Prathamesh Palve D15A - 32

AdvDevOps Case Study 12: Serverless Logging with S3 and Lambda

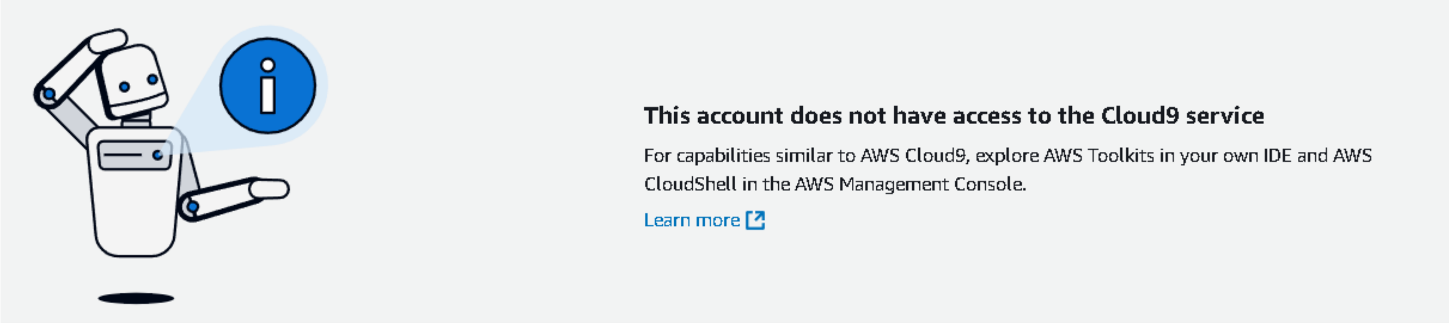
- **Concepts Used**: AWS Lambda, S3, and AWS Cloud9.

- **Problem Statement**: "Set up a Lambda function using AWS Cloud9 that triggers when a text file is uploaded to an S3 bucket. The Lambda function should read the file's content and log it."

- **Tasks**:
  - Create a Lambda function in Python using AWS Cloud9.
  - Configure an S3 bucket as the trigger for the Lambda function.
  - Upload a text file to the S3 bucket and verify that the Lambda function logs the content.

Note**

AWS **Cloud9** has been **discontinued**, so we will now use **EC2** for our development environment.



This account does not have access to the Cloud9 service

For capabilities similar to AWS Cloud9, explore AWS Toolkits in your own IDE and AWS CloudShell in the AWS Management Console.

INTRODUCTION

➢ Case Study Overview

This case study focuses on the integration of AWS Lambda, Amazon S3, and AWS Cloud9 to create a serverless application that automates the processing of text files. The primary objective is to set up a Lambda function that is triggered by an event—specifically, when a text file is uploaded to an S3 bucket. The Lambda function will read the content of the uploaded file and log it for further processing or analysis. This setup exemplifies the ease of using AWS services to build efficient and scalable applications without the need for managing infrastructure.

➢ Key Feature and Application

- **Event-Driven Architecture**: Automatically triggers the Lambda function when a text file is uploaded to S3, enabling real-time data processing without manual intervention.
- **Serverless Computing**: AWS Lambda runs code without the need for server management, allowing developers to focus on writing and deploying code efficiently.
- **Automatic File Processing**: The Lambda function reads and logs the content of uploaded text files, facilitating seamless automation in data workflows.
- **Scalability**: The architecture can handle increasing workloads without performance isues, making it suitable for variable data volumes.
- **Cost-Effectiveness**: Users pay only for the compute time and storage used, optimizing costs for data processing tasks.
- **Practical Applications**: Ideal for data ingestion pipelines, real-time analytics, and automated file processing across various industries

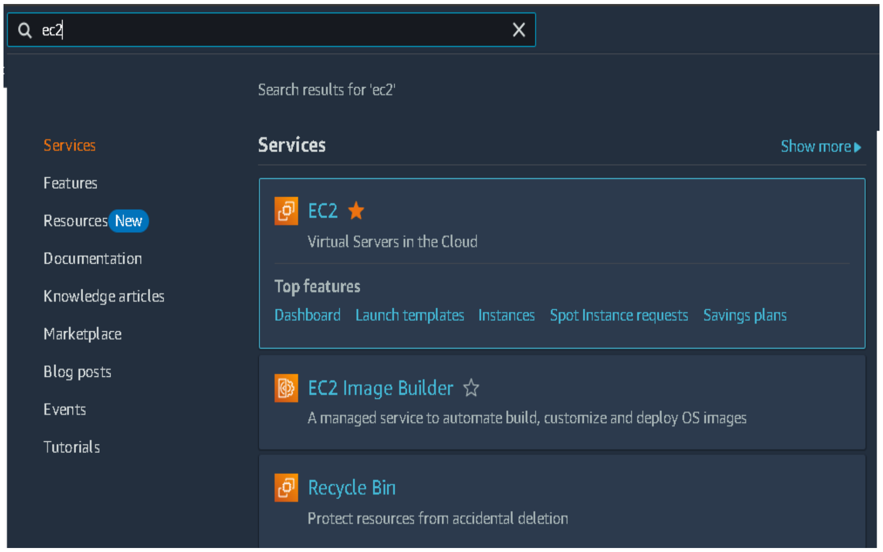
➢ Third-Year Project Integration

This case study can enhance the "TradeNest" project, a MERN stack web application similar to OLX, where seniors sell study materials to juniors. Skills gained from setting up the Lambda function, configuring S3 triggers, and using AWS Cloud9 can be applied to build a more complex system, such as automating document processing or implementing real-time analytics. This integration will deepen understanding of cloud computing and serverless architectures, which are crucial in modern software development.

STEPS:

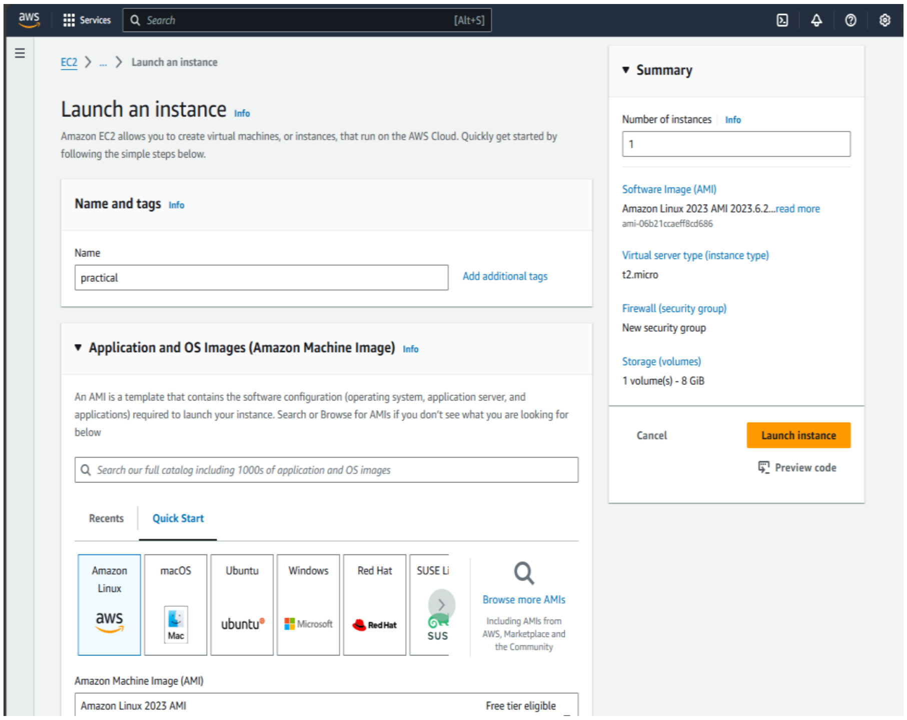
1. Launch an EC2 Instance

1.1 Login to AWS Console and go to EC2 service.



1.2 Click on "Launch Instance".

i. AMI: Choose Amazon Linux 2.
ii. Instance Type: Select t2.micro (eligible for free tier).
iii. Key Pair: Create a new key pair (or select an existing one). You'll need this for SSH access.
iv. Network Settings:
   - ○ Choose default VPC.
   - ○ Security Group: Create a new security group:
     - Inbound Rules:
       - ■ SSH (TCP port 22): Allow from your IP.
       - ■ HTTP (TCP port 80): Optional, allows browser access.
       - ■ HTTPS (TCP port 443): Optional, for secure traffic.
     - Outbound Rules:
       - ■ Allow all outbound traffic (default).



Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: practical

Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Amazon Machine Image (AMI): Amazon Linux 2023 AMI — Free tier eligible

▼ Summary

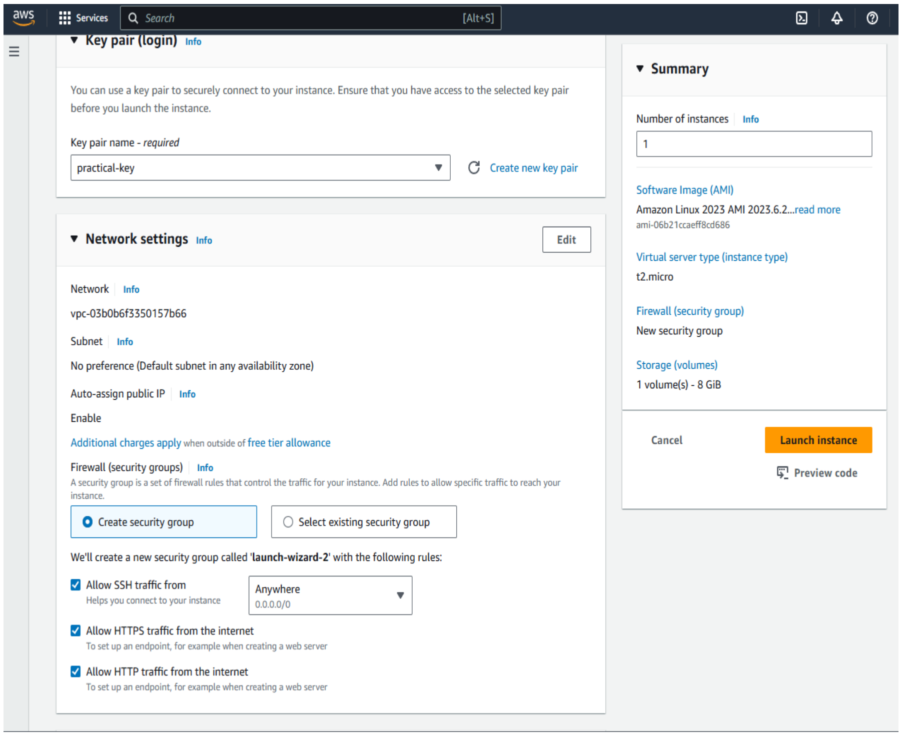
Number of instances Info: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.6.2...read more
ami-06b21ccaeff8cd686

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB



▼ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*: practical-key ▼

Create new key pair

▼ Network settings Info — Edit

Network Info: vpc-03b0b6f3350157b66

Subnet Info: No preference (Default subnet in any availability zone)

