

# Experiment 5

## Automation and Optimization with Amazon S3

Name : **Rushaan Gandhi**

Reg No : **RA2011028010105**

**Aim** : Automate Files backup to aws S3 bucket on Linux machine.

### Procedure :

Steps:

1. Create a S3 bucket.
2. Create a EC2 instance.
3. Give EC2 instance Role to access S3.

The screenshot displays the AWS IAM console interface. On the left, the 'Identity and Access Management (IAM)' sidebar is visible, with 'Roles' selected under the 'Access management' section. The main panel shows a notification banner at the top indicating that the role 'aws-ec2-s3-access' has been successfully created. Below this, the 'Roles' page is displayed, showing a list of roles. The role 'aws-ec2-s3-access' is highlighted, and its details are shown, including its trusted entities (AWS Service: ec2). The bottom of the console features a 'Roles Anywhere' section with a 'Manage' button.

(or you may also grant access to your local linux machine using aws configure cmd and entering your IAM user credentials over there)

4. Connect to your EC2 instance CLI.
5. Type “sudo su” to give access root directory.
6. Create a directory “backup”. Type: mkdir backup
7. Go inside the “backup” directory.
8. Make some test files.

Type : touch a

```
The user-provided path /root/backup does not exist.
[root@ip-172-31-0-253 backup]# aws s3 sync /backup s3://automate-uploadd

The user-provided path /backup does not exist.
[root@ip-172-31-0-253 backup]# aws s3 /backup s3://automate-uploadd
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use. For more information, see the AWS CLI version
2 installation instructions at: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html

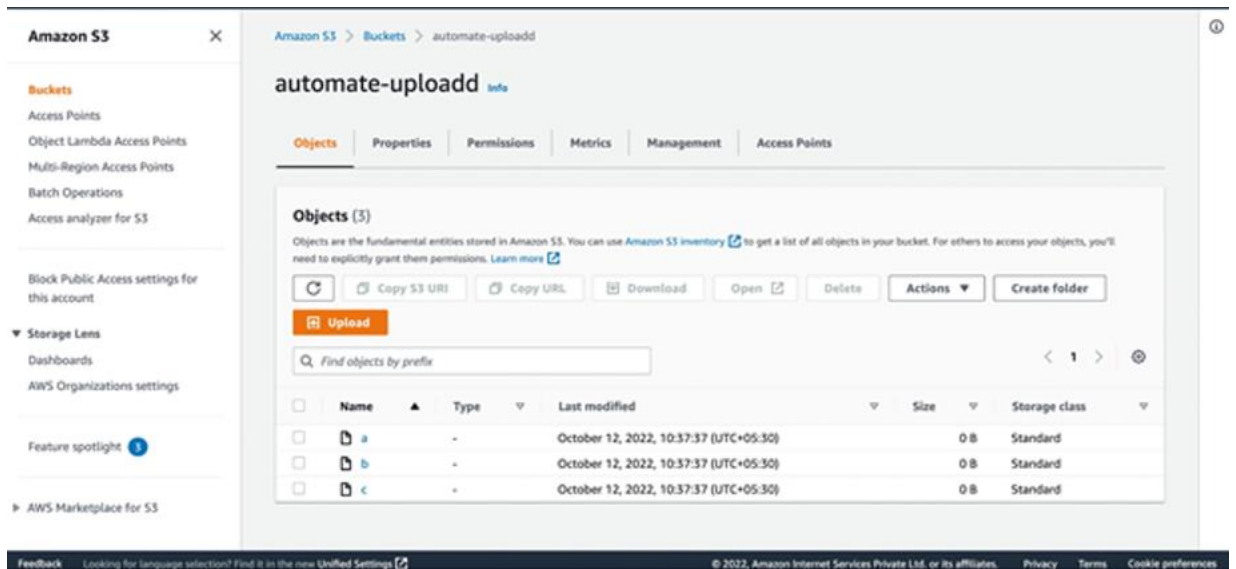
usage: aws [options] <command> [<subcommand> [<subcommand> ...] [parameters]]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: argument subcommand: Invalid choice, valid choices are:

ls                                     website
cp                                     mv
rm                                     sync
mb                                     rb
presign

[root@ip-172-31-0-253 backup]# pwd
/home/ec2-user/backup
[root@ip-172-31-0-253 backup]# aws s3 sync /home/ec2-user/backup s3://automate-uploadd
upload: ./a to s3://automate-uploadd/a
upload: ./c to s3://automate-uploadd/c
upload: ./b to s3://automate-uploadd/b
[root@ip-172-31-0-253 backup]#
```

