**TTL-based cache refresh using AWS Lambda**:

**AWS Lambda Lab: Configuring TTL-Based Cache Refresh for AWS Storage Gateway**

**Objective**

In this lab, you will configure an **AWS Lambda function** to automate the **cache refresh** of an **AWS Storage Gateway File Share** based on a **Time To Live (TTL) value**. This ensures that your gateway updates the cached inventory of files from an **Amazon S3 bucket** at scheduled intervals.

* **Log in** to the AWS Management Console.
* Navigate to **IAM** (Identity and Access Management).
* In the left panel, select **Policies**.
* Click **Create Policy**.
* Choose **JSON** and paste your policy JSON.

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "StorageGatewayPermissions",

"Effect": "Allow",

"Action": "storagegateway:RefreshCache",

"Resource": "\*"

},

{

"Sid": "CloudWatchLogsPermissions",

"Effect": "Allow",

"Action": [

"logs:CreateLogStream",

"logs:CreateLogGroup",

"logs:PutLogEvents"

],

"Resource": "\*"

}

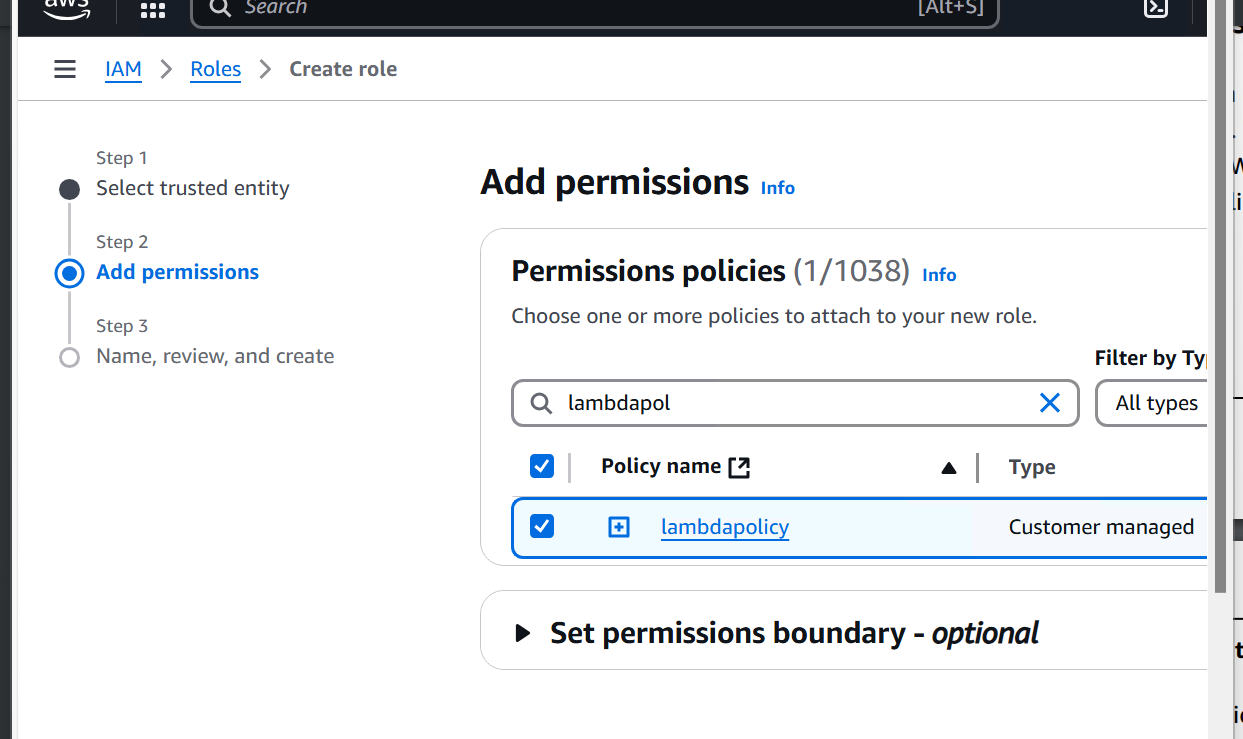
]

}

* Click **Next: Tags** (optional: add tags).
* Click **Next: Review**, provide a **Name :lambdapolicy**, and review the policy.
* Click **Create Policy**.

**Step 1: Create an IAM Role for Lambda**

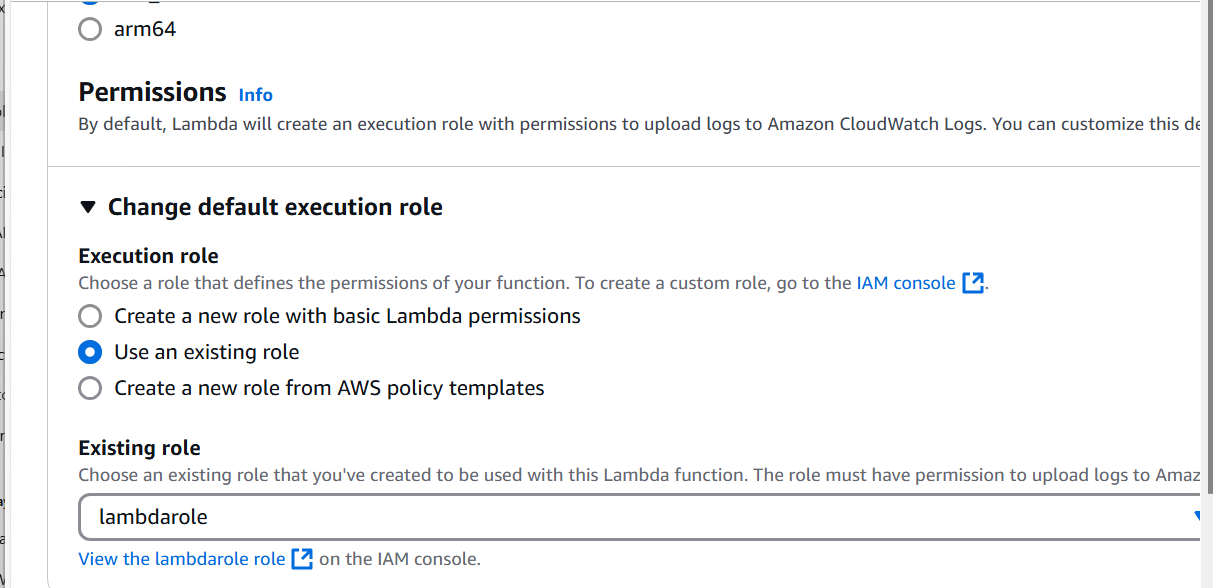
1. Navigate to the **AWS IAM Console**.
2. Create a new IAM Role and **select Lambda** as the trusted entity.



1. Attach the following policy to grant permissions:
2. Save and assign the **IAM Role**.

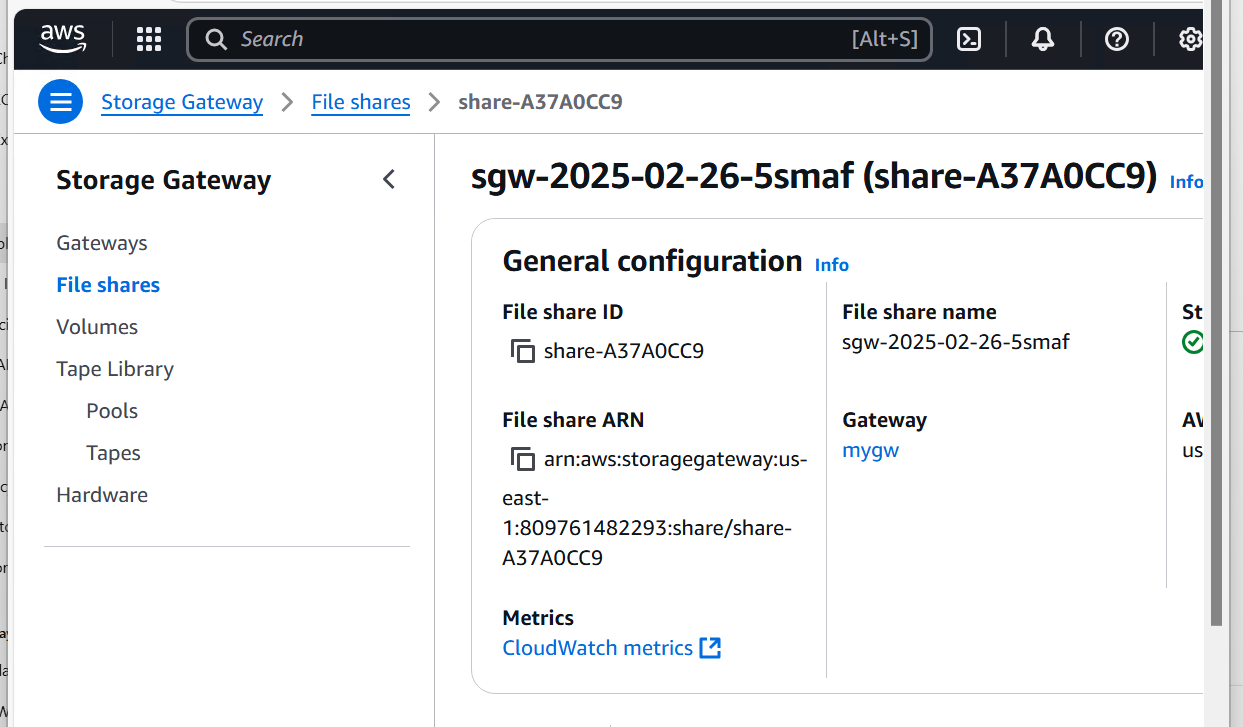
**Step 2: Create an AWS Lambda Function**

1. Navigate to the **AWS Lambda Console**.
2. Click **Create function** → **Author from scratch**.
3. Set:
   * **Function name**: RefreshStorageGatewayCache
   * **Runtime**: Python 3.9
   * **Execution role**: Choose the IAM Role created in Step 1.



1. Click **Create Function**.

Get the file share arn



**Step 3: Add the Lambda Code**

1. Replace the default code with the following:

import json

import boto3

client = boto3.client('storagegateway')

def lambda\_handler(event, context):

print("Event received:", event)

response = client.refresh\_cache(

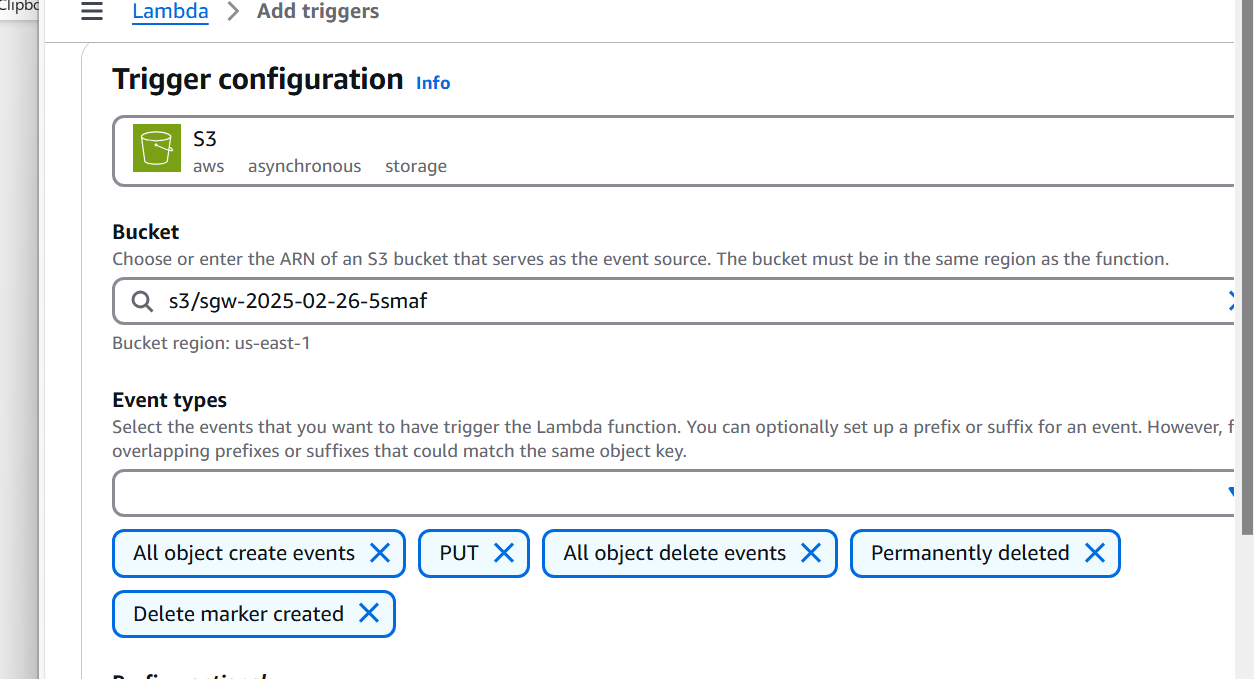
FileShareARN=**'arn:aws:storagegateway:us-east-1:809761482293:share/share-A37A0CC9'**

