Serverless Application with Monitoring

-Report

TOPIC:

- Concepts Used: AWS Lambda, S3, and Nagios.

-Problem Statement: "Create an AWS Lambda function that logs an event when an

image is uploaded to a specific S3 bucket. Set up Nagios to monitor the Lambda

function's execution status and S3 bucket."

- Tasks:

1) Create a Lambda function in Python that logs 'An Image has been added' when

an object is uploaded to an S3 bucket.

2)Configure Nagios to monitor the Lambda function's logs.

3)Upload a test image to the S3 bucket and verify that the function logs the event

and Nagios captures the status.

AIM:

The primary objective of this project is to integrate the monitoring of AWS Lambda functions within Nagios. This includes:

- Monitoring AWS Lambda for function invocations, updates, and other relevant metrics.
- Setting up a custom check script that uses AWS CLI to monitor Lambda functions.
- Configuring Nagios to monitor the health of Lambda functions and trigger alerts based on performance or errors.

STEPS:

1)CREATE AN S3 BUCKET

2)CREATE A LAMBDA FUNCTION AND ADD AN S3 TRIGGER IN IT.

3)WRITE THE PYTHON SCRIPT TO TRIGGER THE LAMBDA FUNCTION WHENEVER AN IMAGE IS UPLOADED IN THE S3 BUCKET

4)CHECK THE CLOUDWATCH LOGS.

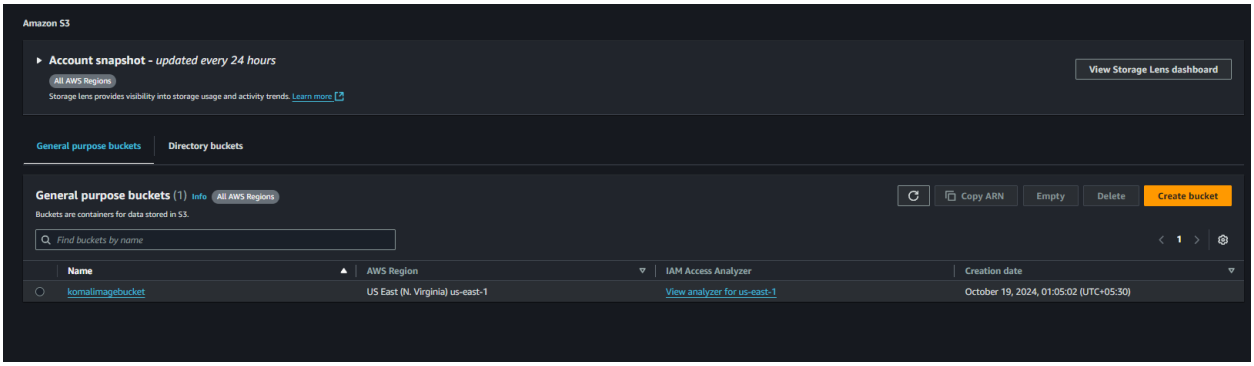
5)MAKE AN EC2 INSTANCE AND INSTALL NAGIOS IN IT.

6)START MONITORING THE LAMBDA FUNCTION VIA THE INSTANCE BY FOLLOWING THE STEPS INCLUDED IN THE DETAILED EXPLAINATION BELOW.

EXECUTION: SERVERLESS MONITORING USING NAGIOS

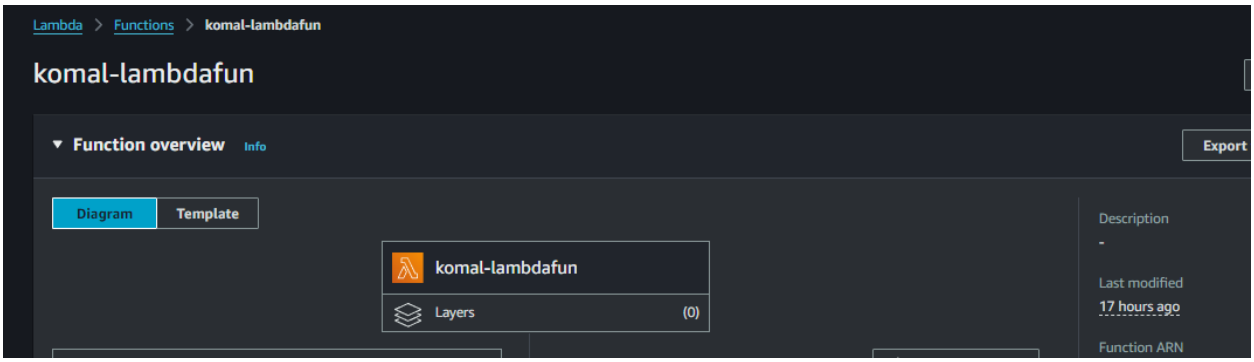
CREATION OF S3 BUCKET:

