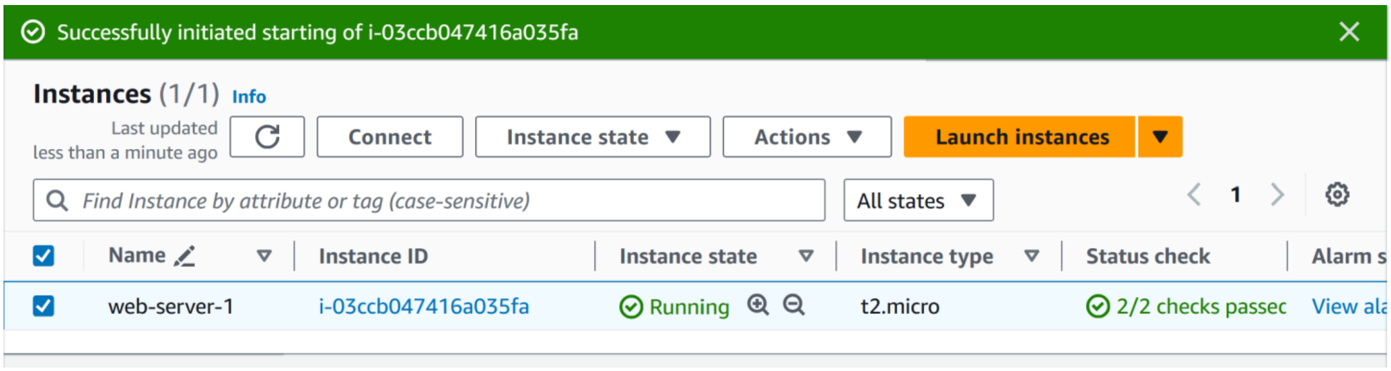
Name: Pranav Chaudhari

Task: Create a python script that automatically takes logs of webserver as backup and save it in S3 bucket and achieve following modification.

1) The ability to purge logs that are older than 7 days.
2) Use SNS service to email when the logs are purged.

Step 1-- Launch EC2 instance: Goto AWS > EC2> launch a new instance, > choose AMI (I am using amazing Linux 2023)> ensure port 80 is enabled from everywhere as we are going to host a website on it and Apache webserver works on port 80, we will also access it through browser.

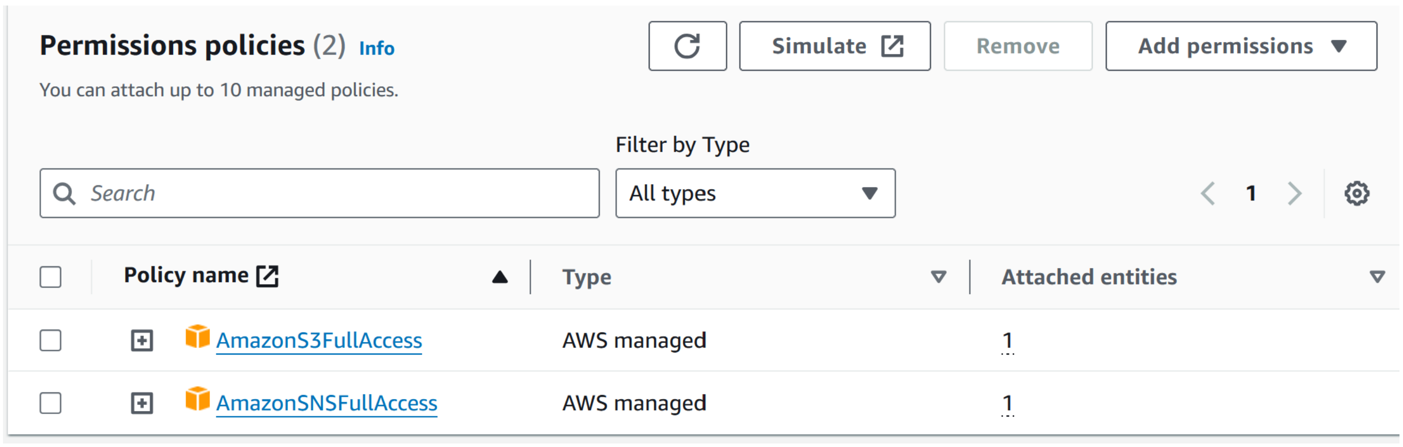


Successfully initiated starting of i-03ccb047416a035fa

Instances (1/1)

Name	Instance ID	Instance state	Instance type	Status check	Alarm s
web-server-1	i-03ccb047416a035fa	Running	t2.micro	2/2 checks passec	View ala

Step 2-- Add SNS and S3 service in a Role and attach it to EC2: As we will use S3 buckets to store logs and SNS service to send email notification we need to give proper policies on a role and attach that role to EC2.



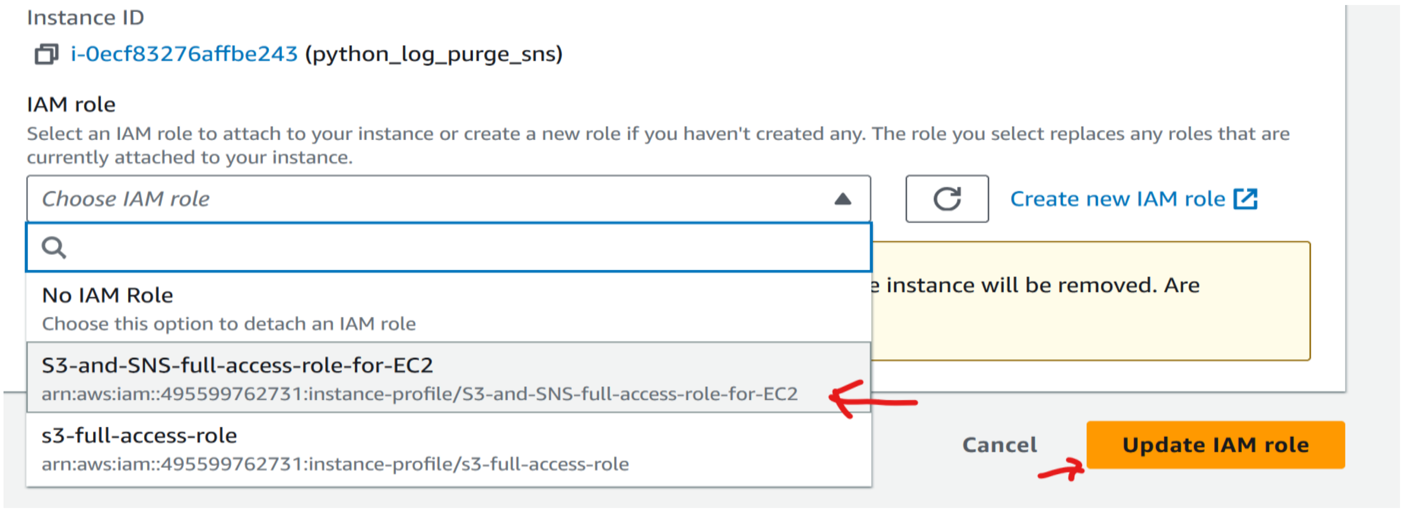
Permissions policies (2)

You can attach up to 10 managed policies.

Policy name	Type	Attached entities
AmazonS3FullAccess	AWS managed	1
AmazonSNSFullAccess	AWS managed	1

Attach this role on EC2 instance.

EC2 > Action > Security > Modify IAM role.



Instance ID

i-0ecf83276affbe243 (python_log_purge_sns)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

Choose IAM role

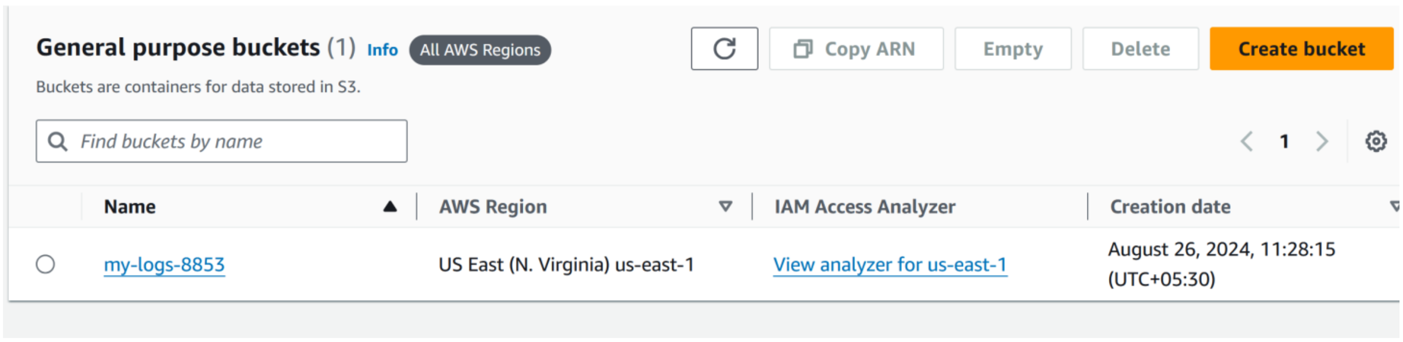
No IAM Role
Choose this option to detach an IAM role

S3-and-SNS-full-access-role-for-EC2
arn:aws:iam::495599762731:instance-profile/S3-and-SNS-full-access-role-for-EC2

s3-full-access-role
arn:aws:iam::495599762731:instance-profile/s3-full-access-role

After this, reboot the EC2 instance to take the affect. EC2 > Instance State > Reboot.

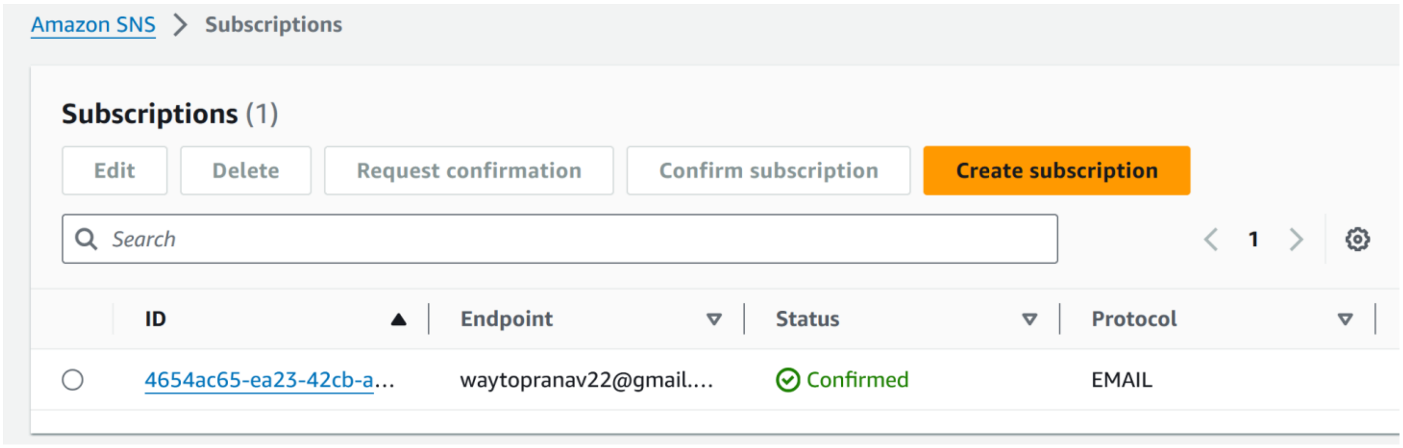
Step 3 -- Create S3 bucket: As it is going to be one time thingy, we are going to need to create a S3 bucket manually as later we are going to use it to store the logs.


General purpose buckets (1)
Info
All AWS Regions
Copy ARN
Empty
Delete
Create bucket
Buckets are containers for data stored in S3.
Find buckets by name
1

Name	AWS Region	IAM Access Analyzer	Creation date
my-logs-8853	US East (N. Virginia) us-east-1	View analyzer for us-east-1	August 26, 2024, 11:28:15 (UTC+05:30)

AWS > S3 > create bucket name.

Step 4 -- Create SNS Topic and subscription and verify it: We need to create SNS topic in order to use this as a means of sending email when our script performs the log deletion.



ID	Endpoint	Status	Protocol
4654ac65-ea23-42cb-a…	waytopranav22@gmail.…	Confirmed	EMAIL

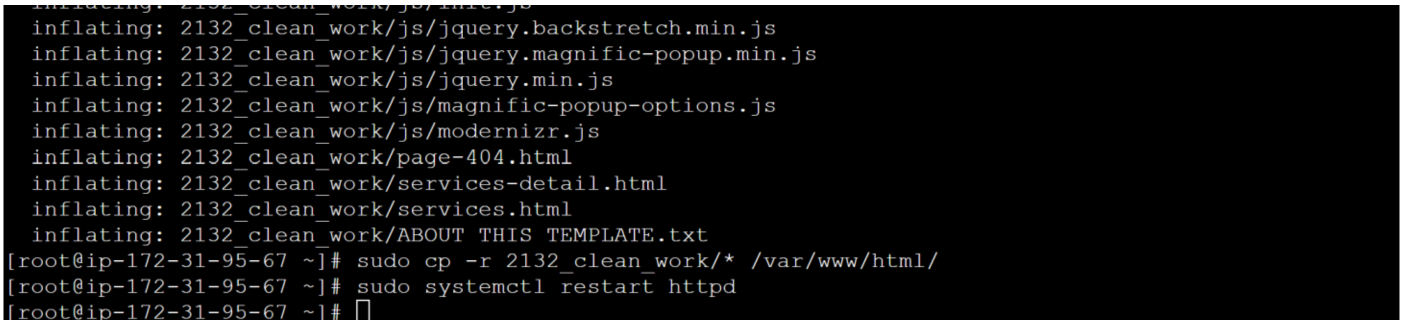
SNS > Topic > Create topic > Standard > name it >

Subscription > Topic ARN (Select the ARN of your SNS Topic) > Protocol (Email) enter email > Create >

Step 5 -- Install necessary packages and content on EC2: To simulate the website hosting and the logs are genuinely occurring from it, we are going to host a website and use the logs to put on s3 bucket later. For this operation we are going to need necessary packages and files. We will need python, boto3 module, crony, httpd, wget and unzip. Since it is Amazon Linux, python comes pre-installed so skipping the installation of Python.

Httpd: using for webserver Apache.
Wget: using to download website content.
Cronie: using to manage the cron jobs to schedule tasks.
Pip: it is a package manager for python to install the modules.
Boto3: it is a library an official AWS SDK for Python, to interact with AWS resources.
Tooplate.com: provides free html templates, using these templates to simulate our website.

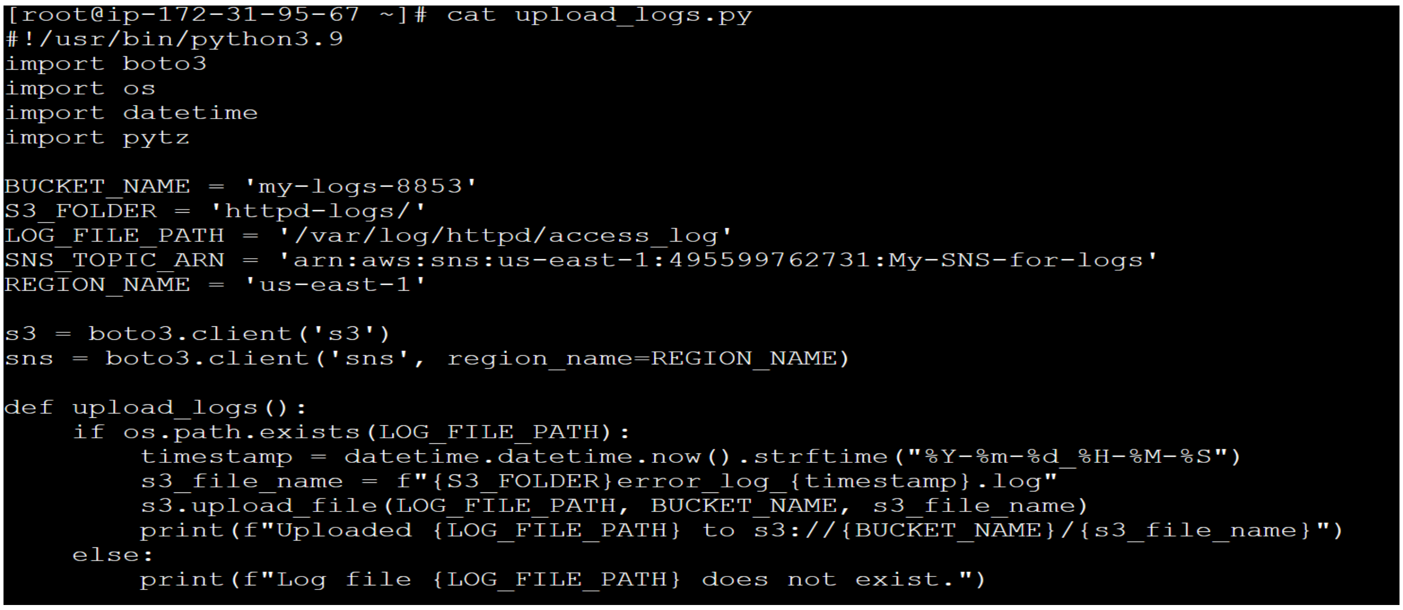
```
sudo yum install httpd wget unzip cronie -y
sudo systemctl start httpd
sudo systemctl enable httpd
sudo systemctl start crond
sudo systemctl enable crond
sudo yum install python3-pip -y
pip install boto3
#download theme
sudo wget https://www.tooplate.com/zip-templates/2132_clean_work.zip
#unzip the file
sudo unzip 2132_clean_work.zip
sudo cp -r 2132_clean_work/* /var/www/html/
sudo systemctl restart httpd
```



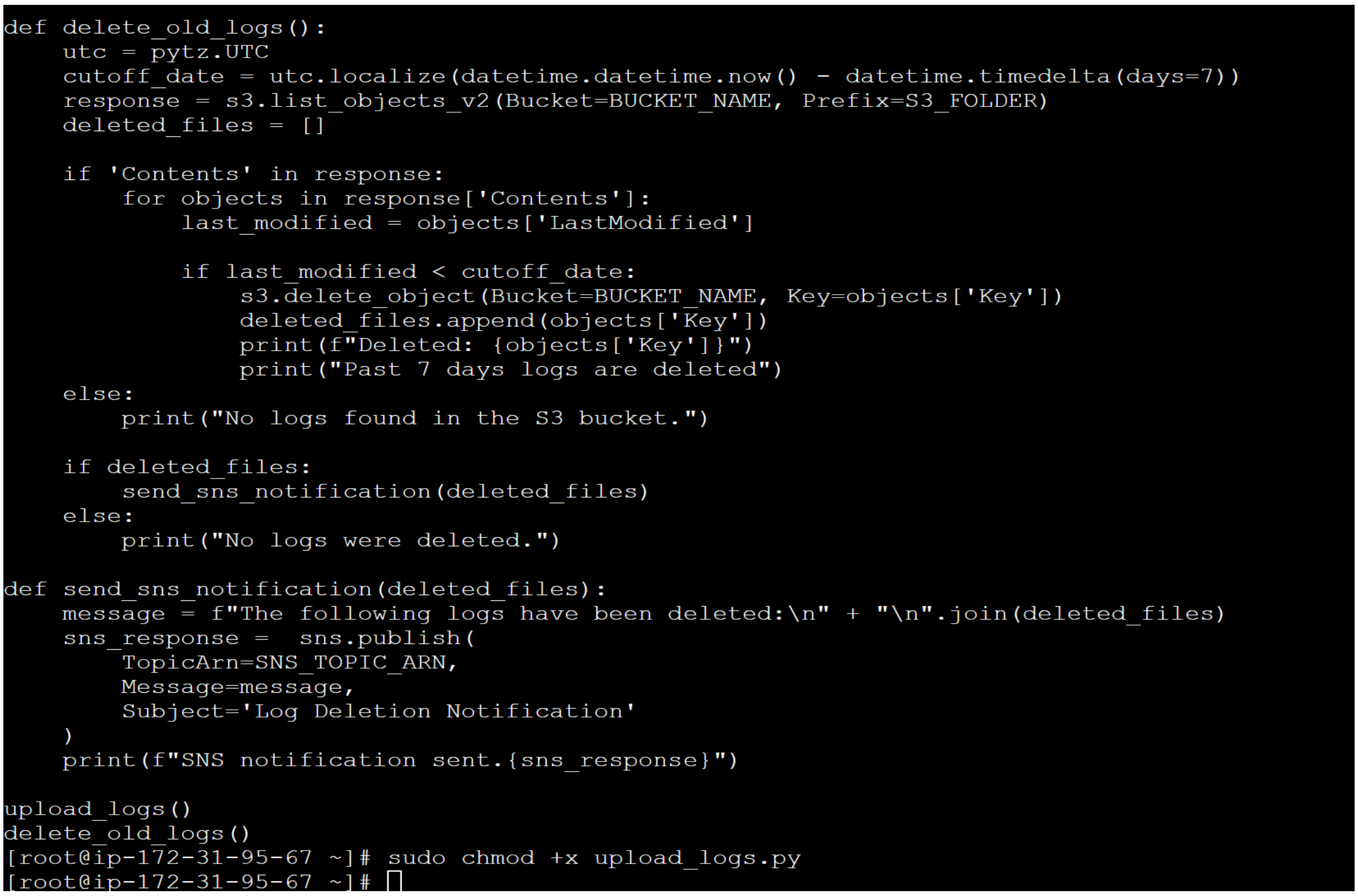
```
  inflating: 2132_clean_work/js/jquery.backstretch.min.js
  inflating: 2132_clean_work/js/jquery.magnific-popup.min.js
  inflating: 2132_clean_work/js/jquery.min.js
  inflating: 2132_clean_work/js/magnific-popup-options.js
  inflating: 2132_clean_work/js/modernizr.js
  inflating: 2132_clean_work/page-404.html
  inflating: 2132_clean_work/services-detail.html
  inflating: 2132_clean_work/services.html
  inflating: 2132_clean_work/ABOUT_THIS_TEMPLATE.txt
[root@ip-172-31-95-67 ~]# sudo cp -r 2132_clean_work/* /var/www/html/
[root@ip-172-31-95-67 ~]# sudo systemctl restart httpd
[root@ip-172-31-95-67 ~]# 
```

Step 6 -- Create python script: Create the script in python and give executable permission to it.

Full script here: https://github.com/techdecipher/cloudformation-templates/blob/main/upload_logs_with_SNS_S3.py



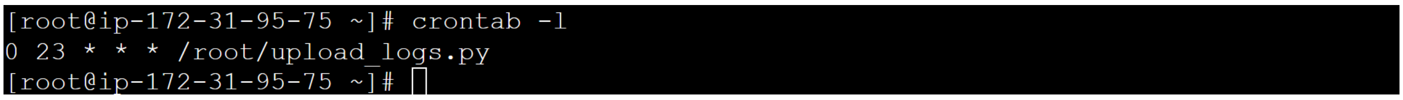
```
[root@ip-172-31-95-67 ~]# cat upload_logs.py
#!/usr/bin/python3.9
import boto3
import os
import datetime
import pytz

BUCKET_NAME = 'my-logs-8853'
S3_FOLDER = 'httpd-logs/'
LOG_FILE_PATH = '/var/log/httpd/access_log'
SNS_TOPIC_ARN = 'arn:aws:sns:us-east-1:495599762731:My-SNS-for-logs'
REGION_NAME = 'us-east-1'

s3 = boto3.client('s3')
sns = boto3.client('sns', region_name=REGION_NAME)

def upload_logs():
    if os.path.exists(LOG_FILE_PATH):
        timestamp = datetime.datetime.now().strftime("%Y-%m-%d_%H-%M-%S")
        s3_file_name = f"{S3_FOLDER}error_log_{timestamp}.log"
        s3.upload_file(LOG_FILE_PATH, BUCKET_NAME, s3_file_name)
        print(f"Uploaded {LOG_FILE_PATH} to s3://{BUCKET_NAME}/{s3_file_name}")
    else:
        print(f"Log file {LOG_FILE_PATH} does not exist.")
```



```
def delete_old_logs():
    utc = pytz.UTC
    cutoff_date = utc.localize(datetime.datetime.now() - datetime.timedelta(days=7))
    response = s3.list_objects_v2(Bucket=BUCKET_NAME, Prefix=S3_FOLDER)
    deleted_files = []

    if 'Contents' in response:
        for objects in response['Contents']:
            last_modified = objects['LastModified']

            if last_modified < cutoff_date:
                s3.delete_object(Bucket=BUCKET_NAME, Key=objects['Key'])
                deleted_files.append(objects['Key'])
                print(f"Deleted: {objects['Key']}")
                print("Past 7 days logs are deleted")
    else:
        print("No logs found in the S3 bucket.")

    if deleted_files:
        send_sns_notification(deleted_files)
    else:
        print("No logs were deleted.")

def send_sns_notification(deleted_files):
    message = f"The following logs have been deleted:\n" + "\n".join(deleted_files)
    sns_response =  sns.publish(
        TopicArn=SNS_TOPIC_ARN,
        Message=message,
        Subject='Log Deletion Notification'
    )
    print(f"SNS notification sent.{sns_response}")

upload_logs()
delete_old_logs()
[root@ip-172-31-95-67 ~]# sudo chmod +x upload_logs.py
[root@ip-172-31-95-67 ~]# 
```

<u>Step 7 -- setup CronJob</u>: Its time we put the file in cron jobs so it runs on its own at certain time. The file will execute every day at 11PM.

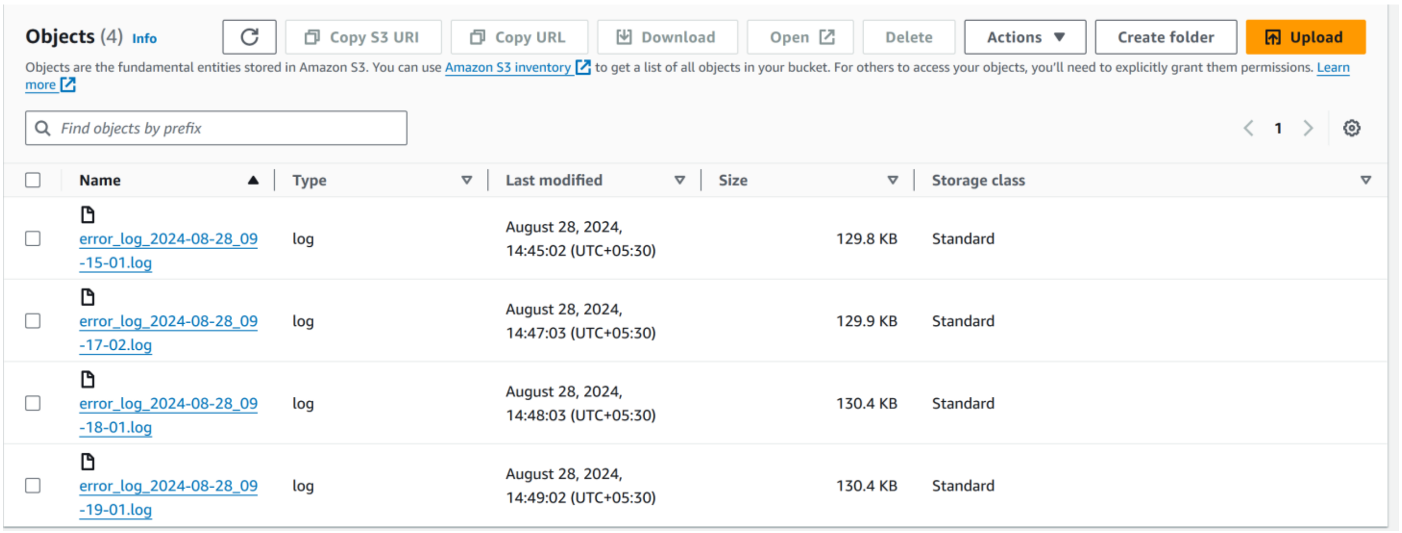
```
Crontab -e
```



```
[root@ip-172-31-95-75 ~]# crontab -l
0 23 * * * /root/upload_logs.py
[root@ip-172-31-95-75 ~]# 
```

Output:

S3 bucket has now list of logs that are save in here automatically by our script.

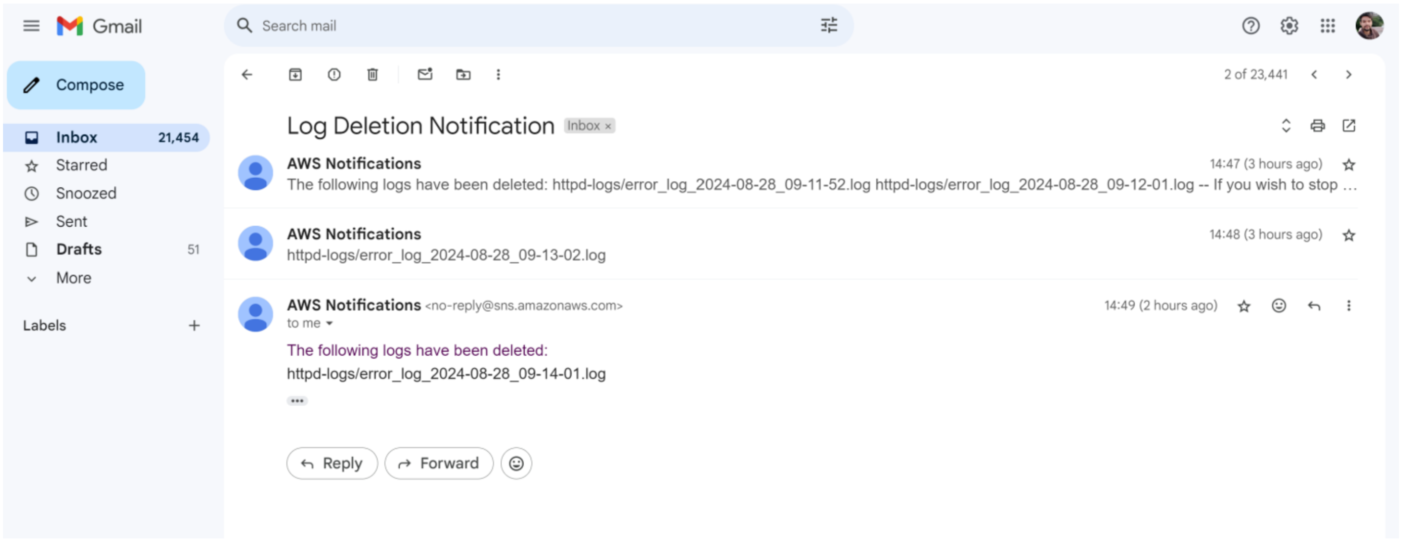


Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
error_log_2024-08-28_09-15-01.log	log	August 28, 2024, 14:45:02 (UTC+05:30)	129.8 KB	Standard
error_log_2024-08-28_09-17-02.log	log	August 28, 2024, 14:47:03 (UTC+05:30)	129.9 KB	Standard
error_log_2024-08-28_09-18-01.log	log	August 28, 2024, 14:48:03 (UTC+05:30)	130.4 KB	Standard
error_log_2024-08-28_09-19-01.log	log	August 28, 2024, 14:49:02 (UTC+05:30)	130.4 KB	Standard

This way we don't have to manually login to the app server or VM to review the logs, one can access s3 using AWS cli or manually and review the logs and see if everything is fine.

<u>SNS Email notification:</u> For testing, I reduced the time of purge log from 7 days to 4 minutes, and setup cronjob to run every minute, started receiving the emails as the script is purging the logs older than 4 minutes, below is the email for reference and proof.



Log Deletion Notification

AWS Notifications 14:47 (3 hours ago)
The following logs have been deleted: httpd-logs/error_log_2024-08-28_09-11-52.log httpd-logs/error_log_2024-08-28_09-12-01.log -- If you wish to stop …

AWS Notifications 14:48 (3 hours ago)
httpd-logs/error_log_2024-08-28_09-13-02.log

AWS Notifications <no-reply@sns.amazonaws.com> 14:49 (2 hours ago)
to me

The following logs have been deleted:
httpd-logs/error_log_2024-08-28_09-14-01.log