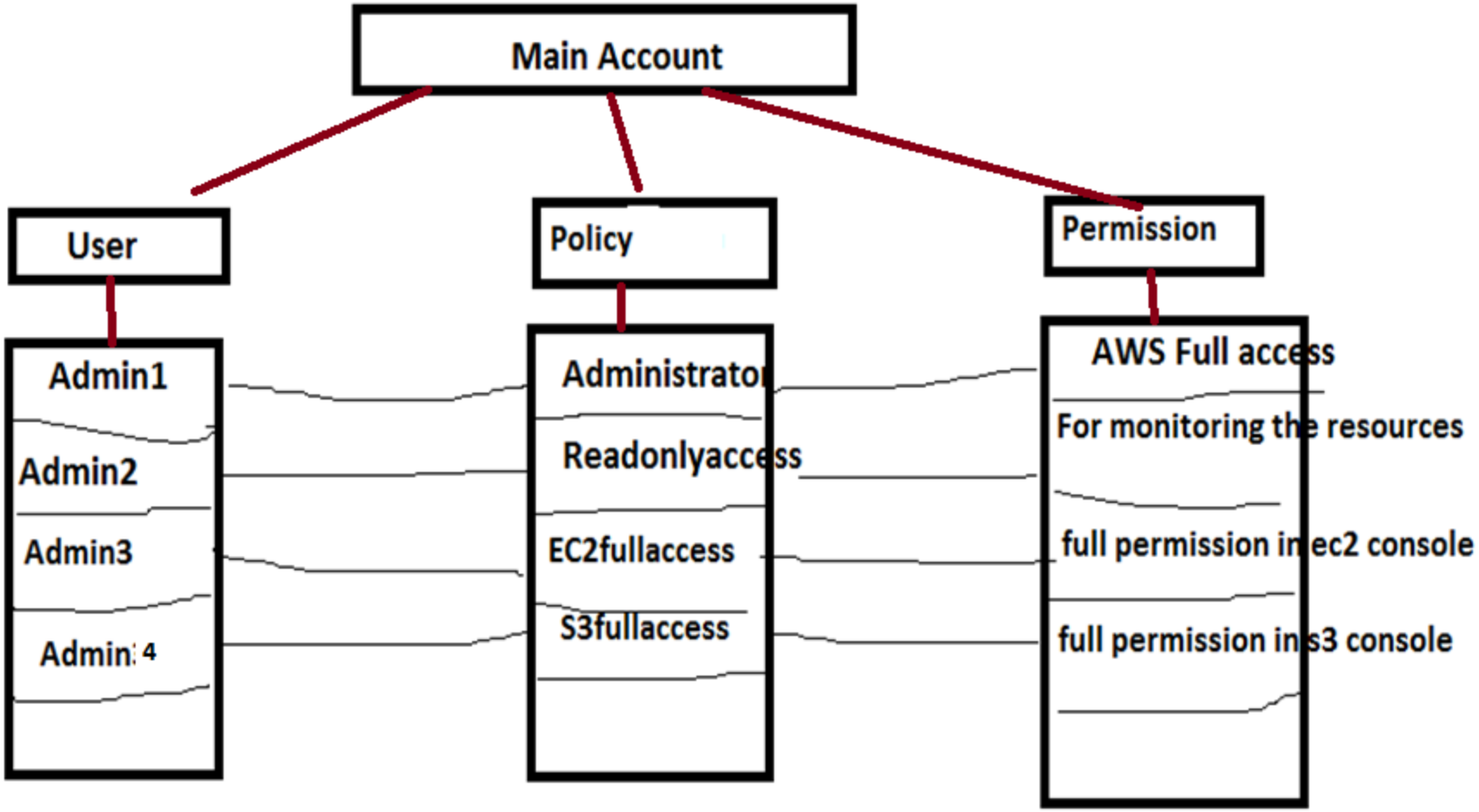
AWS IAM


Main Account
User
Policy
Permission
Admin1
Admin2
Admin3
Admin 4
Administrator
Readonlyaccess
EC2fullaccess
S3fullaccess
AWS Full access
For monitoring the resources
full permission in ec2 console
full permission in s3 console

AWS Account Root User

- IAM stands for Identity Access Management.
- It is used to set users, permissions and roles. It allows you to grant access to the different parts of the aws platform.
- With IAM, Organizations can centrally manage users, security credentials such as access keys, and permissions that control which AWS resources users can access.
- Without IAM, Organizations with multiple users must either create multiple user accounts, each with its own billing and subscriptions to AWS products or share an account with a single security credential. Without IAM, you also don't have control about the tasks that the users can do.
- IAM enables the organization to create multiple users, each with its own security credentials, controlled and billed to a single aws account. IAM allows the user to do only what they need to do as a part of the user's job.

AWS Root User

- When you first create an AWS account, you create an account as a root user identity which is used to sign in to AWS.
- You can sign to the AWS Management Console by entering your email address and password. The combination of email address and password is known as **root user credentials**.
- When you sign in to AWS account as a root user, you have unrestricted access to all the resources in AWS account.
- The Root user can also access the billing information as well as can change the password also.