9. List Them By Cmd–ls



Now to sync these files of backup directory on the S3 bucket. Cmd : `aws s3 sync localfilepath s3://bucketname`

11. Now, we are going to create a cron job in order to automate this process. Cmd : `crontab -e`

Enter the cmd : cron code `aws s3 sync /directory s3://bucketname`

For e.g. : cron code for 1 min is `* * * * *`

(you may use [crontab.guru](https://crontab.guru) to create your own job expression) URL : <https://crontab.guru/>

```

usage: aws [options] <command> [<subcommand> [<subcommand> ...] [parameters]]
To see help text, you can run:

aws help
aws <command> help
aws <command> <subcommand> help
aws: error: argument subcommand: Invalid choice, valid choices are:

ls                               | website
cp                               | mv
rm                               | sync
mb                               | rb
presign

[root@ip-172-31-0-253 backup]# pwd
/home/ec2-user/backup
[root@ip-172-31-0-253 backup]# aws s3 sync /home/ec2-user/backup s3://automate-uploadd
upload: ./a to s3://automate-uploadd/a
upload: ./c to s3://automate-uploadd/c
upload: ./b to s3://automate-uploadd/b
[root@ip-172-31-0-253 backup]# crontab -e
no crontab for root - using an empty one

[1]+  Stopped                  crontab -e
[root@ip-172-31-0-253 backup]# cron code aws s3 sync /home/ec2-user/backup s3://automate-uploadd
bash: cron: command not found
[root@ip-172-31-0-253 backup]# cron code aws s3 sync /backup s3://automate-uploadd
bash: cron: command not found
[root@ip-172-31-0-253 backup]#

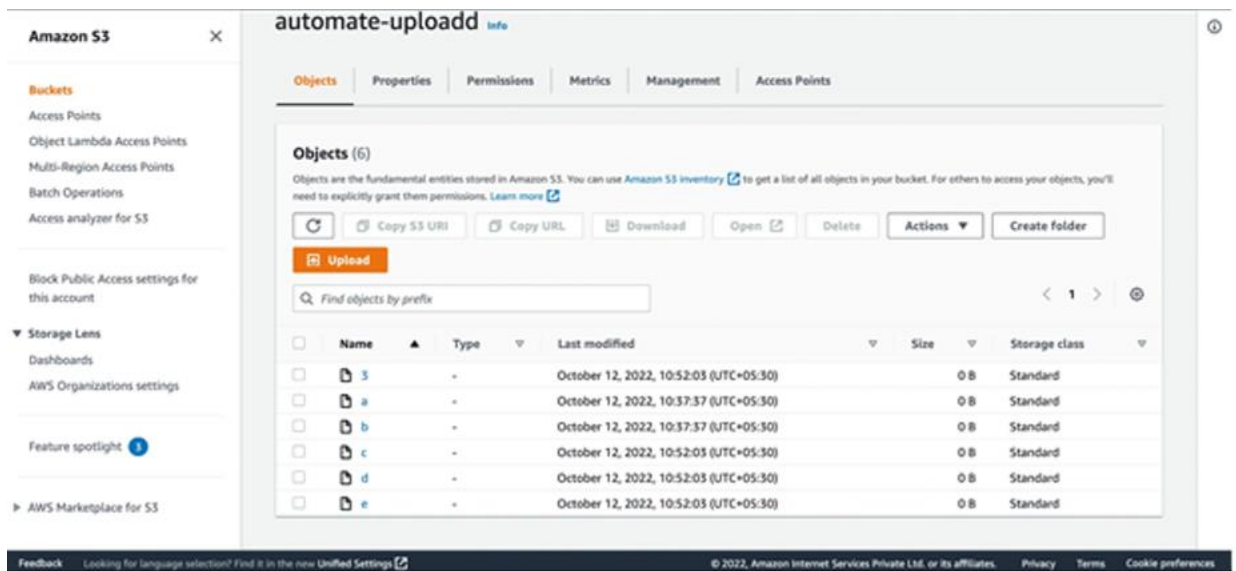
```

## Restart the Crond service

Run “systemctl restart/stop/start cornd.service” to restart/stop/start your cron jobs respectively.

13. Now, we are going to create some test files to check if they are uploaded every minute or not.

14. File d and file e have been updated.



**Result:** We have successfully automated our local files/directory backup on Amazon S3 buckets using crontab.