



Module 9: Lambda Assignment

Problem Statement:

You work for XYZ Corporation. Your corporation wants to launch a new web-based application and they do not want their servers to be running all the time. It should also be managed by AWS. Implement suitable solutions.

Tasks To Be Performed:

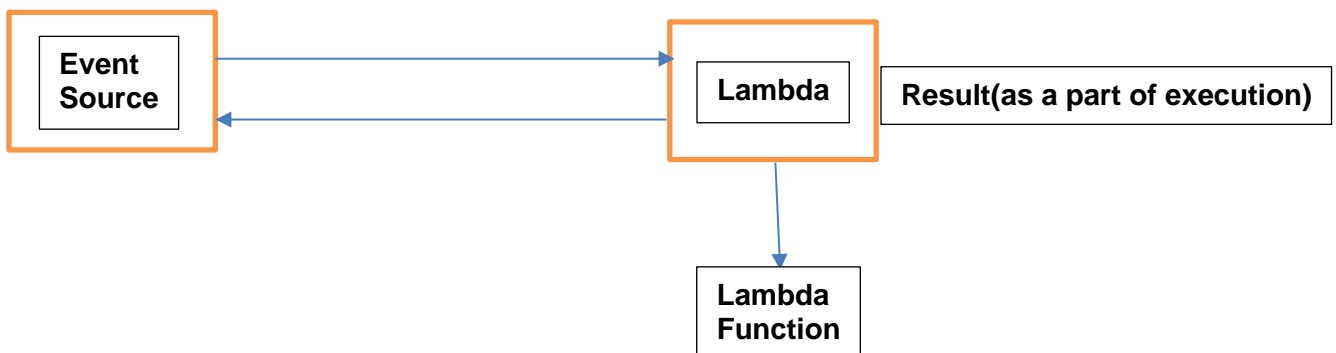
1. Create a sample Python Lambda function.
2. Set the Lambda Trigger as SQS and send a message to test invocations.

Solutions

LAMBDA- It is an event-driven serverless computing service, in simple words Lambda is used for processing pieces of code, and it is also used for integration. Here serverless means without managing the servers or infrastructure or without having visibility into the underlying infrastructure. Here EC2 and Lambda both compute services but the difference is that EC2 is not serverless, we have to manage the infrastructure when we launch an EC2, but that is not the case for Lambda.

Lambda has an identifier called Lambda function, Lambda function is nothing but a piece of code written by the developer and it provides support for different programming languages.

Lambda Architecture



Event source means the source which will invoke/call the Lambda, Lambda will run the Lambda function which will develop by the developer and the result will be there as a part of execution. Since Lambda needs to be invoked/called, it needs to read the data or Lambda will be needed to interact with the source, e.g. let's say I have the S3 bucket and I also have Lambda function. I need to print the content type of the object added to the S3 bucket. Content type means file extension, which means as soon as there is any put event in the S3 bucket, S3 will invoke the Lambda, Lambda will read the content and execute the Lambda function and according to the codes, Lambda will give the result.

Here we will create a bucket first, Management console → S3 → Create Bucket → Bucket Name → Region → Create bucket.

Steps to create Lambda function

Since it is a regional service, so we must select a valid region.

Go to management console → Search Lambda → Create a function → Author from scratch → Function Name → Runtime (Python3.9) → Architecture (x86_64) → Change default execution role (Here Lambda has to integrate with the AWS resources and also Lambda has to read the objects from the S3, and second thing is that lambda has to print the content type i.e. the file extension and the print statement will be available in the Cloudwatch logs, so Lambda needs the access to Cloudwatch, so we give admin access permission that lambda needs), so for that, I need to go to the IAM console and then Roles → Create Roles → AWS Service → use case (Lambda) → Next → permissions → Role Name (DemoLambda) → Create Role.

Now go to the Lambda function → Select use an existing role → Existing role (select the role that we created just as 'DemoLambda') → Create Function.

Now for adding a trigger, we need to create an S3 bucket first, so after creating the S3 bucket go to Lambda function → Click on Add trigger → Source (S3) → Put the bucket that you already created → Tick the I acknowledge → Add (Now trigger got added to the Lambda function)

Putting up the Lambda code

Go to the Lambda function → code → Write the codes → Deploy

```
import json
import boto3
import urllib

def lambda_handler(event, context):
    s3_client = boto3.client('s3')
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']
    key = urllib.parse.unquote_plus(key, encoding='utf-8')

    message = 'File' + key + ' is successfully uploaded in bucket ' + bucket_name
    print(message)

    response = s3_client.get_object(Bucket=bucket_name, Key=key)
    contents = response["Body"].read().decode()
    contents = json.loads(contents)

    print("The data in the file is: \n", contents)
```

Now upload a json file to the Bucket, go to the Lambda function → Monitor → View in Cloud watch, and here logs group will be created in the Lambda function. In the log stream, we will be able to see that object has been uploaded.

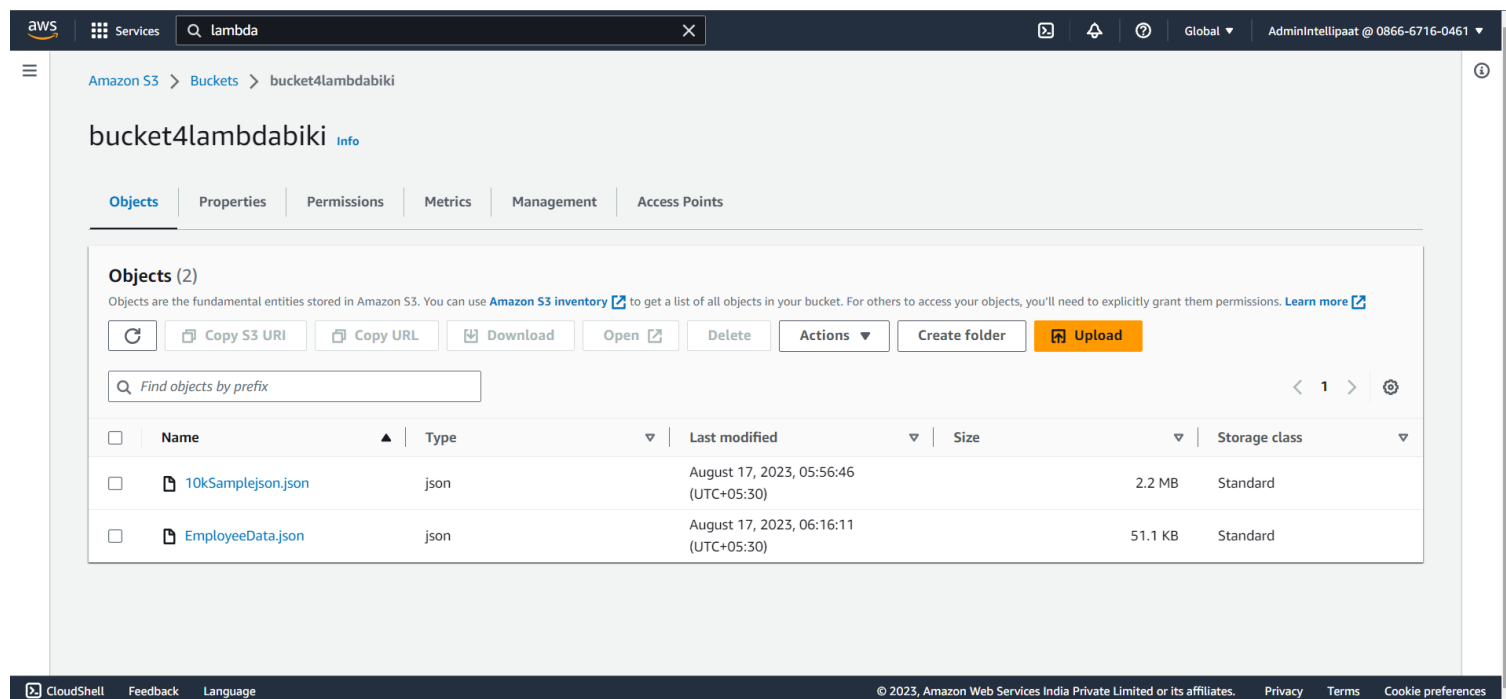
Now it's time to add the destination, but before going to add the destination we have to create SQS queue.

Steps to create SQS

- Go to the management console, search SQS.
- Create Queue.
- Put the Name.
- Server-side encryption- Disabled.
- Create Queue.