Auto-assign public IP Info: Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

We'll create a new security group called '**launch-wizard-2**' with the following rules:

☑ Allow SSH traffic from — Anywhere ▼ 0.0.0.0/0
Helps you connect to your instance

☑ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

☑ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

▼ Summary

Number of instances Info: 1

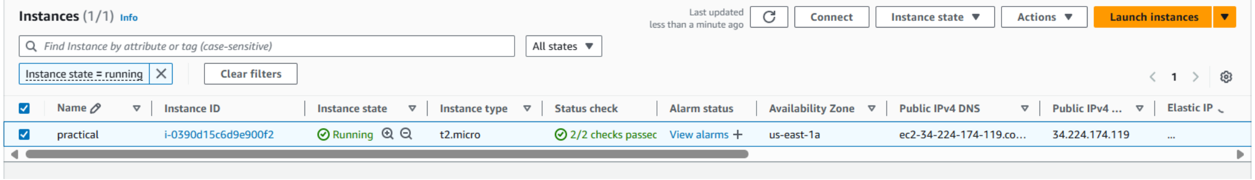
Software Image (AMI): Amazon Linux 2023 AMI 2023.6.2...read more
ami-06b21ccaeff8cd686

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

1.3 Launch the instance and wait for it to be ready.

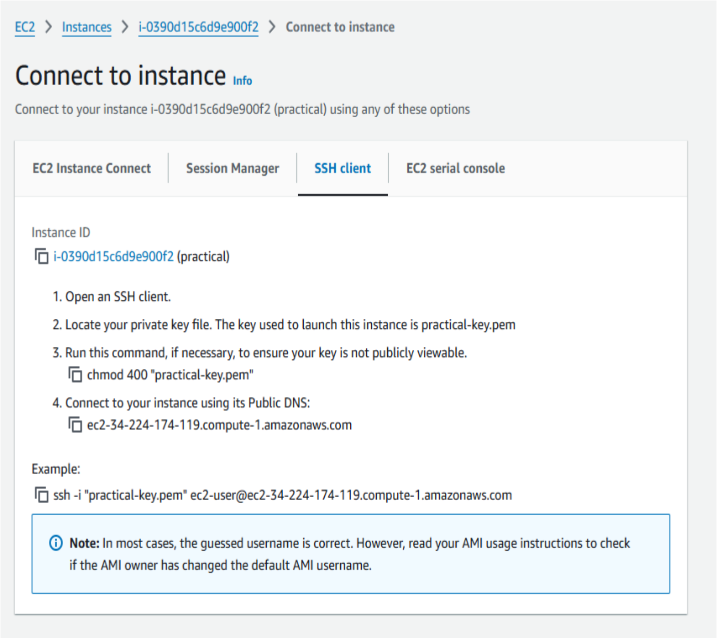


Instances (1/1) Info
Last updated less than a minute ago
Connect
Instance state ▼
Actions ▼
Launch instances ▼
Find Instance by attribute or tag (case-sensitive)
All states ▼
Instance state = running ✕
Clear filters
1

☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
☑	practical	i-0390d15c6d9e900f2	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1a	ec2-34-224-174-119.co...	34.224.174.119	...

1.4 Connect to the EC2 instance via SSH:

```
ssh -i <your-key.pem> ec2-user@<your-ec2-public-dns>
```



Connect to instance Info

Connect to your instance i-0390d15c6d9e900f2 (practical) using any of these options

Instance ID

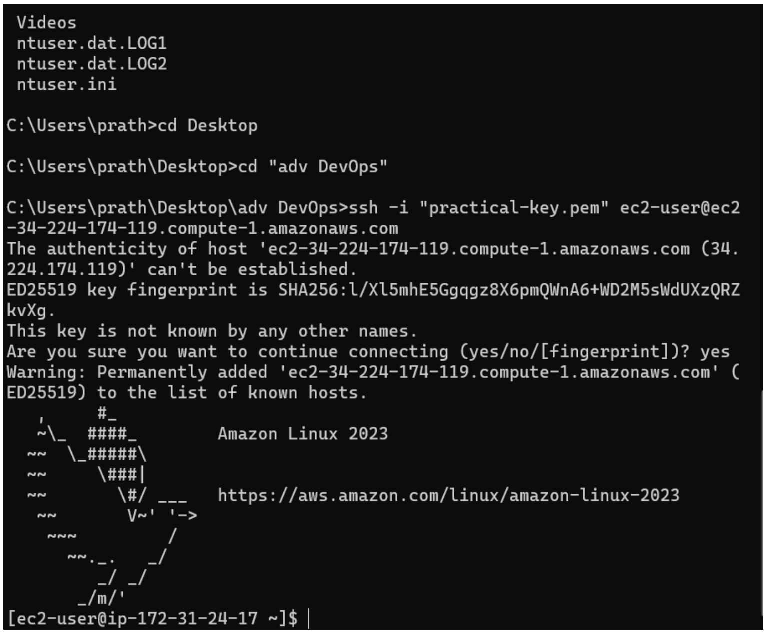
i-0390d15c6d9e900f2 (practical)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is practical-key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   `chmod 400 "practical-key.pem"`
4. Connect to your instance using its Public DNS:
   `ec2-34-224-174-119.compute-1.amazonaws.com`

Example:

`ssh -i "practical-key.pem" ec2-user@ec2-34-224-174-119.compute-1.amazonaws.com`

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

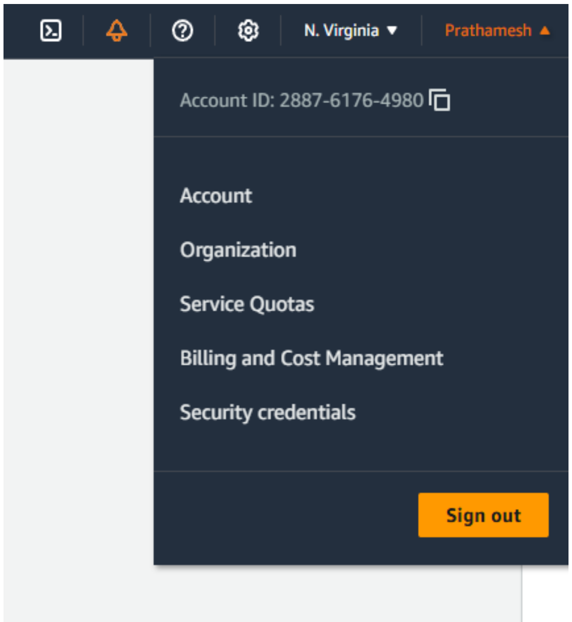


```
 Videos
 ntuser.dat.LOG1
 ntuser.dat.LOG2
 ntuser.ini

C:\Users\prath>cd Desktop

C:\Users\prath\Desktop>cd "adv DevOps"

C:\Users\prath\Desktop\adv DevOps>ssh -i "practical-key.pem" ec2-user@ec2
-34-224-174-119.compute-1.amazonaws.com
The authenticity of host 'ec2-34-224-174-119.compute-1.amazonaws.com (34.
224.174.119)' can't be established.
ED25519 key fingerprint is SHA256:l/Xl5mhE5Ggqgz8X6pmQWnA6+WD2M5sWdUXzQRZ
kvXg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-34-224-174-119.compute-1.amazonaws.com' (
ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-24-17 ~]$
```

2. Create Access keys for Root user

2.1 **Access the Root User Security Credentials**:

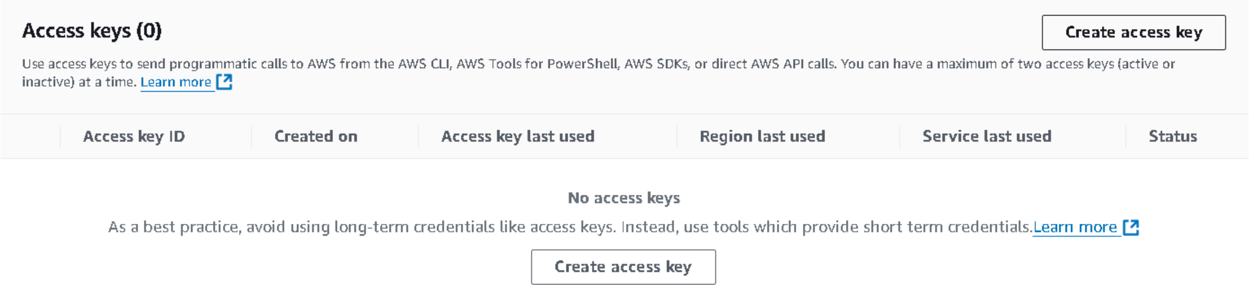
- In the top-right corner of AWS Management Console, click on your account name or email address, and then click **Security Credentials** from the dropdown menu.



Account ID: 2887-6176-4980

2.2 **Manage Root Access Keys**:

- Scroll down to the **Access keys for the root account** section.
- If you don’t have any existing access keys, click on **Create New Access Key**.
    - This will generate an **Access Key ID** and a **Secret Access Key** for your root user.
- **Download** the keys or **copy** them immediately. You won’t be able to see the **Secret Access Key** again after closing this page.



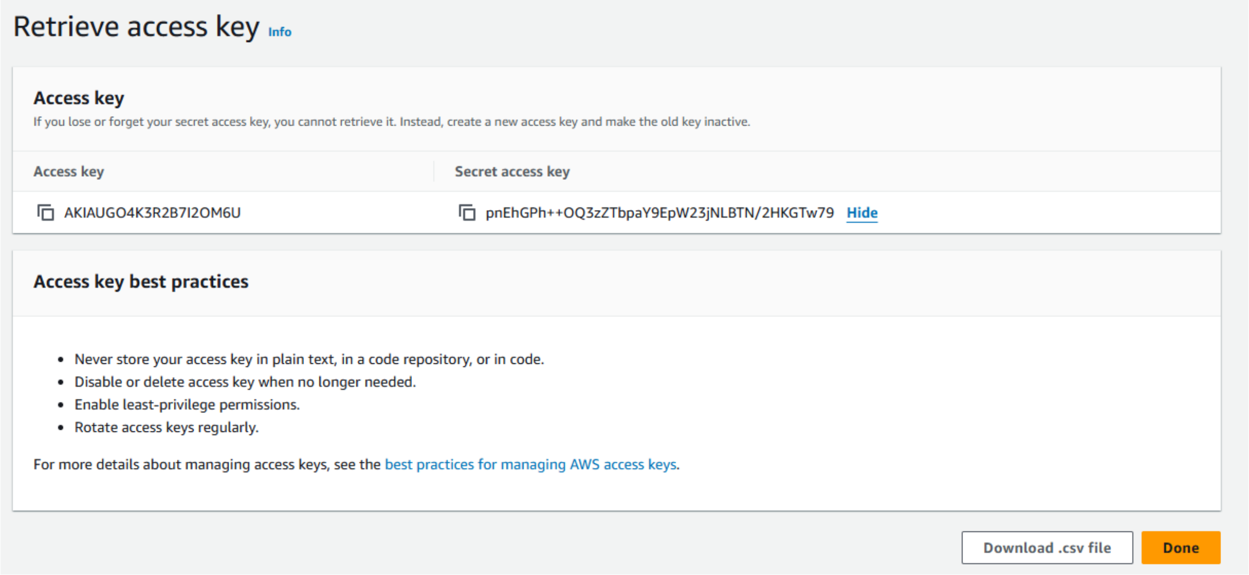
Access keys (0)

