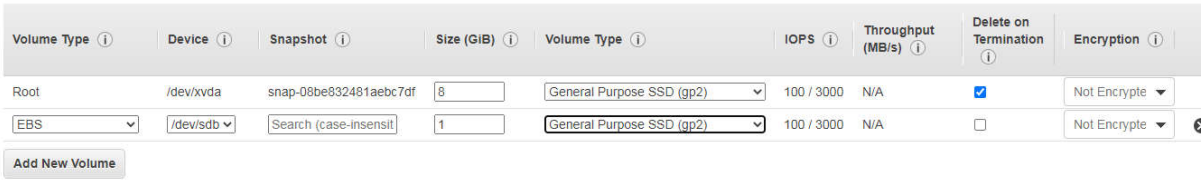
Elastic Block Storage

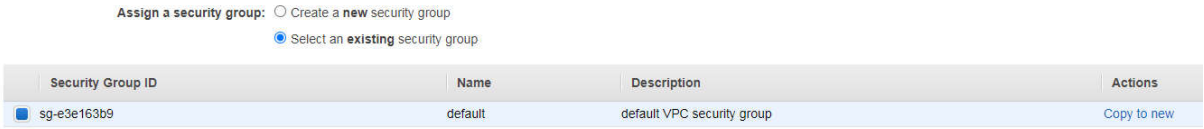
Step1: Create an Ec2 instance with EBS volumes

Launch an Amazon Linux 2 AMI (HVM), SSD Volume Type - (64-bit Arm)

Change the voulme type:



Select the security



Create the instance with the extra EBS Volume

Step2: Connect to the instance using SSH

Check the Volumes :

Lists information about all or the specified block devices. **lsblk**

```
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-23-149 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdb    202:16   0   1G  0 disk
[ec2-user@ip-172-31-23-149 ~]$
[ec2-user@ip-172-31-23-149 ~]$
```

Determine file type **-s** option causes **file** to also read argument files which are block or character special files. **sudo file -s /dev/xvda1**

```
[ec2-user@ip-172-31-23-149 ~]$ sudo file -s /dev/xvda1
/dev/xvda1: SGI XFS filesystem data (blksz 4096, inosz 512, v2 dirs)
[ec2-user@ip-172-31-23-149 ~]$ sudo mkfs -t ext4 /dev/xvdb
mke2fs 1.42.9 (28-Dec-2013)
```

Build a file system. **mkfs -**

Build a Linux file system.

sudo mkfs -t ext4 /dev/xvdb

```
[ec2-user@ip-172-31-23-149 ~]$ sudo mkfs -t ext4 /dev/xvdb
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
65536 inodes, 262144 blocks
13107 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=268435456
8 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

Make a Directory : **sudo mkdir data**

```
[ec2-user@ip-172-31-23-149 ~]$ sudo mkdir data
[ec2-user@ip-172-31-23-149 ~]$ ls
data
```

Mount the file system:

sudo mount /dev/xvdb data

```
[ec2-user@ip-172-31-23-149 ~]$ sudo  mount  /dev/xvdb   data
[ec2-user@ip-172-31-23-149 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdb    202:16   0   1G  0 disk /home/ec2-user/data
[ec2-user@ip-172-31-23-149 ~]$
```

Make a File in new volume : **sudo nano file1.txt**

