AUTOMATED FILE SYNCHRONIZATION WITH AWS S3

By P.Lokesh

OVERVIEW :

In this Project we"ll explore the power of automation using Shell Scripting in conjunction with AWS Command Line Interface(CLI).The object is to create a robust solution for regularly synchronising local file with an Amazon S3 bucket.To make this process even more hands-free we"ll use cron jobs to schedule these operations at specific intervals.

TECHNOLOGIES USED:

AWS CLI : The AWS Command Line Interface (AWS CLI) is an open source tool from Amazon Web Services (AWS). You can use it to interact with AWS services using commands in your command line shell.

Shell Scripting: We'll be using bash scripting to create a script that automates the file synchronisation process.This script will be responsible for interacting with AWS CLI to sync local files with an S3 Bucket.

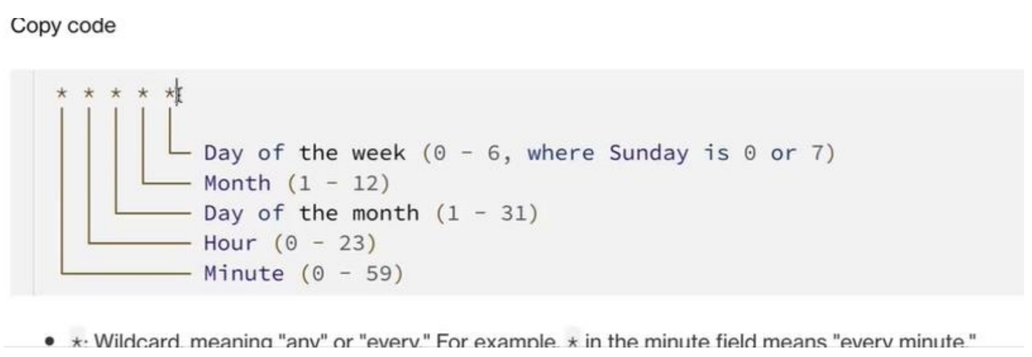
Cron Job: Cron is a time based job scheduler in unix -like operating systems , we'll leverage cron jobs to automate the execution of our shell script at predefined intervals.

What is Cron?

The cron command-line utility is a job scheduler on Unix-like operating systems. Users who set up and maintain software environments use cron to schedule jobs (commands or shell scripts), also known as cron jobs, to run periodically at fixed times, dates, or intervals.

Cron syntax :

The syntax of a cron job is composed of five fields , representing minute , hour , day of the month , month , day of the week. Each field uses a numeric value wildcard symbol.



Copy code

```
* * * * *
| | | | |
| | | | └── Day of the week (0 - 6, where Sunday is 0 or 7)
| | | └──── Month (1 - 12)
| | └────── Day of the month (1 - 31)
| └──────── Hour (0 - 23)
└────────── Minute (0 - 59)
```

- *: Wildcard, meaning "any" or "every." For example, * in the minute field means "every minute."

Your cron job

Your cron job is scheduled to run the synchronisation script every minute. Lets breakdown the line:

1. Add the cron job:

   Add the following line to the crontab file to run the sync command every minute:

``bash

*** * * * * /path/to/your/script.sh >> /path/to/your/logfile.log 2>&1**

``

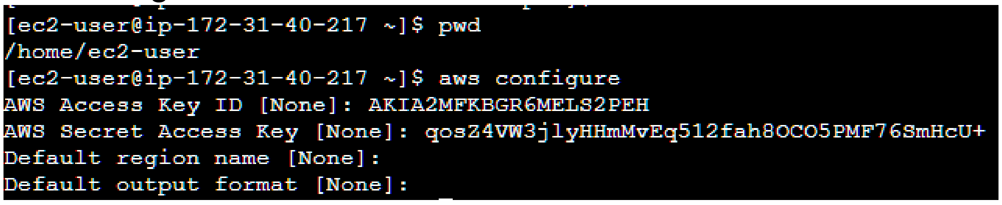
Replace /path/to/your/script.sh with the actual path to your synchronization script. The >> /path/to/your/logfile.log 2>&1 part is optional but useful for logging any output or errors from the script.

Step1:

AWS configuration:

Before dividing into scripting , setup your AWS credentials & ensure that AWS CLI is installed on your system. Aws credentials including access keys , secret access keys , default region are essential for authentication.

1. Go to AWS console.Search for EC2 & launch the instance.
2. Create a new directory by using the command:
   mkdir shell-scripts
3. Change the command to shell-scripts.
   cd shell-scripts
4. Create shellscripting file by using the command
   touch sync s3.sh
   cd ..
   pwd
5. The files which are creating in local user that all should be in my s3 bucket.
6. We have to configure AWS by using the command
   aws configure



```
[ec2-user@ip-172-31-40-217 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-40-217 ~]$ aws configure
AWS Access Key ID [None]: AKIA2MFKBGR6MELS2PEH
AWS Secret Access Key [None]: qosZ4VW3jlyHHmMvEq512fah8OCO5PMF76SmHcU+
Default region name [None]:
Default output format [None]:
```

Now we have configured AWS.

7. Create s3 bucket all your local files should be synchronised to particular bucket.

Step2:

Shell script for file synchronisation

Create bash script that uses AWS CLI ,to sync local files with a specified S3 bucket.this script should handle AWS credentials , specify the s3 bucket details , define the local directory to be backedup, optionally allows for the deletion of local files after upload.

```
#!/bin/bash
#S3 bucket details
S3_BUCKET_NAME="shell-scripting"
S3_BACKUP_FOLDER="ec2-backup"
#Local directory to be backed up
LOCAL_DIR="/home/ec2-user"
# AWS CLI command
AWS_CLI_COMMAND="aws s3 sync $LOCAL_DIR
s3://$S3_BUCKET_NAME/$S3_BACKUP_FOLDER/"
# Run the AWS CLI command
$AWS_CLI_COMMAND
```

To run this file we have to change the command

```
Chmod +x folder/file_name/
```

STEP 3:

CRON JOB SETUP

To run the cron tab in your ec2 linux machine , have to install the cron by using the following command

```
sudo yum install cronie cronie-anacron
```

Now , lets create a cron job to run this script every minute:

Open the crontab file for editing:

```
crontab -e
```

```
 # Every Minute
* * * * * /path/to/your/script.sh >> /path/to/your/logfile.log 2>&1
# Every 5  Minutes
*/5  * * * * /path/to/your/script.sh >> /path/to/your/logfile.log 2>&1
# Every Hour
0 * * * * /path/to/your/script.sh >> /path/to/your/logfile.log 2>&1
# Daily at 12 AM
0 0 * * * /path/to/your/script.sh >> /path/to/your/logfile.log 2>&1
```

Add the following command line to run the script every minute & log the output :

```
* * * * * /path/to/your/script.sh >> /path/to/your/logfile.log 2>&1
```

Replace /path/to/your/script.sh with the actual path to your synchronization script. The >> /path/to/your/logfile.log 2>&1 part is optional but useful for logging any output or errors from the script.

4. **Verification :**

After setting up the cronjob use crontab -l , to verify that your scheduled taks are correctly listed.

Conclusion :

This project showcases the synergy between AWS CLI , shell scripting , cronjobs , offering an automated solution for file synchronisation with Amazon S3.By following these steps , you empower your system to regularly & efficiently manage data backups , providing a scalable & robust solution for cloud-based file storage.

Remember to adhere to security best practices especially when handling AWS credentials & consider incorporating IAM roles with the principle of least privilege.Happy Scripting!