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

No access keys

As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials.Learn more



Retrieve access key

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAUGO4K3R2B7I2OM6U	pnEhGPh++OQ3zZTbpaY9EpW23jNLBTN/2HKGTw79 Hide

Access key best practices

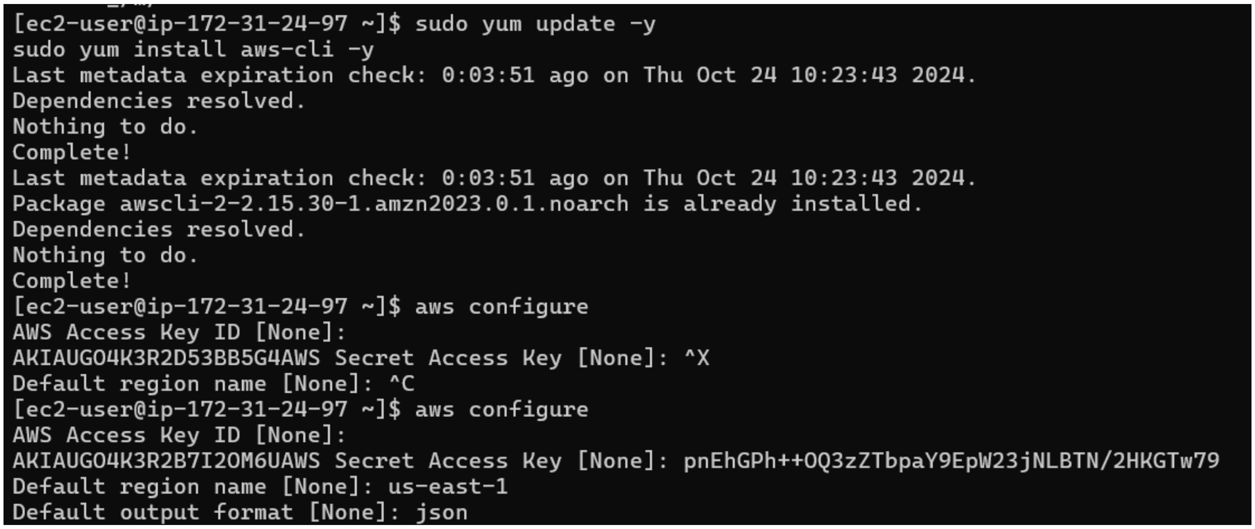
- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

3. Install AWS CLI and Configure EC2

3.1 Update packages and install AWS CLI:

sudo yum update -y
sudo yum install aws-cli -y



```
[ec2-user@ip-172-31-24-97 ~]$ sudo yum update -y
sudo yum install aws-cli -y
Last metadata expiration check: 0:03:51 ago on Thu Oct 24 10:23:43 2024.
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 0:03:51 ago on Thu Oct 24 10:23:43 2024.
Package awscli-2-2.15.30-1.amzn2023.0.1.noarch is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-24-97 ~]$ aws configure
AWS Access Key ID [None]:
AKIAUGO4K3R2D53BB5G4AWS Secret Access Key [None]: ^X
Default region name [None]: ^C
[ec2-user@ip-172-31-24-97 ~]$ aws configure
AWS Access Key ID [None]:
AKIAUGO4K3R2B7I2OM6UAWS Secret Access Key [None]: pnEhGPh++OQ3zZTbpaY9EpW23jNLBTN/2HKGTw79
Default region name [None]: us-east-1
Default output format [None]: json
```

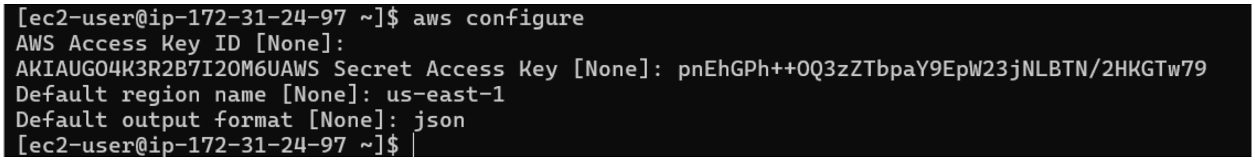
3.2 Configure AWS CLI:

aws configure

Enter your:

- AWS **Access Key ID**

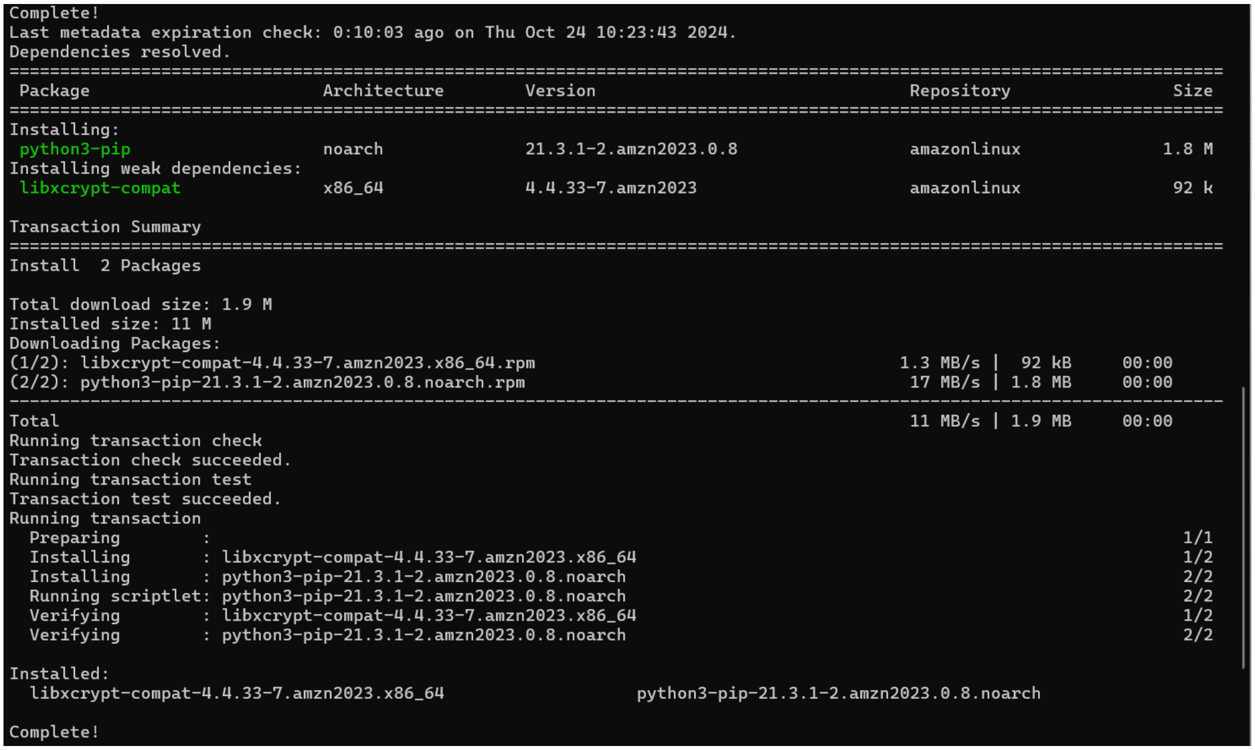
- AWS **Secret Access Key**
- Region (e.g., `us-east-1`)
- Output format: `json`



```
[ec2-user@ip-172-31-24-97 ~]$ aws configure
AWS Access Key ID [None]:
AKIAUGO4K3R2B7I2OM6UAWS Secret Access Key [None]: pnEhGPh++OQ3zZTbpaY9EpW23jNLBTN/2HKGTw79
Default region name [None]: us-east-1
Default output format [None]: json
[ec2-user@ip-172-31-24-97 ~]$ |
```

3.3 **Install Python and pip** (since Lambda uses Python):

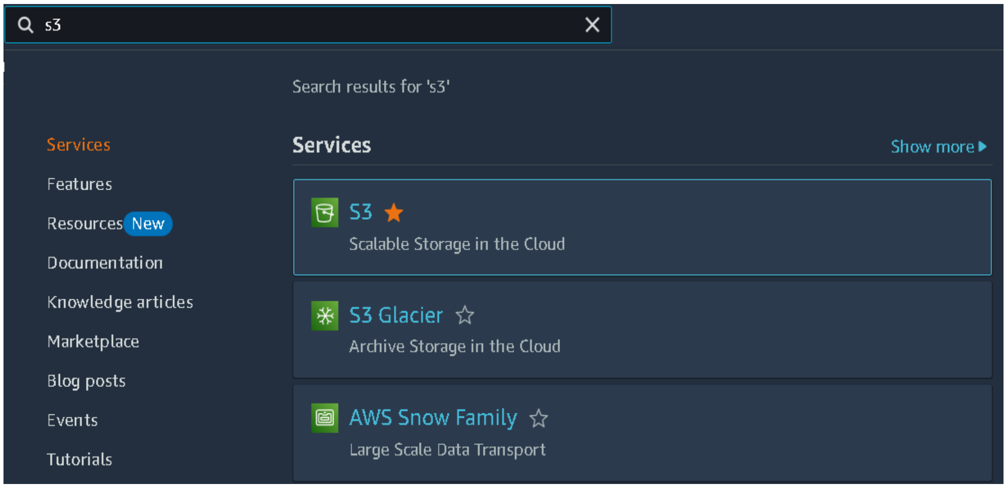
sudo yum install python3 -y
sudo yum install python3-pip -y



```
Complete!
Last metadata expiration check: 0:10:03 ago on Thu Oct 24 10:23:43 2024.
Dependencies resolved.
=========================================================================================================================
 Package                       Architecture        Version                               Repository                 Size
=========================================================================================================================
Installing:
 python3-pip                   noarch              21.3.1-2.amzn2023.0.8                 amazonlinux               1.8 M
Installing weak dependencies:
 libxcrypt-compat              x86_64              4.4.33-7.amzn2023                     amazonlinux                92 k

Transaction Summary
=========================================================================================================================
Install  2 Packages

Total download size: 1.9 M
Installed size: 11 M
Downloading Packages:
(1/2): libxcrypt-compat-4.4.33-7.amzn2023.x86_64.rpm                                    1.3 MB/s |  92 kB      00:00
(2/2): python3-pip-21.3.1-2.amzn2023.0.8.noarch.rpm                                      17 MB/s | 1.8 MB      00:00
-------------------------------------------------------------------------------------------------------------------------
Total                                                                                    11 MB/s | 1.9 MB      00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                 1/1
  Installing       : libxcrypt-compat-4.4.33-7.amzn2023.x86_64                                                       1/2
  Installing       : python3-pip-21.3.1-2.amzn2023.0.8.noarch                                                        2/2
  Running scriptlet: python3-pip-21.3.1-2.amzn2023.0.8.noarch                                                        2/2
  Verifying        : libxcrypt-compat-4.4.33-7.amzn2023.x86_64                                                       1/2
  Verifying        : python3-pip-21.3.1-2.amzn2023.0.8.noarch                                                        2/2

Installed:
  libxcrypt-compat-4.4.33-7.amzn2023.x86_64                   python3-pip-21.3.1-2.amzn2023.0.8.noarch

Complete!
```