IAM Roles

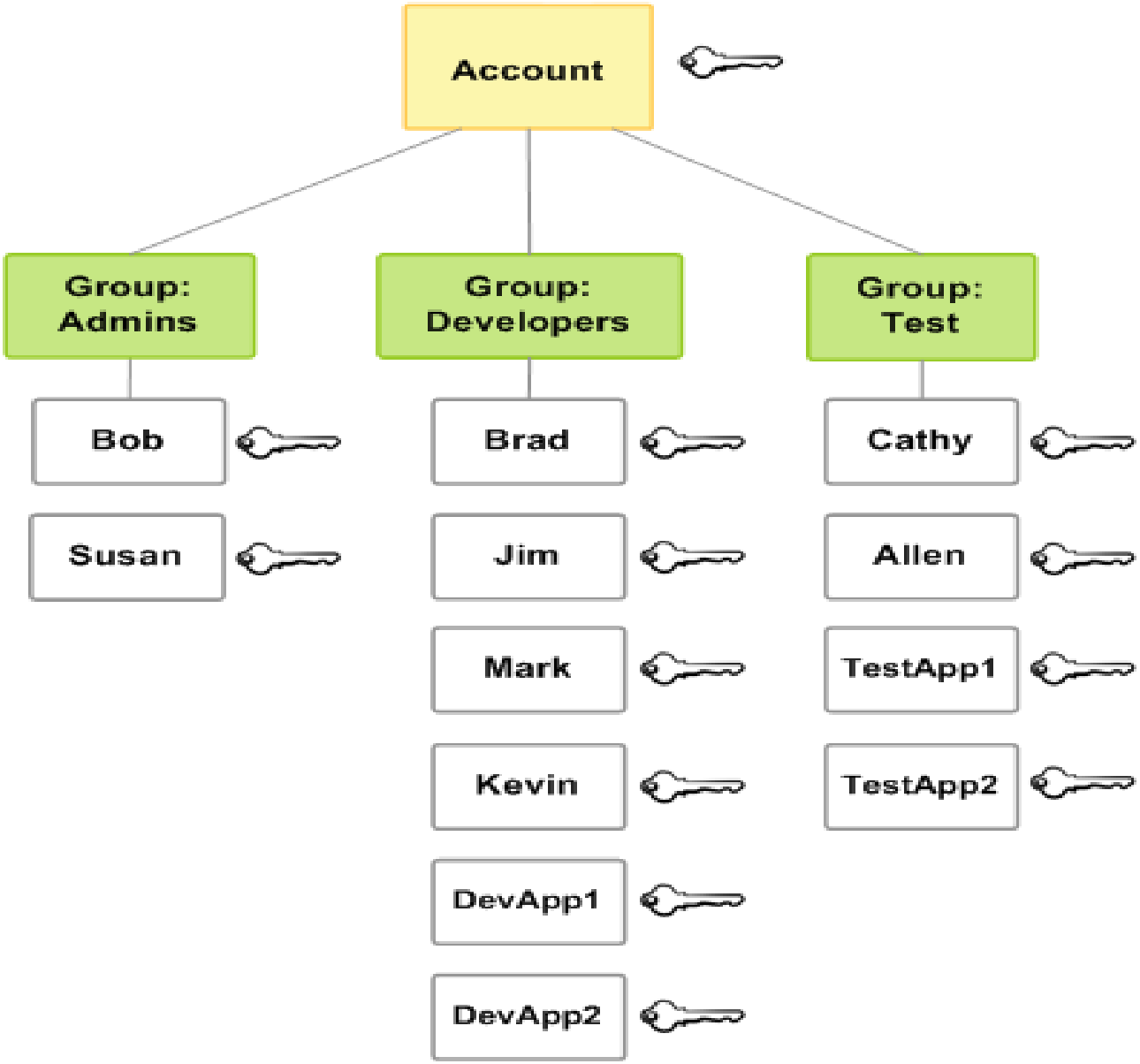
- A role is a set of permissions that grant access to actions and resources in AWS. These permissions are attached to the role, not to an IAM User or a group.
- An IAM User can use a role in the same AWS account or a different account.
- You can use the roles to delegate access to users, applications or services that generally do not have access to your AWS resources.

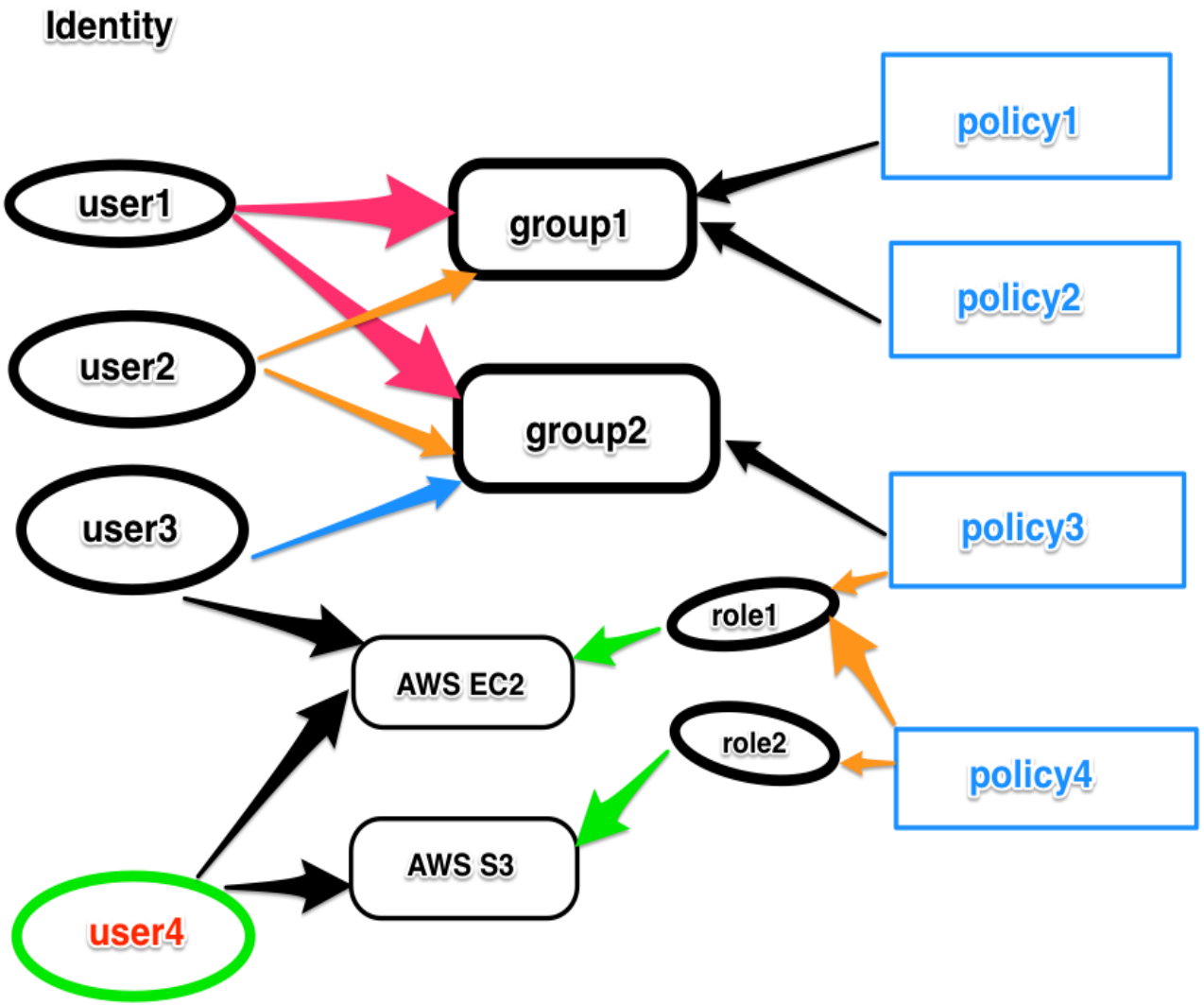
- A role is not uniquely associated with a single person; it can be used by anyone who needs it

IAM Components

1) Users –Entity to manage different aws resource
2) Group –set of users
3) Policy --permission
4) Role –set of policy
5) Password

- ✓ AWS Console Management password
- ✓ Auto generated password
- ✓ Multi factor Authentication(MFA)

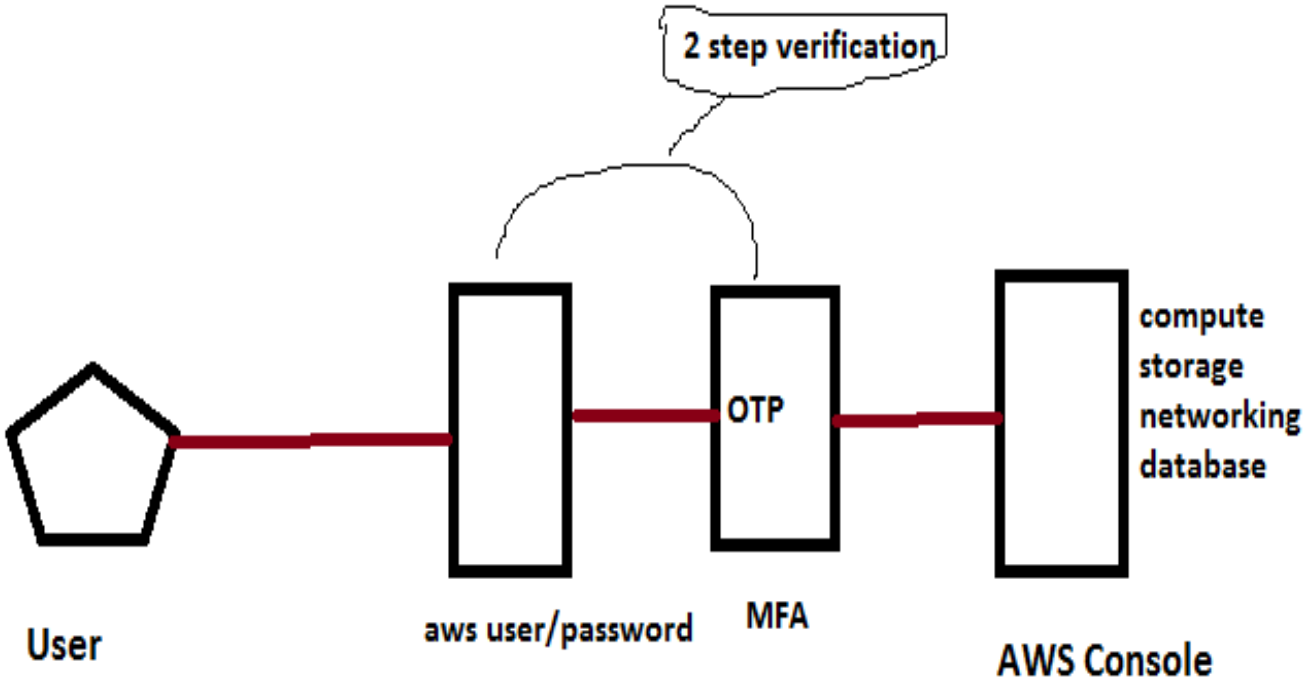

Account
Group: Admins
Group: Developers
Group: Test
Bob
Susan
Brad
Jim
Mark
Kevin
DevApp1
DevApp2
Cathy
Allen
TestApp1
TestApp2


Identity
policy1
user1
group1
policy2
user2
group2
user3
policy3
role1
AWS EC2
role2
policy4
AWS S3
user4

AWS IAM Hands on

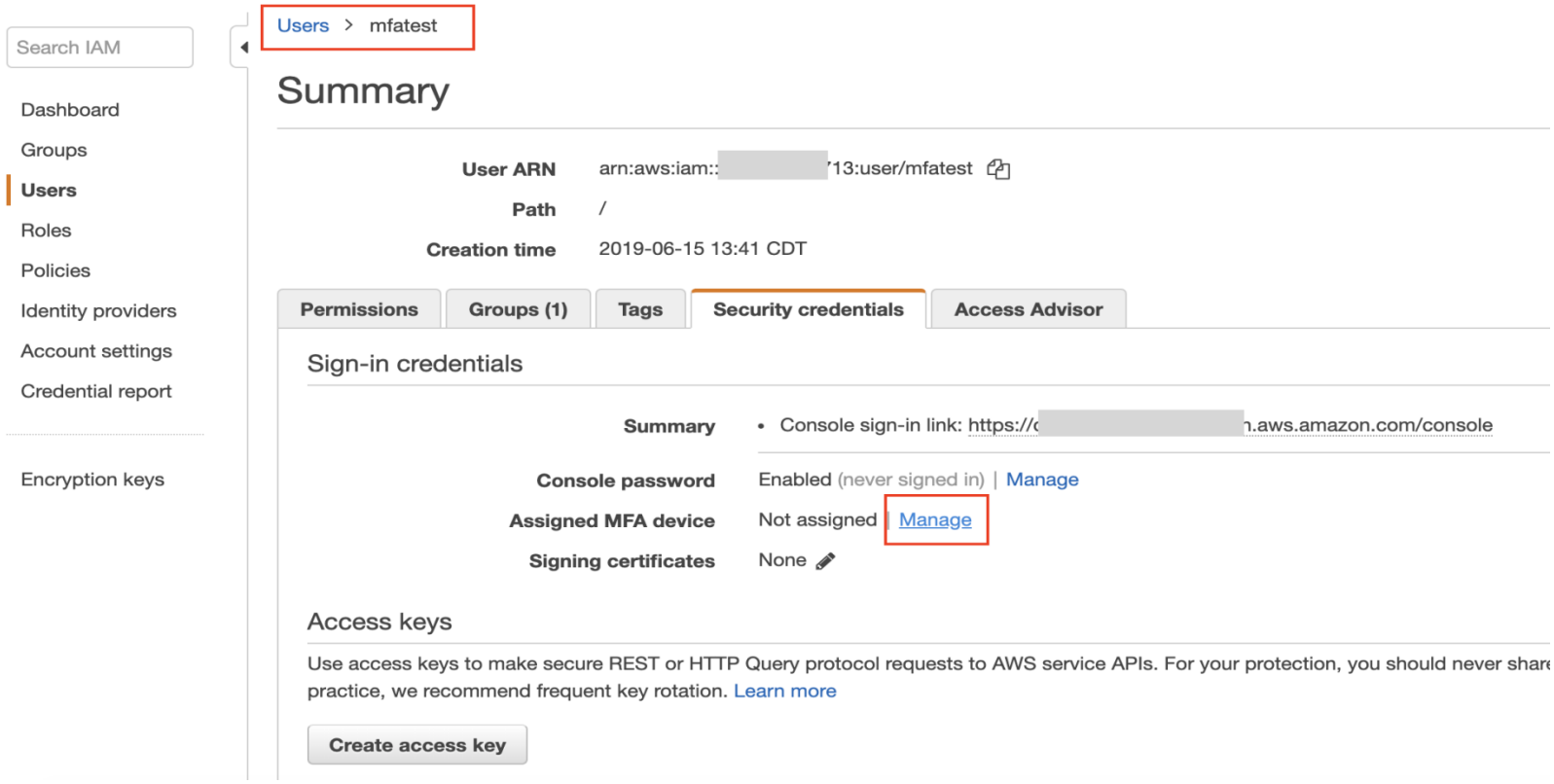
1) Create an IAM user and assign full access in aws console
2) Create an IAM user and assign full access in ec2 console only
3) Create an IAM user and assign full access in S3 console only
4) Create an IAM user and assign read only access in aws console
5) How to add multiple permission to an user
6) Create group –add user—assign policy
7) Configure custom policy
8) Creating Roles
10) Autogenerated password
10) Configure MFA – 2 step authentication for user aws console login
11) Configure MFA – 2 step authentication for user ec2 instance login
12) Sync S3 bucket with EC2 instance

MFA -Muti Factor Authentication



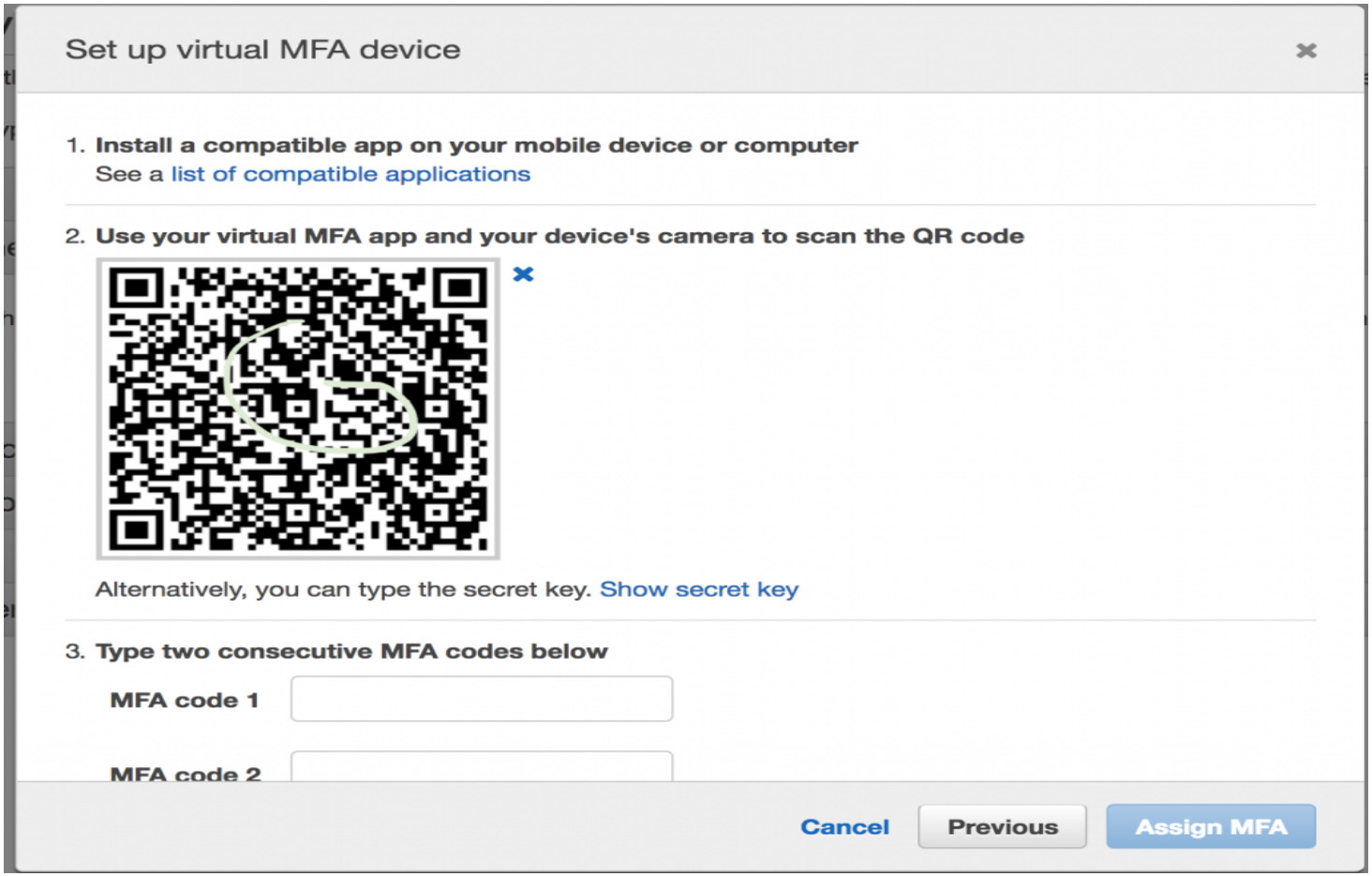
Lab: Configure MFA

1) This is two step authentication
2) Create an user in IAM
3) After creating --Open it – security credential –Assign MFA– Manage--

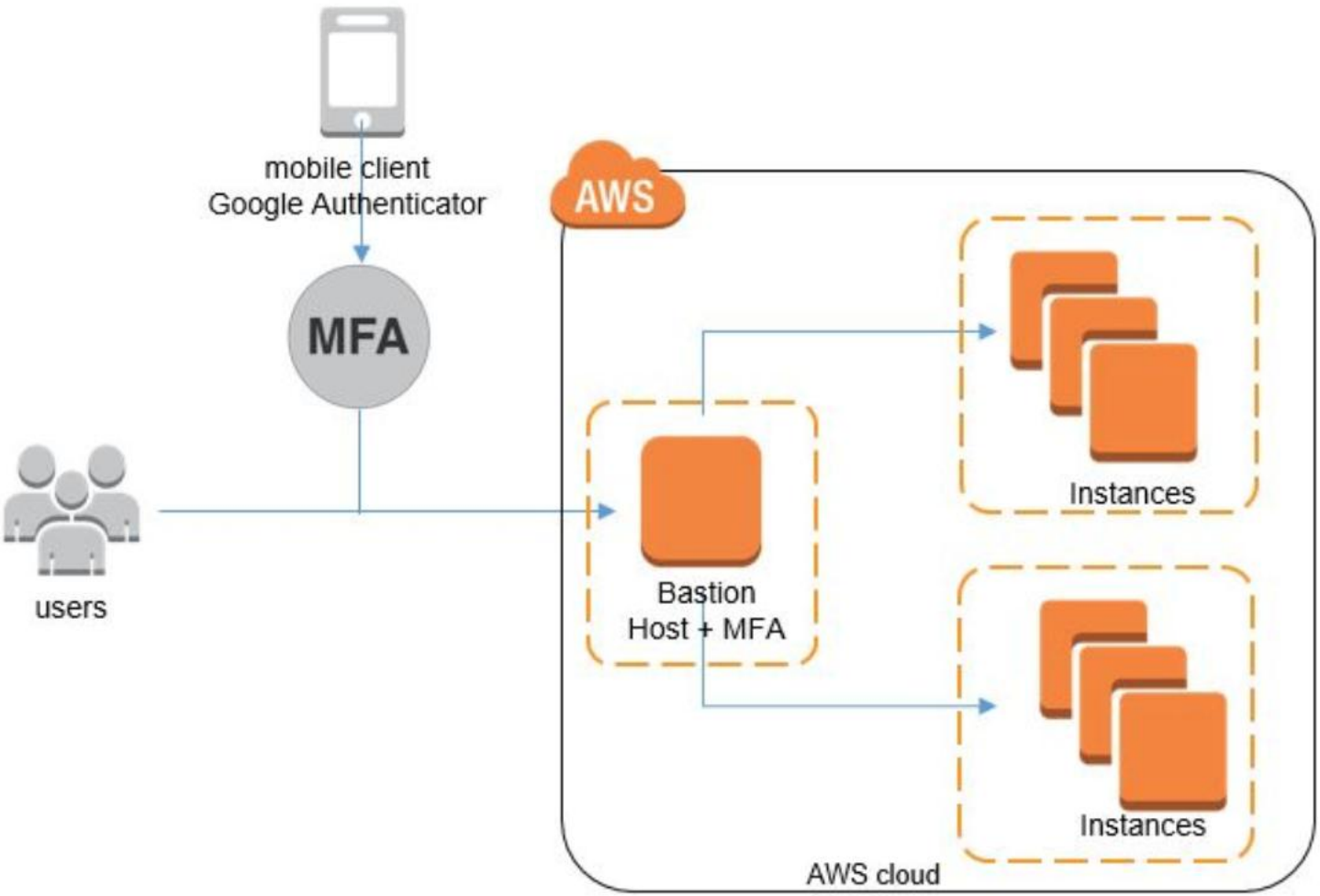


Lab: Configure MFA

Now select Virtual MFA– Download google authenticator from play store in android mbile –open and scan this code –one by one two code will displayed there—put these code here--- Assign MFA



Configure MFA for EC2 Instance



Lab: Configure MFA

- Why SSH Security is Important?
- What is Two-factor Authentication (or) Multi-Factor Authentication
- How to setup MFA for SSH in AWS EC2 – AWS MFA Setup
  - Step1: Install EPEL Repo on the EC2 instance
  - Step2: Install Google Authenticator on the EC2 instance
  - Step3: Configure EC2 SSH to use Google Authentication module
  - Step4: Configure Google Authenticator
  - Step5: Restart SSH services on the EC2 server
  - Step6: SSH to validate the AWS MFA setup.

Lab: Configure MFA

1) # yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

2) # yum install google-authenticator.x86_64 –y

3) #vi /etc/pam.d/sshd

#auth substack password-auth --- comment on this line

auth required pam_google_authenticator.so -----add this line at the end

4) #vi /etc/ssh/sshd_config

ChallengeResponseAuthentication yes

#ChallengeResponseAuthentication no

AuthenticationMethods publickey,keyboard-interactive

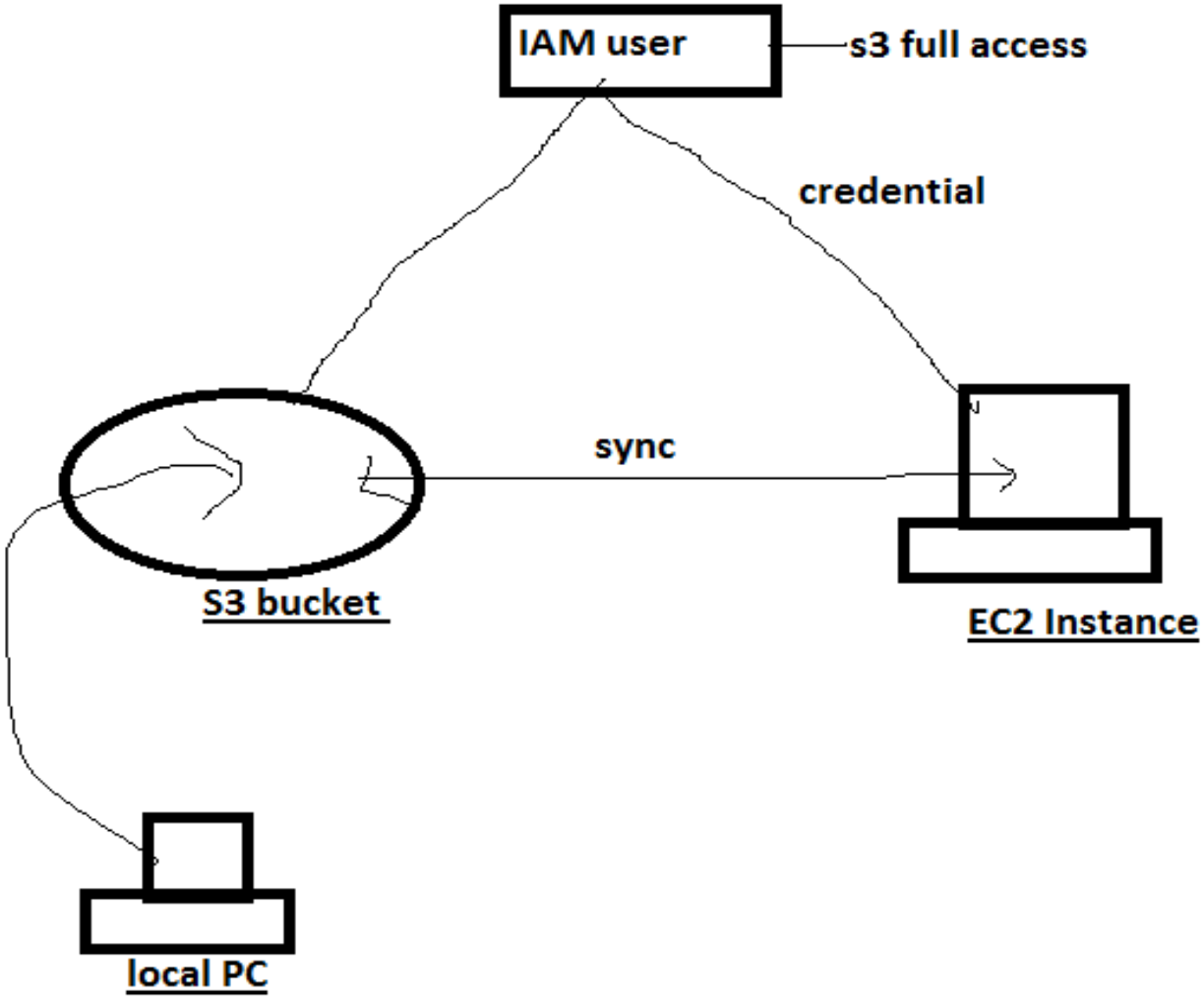
5) **Install Google Authenticator in mobile**

6) # google-authenticator --- scan the QR code from mobile

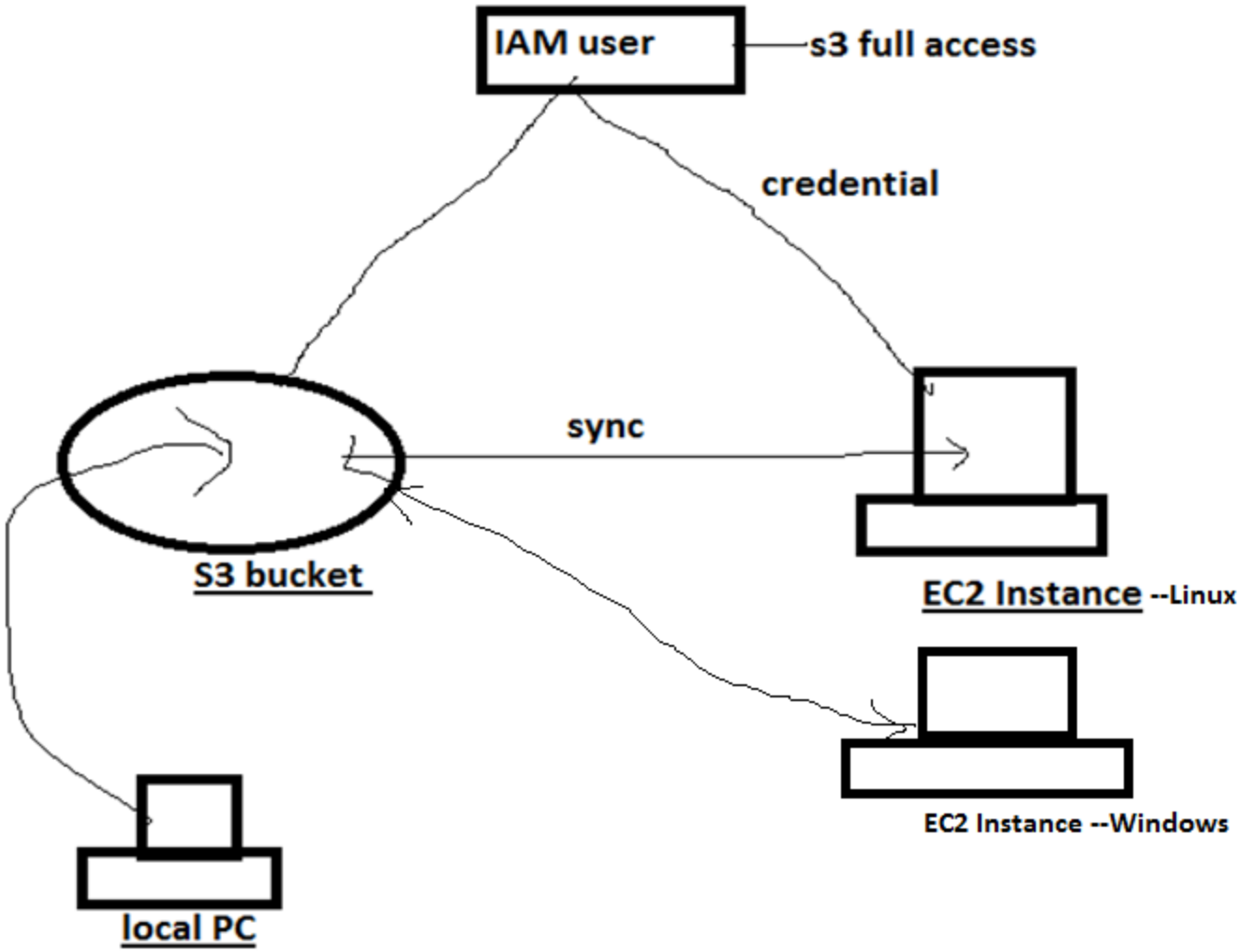
Press Y, Y, N, Y

7) # systemctl restart sshd

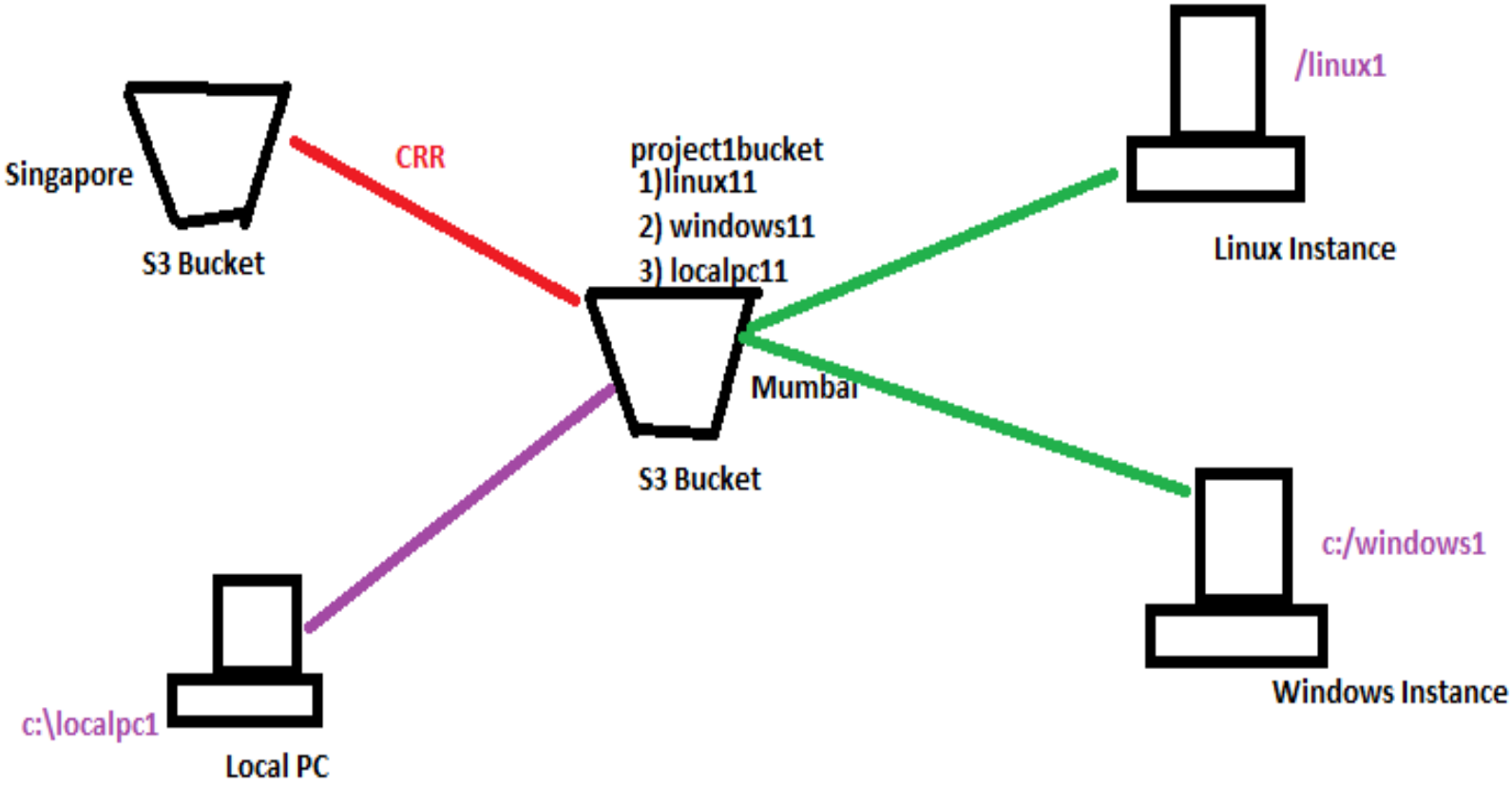
Sync S3 bucket with EC2 instance



Sync S3 bucket with EC2 instance



Final Practise/Test lab 1



Sync S3 bucket with EC2 instance (Linux) --Steps

1) Open IAM –create an user and assign Amazons3fullaccess policy there.

2) Open this user –Security credential –create access key –download access key.csv file –open it with excel and note down access key and secret access key.

3) Open S3 and create one bucket (indiabucket) –open the bucket—create 3 folder there(lab1, lab2, lab3)

4)Launch Amazon linux2 AMI ---Open it – type command “aws configure” and give the data.

Access key:

Secret access key:

Default region: ap-south-1

Output format: JSON

```
$ mkdir lab11
$ cd lab11
$ touch file1 file2 file2
$ aws   s3 sync  /home/ec2-user/lab11      s3://indiabucket/lab1
```

Now check the data in S3 bucket

Upload some files in S3 bucket—Come to EC2 instance

```
$  aws  s3 sync    s3://indiabucket/lab1       /home/ec2-user/lab11
$ ls
```

Sync S3 bucket with EC2 instance (Windows) --Steps

1) Open IAM –create an user and assign Amazons3fullaccess policy there.

2) Open this user –Security credential –create access key –download access key.csv file –open it with excel and note down access key and secret access key.

3) Open S3 and create one bucket (indiabucket) –open the bucket—create 3 folder there(lab1, lab2, lab3)

4)Launch Windows instance ---Open it – Download and install “Amazon CLI “

Open cmd– type command “aws configure” and give the data.

Access key:

Secret access key:

Default region: ap-south-1

Output format: JSON

C:\ mkdir lab22

C:\ cd lab22

C:\ touch file1 file2 file2

C:\ aws s3 sync c:\lab22 s3://indiabucket/lab2

Now check the data in S3 bucket

Upload some files in S3 bucket—Come to EC2 instance

C:\ aws s3 sync s3://indiabucket/lab2 c:\lab22

C:\ ls

How to Schedule the File sync (Auto Backup)(Windows)

Windows

1) Create one batch file and write the command there

aws s3 sync c:\awsdata2\ s3://awsbatch100/windows1

Save file--- file name: s3sync.bat, save as type: all files

2) Open task Scheduler – create basic task—name:test1 –next—select daily-next-set time –next—start a program– browse and select created batch file –ok—finish.

3) Now create some files in c:\awsdata2\ folder and wait for that scheduled time –then check the output in s3 bucket.

How to Schedule the File sync (Auto Backup)(Linux)

Linux

1) Create one shell script file and write the command there

$ pwd

/home/ec2-user

$nano test1.sh

aws s3 sync /home/ec2-user/linux11 s3://awsbatch100/linux1

Save and exit

2) Chmod u+x test1.sh

3) crontab –e

- * * * * sh /home/ec2-user/test1.sh

3) Now create some files in /home/ec2-user/linux11 folder and wait for that scheduled time –then check the output in s3 bucket.

Note: the given time is to run the script in every one minue