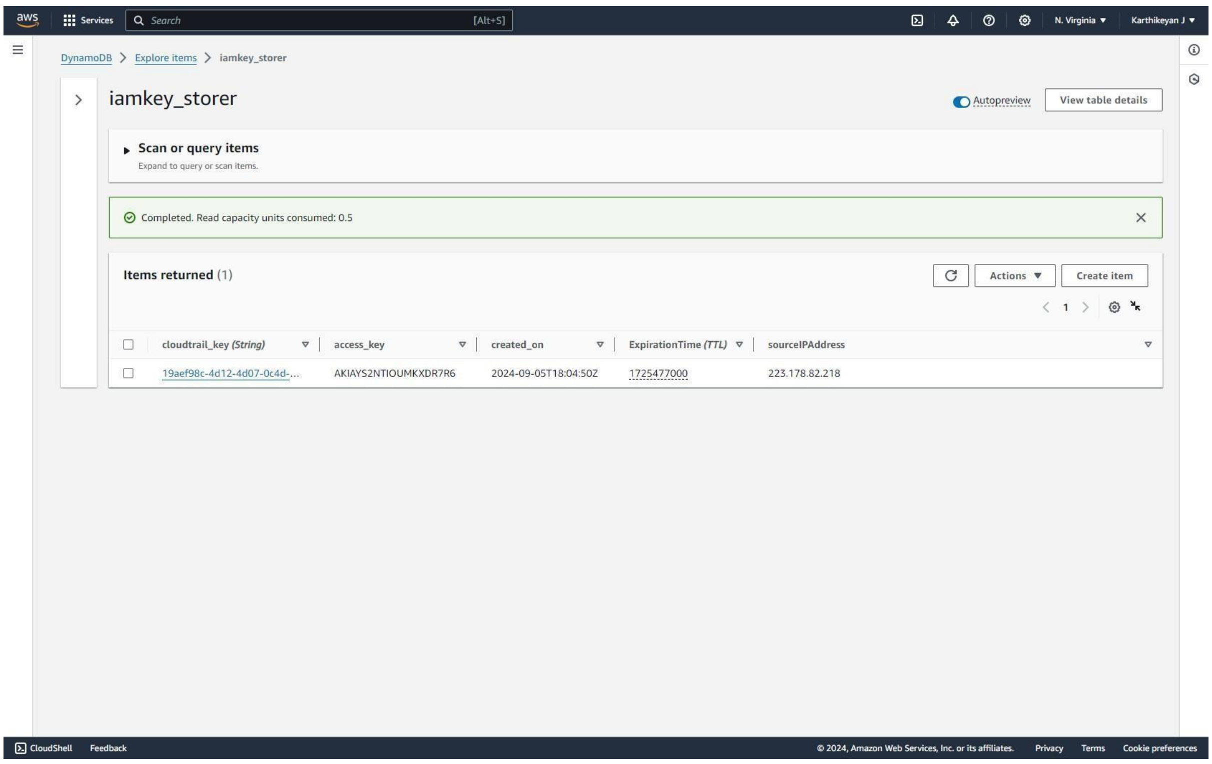
Function ARN
arn:aws:lambda:us-east-1:590183875496:function:ExpiryKeyRotation-Creation

Function URL Info
-

EventBridge (CloudWatch Events): ExpiryKeyRotation-EventTrigger
arn:aws:events:us-east-1:590183875496:rule/ExpiryKeyRotation-EventTrigger
Rule state: ENABLED

5. Set Up Time-to-Live (TTL) in DynamoDB

- Navigate to **DynamoDB** and create a table for storing key information:
  - Set **cloudtrail_key** as the partition key.
- In the table settings, enable **Time-to-Live (TTL)** and configure it to use a dedicated attribute (e.g., ttl).
- This TTL attribute should be a Unix timestamp representing the expiration time (90 days from creation).



iamkey_storer

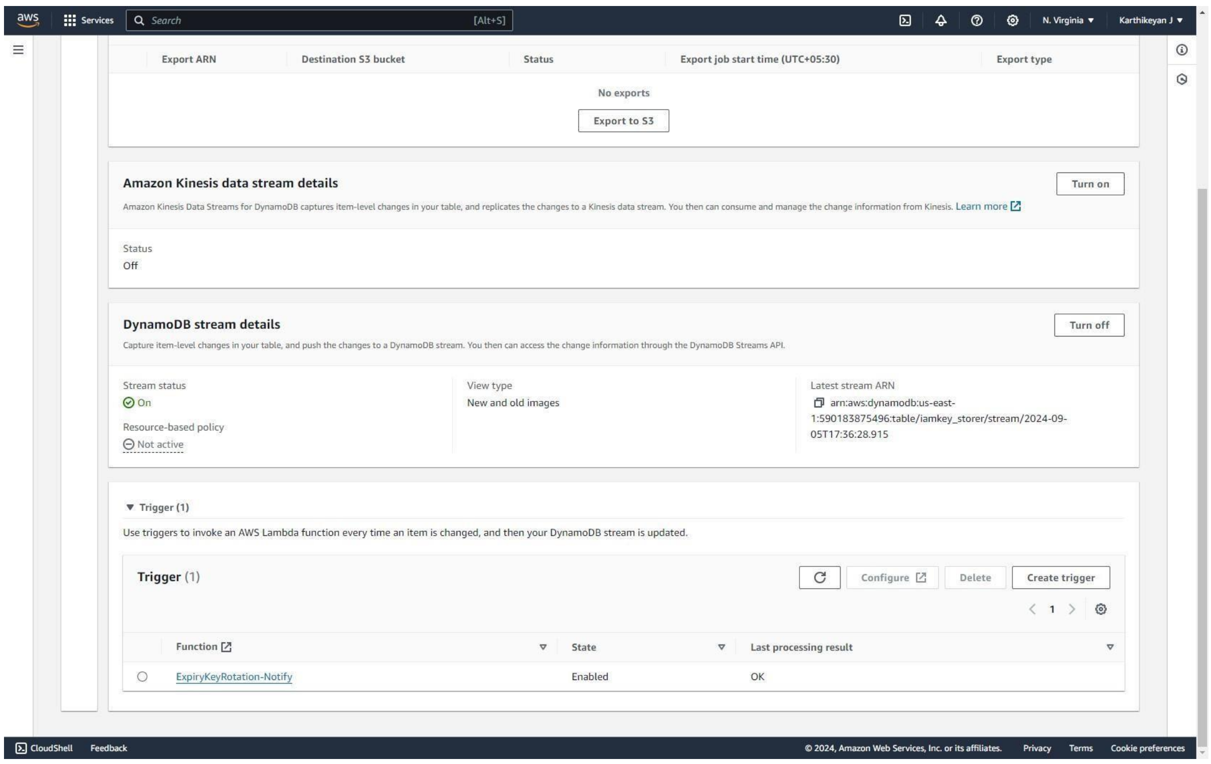
Completed. Read capacity units consumed: 0.5

Items returned (1)

cloudtrail_key (String)	access_key	created_on	ExpirationTime (TTL)	sourceIPAddress
19aef98c-4d12-4d07-0c4d-...	AKIAYS2NTIOUMKXDR7R6	2024-09-05T18:04:50Z	1725477000	223.178.82.218

6. Enable DynamoDB Streams for Monitoring Changes

- In the **DynamoDB Console**, enable **DynamoDB Streams** for the table:
  - This will capture any changes to the table, including TTLtriggered deletions.
- Configure a stream ARN and set it as a trigger for a Lambda function that sends an email notification upon deletion (for expired keys).



Export ARN	Destination S3 bucket	Status	Export job start time (UTC+05:30)	Export type

No exports

Amazon Kinesis data stream details

Amazon Kinesis Data Streams for DynamoDB captures item-level changes in your table, and replicates the changes to a Kinesis data stream. You then can consume and manage the change information from Kinesis. Learn more

Status
Off

DynamoDB stream details

Capture item-level changes in your table, and push the changes to a DynamoDB stream. You then can access the change information through the DynamoDB Streams API.

Stream status
On

Resource-based policy
Not active

View type
New and old images

Latest stream ARN
arn:aws:dynamodb:us-east-1:590183875496:table/iamkey_storer/stream/2024-09-05T17:36:28.915

Trigger (1)

Use triggers to invoke an AWS Lambda function every time an item is changed, and then your DynamoDB stream is updated.

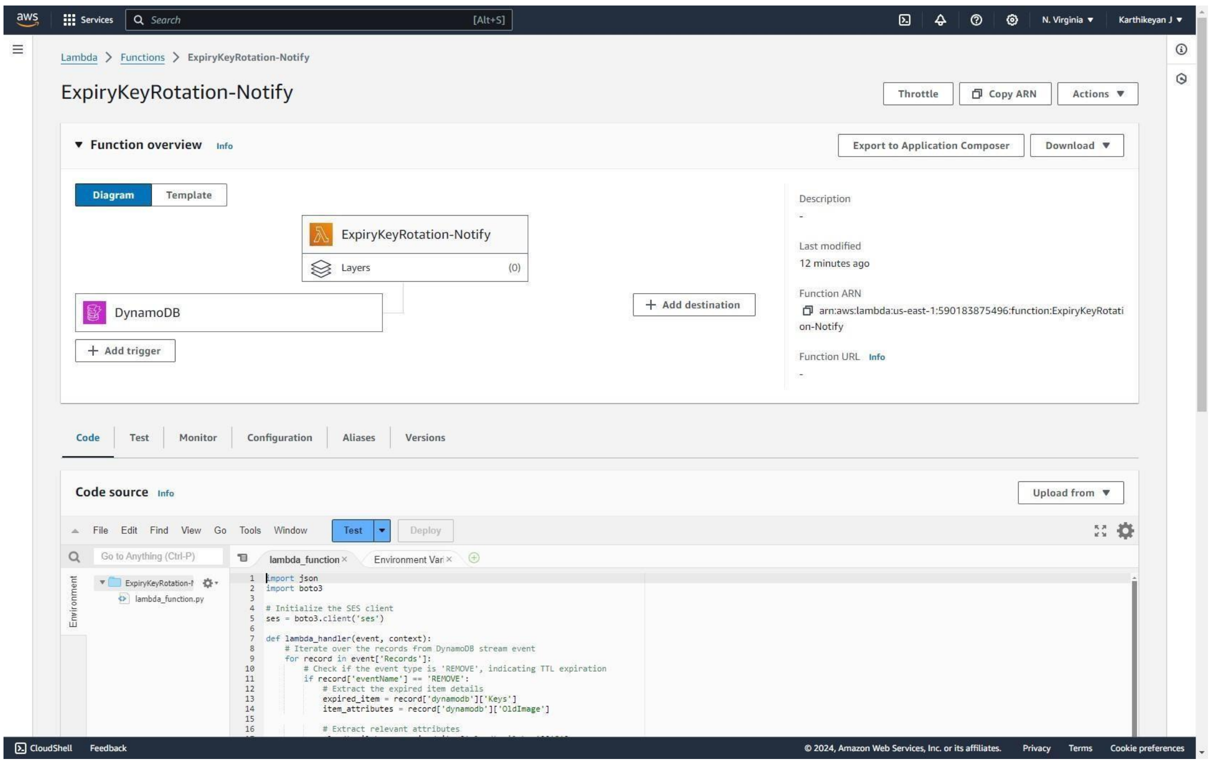
Trigger (1)

Function	State	Last processing result
ExpiryKeyRotation-Notify	Enabled	OK

8. Create the Lambda Function for Email Notification

- Create another Lambda function (ExpiryKeyRotation-Notify):
  - This function will be triggered by DynamoDB Streams when a TTL event occurs (key expiration).
  - Use AWS SDK to send an email via **Amazon SES**:
    - Configure the SES client to specify the sender and recipient email addresses.
    - SES must be verified and have appropriate permissions for sending emails.
    - Example email body:

      "Your IAM access Key with ID <access_key> has expired. Please rotate your credential or remove unused keys."