4. Create and S3 Bucket

4.1 In the AWS Management Console, go to **S3**.



Search results for 's3'

Services

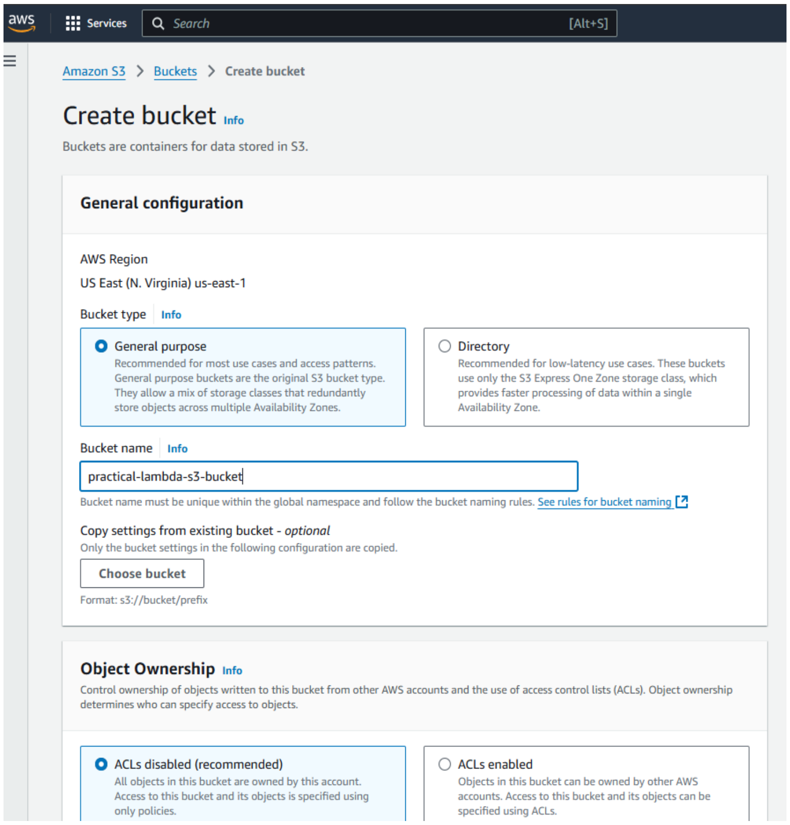
S3 ★
Scalable Storage in the Cloud

S3 Glacier ☆
Archive Storage in the Cloud

AWS Snow Family ☆
Large Scale Data Transport

4.2 Click **Create bucket**:

- **Bucket Name**: Give a unique name (e.g., `lambda-s3-trigger-bucket`).
- **Region**: Keep the same as your AWS Configuration (e.g., `us-east-1`).
- Keep other settings default.



Create bucket

Buckets are containers for data stored in S3.

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type

● General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

○ Directory
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name: practical-lambda-s3-bucket

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Format: s3://bucket/prefix

Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

● ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

○ ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

4.3 Create the bucket.

5. Create the Lambda Function code.

5.1 **On your EC2 instance**, create the Python Lambda function code:

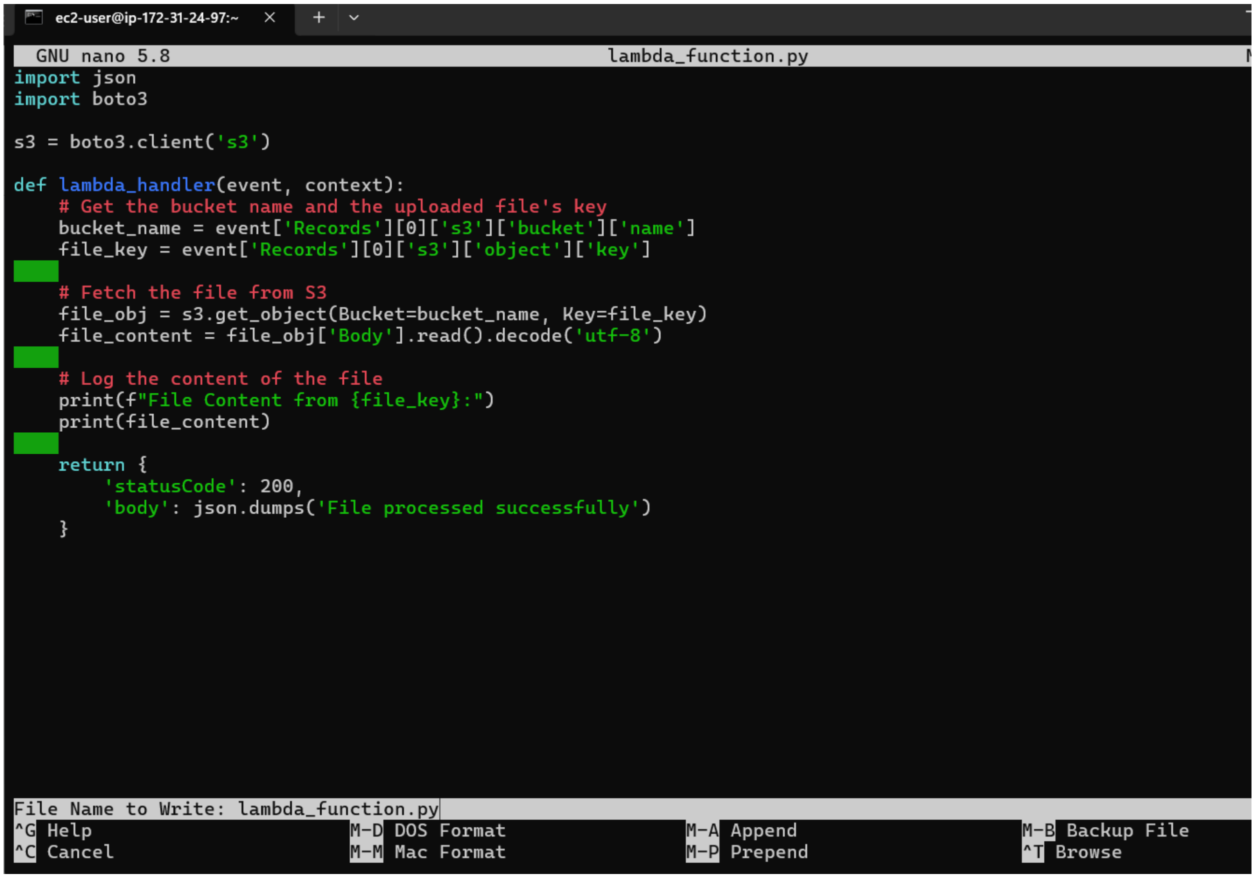
```
nano lambda_function.py
```

```
[ec2-user@ip-172-31-24-97 ~]$ nano lambda_function.py
```

5.2 **Write the following Lambda function** to read the uploaded file from S3:

```
import json
import boto3

s3 = boto3.client('s3')

def lambda_handler(event, context):
    # Get the bucket name and the uploaded file's key
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    file_key = event['Records'][0]['s3']['object']['key']

    # Fetch the file from S3
    file_obj = s3.get_object(Bucket=bucket_name, Key=file_key)
    file_content = file_obj['Body'].read().decode('utf-8')

    # Log the content of the file
    print(f"File Content from {file_key}:")
    print(file_content)

    return {
        'statusCode': 200,
        'body': json.dumps('File processed successfully')
    }
```

5.3 Press `Ctrl+X`, then `Y`, and hit `Enter`.

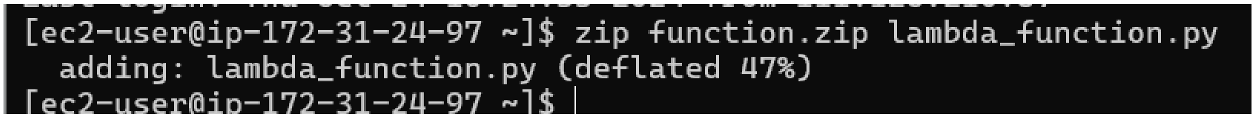


```
  GNU nano 5.8                          lambda_function.py
import json
import boto3

s3 = boto3.client('s3')

def lambda_handler(event, context):
    # Get the bucket name and the uploaded file's key
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    file_key = event['Records'][0]['s3']['object']['key']

    # Fetch the file from S3
    file_obj = s3.get_object(Bucket=bucket_name, Key=file_key)
    file_content = file_obj['Body'].read().decode('utf-8')

    # Log the content of the file
    print(f"File Content from {file_key}:")
    print(file_content)

    return {
        'statusCode': 200,
        'body': json.dumps('File processed successfully')
    }

File Name to Write: lambda_function.py
^G Help                M-D DOS Format          M-A Append              M-B Backup File
^C Cancel              M-M Mac Format          M-P Prepend             ^T Browse
```

6. Deploy the Lambda function from EC2

6.1 Package the Lambda function:

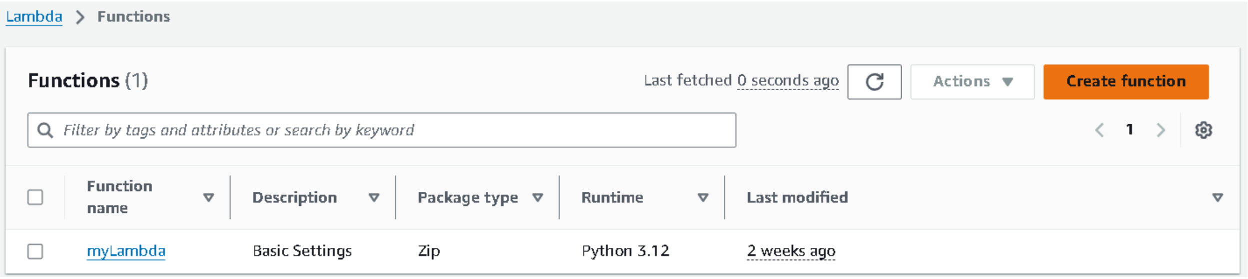
zip function.zip lambda_function.py



```
[ec2-user@ip-172-31-24-97 ~]$ zip function.zip lambda_function.py
  adding: lambda_function.py (deflated 47%)
[ec2-user@ip-172-31-24-97 ~]$
```

6.2 **Create a Lambda function in AWS Console**:

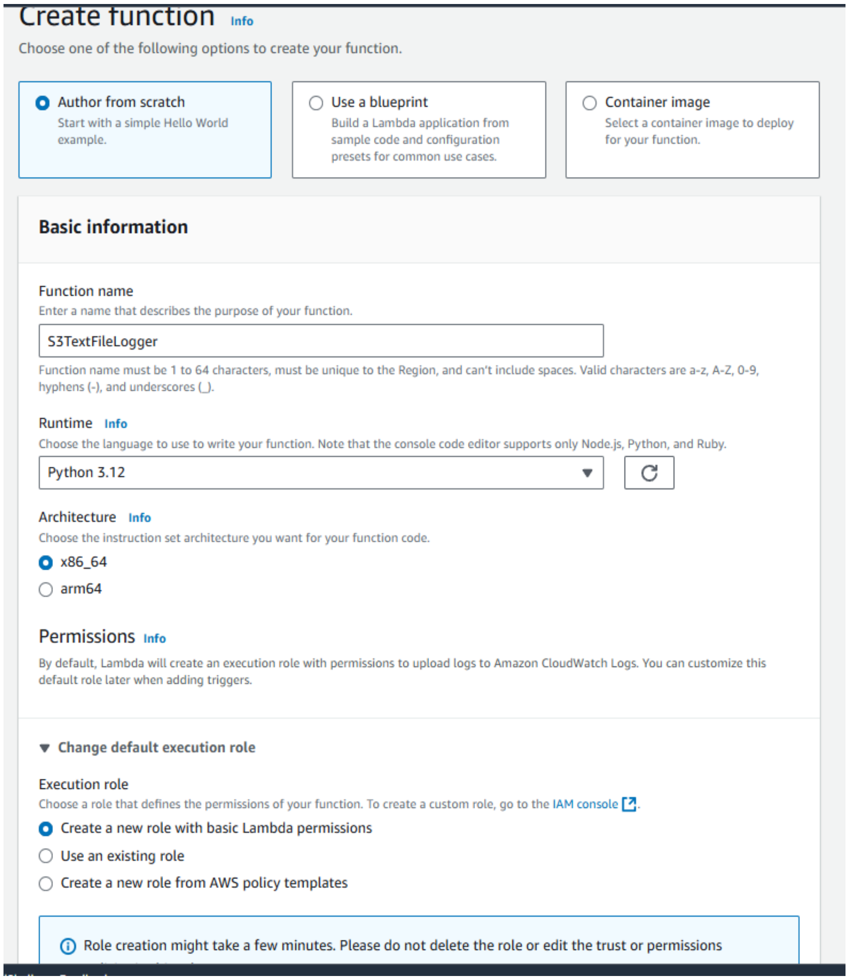
- Go to **Lambda** > **Create Function**.



Functions (1)   Last fetched 0 seconds ago

□	Function name	Description	Package type	Runtime	Last modified
□	myLambda	Basic Settings	Zip	Python 3.12	2 weeks ago

- Choose **Author from Scratch**:
  - **Function Name**: `S3TextFileLogger`
  - **Runtime**: Python 3.12
  - **Execution Role**: Select "Create a new role with basic Lambda permissions."



Create function Info

Choose one of the following options to create your function.

● Author from scratch
Start with a simple Hello World example.

○ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

○ Container image
Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.

S3TextFileLogger

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.12 ▼

Architecture Info
Choose the instruction set architecture you want for your function code.

● x86_64
○ arm64

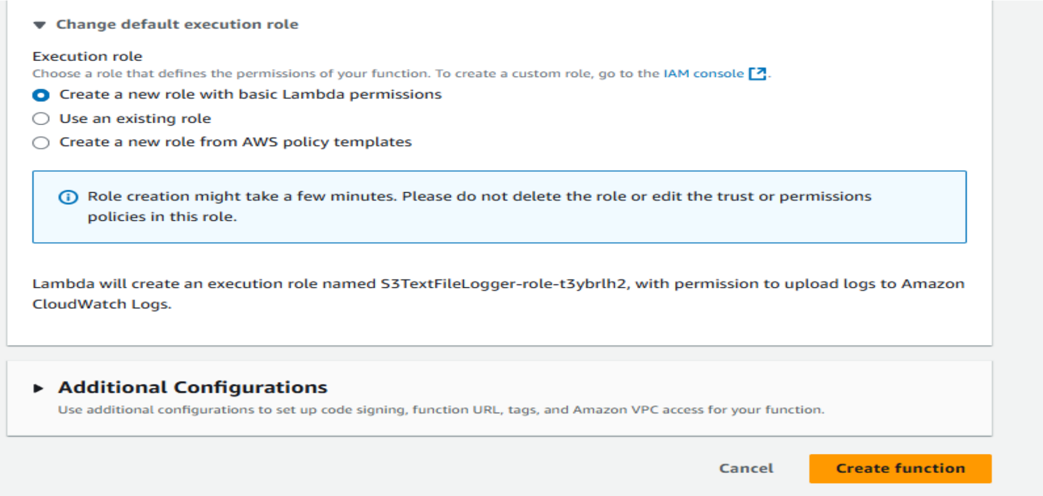
Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

● Create a new role with basic Lambda permissions
○ Use an existing role
○ Create a new role from AWS policy templates

ⓘ Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions



▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

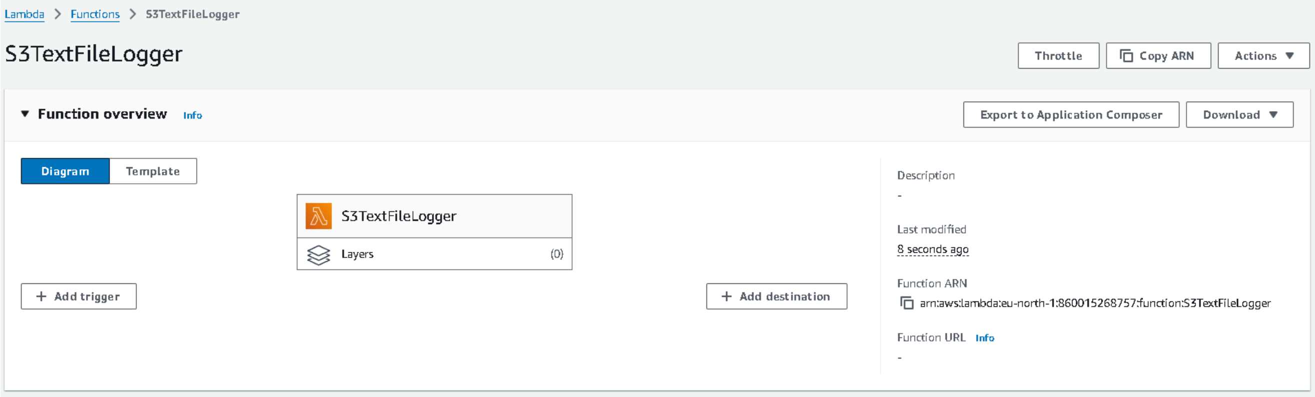
● Create a new role with basic Lambda permissions
○ Use an existing role
○ Create a new role from AWS policy templates

ⓘ Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named S3TextFileLogger-role-t3ybrlh2, with permission to upload logs to Amazon CloudWatch Logs.

▶ Additional Configurations
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

- Click **Create Function**.



S3TextFileLogger

Description
-

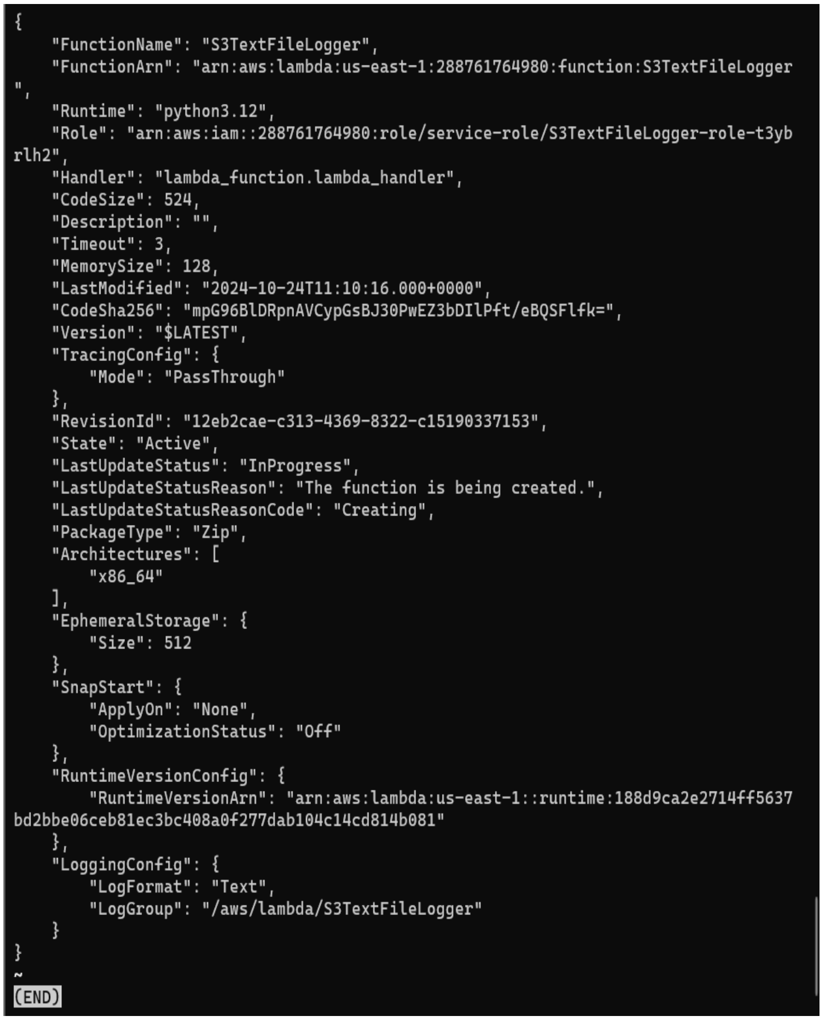
Last modified
8 seconds ago

Function ARN
arn:aws:lambda:eu-north-1:860015268757:function:S3TextFileLogger

Function URL Info
-

6.3 Upload the function code from EC2 using the AWS CLI:

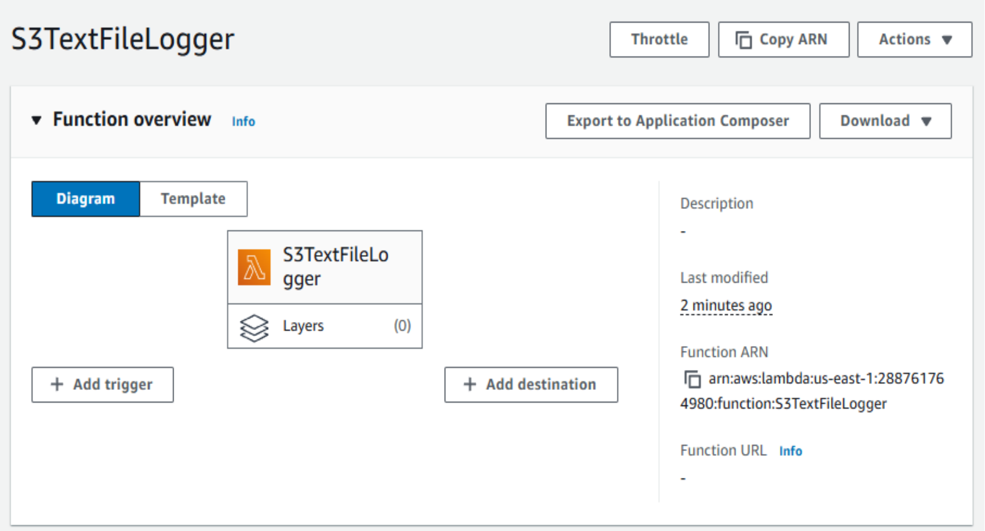
aws lambda update-function-code --function-name S3TextFileLogger --zip-file fileb://function.zip



```
{
    "FunctionName": "S3TextFileLogger",
    "FunctionArn": "arn:aws:lambda:us-east-1:288761764980:function:S3TextFileLogger
",
    "Runtime": "python3.12",
    "Role": "arn:aws:iam::288761764980:role/service-role/S3TextFileLogger-role-t3yb
rlh2",
    "Handler": "lambda_function.lambda_handler",
    "CodeSize": 524,
    "Description": "",
    "Timeout": 3,
    "MemorySize": 128,
    "LastModified": "2024-10-24T11:10:16.000+0000",
    "CodeSha256": "mpG96BlDRpnAVCypGsBJ30PwEZ3bDIlPft/eBQSFlfk=",
    "Version": "$LATEST",
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "RevisionId": "12eb2cae-c313-4369-8322-c15190337153",
    "State": "Active",
    "LastUpdateStatus": "InProgress",
    "LastUpdateStatusReason": "The function is being created.",
    "LastUpdateStatusReasonCode": "Creating",
    "PackageType": "Zip",
    "Architectures": [
        "x86_64"
    ],
    "EphemeralStorage": {
        "Size": 512
    },
    "SnapStart": {
        "ApplyOn": "None",
        "OptimizationStatus": "Off"
    },
    "RuntimeVersionConfig": {
        "RuntimeVersionArn": "arn:aws:lambda:us-east-1::runtime:188d9ca2e2714ff5637
bd2bbe06ceb81ec3bc408a0f277dab104c14cd814b081"
    },
    "LoggingConfig": {
        "LogFormat": "Text",
        "LogGroup": "/aws/lambda/S3TextFileLogger"
    }
}
~
(END)
```

7. Configure S3 as the Trigger

7.1 **In Lambda console**, go to the **Function Overview** section and click **Add Trigger**.



S3TextFileLogger

S3TextFileLogger

Description
-

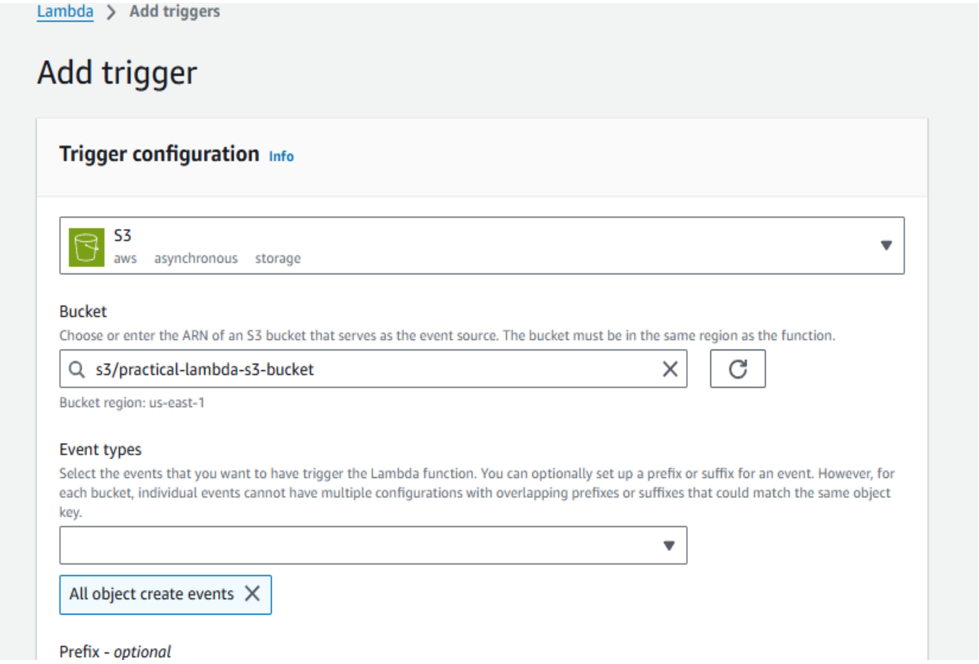
Last modified
2 minutes ago

Function ARN
arn:aws:lambda:us-east-1:288761764980:function:S3TextFileLogger

Function URL
-

7.2 Choose **S3** as the trigger:

- Select your bucket (`practical-lambda-s3-bucket`).
- **Event type**: Choose **All object create events**.



Add trigger

Trigger configuration

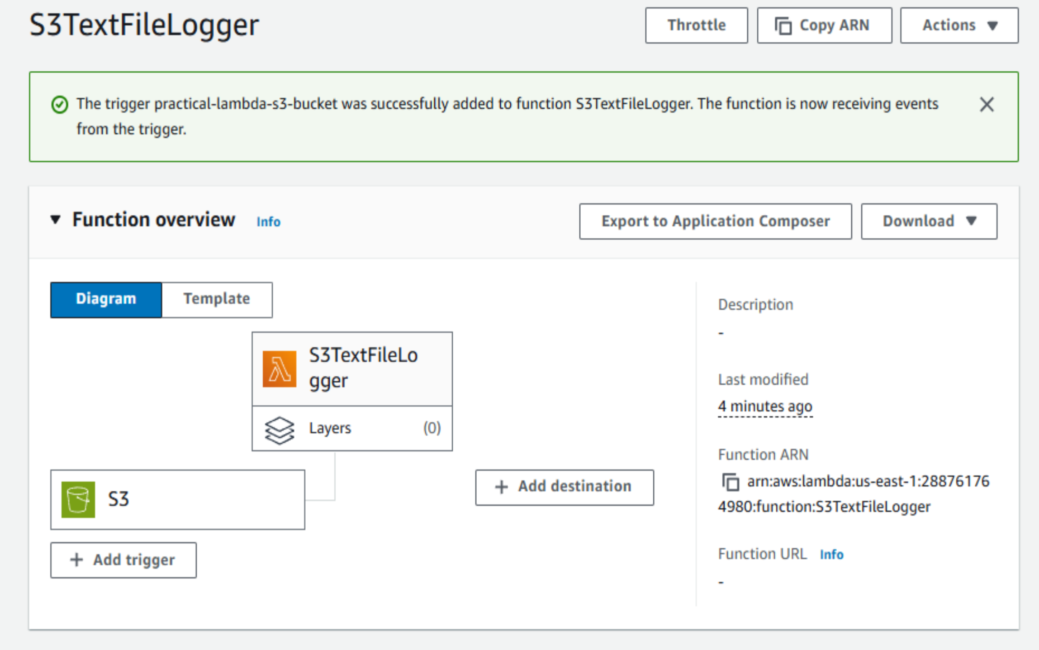
S3
aws asynchronous storage

Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.
s3/practical-lambda-s3-bucket
Bucket region: us-east-1

Event types
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
All object create events

Prefix - *optional*

7.3 Click **Add** to enable the trigger.



S3TextFileLogger

The trigger practical-lambda-s3-bucket was successfully added to function S3TextFileLogger. The function is now receiving events from the trigger.

Description
-

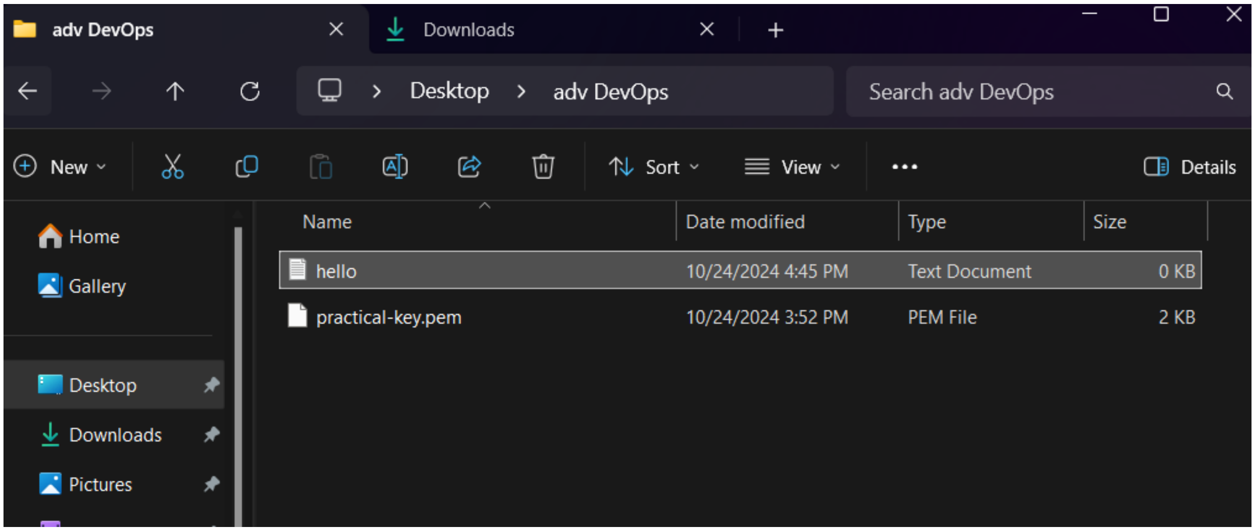
Last modified
4 minutes ago

Function ARN
arn:aws:lambda:us-east-1:288761764980:function:S3TextFileLogger

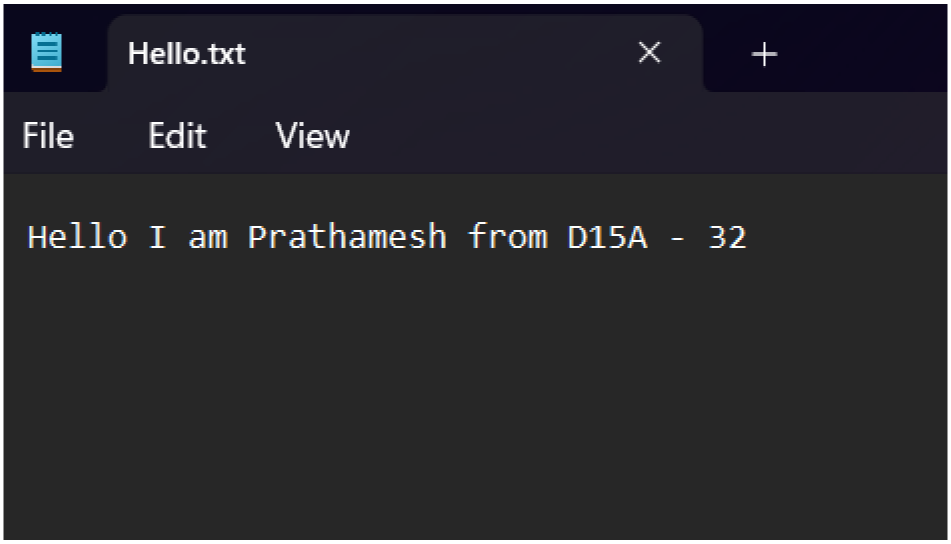
Function URL
-

8. Upload a File and Test

8.1 Create a text file in your local host with some content.



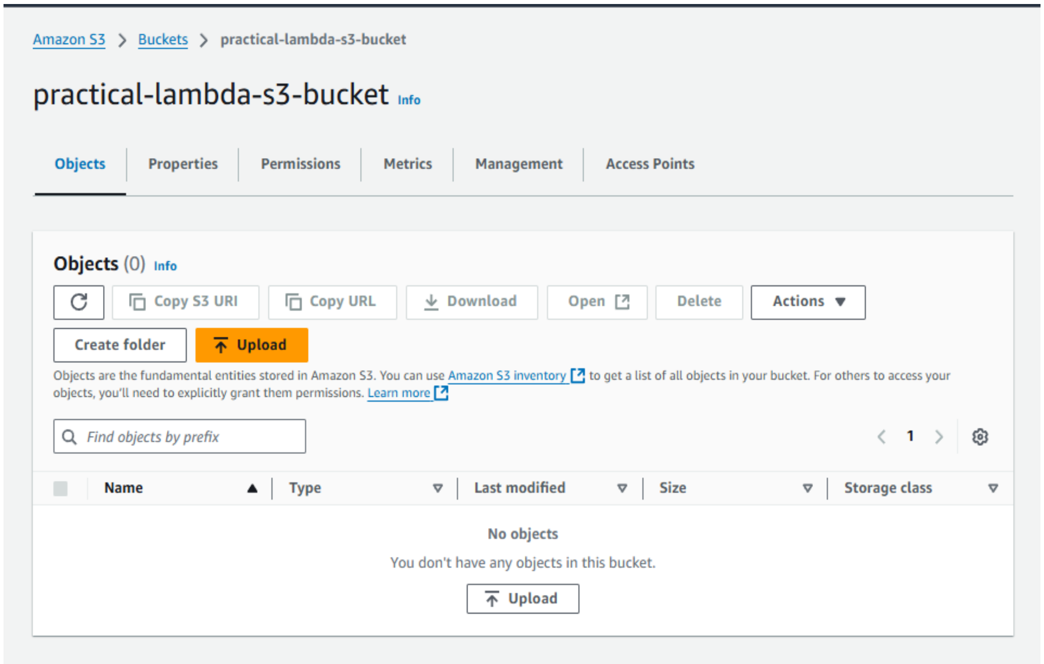
Name	Date modified	Type	Size
hello	10/24/2024 4:45 PM	Text Document	0 KB
practical-key.pem	10/24/2024 3:52 PM	PEM File	2 KB



```
Hello I am Prathamesh from D15A - 32
```

8.2 **Upload a text file** to your S3 bucket:

- Go to **S3** > your bucket > **Upload**.

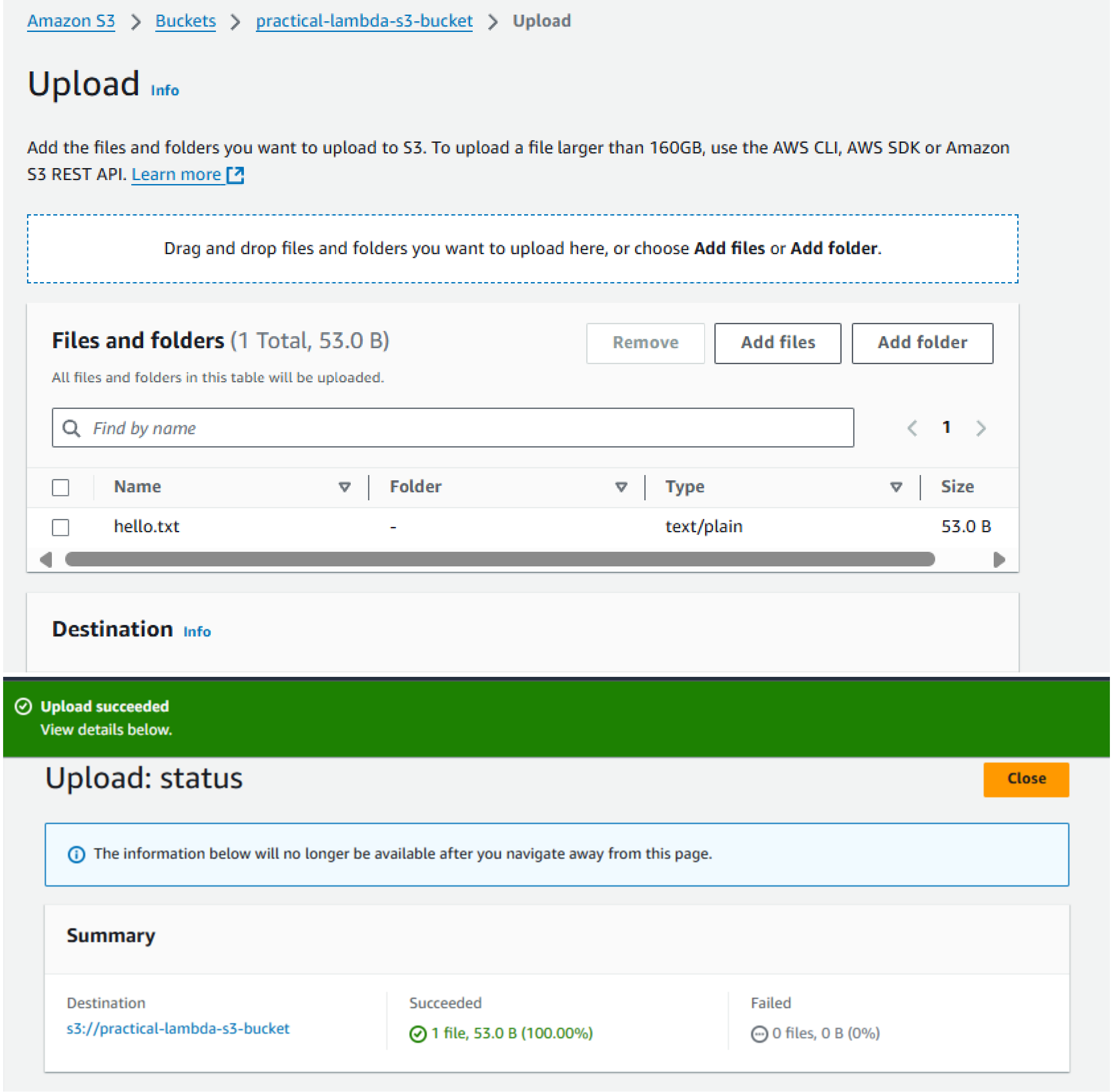


Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

No objects

You don't have any objects in this bucket.

- Upload a `.txt` file with some content (e.g., `hello.txt`)



Amazon S3 > Buckets > practical-lambda-s3-bucket > Upload

Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (1 Total, 53.0 B)

All files and folders in this table will be uploaded.

□	Name	Folder	Type	Size
□	hello.txt	-	text/plain	53.0 B

Destination Info

Upload succeeded
View details below.

Upload: status

The information below will no longer be available after you navigate away from this page.

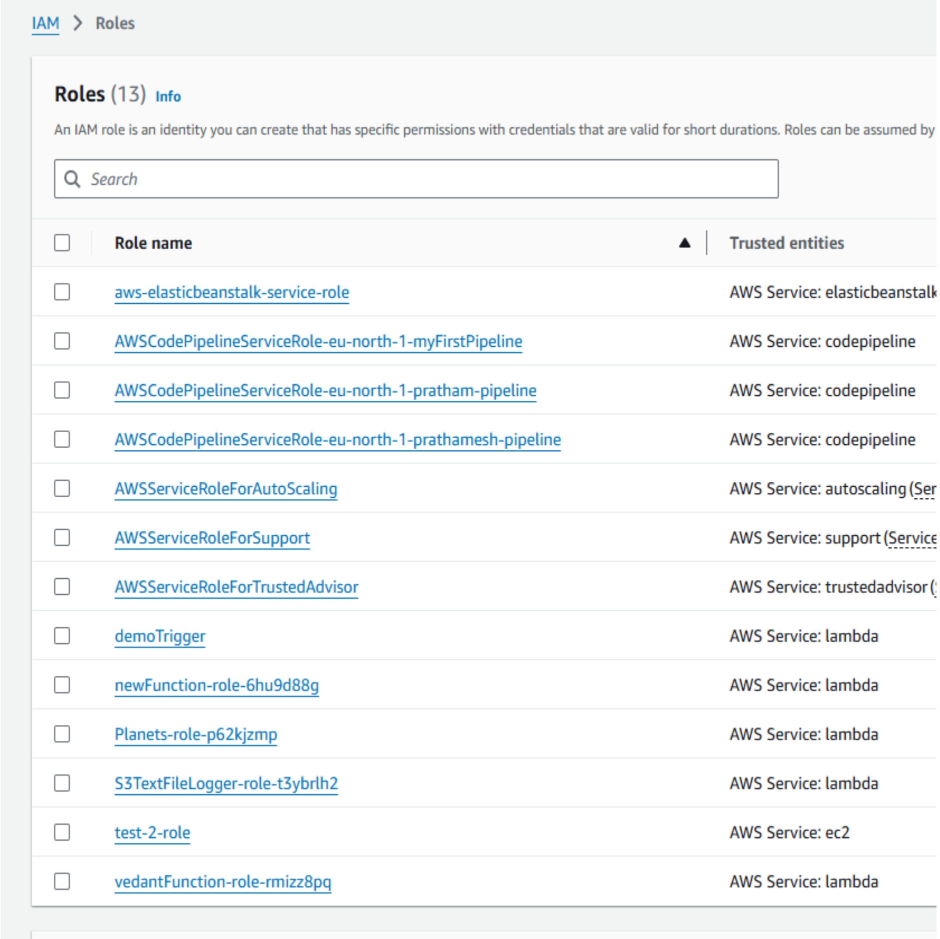
Summary

Destination	Succeeded	Failed
s3://practical-lambda-s3-bucket	1 file, 53.0 B (100.00%)	0 files, 0 B (0%)

The Lambda function will automatically run when the file is uploaded.