```
[ec2-user@ip-172-31-23-149 ~]$ vim sample1.txt
[ec2-user@ip-172-31-23-149 ~]$ ls
data  sample1.txt
[ec2-user@ip-172-31-23-149 ~]$
```

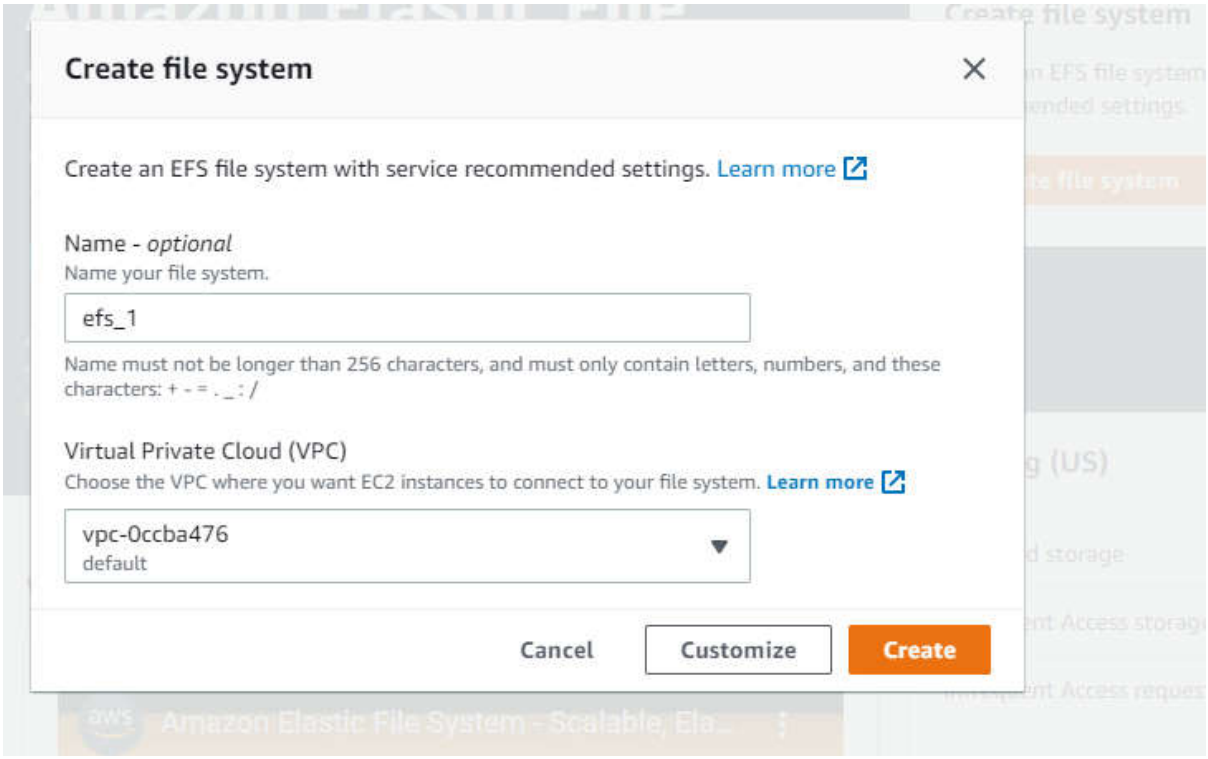
Unmount a file system : **sudo umount -l data**

```
[ec2-user@ip-172-31-23-149 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdb    202:16   0   1G  0 disk
[ec2-user@ip-172-31-23-149 ~]$
```

Creating an EFS

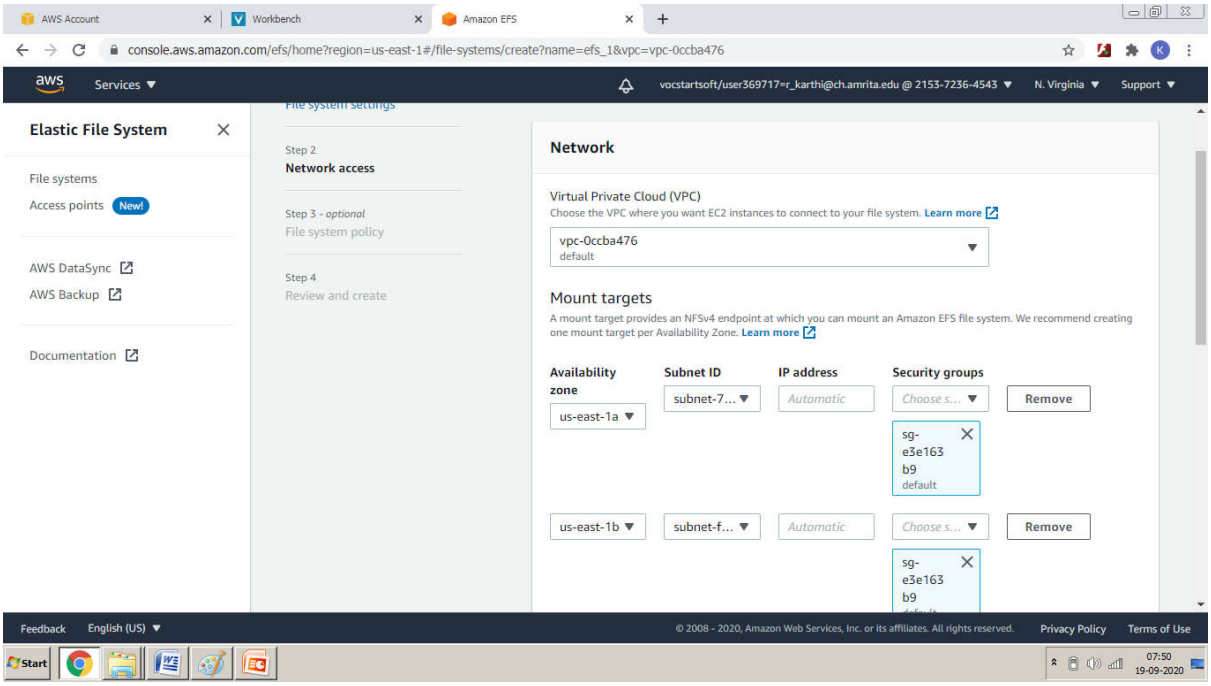
Step1: Create a file system

Select the default VPC



Choose Customise the view the other settings

Network Access

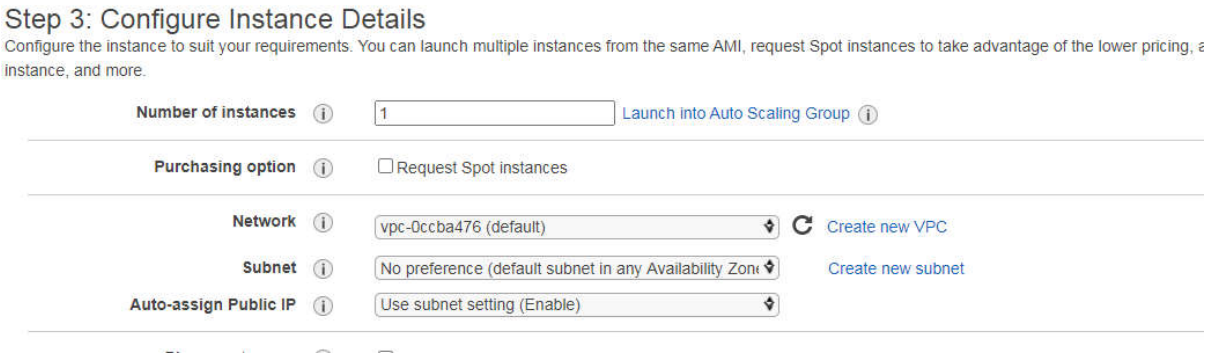


Set all as default and create the EFS

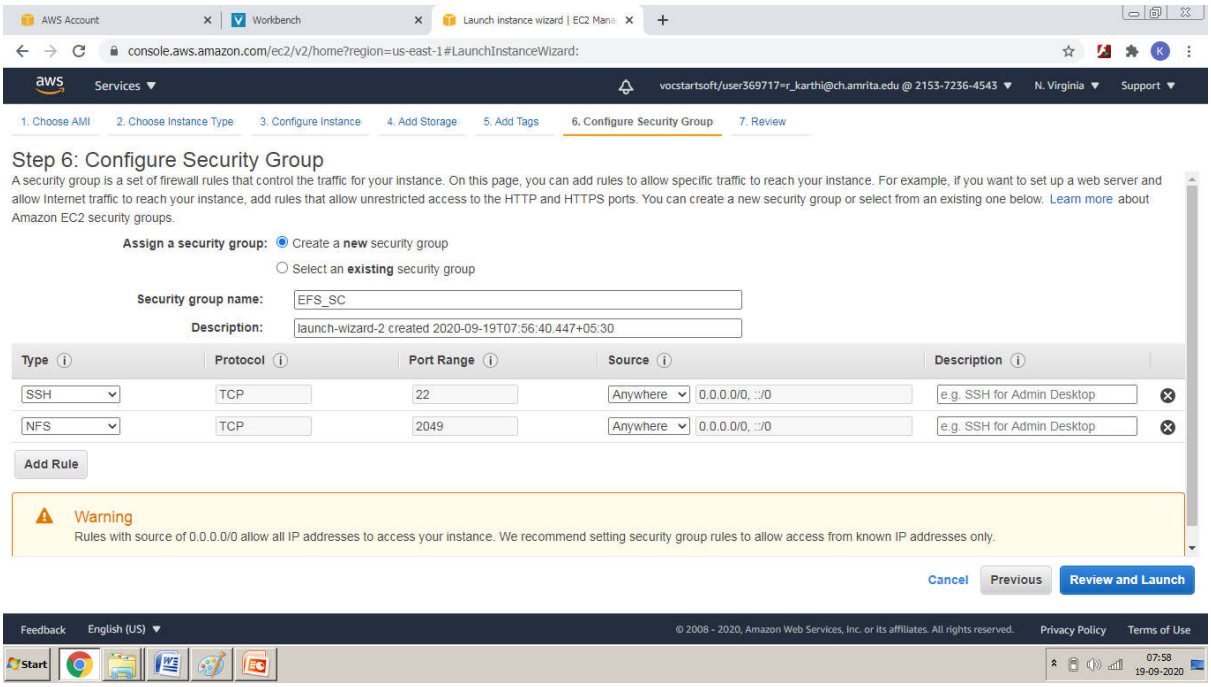
Name	File system ID	Encrypted	Total size	Size in EFS Standard	Size in EFS IA	Provisioned Throughput (MiB/s)
efs_1	fs-5543d9d7	Encrypted	6 KiB	6 KiB	0 Bytes	-

Step2: Create an instance and Connect to EFS

Launch an Linux instance and set the network to default VPC



Set the security group



Create the instance

Step3: Connect to the EFS using ssh

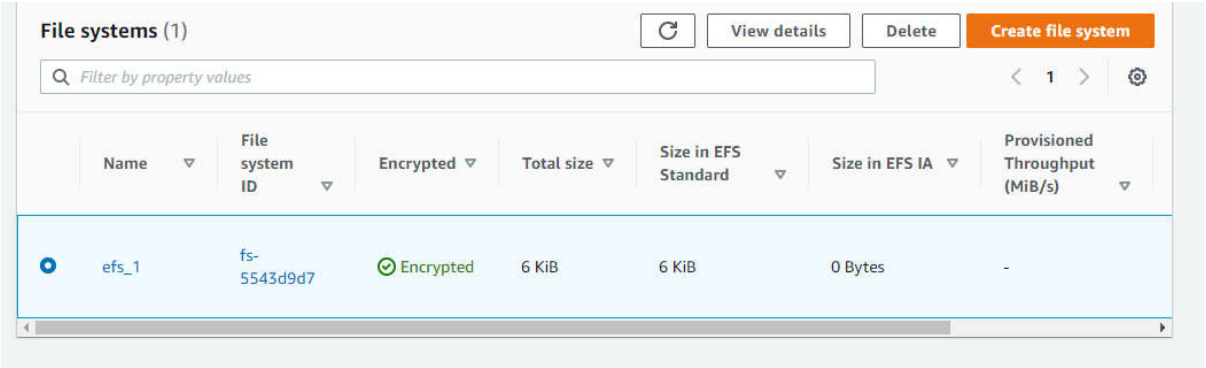
```
       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
No packages needed for security; 1 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-21-112 ~]$ ls
[ec2-user@ip-172-31-21-112 ~]$ ls
```

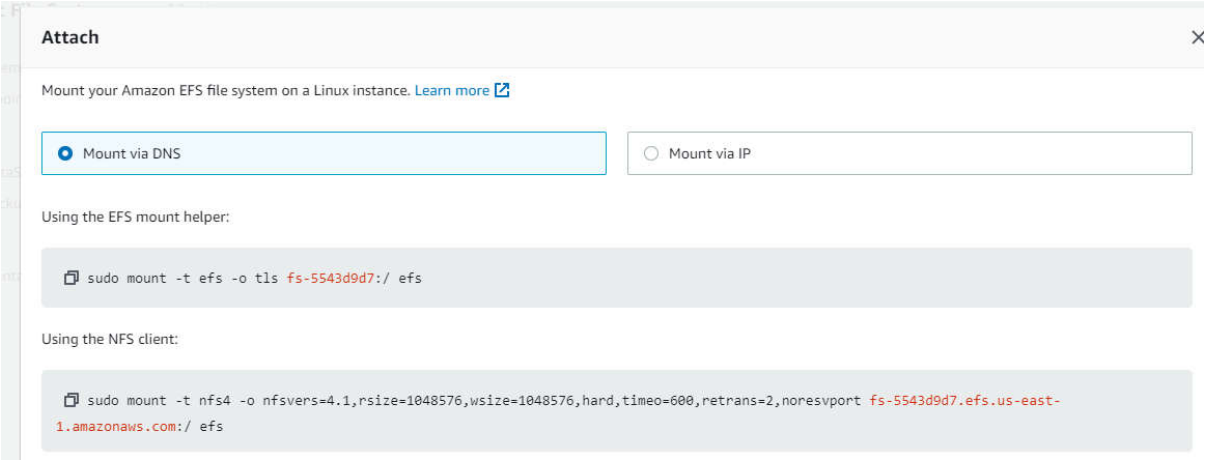
Make a directory to mount the EFS drive to the instance

mkdir efs2

```
[ec2-user@ip-172-31-21-112 ~]$ mkdir efs2
[ec2-user@ip-172-31-21-112 ~]$ ls
efs2
```

Get the Mount details from EFS – view details – attach

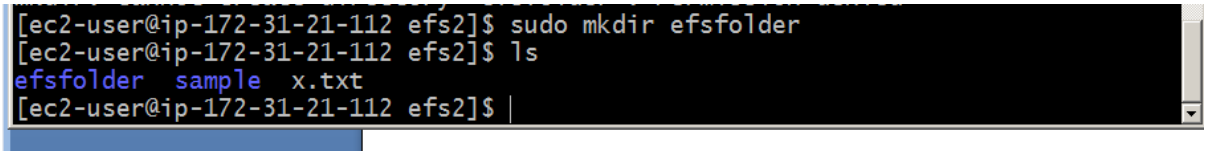




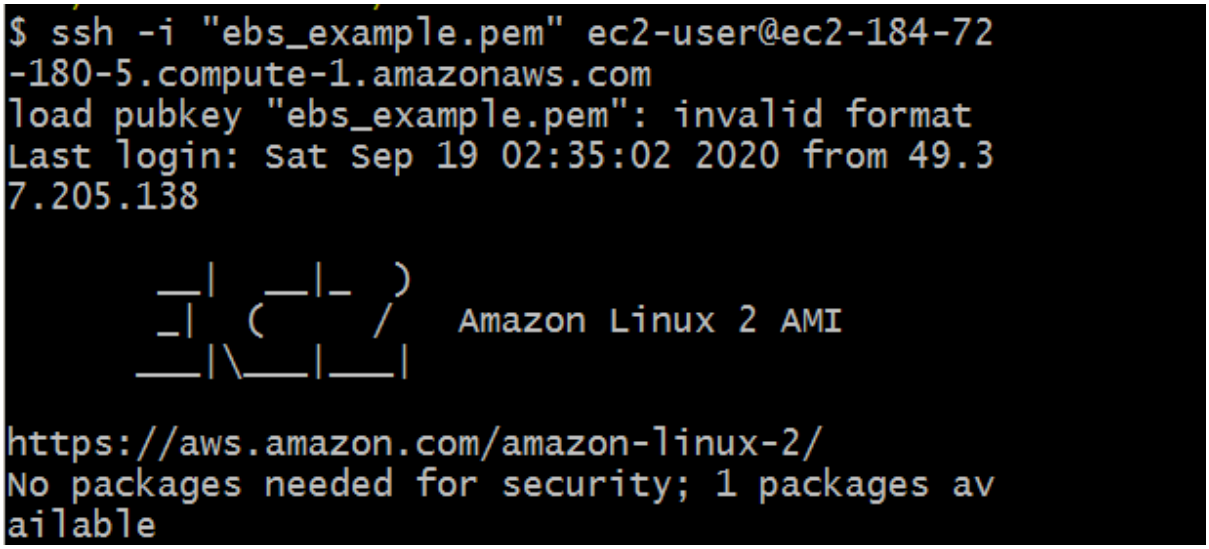
Mount the EFS device to the instance

```
[ec2-user@ip-172-31-21-112 ~]$ sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,w
size=1048576,hard,timeo=600,retrans=2,noresvport fs-5543d9d7.efs.us-east-1.amazo
naws.com:/  /home/ec2-user/efs2
```

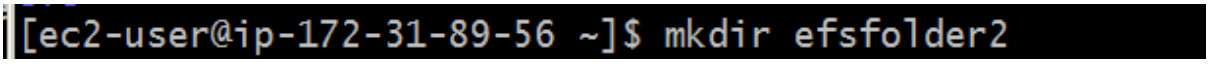
Make a folder in EFS drive - efsfolder



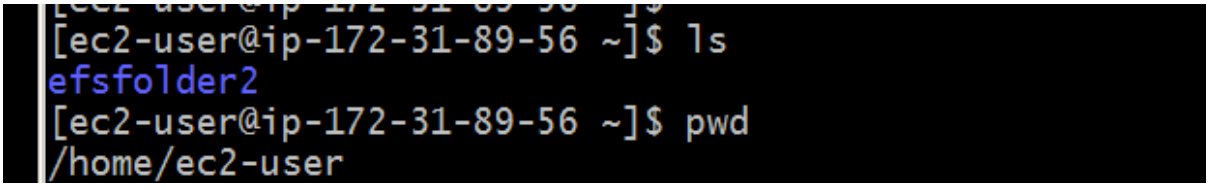
Create an other instance and connect to the efs device



Make a new folder to mount the efs

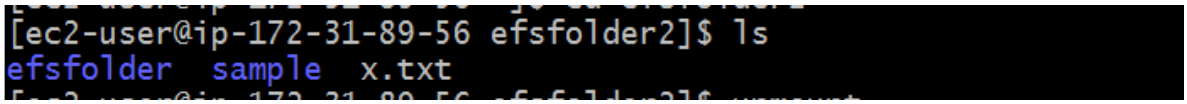


Check the folder

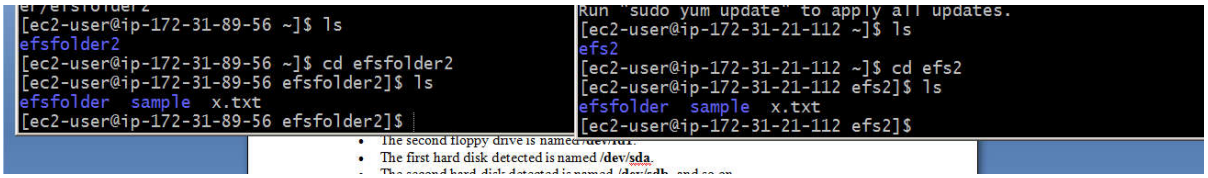


Mount the EFS to ec2 instance in folder efsfolder2

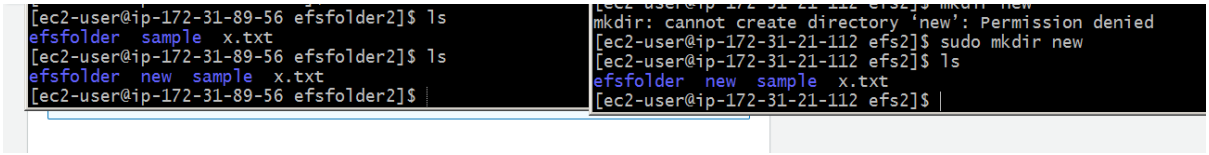
The same information in ec1 is available for ec2 instance



In both the EC2 instance the files are available for update.



A new folder is added which is visible in both EC2 instances



GCP using command line

gcloud auth login

gcloud config set project

gsutil mb gs://karth

for File System

https://cloud.google.com/filestore/docs/quickstart-console

C.4. Device Names in Linux

Linux disks and partition names may be different from other operating systems. You need to know the names that Linux uses when you create and mount partitions. Here's the basic naming scheme:

- The first floppy drive is named **/dev/fd0**.
- The second floppy drive is named **/dev/fd1**.
- The first hard disk detected is named **/dev/sda**.
- The second hard disk detected is named **/dev/sdb**, and so on.

- The first SCSI CD-ROM is named **/dev/scd0**, also known as **/dev/sr0**.

The partitions on each SCSI disk are represented by appending a decimal number to the disk name: **sda1** and **sda2** represent the first and second partitions of the first SCSI disk drive in your system.

Here is a real-life example. Let's assume you have a system with 2 SCSI disks, one at SCSI address 2 and the other at SCSI address 4. The first disk (at address 2) is then named **sda**, and the second **sdb**. If the **sda** drive has 3 partitions on it, these will be named **sda1**, **sda2**, and **sda3**. The same applies to the **sdb** disk and its partitions.

Note that if you have two SCSI host bus adapters (i.e., controllers), the order of the drives can get confusing. The best solution in this case is to watch the boot messages, assuming you know the drive models and/or capacities.

xvd means **X**en **V**irtual **D**isk on a Xen Server. /dev/xvd is the standard name for **X**en **v**irtual **d**isk, by analogy with the **hd*** in IDE and **sd*** in SCSI. The first virutal disk is /dev/xvda, which can be partioned into /dev/xvda1 (just as /dev/sda1 is the first partition of the first SCSI or SCSI-like storage device). In short, treat xvda1 exactly as you would sda1 on a regular PC.