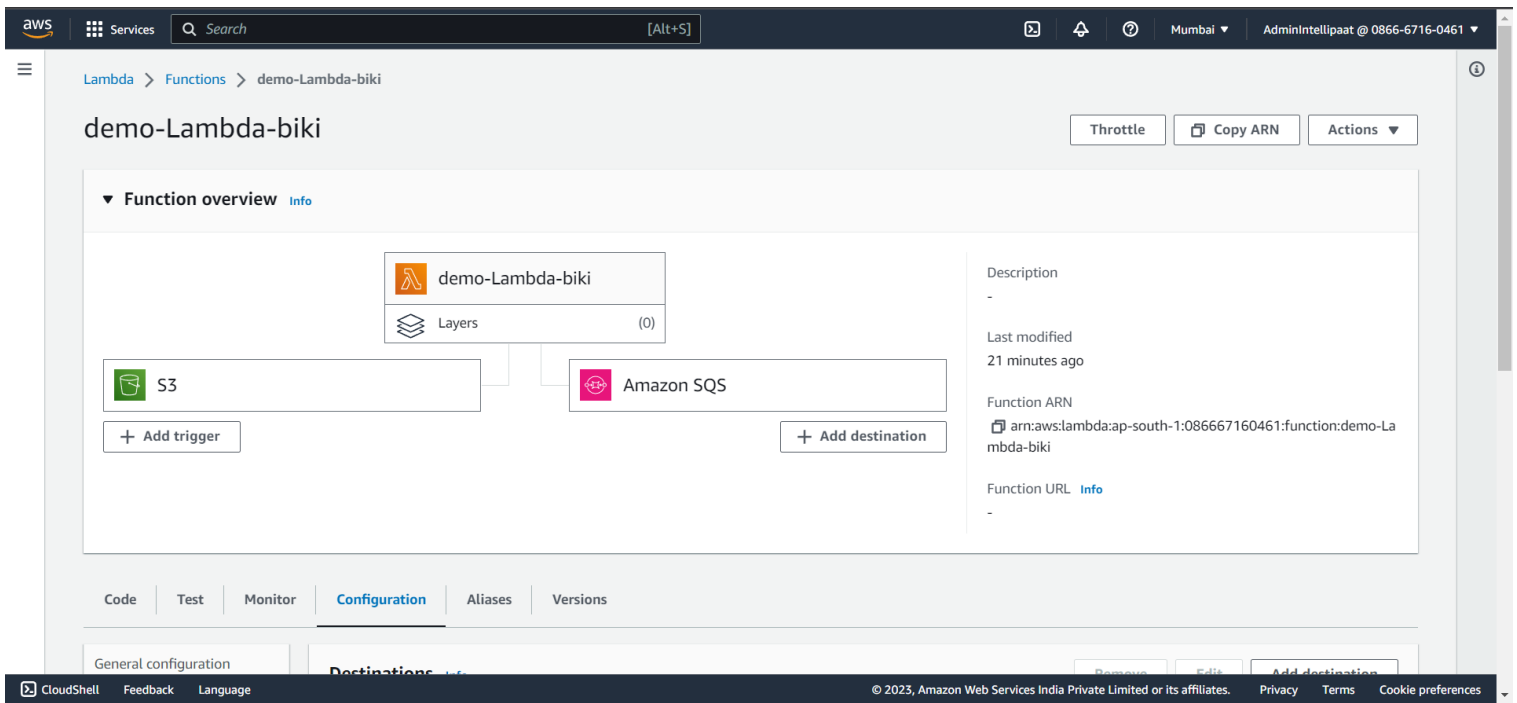
Now, time to add a Destination, click on Add Destination → source(Async) → Condition(on success) → Destination type(SQS queue) → refresh the destination, and you will see the SQS.

Results



The screenshot displays the AWS Management Console interface for the bucket 'bucket4lambdabiki'. The 'Objects' tab is selected, showing a list of two JSON files: '10kSamplejson.json' and 'EmployeeData.json'. The console includes a search bar, action buttons like 'Copy S3 URI', 'Download', and 'Upload', and a table with columns for Name, Type, Last modified, Size, and Storage class.

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	10kSamplejson.json	json	August 17, 2023, 05:56:46 (UTC+05:30)	2.2 MB	Standard
<input type="checkbox"/>	EmployeeData.json	json	August 17, 2023, 06:16:11 (UTC+05:30)	51.1 KB	Standard



The screenshot shows the AWS Lambda console for the function 'demo-Lambda-biki'. The 'Configuration' tab is selected, displaying the 'Function overview' section. This section includes a diagram of the function's architecture, showing it is triggered by an S3 bucket and has an Amazon SQS queue as a destination. The function is currently in the 'Layers' section, showing no layers are attached. The right-hand side of the console provides details about the function, including its description, last modified time (21 minutes ago), and its ARN. The bottom of the console shows the 'General configuration' and 'Destinations' sections.

demo-Lambda-biki

Throttle Copy ARN Actions

▼ Function overview Info

demo-Lambda-biki

Layers (0)

S3

+ Add trigger

Amazon SQS

+ Add destination

Description

-

Last modified

21 minutes ago

Function ARN

arn:aws:lambda:ap-south-1:086667160461:function:demo-Lambda-biki

Function URL Info

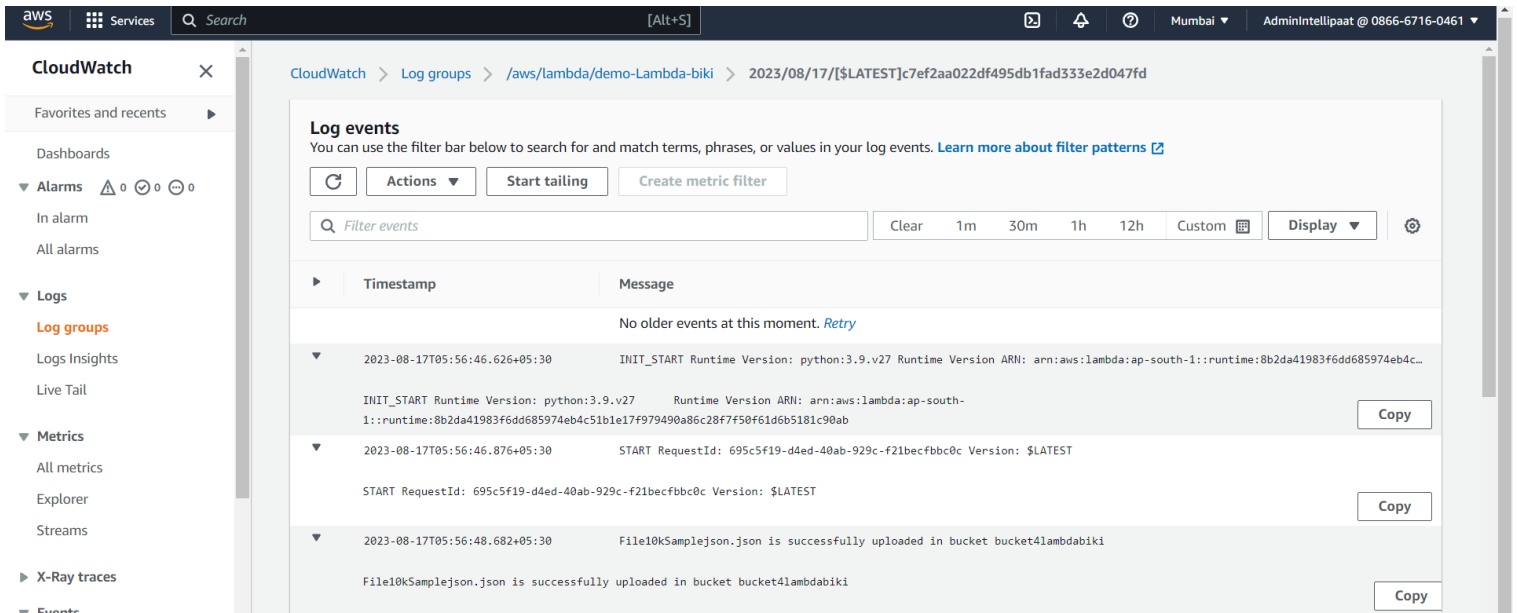
-

Code Test Monitor Configuration Aliases Versions

General configuration Destinations

CloudShell Feedback Language

© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences



The screenshot shows the AWS CloudWatch console for the log group '/aws/lambda/demo-Lambda-biki'. The 'Log events' section is displayed, showing a list of log events. The first event is an 'INIT_START' event, followed by a 'START' event, and then a 'File10kSamplejson.json' event. Each event has a timestamp and a message. The console also includes a search bar, filters, and a 'Start tailing' button.

CloudWatch Log groups /aws/lambda/demo-Lambda-biki 2023/08/17/[\$LATEST]c7ef2aa022df495db1fad333e2d047fd

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](#)

Filter events

Clear 1m 30m 1h 12h Custom Display

Timestamp Message

No older events at this moment. [Retry](#)

2023-08-17T05:56:46.626+05:30 INIT_START Runtime Version: python:3.9.v27 Runtime Version ARN: arn:aws:lambda:ap-south-1::runtime:8b2da41983f6dd685974eb4c...

2023-08-17T05:56:46.876+05:30 START RequestId: 695c5f19-d4ed-40ab-929c-f21becfbbc8c Version: \$LATEST

2023-08-17T05:56:48.682+05:30 File10kSamplejson.json is successfully uploaded in bucket bucket41lambdabiki

Copy

Copy

Copy

CloudWatch

Favorites and recents

Dashboards

Alarms 0 0 0 0

In alarm

All alarms

Logs

Log groups

Logs Insights

Live Tail

Metrics

All metrics

Explorer

Streams

X-Ray traces

Events

aws

Services

Search

[Alt+S]

Global

AdminIntelliPaat @ 0866-6716-0461

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analyzers

Settings

Credential report

Demo-lambda_role

Delete

Allows Lambda functions to call AWS services on your behalf.

Summary

Edit

Creation date

August 17, 2023, 05:35 (UTC+05:30)

ARN

arn:aws:iam::086667160461:role/Demo-lambda_role

Last activity

None

Maximum session duration

1 hour

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

Permissions policies (1)

Info

You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter.

Policy name

Type

Description

AdministratorAccess

AWS managed - job function

Provides full access to AWS services and resources.

CloudShell

Feedback

Language

© 2023, Amazon Web Services India Private Limited or its affiliates.

Privacy

Terms

Cookie preferences

Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved