

Step 1:- Create the S3 Bucket and list of bucket

The screenshot shows the AWS S3 Buckets page. At the top, there are tabs for "General purpose buckets" (4) and "Directory buckets". Below the tabs, there is a search bar labeled "Find buckets by name". A table lists four buckets:

Name	AWS Region	Creation date
adish-static-site01	Europe (London) eu-west-2	September 7, 2025, 17:33:59 (UTC+05:30)
cdk-hnb659fds-assets-133562846627-ap-south-1	Asia Pacific (Mumbai) ap-south-1	June 17, 2025, 09:18:36 (UTC+05:30)
mynginxserverbucket	Asia Pacific (Mumbai) ap-south-1	August 31, 2025, 18:41:33 (UTC+05:30)
s3-bucket-cleanup-adish	Asia Pacific (Mumbai) ap-south-1	October 8, 2025, 20:12:41 (UTC+05:30)

On the right side of the page, there are three informational boxes: "Account snapshot", "External access summary - new", and "Storage Lens provides visibility into storage usage and activity trends".

Step 2 : - Create IAM Role

The screenshot shows the AWS IAM Roles page. On the left, there is a sidebar with "Identity and Access Management (IAM)" and "Access management" sections. The main area displays the details for the "Lambda_function-role-9i5yg9oh" role:

- Summary**: Creation date: June 05, 2025, 14:38 (UTC+05:30); Last activity: 34 minutes ago.
- ARN**: arn:aws:iam::133562846627:role/service-role/Lambda_function-role-9i5yg9oh
- Permissions**: Maximum session duration: 1 hour.

Below the summary, there is a section for "Permissions policies (8)". It includes a search bar, a "Filter by Type" dropdown set to "All types", and buttons for "Simulate", "Remove", and "Add permissions".

The screenshot shows the "Add permissions" page for the Lambda function role. It lists "Other permissions policies (2/1075)" that can be attached:

Policy name	Type	Description
AmazonDMSRedshiftS3Role	AWS managed	Provides access to manage S3 settings...
AmazonS3ObjectLambdaExecutionRolePolicy	AWS managed	Provides AWS Lambda functions permis...
AmazonS3OutpostsReadOnlyAccess	AWS managed	Provides read only access to Amazon S...
AmazonS3ReadOnlyAccess	AWS managed	Provides read only access to all bucket...
AmazonS3TablesLakeFormationServiceRole	AWS managed	This managed policy grants AWS Lake ...
AmazonS3TablesReadOnlyAccess	AWS managed	Provides read only access to all S3 tabl...
AWSBackupServiceRolePolicyForS3Backup	AWS managed	Policy containing permissions necessar...
AWSBackupServiceRolePolicyForS3Restore	AWS managed	Policy containing permissions necessar...

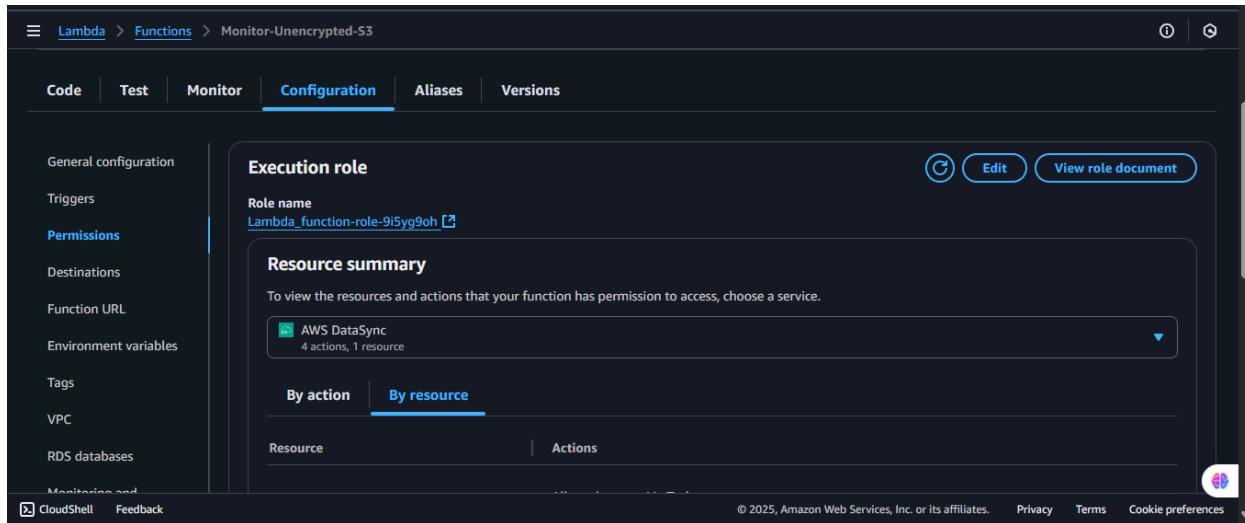
At the bottom of the page, there are links for "CloudShell" and "Feedback".

Step 3 :- Create Lambda Function

The screenshot shows the 'Create function' wizard on the AWS Lambda console. The first step, 'Basic information', is selected. It includes fields for 'Function name' (set to 'Monitor-Unencrypted-S3'), 'Runtime' (set to 'Python 3.13'), and 'Architecture' (set to 'Lambda@edge'). Other options like 'Author from scratch' and 'Container image' are also shown.

The screenshot shows the second step of the wizard, 'Permissions'. It allows users to change the default execution role. Under 'Execution role', 'Use an existing role' is selected, and 'service-role/Lambda_function-role-9i5yg9oh' is chosen. There's also a note about creating a custom role via the IAM console.

The screenshot shows the configuration page for the 'Monitor-Unencrypted-S3' function. The 'Configuration' tab is active. On the left, a sidebar lists settings like Triggers, Permissions, Destinations, Function URL, Environment variables, Tags, VPC, and RDS databases. The main area displays 'General configuration' with fields for Description (empty), Memory (128 MB), Timeout (1 min 0 sec), SnapStart (None), and Ephemeral storage (512 MB). An 'Edit' button is visible at the top right of this section.



```
import boto3

def lambda_handler(event, context):
    s3 = boto3.client('s3')
    buckets = s3.list_buckets()["Buckets"]
    unencrypted_buckets = []
    print("Listing all buckets and checking encryption:")
    for bucket in buckets:
        name = bucket["Name"]
        print(f"Checking bucket: {name}") # Print all bucket names to
    console/logs
    try:
        enc = s3.get_bucket_encryption(Bucket=name)
        # Check if encryption rules exist
        rules = enc["ServerSideEncryptionConfiguration"]["Rules"]
        if not rules:
            unencrypted_buckets.append(name)
    except s3.exceptions.ClientError as e:
        # If encryption configuration is not found, treat as
    unencrypted
        if e.response['Error']['Code'] ==
'ServerSideEncryptionConfigurationNotFoundError':
            unencrypted_buckets.append(name)
    print("Buckets without server-side encryption:", unencrypted_buckets)
    return {'unencrypted_buckets': unencrypted_buckets}
```

Lambda > Functions > Monitor-Unencrypted-S3

Code Test Monitor Configuration Aliases Versions

Code source Info

EXPLORER MONITOR-UNENCRYPTED-S3 lambda_function.py

DEPLOY Deploy (Ctrl+Shift+U) Test (Ctrl+Shift+I)

lambda_function.pyimport boto3
def lambda_handler(event, context):
 s3 = boto3.client('s3')
 buckets = s3.list_buckets()["Buckets"]
 unencrypted_buckets = []
 print("Listing all buckets and checking encryption:")
 for bucket in buckets:
 name = bucket["Name"]