)

print("Cache Refresh Response:", response)

return "File Share Cache Refreshed Successfully"

1. Replace your-region, your-account-id, and share-XYZ with **your actual FileShareARN**.
2. Click **Deploy**.
3. Optional: add a trigger for Amazon S3 and select the event ObjectCreated or ObjectRemoved.



**Step 4: Test the Setup**

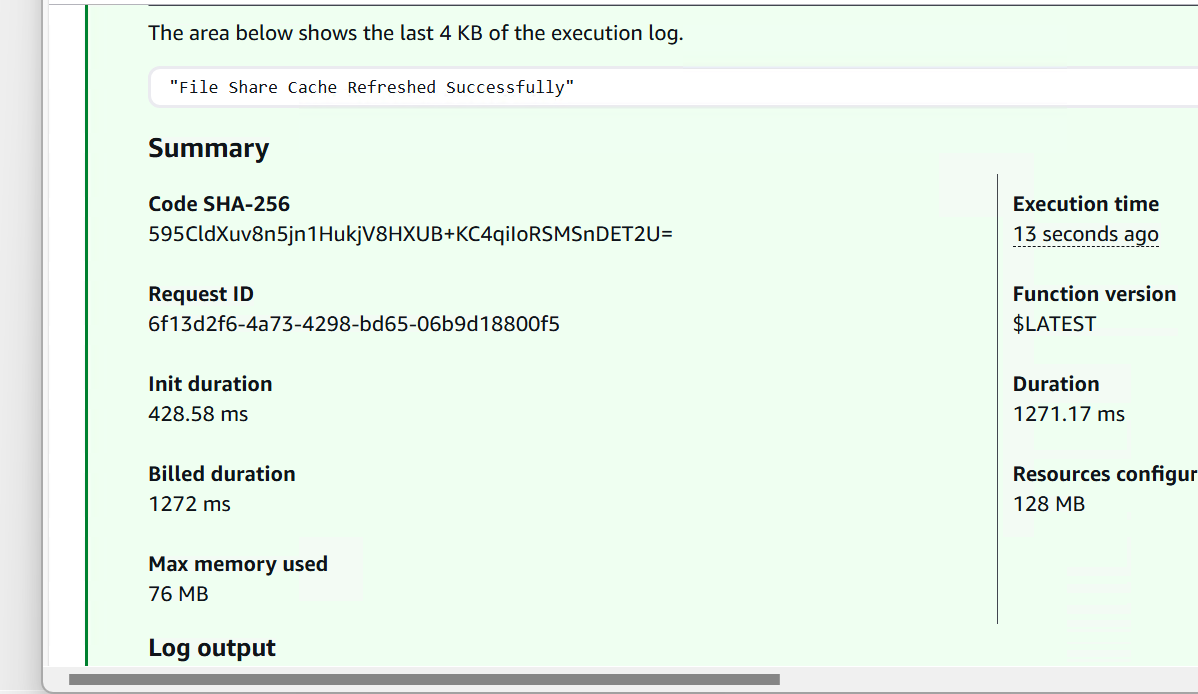
1. In the Lambda Console, click **Test**.
2. Create a test event with the following JSON:

{

"test": "manual-run"

}

1. Click **Test** and check **CloudWatch Logs** for success messages.



**Step [optional]: Create a CloudWatch Rule for Scheduled Refresh**

1. Navigate to **Amazon CloudWatch Console**->event
2. Click **Rules** → **Create Rule**.
3. Select **Event Source** → **Schedule**:
   * Choose **Fixed rate** of **30 minutes** (or set a **cron expression** for hourly refresh).
4. Select **Target** → Choose **AWS Lambda Function**.
5. Select the Lambda function **RefreshStorageGatewayCache**.
6. Click **Create Rule**.