9. Edit the permissions of the s3 bucket to rectify the access denied problem.

9.1 click on IAM console and find S3TextFileLogger role



IAM > Roles

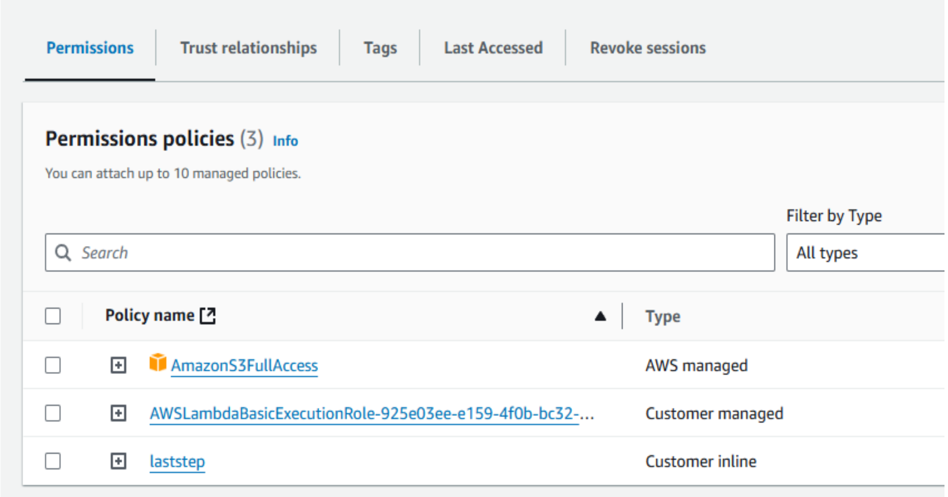
Roles (13) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by

Search

□	Role name ▲	Trusted entities
□	aws-elasticbeanstalk-service-role	AWS Service: elasticbeanstalk
□	AWSCodePipelineServiceRole-eu-north-1-myFirstPipeline	AWS Service: codepipeline
□	AWSCodePipelineServiceRole-eu-north-1-pratham-pipeline	AWS Service: codepipeline
□	AWSCodePipelineServiceRole-eu-north-1-prathamesh-pipeline	AWS Service: codepipeline
□	AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Ser
□	AWSServiceRoleForSupport	AWS Service: support (Service
□	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (
□	demoTrigger	AWS Service: lambda
□	newFunction-role-6hu9d88g	AWS Service: lambda
□	Planets-role-p62kjzmp	AWS Service: lambda
□	S3TextFileLogger-role-t3ybrlh2	AWS Service: lambda
□	test-2-role	AWS Service: ec2
□	vedantFunction-role-rmizz8pq	AWS Service: lambda

9.2 add permission of AmazonS3FullAccess



Permissions policies (3) Info

You can attach up to 10 managed policies.

Filter by Type

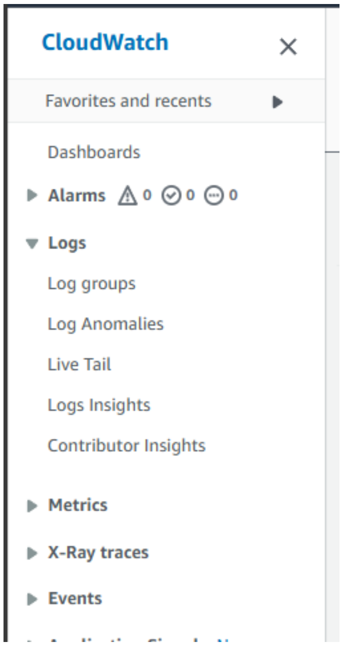
Search

All types

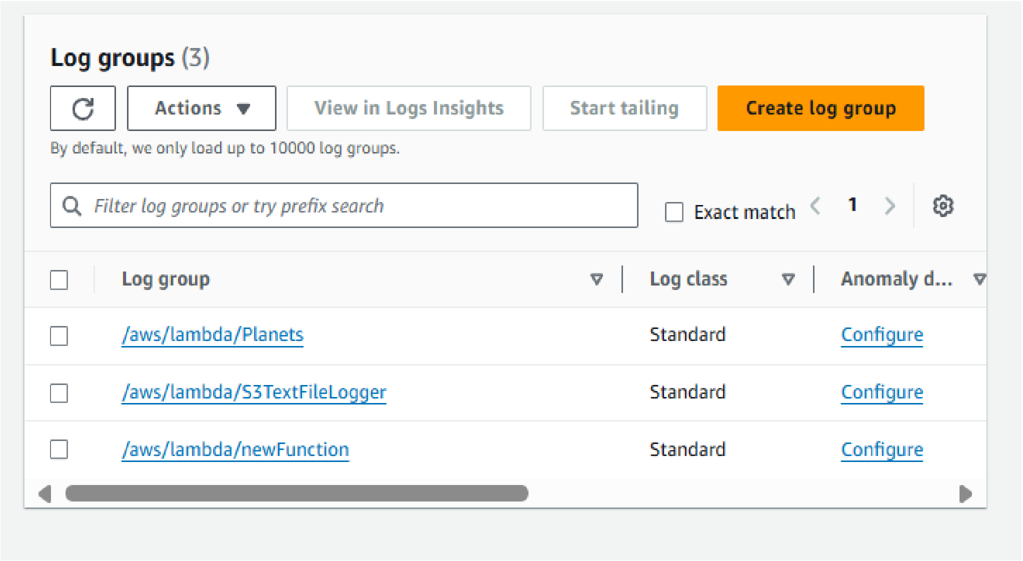
□		Policy name ▲	Type
□	⊞	AmazonS3FullAccess	AWS managed
□	⊞	AWSLambdaBasicExecutionRole-925e03ee-e159-4f0b-bc32-...	Customer managed
□	⊞	laststep	Customer inline

10. Check Logs in CloudWatch

10.1 In the AWS Console, go to **CloudWatch** > **Logs**.



10.2 Under **Log Groups**, find the log group for your Lambda function (`/aws/lambda/S3TextFileLogger`).

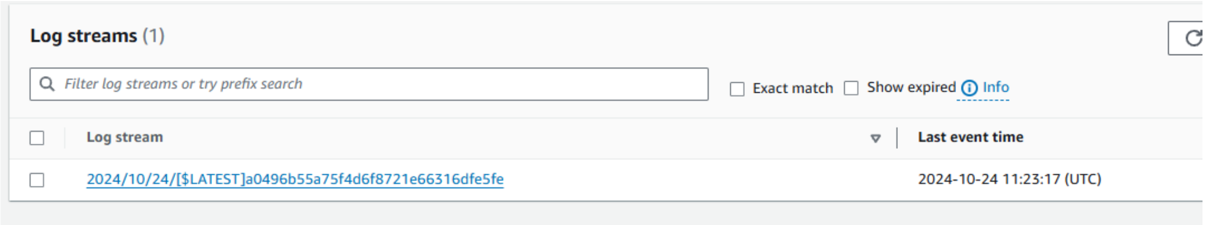


Log groups (3)

By default, we only load up to 10000 log groups.

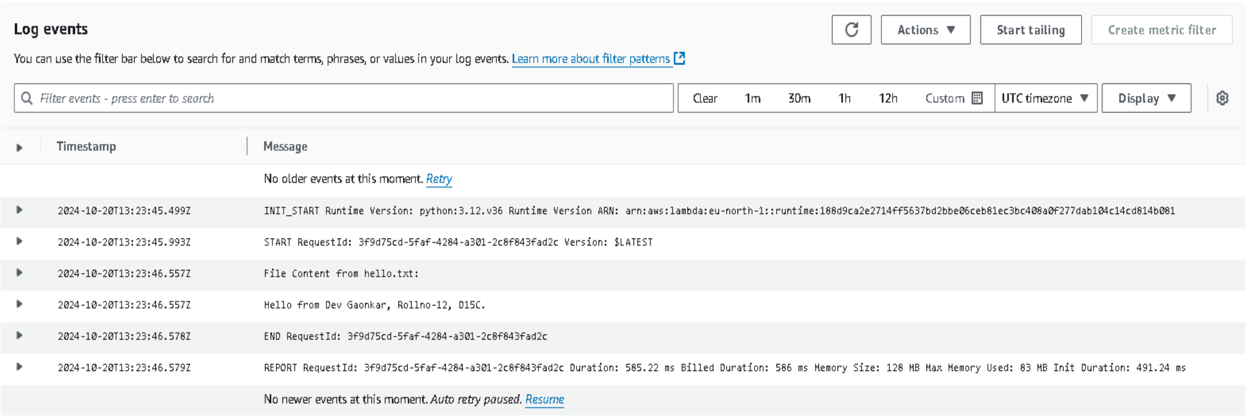
☐	Log group	Log class	Anomaly d...
☐	/aws/lambda/Planets	Standard	Configure
☐	/aws/lambda/S3TextFileLogger	Standard	Configure
☐	/aws/lambda/newFunction	Standard	Configure

10.3 Open the latest log stream to see the file content logged by the Lambda function.



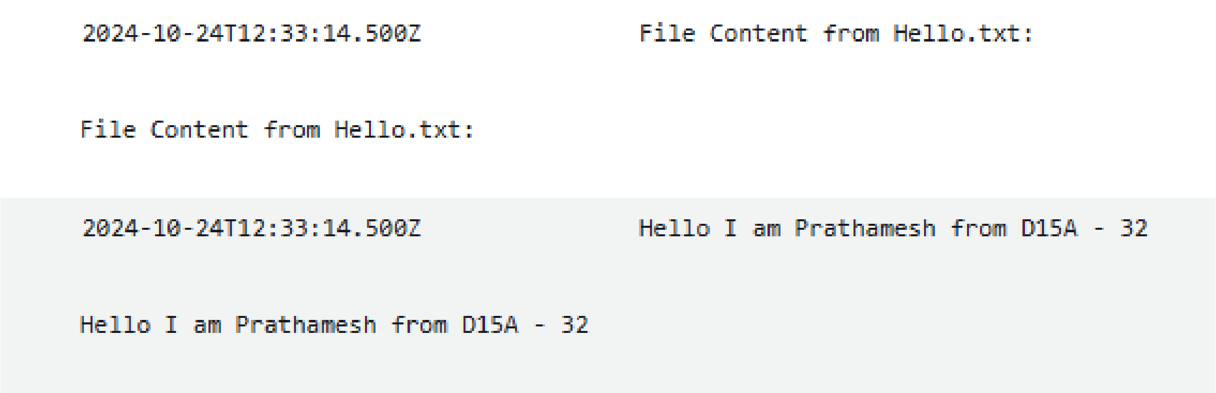
Log streams (1)
Filter log streams or try prefix search
□ Exact match □ Show expired Info

□	Log stream	Last event time
□	2024/10/24/[$LATEST]a0496b55a75f4d6f8721e66316dfe5fe	2024-10-24 11:23:17 (UTC)



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
2024-10-20T13:23:45.499Z    INIT_START Runtime Version: python:3.12.v36 Runtime Version ARN: arn:aws:lambda:eu-north-1::runtime:188d9ca2e2714ff5637bd2bbe06ceb81ec3bc408a0f277dab104c14cd814b081
2024-10-20T13:23:45.993Z    START RequestId: 3f9d75cd-5faf-4284-a301-2c8f843fad2c Version: $LATEST
2024-10-20T13:23:46.557Z    File Content from hello.txt:
2024-10-20T13:23:46.557Z    Hello from Dev Gaonkar, Rollno-12, D15C.
2024-10-20T13:23:46.578Z    END RequestId: 3f9d75cd-5faf-4284-a301-2c8f843fad2c
2024-10-20T13:23:46.579Z    REPORT RequestId: 3f9d75cd-5faf-4284-a301-2c8f843fad2c Duration: 585.22 ms Billed Duration: 586 ms Memory Size: 128 MB Max Memory Used: 83 MB Init Duration: 491.24 ms
                            No newer events at this moment. Auto retry paused. Resume
```



```
2024-10-24T12:33:14.500Z          File Content from Hello.txt:

File Content from Hello.txt:

2024-10-24T12:33:14.500Z          Hello I am Prathamesh from D15A - 32

Hello I am Prathamesh from D15A - 32
```