ExpiryKeyRotation-Notify

Function overview Info

Description
-

Last modified
12 minutes ago

Function ARN
arn:aws:lambda:us-east-1:590183875496:function:ExpiryKeyRotation-Notify

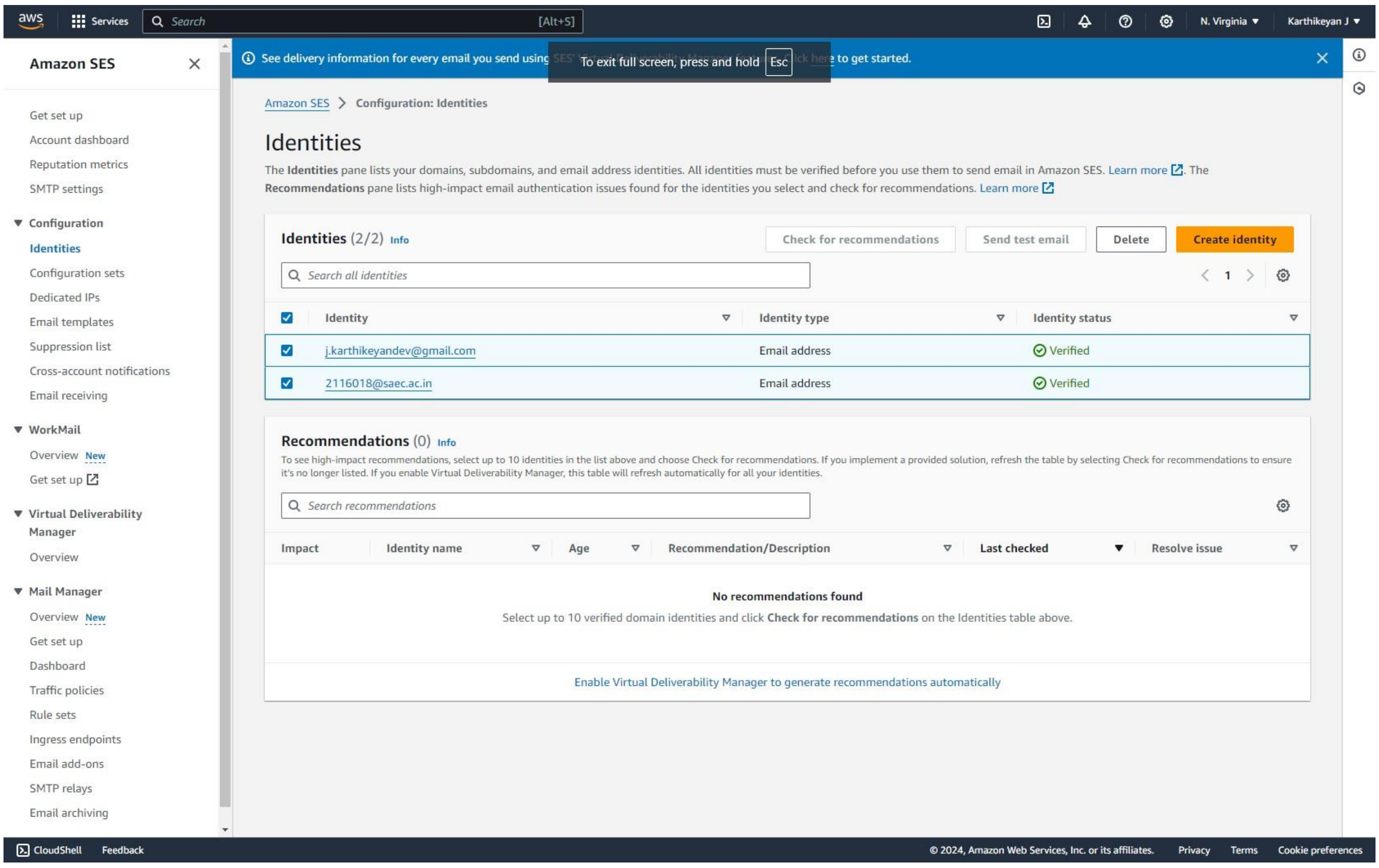
Function URL Info
-

Code source Info

```python
import json
import boto3

# Initialize the SES client
ses = boto3.client('ses')

def lambda_handler(event, context):
    # Iterate over the records from DynamoDB stream event
    for record in event['Records']:
        # Check if the event type is 'REMOVE', indicating TTL expiration
        if record['eventName'] == 'REMOVE':
            # Extract the expired item details
            expired_item = record['dynamodb']['Keys']
            item_attributes = record['dynamodb']['OldImage']

            # Extract relevant attributes
```

9. Set Up Amazon SES for Email Notifications

- In **Amazon SES**, verify the sender and recipient email addresses.
- Configure the necessary permissions and policies to allow SES to send emails.
- Link the **Notification Lambda function** to SES for sending the email upon key expiration.



See delivery information for every email you send using SES' To exit full screen, press and hold Esc ck here to get started.

Amazon SES > Configuration: Identities

Identities

The **Identities** pane lists your domains, subdomains, and email address identities. All identities must be verified before you use them to send email in Amazon SES. Learn more. The **Recommendations** pane lists high-impact email authentication issues found for the identities you select and check for recommendations. Learn more

Identities (2/2) Info

Check for recommendations | Send test email | Delete | Create identity

Search all identities

	Identity	Identity type	Identity status
✓	j.karthikeyandev@gmail.com	Email address	Verified
✓	2116018@saec.ac.in	Email address	Verified

Recommendations (0) Info

To see high-impact recommendations, select up to 10 identities in the list above and choose Check for recommendations. If you implement a provided solution, refresh the table by selecting Check for recommendations to ensure it's no longer listed. If you enable Virtual Deliverability Manager, this table will refresh automatically for all your identities.

Search recommendations

Impact	Identity name	Age	Recommendation/Description	Last checked	Resolve issue

No recommendations found

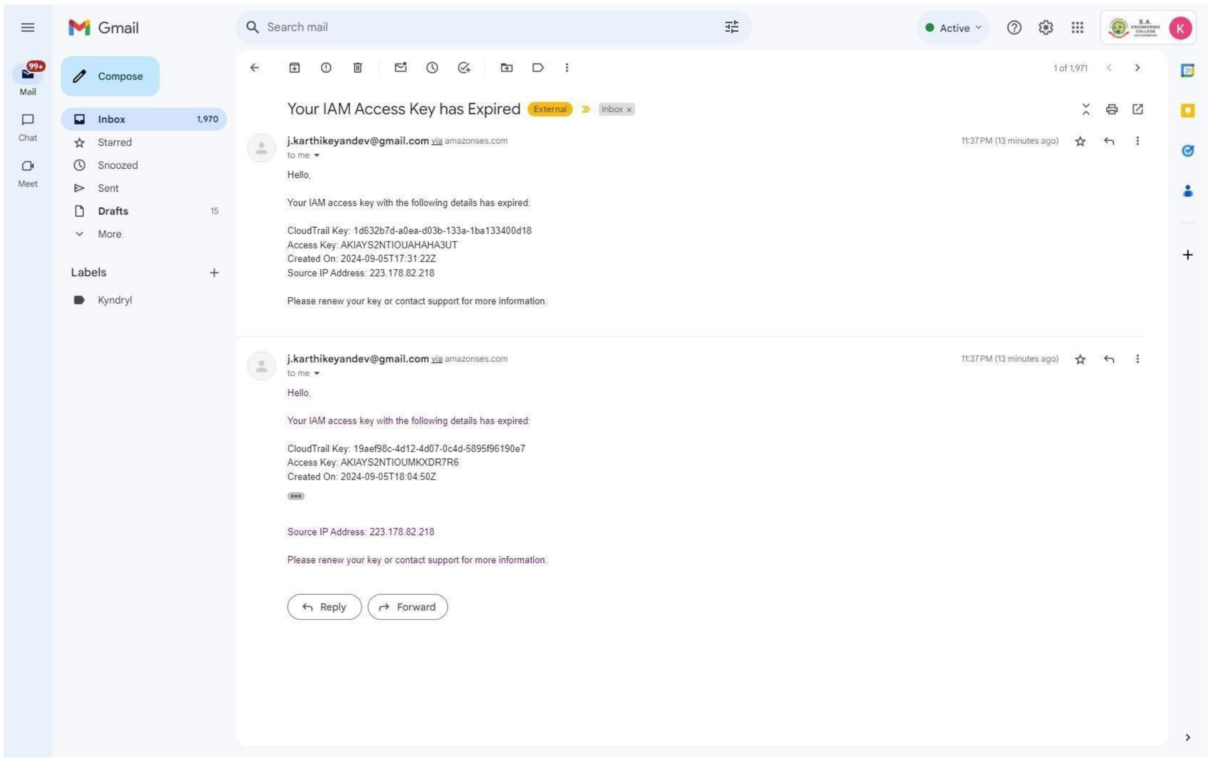
Select up to 10 verified domain identities and click **Check for recommendations** on the Identities table above.

Enable Virtual Deliverability Manager to generate recommendations automatically

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

10. Test the Setup

- Test the flow by creating a new access key using AWS CLI or Console.
- Verify that the CloudTrail logs the event, EventBridge triggers the Lambda function, and the key is stored in DynamoDB with the TTL attribute.
- After key expiration (or by manually updating the TTL for testing), confirm that SES sends the expiration email.



Your IAM Access Key has Expired

j.karthikeyandev@gmail.com via amazonses.com
to me
11:37 PM (13 minutes ago)

Hello,

Your IAM access key with the following details has expired:

CloudTrail Key: 1d632b7d-a0ea-d03b-133a-1ba133400d18
Access Key: AKIAYS2NTIOUAHAHA3UT
Created On: 2024-09-05T17:31:22Z
Source IP Address: 223.178.82.218

Please renew your key or contact support for more information.

j.karthikeyandev@gmail.com via amazonses.com
to me
11:37 PM (13 minutes ago)

Hello,

Your IAM access key with the following details has expired:

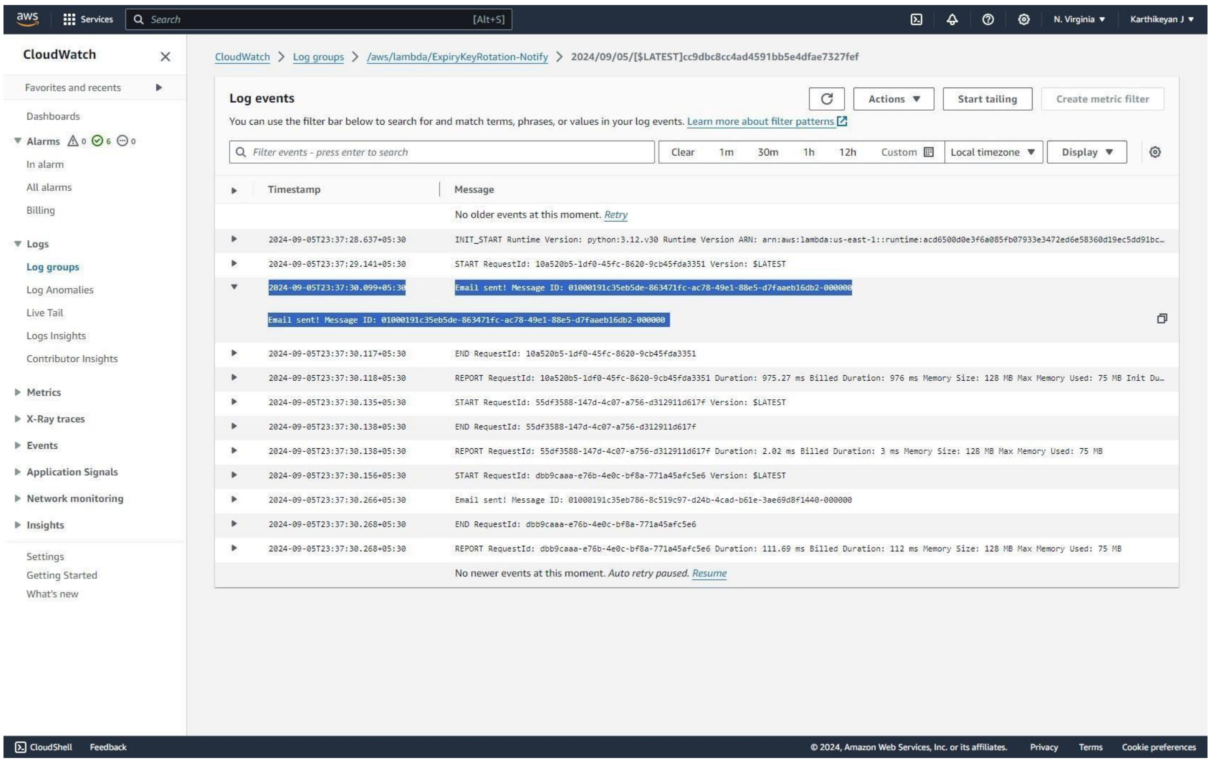
CloudTrail Key: 19aef98c-4d12-4d07-0c4d-5895f96190e7
Access Key: AKIAYS2NTIOUMKXDR7R6
Created On: 2024-09-05T18:04:50Z

Source IP Address: 223.178.82.218

Please renew your key or contact support for more information.

11. Security and Permissions

- Ensure all necessary IAM roles and policies are in place for the Lambda functions to interact with CloudTrail, DynamoDB, and SES.
- Use **AWS Key Management Service (KMS)** if you require encryption for sensitive data like access keys stored in DynamoDB.



You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                       Message
                                No older events at this moment. Retry
2024-09-05T23:37:28.637+05:30   INIT_START Runtime Version: python:3.12.v30 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:acd6500d0e3f6a085fb07933e3472ed6e58360d19ec5dd91bc…
2024-09-05T23:37:29.141+05:30   START RequestId: 10a520b5-1df0-45fc-8620-9cb45fda3351 Version: $LATEST
2024-09-05T23:37:30.099+05:30   Email sent! Message ID: 01000191c35eb5de-863471fc-ac78-49e1-88e5-d7faaeb16db2-000000
    Email sent! Message ID: 01000191c35eb5de-863471fc-ac78-49e1-88e5-d7faaeb16db2-000000
2024-09-05T23:37:30.117+05:30   END RequestId: 10a520b5-1df0-45fc-8620-9cb45fda3351
2024-09-05T23:37:30.118+05:30   REPORT RequestId: 10a520b5-1df0-45fc-8620-9cb45fda3351 Duration: 975.27 ms Billed Duration: 976 ms Memory Size: 128 MB Max Memory Used: 75 MB Init Du…
2024-09-05T23:37:30.135+05:30   START RequestId: 55df3588-147d-4c07-a756-d312911d617f Version: $LATEST
2024-09-05T23:37:30.138+05:30   END RequestId: 55df3588-147d-4c07-a756-d312911d617f
2024-09-05T23:37:30.138+05:30   REPORT RequestId: 55df3588-147d-4c07-a756-d312911d617f Duration: 2.02 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 75 MB
2024-09-05T23:37:30.156+05:30   START RequestId: dbb9caaa-e76b-4e0c-bf8a-771a45afc5e6 Version: $LATEST
2024-09-05T23:37:30.266+05:30   Email sent! Message ID: 01000191c35eb786-8c519c97-d24b-4cad-b61e-3ae69d8f1440-000000
2024-09-05T23:37:30.268+05:30   END RequestId: dbb9caaa-e76b-4e0c-bf8a-771a45afc5e6
2024-09-05T23:37:30.268+05:30   REPORT RequestId: dbb9caaa-e76b-4e0c-bf8a-771a45afc5e6 Duration: 111.69 ms Billed Duration: 112 ms Memory Size: 128 MB Max Memory Used: 75 MB
                                No newer events at this moment. Auto retry paused. Resume
```

Conclusion

This setup ensures that IAM access keys are automatically rotated every 90 days. When a key is created, it is logged via CloudTrail and processed through EventBridge to a Lambda function that stores the details in DynamoDB. A Timeto-Live (TTL) attribute is added to automatically expire the keys after 90 days, triggering another Lambda function that sends a notification via SES when the key expires.

1.ExpiryKeyRotation-Creation (Lambda Function Code):

```
import json from datetime import
datetime, timedelta import boto3

#Create a DynamoDB client table_name='iamkey_storer'
region_name='us-east-1'
client_dynamo=boto3.resource('dynamodb',region_name=region_na
me) dynamodb=client_dynamo.Table(table_name)

#Function to create an item in DynamoDB table
def put_item_to_dynamodb(item):
dynamodb.put_item(Item=item)

def lambda_handler(event, context):
#TODO implement
print(event) data_to_insert={}
data_to_insert['access_key']=event['detail']['responseElements']['accessKey']['accessKeyI
d'] creation_time_str=event['detail']['responseElements']['accessKey']['createDate']
creation_time = datetime.strptime(creation_time_str, '%b %d, %Y %I:%M:%S %p')
data_to_insert['created_on'] = creation_time.strftime('%Y-%m-%dT%H:%M:%SZ')
data_to_insert['sourceIPAddress']=event['detail']['sourceIPAddress']
data_to_insert['cloudtrail_key']=event['id']

put_item_to_dynamodb(data_to_insert)

return {
'statusCode': 200,
'body': json.dumps('Hello from Lambda!')
}
```

2.ExpiryKeyRotation-Notify (Lambda Function Code):

```
import json
import boto3


# Initialize the SES client
ses = boto3.client('ses')


def lambda_handler(event, context):

   # Iterate over the records from DynamoDB stream event
for record in event['Records']:
      # Check if the event type is 'REMOVE', indicating TTL
expiration        if record['eventName'] == 'REMOVE':
#Extract the expired item details
 expired_item = record['dynamodb']['Keys']
item_attributes = record['dynamodb']['OldImage']


         # Extract relevant attributes
 cloudtrail_key = expired_item['cloudtrail_key']['S']
source_ipaddress = item_attributes['source_ipaddress']['S']
current_time = item_attributes['current_time']['S']


         # Create the email content

 subject = "Your IAM Access Key has Expired"

(MENTION THE BODY CONTENT OF THE MESSAGE)

       )

         # Define SES email parameters
response = ses.send_email(
            Source='your-verified-email@example.com',  # Replace with your verified sender email

            Destination={

               'ToAddresses': ['j.karthikeyandev@gmail.com'],  # Replace with recipient email

            },
            Message={

                'Subject': {

                    'Data': subject,

                    'Charset': 'UTF-8'

                },

                'Body': {

                    'Text': {

                        'Data': body_text,

                        'Charset': 'UTF-8'

                    }

                }

            }

        )


        # Log the email response
print(f"Email sent! Message ID: {response['MessageId']}")
    return {

        'statusCode': 200,

        'body': json.dumps('Lambda function executed successfully!')

    }
```