CREATE AN S3 BUCKET OF IN AWS

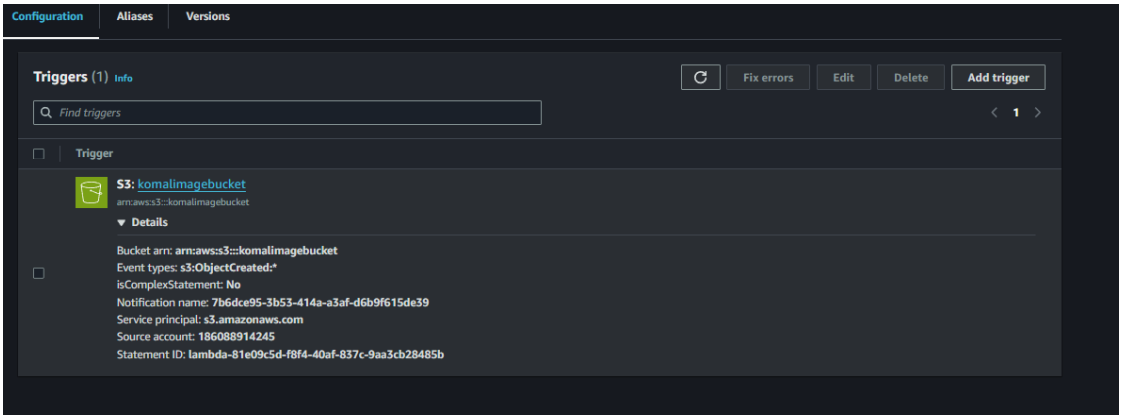


CREATION OF LAMBDA FUNCTION

CREATE A NEW LAMBDA FUNCTION AND CONFIGURE IT.



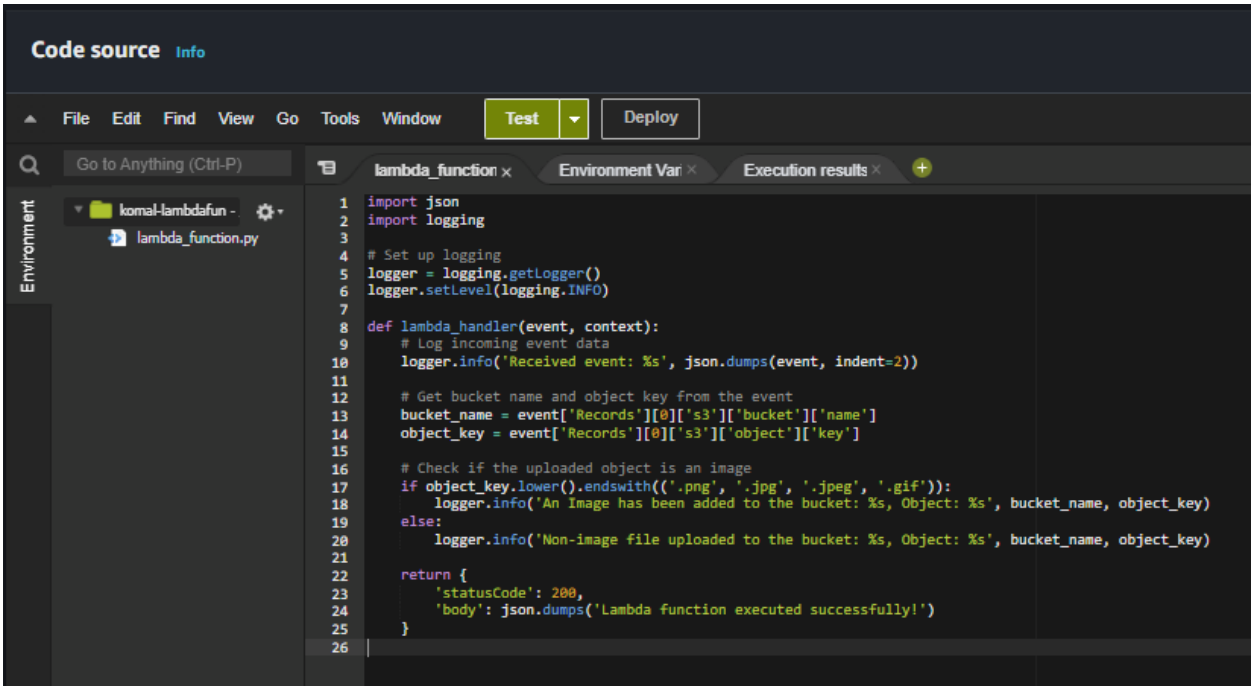
ADD AN S3 TRIGGER TO GET UPDATED WHEN A NEW IMAGE IS UPLOADED IN THE S3 BUCKET



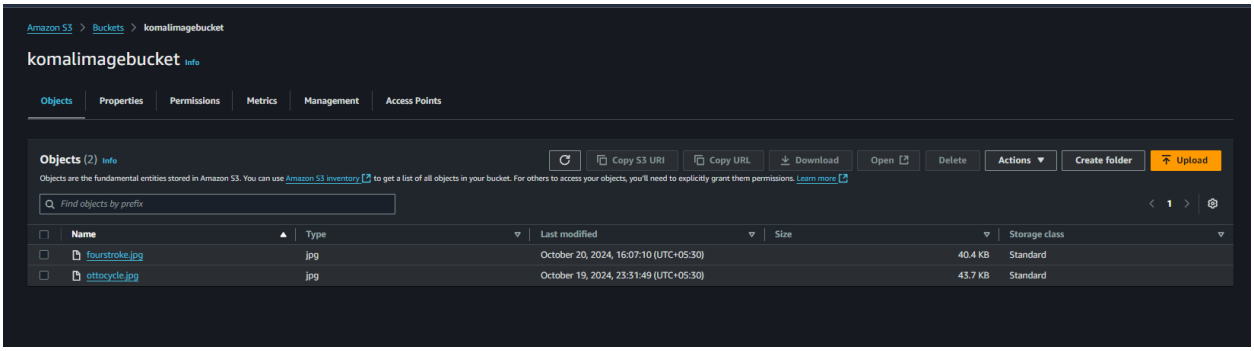
UPDATE THE PYTHON SCRIPT IN THE CODE SOURCE TO WRITE A CODE THAT PRINTS “AN IMAGE IS UPLOADED” WHEN THE LAMBDA FUNCTION IS TRIGGERED BY S3.

Write the below code in the script.

```
import json

import logging

# Set up logging

logger = logging.getLogger()

logger.setLevel(logging.INFO)


def lambda_handler(event, context):

    # Log incoming event data

    logger.info('Received event: %s', json.dumps(event, indent=2))


    # Get bucket name and object key from the event

    bucket_name = event['Records'][0]['s3']['bucket']['name']

    object_key = event['Records'][0]['s3']['object']['key']


    # Check if the uploaded object is an image

    if object_key.lower().endswith(('.png', '.jpg', '.jpeg', '.gif')):

        logger.info('An Image has been added to the bucket: %s, Object: %s', bucket_name, object_key)

    else:

        logger.info('Non-image file uploaded to the bucket: %s, Object: %s', bucket_name, object_key)


    return {

        'statusCode': 200,

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

    }
```

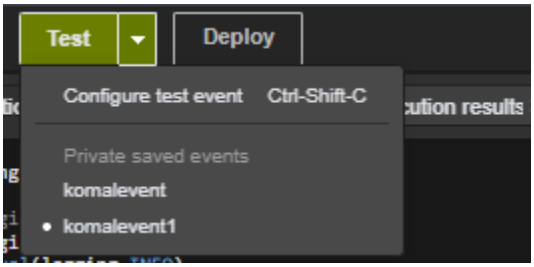


1)UPLOAD AN IMAGE IN THE S3 BUCKET:



NOW GO BACK TO LAMBDA FUNCTION AND

DEPLOY THE CODE BY CLICKING ON THE DEPLOY OPTION AND TEST IT.

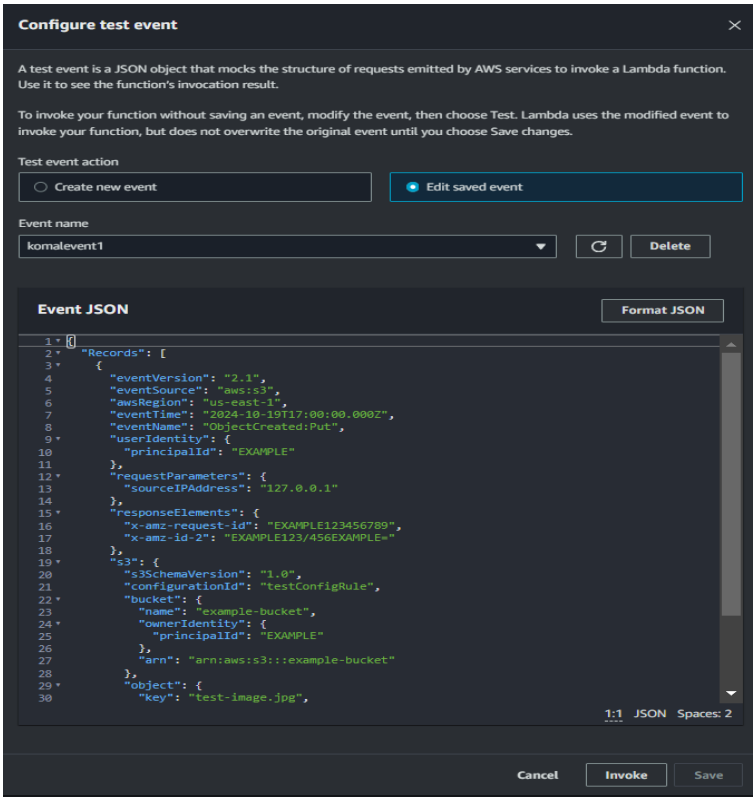


-USE THE SUITABLE EVENT FOR YOUR CODE.

THE EVENT SHOULD INCLUDE BELOW SCRIPT.


```
{
  "Records": [
    {
      "eventVersion": "2.1",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "2024-10-19T17:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/456EXAMPLE="
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "example-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::example-bucket"
```

```
      },
      "object": {
        "key": "test-image.jpg",
        "size": 1024,
        "eTag": "0123456789abcdef0123456789abcdef",
        "sequencer": "0A1B2C3D4E5F678901"
      }
    }
  }
]}
```

THE RESULT OF EXECUTION WILL LOOK LIKE:

```
lambda_function   Environment Var   Execution result:

Execution results

Test Event Name
komalevent1

Response
{
  "statusCode": 200,
  "body": "\"Lambda function executed successfully!\""
}

Function Logs
START RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Version: $LATEST
[INFO]  2024-10-20T11:30:12.312Z    9dccd201-07da-4a2c-8b7d-119fe2a3ed99    Received event: {
  "Records": [
    {
      "eventVersion": "2.1",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "2024-10-19T17:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/456EXAMPLE="
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "komalimagebucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::komalimagebucket"
        },
        "object": {
          "key": "ottocycle.jpg",
          "size": 1024,
          "eTag": "0123456789abcdef0123456789abcdef",
```

```
[INFO]  2024-10-20T11:30:12.313Z    9dccd201-07da-4a2c-8b7d-119fe2a3ed99    An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg
END RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99
REPORT RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99  Duration: 4.97 ms   Billed Duration: 5 ms   Memory Size: 128 MB Max Memory Used: 40 MB

Request ID
9dccd201-07da-4a2c-8b7d-119fe2a3ed99
```

-*the output of this code is of before the image fourstroke.jpg was uploaded.*

2)CHECK THE CLOUDWATCH LOGS

Once the image is uploaded. The lambda function is triggered and it can be checked in the logs.

Go to cloudwatch>log group name>log stream

```
CloudWatch > Log groups > /aws/lambda/komal-lambdafun > 2024/10/20/[$LATEST]a1eb2c02645f446ab79825d2058cc189

Log events
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
Actions ▼   Start tailing   Create metric fil
Filter events - press enter to search
Clear  1m  30m  1h  12h  Custom  Local timezone ▼  Display ▼

Timestamp                    Message
                             No older events at this moment. Retry
2024-10-20T16:58:05.921+05:30   INIT_START Runtime Version: python:3.8.v59 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:31ddccb9689d29d2fa6f49a11d508466fad8a5c21d1a9337d4cf257bcef8ebb0
2024-10-20T16:58:06.097+05:30   START RequestId: 7d1bf0a5-a3df-4dc2-93eb-1487c5960a33 Version: $LATEST
2024-10-20T16:58:06.098+05:30   [INFO] 2024-10-20T11:28:06.098Z 7d1bf0a5-a3df-4dc2-93eb-1487c5960a33 Received event: { "Records": [ { "eventVersion": "2.1", "eventSource": "aws:s3", "awsRegion": "us-east-1", "eventTime": "2024-10-19T17:00:00.000Z", "eventName": "Obje
2024-10-20T16:58:06.098+05:30   [INFO] 2024-10-20T11:28:06.098Z 7d1bf0a5-a3df-4dc2-93eb-1487c5960a33 An Image has been added to the bucket: example-bucket, Object: test-image.jpg
2024-10-20T16:58:06.118+05:30   END RequestId: 7d1bf0a5-a3df-4dc2-93eb-1487c5960a33
2024-10-20T16:58:06.118+05:30   REPORT RequestId: 7d1bf0a5-a3df-4dc2-93eb-1487c5960a33 Duration: 2.54 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 40 MB Init Duration: 174.16 ms
2024-10-20T17:00:12.312+05:30   START RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Version: $LATEST
2024-10-20T17:00:12.312+05:30   [INFO] 2024-10-20T11:30:12.312Z 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Received event: { "Records": [ { "eventVersion": "2.1", "eventSource": "aws:s3", "awsRegion": "us-east-1", "eventTime": "2024-10-19T17:00:00.000Z", "eventName": "Obje
2024-10-20T17:00:12.313+05:30   [INFO] 2024-10-20T11:30:12.313Z 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg
[INFO]  2024-10-20T11:30:12.313Z    9dccd201-07da-4a2c-8b7d-119fe2a3ed99    An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg
2024-10-20T17:00:12.317+05:30   END RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99
2024-10-20T17:00:12.317+05:30   REPORT RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Duration: 4.97 ms Billed Duration: 5 ms Memory Size: 128 MB Max Memory Used: 40 MB
                             No newer events at this moment. Auto retry paused. Resume
```

```
2024-10-20T17:00:12.313+05:30        [INFO] 2024-10-20T11:30:12.313Z 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg

[INFO]  2024-10-20T11:30:12.313Z       9dccd201-07da-4a2c-8b7d-119fe2a3ed99    An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg
```

AFTER UPLOADING ONE MORE IMAGE:

```
2024-10-20T16:07:11.478+05:30        [INFO] 2024-10-20T10:37:11.478Z e1324b12-b1ee-437a-bf9b-3cf4a16e471d An Image has been added to the bucket: komalimagebucket, Object: fourstroke.jpg

[INFO]  2024-10-20T10:37:11.478Z       e1324b12-b1ee-437a-bf9b-3cf4a16e471d    An Image has been added to the bucket: komalimagebucket, Object: fourstroke.jpg
```

3)FOR MONITORING THE LAMBDA FUNCTION BY USING NAGIOS ON EC2:

PREREQUISITES: INSTALL AND CONFIGURE NAGIOS ON YOUR EC2 INSTANCE.

1)GO TO THE LIBEXEC DIRECTORY *cd /usr/local/nagios/libexec*

```
[ec2-user@ip-172-31-43-5 ~]$ cd /usr/local/nagios/libexec
[ec2-user@ip-172-31-43-5 libexec]$ sudo chmod +x check_lambda.sh
```

2) RUN THE COMMAND *sudo nano check_lambda.sh*

3) WRITE THE BELOW CODE IN THE SCRIPT

```
#!/bin/bash

# AWS Lambda function name

LAMBDA_FUNCTION_NAME="komal-lambdafun"

# S3 bucket name (replace this with your actual bucket name)

S3_BUCKET_NAME="your-s3-bucket-name"

# Log group for the Lambda function (CloudWatch logs)

LOG_GROUP_NAME="/aws/lambda/$LAMBDA_FUNCTION_NAME"

# Get the current time to filter logs from this point onward

CURRENT_TIME=$(date -u +"%Y-%m-%dT%H:%M:%S.000Z")

# Monitor Lambda for S3 trigger (image upload)

echo "Monitoring Lambda function '$LAMBDA_FUNCTION_NAME' for image uploads to S3 bucket '$S3_BUCKET_NAME'."

# Monitor S3 bucket for any new object (image upload)

echo "Checking if there is a new image uploaded to the S3 bucket..."

LATEST_OBJECT=$(aws s3api list-objects --bucket $S3_BUCKET_NAME --query 'Contents[?contains(Key, `.jpg`) || contains(Key, `.png`)] | sort_by(@, &LastModified)[-1].Key' --output text)
```

```
if [ "$LATEST_OBJECT" == "None" ]; then
    echo "No image found in the bucket '$S3_BUCKET_NAME'."
    exit 1
else
    echo "Latest image uploaded: $LATEST_OBJECT"
fi
# Check the Lambda function's logs for recent activity (triggered by the S3 event)
echo "Checking Lambda function logs for the latest execution..."
# Fetch log streams (sorted by latest event time)
LATEST_LOG_STREAM=$(aws logs describe-log-streams --log-group-name $LOG_GROUP_NAME --order-by LastEventTime --descending --limit 1 --query 'logStreams[0].logStreamName' --output text)

if [ "$LATEST_LOG_STREAM" == "None" ]; then
    echo "No logs found for Lambda function '$LAMBDA_FUNCTION_NAME'."
    exit 2
else
    echo "Fetching logs from stream: $LATEST_LOG_STREAM"
fi
# Get logs for the latest Lambda execution
LOG_EVENTS=$(aws logs get-log-events --log-group-name $LOG_GROUP_NAME --log-stream-name $LATEST_LOG_STREAM --start-time $(date --date="$CURRENT_TIME" +%s%3N))

# Check if log events were found
if [ -z "$LOG_EVENTS" ]; then
    echo "No log events found for Lambda function execution after image upload."
    exit 3
else
    echo "Log events from Lambda function triggered by S3 upload:"
    echo "$LOG_EVENTS"
```

exit 0

fi

-IT MONITORS YOUR LAMBDA FUNCTION AND THE S3 BUCKET'S LATEST CHANGES.

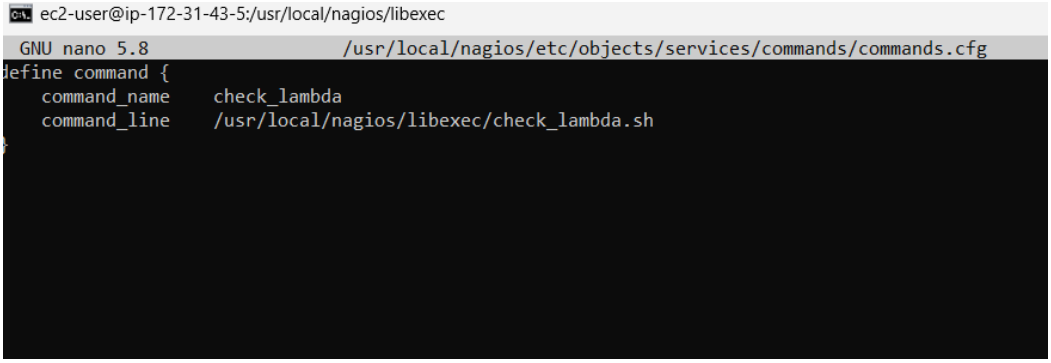
3)RUN *sudo chmod +x check_lambda.sh*

4) RUN THE FOLLOWING COMMAND TO ACCESS THE COMMANDS SCRIPT

sudo nano /usr/local/nagios/etc/objects/services/commands/commands.cfg

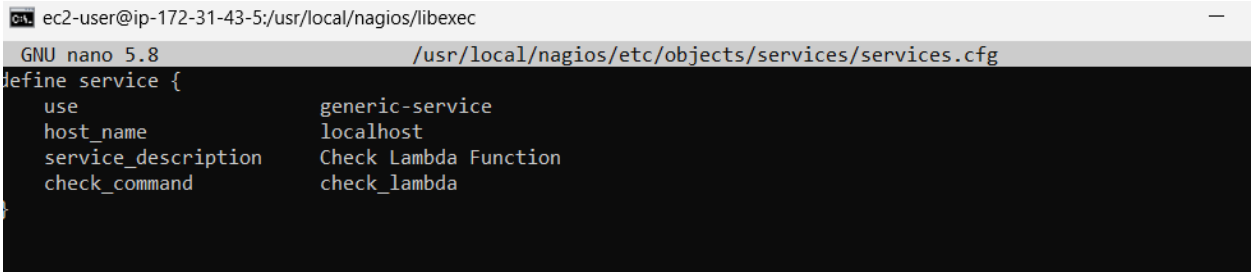
5)WRITE THE BELOW CODE IN THE SCRIPT

```
define command {
    command_name    check_lambda
    command_line    /usr/local/nagios/libexec/check_lambda.sh
}
```



6)RUN THE FOLLOWING COMMAND TO ACCESS THE SERVICES SCRIPT

sudo nano /usr/local/nagios/etc/objects/services/services.cfg



7)WRITE THE BELOW CODE IN THE SCRIPT

```
define service {
    use                 generic-service
    host_name           localhost
```

service_description Check Lambda Function

check_command check_lambda

}

8)SET THE AWS CREDENTIALS TO ACCESS AND MONITOR THE CREATED LAMBDA FUNCTION.

export AWS_SECRET_ACCESS_KEY="your_secret_key"

export AWS_SESSION_TOKEN="your_session_token"

export AWS_SESSION_TOKEN="your_session_token"

```
ec2-user@ip-172-31-43-5 libexec]$ export AWS_ACCESS_KEY_ID="ASIASWU6DPVCRK2RKYT7"
ec2-user@ip-172-31-43-5 libexec]$   export AWS_SESSION_TOKEN="IQoJb3JpZ2luX2VjEAcaCXVzLXdlc3QtMiJIMEYCIQCd+3eoNv+R4fYRdMXGn8EJZybHYYtYLCkcgsy+368WbQIhAJo76ofF9/13D7vdWLAMgjs0DaO9wtDvrUjhOm
2bgnVKrUCCHAQARoMMTg2MDg4OTE0MjQ1IgwL6krnu8dc2j4+R0cqkgIzb4mKEu3h7CcxacQ/K4ttwsRkWxcB23W8utmDOMKntu1oMT9oSPglenSAEGjUm56TbNo7jpMk3cJZ9h0t2aGS8DUgLO66Swg+ce9EQRPJsrnbYiGnFTR9yDu579BPgfipIBu
3V9iW+Y/N75f1VEbT3yLTevmQAXoX59RurOdMFMf1rfu1MN8JkgcAf8JeJr5BJIeMfnNtZcKPMGtoTopBMir3QVjplZZ+ABmJES+9CtAwitbfVopQILFAw2uXqRGPNtTv4eBrW0ZWbA2Zfn0LBunEpFr+jOg4i+EqE7YEVniWN2cnpJtqIDVoCnQNJ+E
9G5S5G9zYLjlmSKpH5MHojDRmcFs649qwJ+Snaa3hp80MPjO0rgGOpwBNni6lfZKV5IUd4/EtBl5dgLI9eFTTTICYcdaPnCD0F2WUf3VzVa1ibCdl00rSbwTtbEXsXx41PBmTj2GmZ4qwmjR2ftaXkF9uwSMxQKZkrebOOv7c6II/v85OpJQAEOLC3ald
WQt/AeS9+2DkDrlI6LHdJv+seEqCmNhQ3DEoKDZ1YXR93wHUQy3T6Dfv4IHC+vcQYzafvIzpks"TZUKk0XeF1qMYWojXAeDtlCjQQdoB%2BlhGP1awBsSln1tqA1CzLXC%2FxK2TQxG%2Fw29L1FGvzq%2BTREzcgLPEXz0BdwwmrGuWBBr2%2B0gMC5
ec2-user@ip-172-31-43-5 libexec]$ oEb50M7IzTZrAq%2FE9XWcn6sFvQ4d9KEZsr7Ep64GYz1a4JU6EWpaLYQQA1Xn10utbCbVoH6VZOR9knGhBxIFScLxDZ%2BoXs8Nycsai7jtazNvWF7sIUHmsNC8zRXl0%2Bw450hK%2Fkb%2FaSmHZ%2F
ec2-user@ip-172-31-43-5 libexec]$ ^CuhL96Sq8wGlovqK0mFYv3SeExWCrl4febQrCxM9Lk%2BEH554XksEIQKAk%2B2LcmvH35rK98VUY3V3MzrQC4%2F3ueNB0x1VPInqEkuWNuCT6KIDGs0lrnpmBa%2BkI85TrFW6hF8iDxFP27OW11bIk
ec2-user@ip-172-31-43-5 libexec]$ ^C2EhbTG3GiSMYQSwv%2F0XTgI3UlphHmPDsN3pyjmcwsZTTUm6H%2Bz8RlWpjnlP%2FDzzcocdgnleJUAviF%2FD08tWUA4z4ZAcsxEKXtbo9Ic4yVZH8FcL%2FOeD790kK3i3zZAIYMX6yjBMYFs4uFj
```

```
ec2-user@ip-172-31-43-5 libexec]$ export AWS_SECRET_ACCESS_KEY="uzjnupR3BdeJnPktvBZxxxzc4GHiCdmWJ9T+wGKL"
```

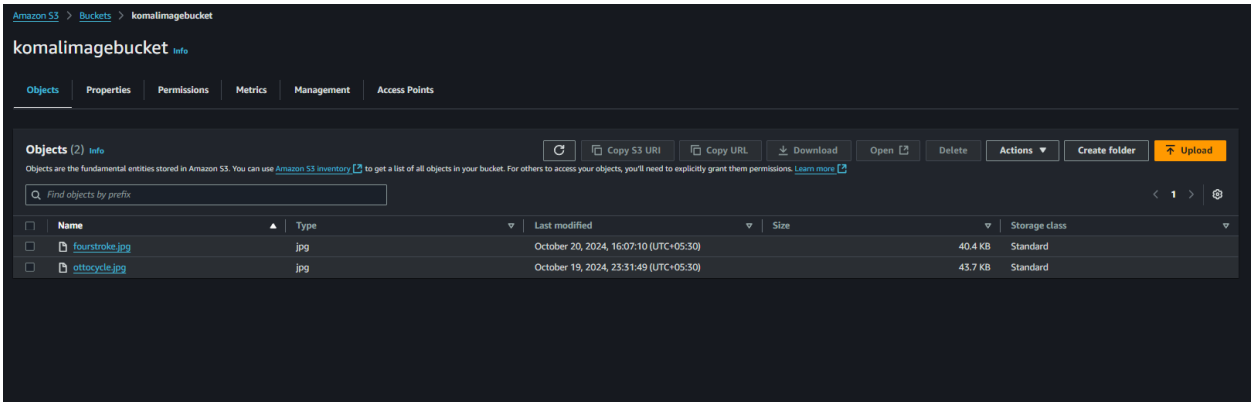
9) RESTART NAGIOS TO GET THE CHANGES UPDATED *sudo systemctl restart nagios*

10)RUN BELOW COMMAND TO CHECK THE MONITORING OF YOUR LAMBDA FUNCTION

./check_lambda.sh

```
Monitoring Lambda function 'komal-lambdafun' for image uploads to S3 bucket 'komalimagebucket'.
Checking if there is a new image uploaded to the S3 bucket...
Latest image uploaded: ottocycle.jpg
Checking Lambda function logs for the latest execution...
Fetching logs from stream: 2024/10/19/[$LATEST]6fffb263855042b295ab4aabd7f7f3fb
Log events from Lambda function triggered by S3 upload:
{
    "events": [],
    "nextForwardToken": "f/38567525873567295377391290756440763726736881459105103872/s",
    "nextBackwardToken": "b/38567365242593073582827578416871117500402558849515520000/s"
}
```

ALSO TRY UPLOADING A NEW IMAGE IN THE S3 BUCKET TO CHECK IF THE MONITORING IS WORKING



```
Monitoring Lambda function 'komal-lambdafun' for image uploads to S3 bucket 'komalimagebucket'.
Checking if there is a new image uploaded to the S3 bucket...
Latest image uploaded: fourstroke.jpg
Checking Lambda function logs for the latest execution...
Fetching logs from stream: 2024/10/20/[$LATEST]2385120a48774ac5aac4a8c8a8f44d49
Log events from Lambda function triggered by S3 upload:
{
    "events": [],
    "nextForwardToken": "f/38567529563337092200275542639233030527771534431094833152/s",
    "nextBackwardToken": "b/38567368944516776538911019911800350760030568842264576000/s"
}
[ec2-user@ip-172-31-43-5 libexec]$
Broadcast message from root@localhost (Sun 2024-10-20 10:37:49 UTC):

The system will power off now!

Connection to ec2-98-83-23-82.compute-1.amazonaws.com closed by remote host.
Connection to ec2-98-83-23-82.compute-1.amazonaws.com closed.
```

-IT ALSO NOTIFIES YOU ABOUT SYSTEM BEING POWERED OFF BY THE HOST.

11) RUN *aws lambda get-function --function-name your_function_name* TO GET THE CONFIGURATION OF THE LAMBDA FUNCTION.

```
[ec2-user@ip-172-31-43-5 libexec]$ aws lambda get-function --function-name komal-lambdafun
{
    "Configuration": {
        "FunctionName": "komal-lambdafun",
        "FunctionArn": "arn:aws:lambda:us-east-1:186088914245:function:komal-lambdafun",
        "Runtime": "python3.8",
        "Role": "arn:aws:iam::186088914245:role/LabRole",
        "Handler": "lambda_function.lambda_handler",
        "CodeSize": 557,
        "Description": "",
        "Timeout": 3,
        "MemorySize": 128,
        "LastModified": "2024-10-19T17:57:43.000+0000",
        "CodeSha256": "SUR6xL7BG9bHk3QQx3UFhBHbP1jj0Cm2cuJjsk9Wr94=",
        "Version": "$LATEST",
        "TracingConfig": {
            "Mode": "PassThrough"
        },
        "RevisionId": "0376bc2b-e931-4909-b7cc-e09fcd5ea146",
        "State": "Active",
        "LastUpdateStatus": "Successful",
        "PackageType": "Zip",
        "Architectures": [
            "x86_64"
        ],
        "EphemeralStorage": {
            "Size": 512
        },
        "SnapStart": {
            "ApplyOn": "None",
            "OptimizationStatus": "Off"
        },
        "RuntimeVersionConfig": {
            "RuntimeVersionArn": "arn:aws:lambda:us-east-1::runtime:31ddccb9689d29d2fa6f49a11d508466fad8a5c21d1a9337d4cf257bcef8ebb0"
        },
        "LoggingConfig": {
            "LogFormat": "Text",
            "LogGroup": "/aws/lambda/komal-lambdafun"
        }
    },
    "Code": {
        "RepositoryType": "S3",
        "Location": "https://prod-04-2014-tasks.s3.us-east-1.amazonaws.com/snapshots/186088914245/komal-lambdafun-dbc2793b-3ebf-49f3-afe9-5f2cbaefcfc0?versionId=Q2bTAvhlgWg2vaW86p1BcmWhJVoA
86.t&X-Amz-Security-Token=IQoJb3JpZ2luX2VjEAoaCXVzLWVhc3QtMSJHMEUCICPv8uNF8s2bYXCQ%2BHZN84NoLd2TRf7VER7dED3sA5uIAiEA91gyQiNPv9TVI8Sk4AJt9s5dyHBc%2FVXTvtfXsSZ3PUcqsgUIchAAGgw3NDk2NzgSMDI4Mzk
```