PROBLEMS OUTPUT CODE REFERENCE LOG TERMINAL

Open in Visual Studio Code Upload from

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Lambda > Functions > Monitor-Unencrypted-S3

EXPLORER MONITOR-UNENCRYPTED-S3 lambda_function.py

DEPLOY Deploy (Ctrl+Shift+U) Test (Ctrl+Shift+I)

TEST EVENTS [SELECTED: MONITOR] Create new test event Private saved events monitor

lambda_function.pyimport boto3
def lambda_handler(event, context):
 s3 = boto3.client('s3')
 buckets = s3.list_buckets()["Buckets"]
 unencrypted_buckets = []
 print("Listing all buckets and checking encryption:")
 for bucket in buckets:
 name = bucket["Name"]

PROBLEMS OUTPUT CODE REFERENCE LOG TERMINAL

Status: Succeeded Test Event Name: monitor

Response:

```
{ "unencrypted_buckets": [] }
```

Function Logs:

```
START RequestId: 479b25f3-115d-4bd2-be7c-b189eb8b80de Version: $LATEST
```

Successfully updated the function Monitor-Unencrypted-S3.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Lambda > Functions > Monitor-Unencrypted-S3

EXPLORER MONITOR-UNENCRYPTED-S3 lambda_function.py

DEPLOY Deploy (Ctrl+Shift+U) Test (Ctrl+Shift+I)

TEST EVENTS [SELECTED: MONITOR] Create new test event Private saved events monitor

lambda_function.pyimport boto3
def lambda_handler(event, context):
 s3 = boto3.client('s3')
 buckets = s3.list_buckets()["Buckets"]
 unencrypted_buckets = []
 print("Listing all buckets and checking encryption:")
 for bucket in buckets:
 name = bucket["Name"]

PROBLEMS OUTPUT CODE REFERENCE LOG TERMINAL

Function Logs:

```
START RequestId: 479b25f3-115d-4bd2-be7c-b189eb8b80de Version: $LATEST
```

Listing all buckets and checking encryption:

```
Checking bucket: adish-static-site01  
Checking bucket: cdk-hnb659fds-assets-133562846627-ap-south-1  
Checking bucket: mynginxserverbucket  
Checking bucket: s3-bucket-cleanup-adish  
Buckets without server-side encryption: []
```

```
END RequestId: 479b25f3-115d-4bd2-be7c-b189eb8b80de
```

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Lambda > Functions > Monitor-Unencrypted-S3

EXPLORER

MONITOR-UNENCRYPTED-S3

lambda_function.py

DEPLOY

Deploy (Ctrl+Shift+U)

Test (Ctrl+Shift+I)

TEST EVENTS [SELECTED: MONITOR]

- + Create new test event
- Private saved events

monitor

lambda_function.py

```
1 import boto3
2
3 def lambda_handler(event, context):
4     s3 = boto3.client('s3')
5     buckets = s3.list_buckets()['Buckets']
6     unencrypted_buckets = []
7     print("Listing all buckets and checking encryption:")
8     for bucket in buckets:
9         name = bucket['Name']
```

PROBLEMS OUTPUT CODE REFERENCE LOG TERMINAL

Checking bucket: cdk-hnb659fds-assets-133562846627-ap-south-1
Checking bucket: mynginxserverbucket
Checking bucket: s3-bucket-cleanup-adish
Buckets without server-side encryption: []
END RequestId: 479b25f3-115d-4bd2-be7c-b189eb8b80de
REPORT RequestId: 479b25f3-115d-4bd2-be7c-b189eb8b80de Duration: 5079.84 ms Billed Duration: 5335 ms Memory Size: 128 MB Max Memory Used: 92 MB Init Duration: 254.81 ms
Request ID: 479b25f3-115d-4bd2-be7c-b189eb8b80de

Ln 20, Col 49 Spaces: 4 UTF-8 LF Python Lambda Layout: US

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences