Amazon EC2 Storage & Data Management

Elastic Block Store (EBS):

AWS EBS (Elastic Block Store) is a block-level storage service provided by Amazon Web Services (AWS). It offers persistent block storage volumes that can be attached to Amazon EC2 instances, providing durable and low-latency storage for applications.

Here are some basic concepts related to AWS EBS:

1. Volume: An EBS volume is a block-level storage device that can be attached to an EC2 instance. It provides a specific amount of storage capacity, ranging from 1 GB to 16 TB, and appears as a block device to the instance.
2. Snapshots: EBS snapshots are point-in-time copies of EBS volumes. They capture the data and configuration of the volume at a specific moment. Snapshots are stored in Amazon S3 and can be used to create new volumes or restore existing volumes.
3. Volume Types: AWS EBS offers different types of volumes optimized for various use cases:

a. General Purpose (SSD): Provides a balance of price and performance for a wide range of workloads.

b. Provisioned IOPS (SSD): Offers high-performance storage with a consistent level of I/O operations per second (IOPS).

c. Throughput Optimized (HDD): Suitable for large, sequential workloads with high throughput requirements.

d. Cold HDD: Offers low-cost storage for infrequently accessed workloads.

e. Magnetic: The original EBS storage type, now being phased out, offering the lowest cost per gigabyte.

4. Elastic Volumes: AWS EBS supports elastic volumes, which allow you to dynamically adjust the size, performance, and volume type of an existing EBS volume without requiring any downtime or data migration.
5. Encryption: EBS volumes can be encrypted using AWS Key Management Service (KMS) keys. Encryption provides an additional layer of data security and ensures that the data at rest is protected.
6. Availability and Durability: EBS volumes are designed for durability and availability. They are automatically replicated within an Availability Zone (AZ) to protect against component failure. You can also create snapshots to back up your data and store them in multiple AZs or regions.

By leveraging EBS, you can have flexible and scalable block storage for your EC2 instances, ensuring data persistence and high performance for your applications running in the AWS cloud.

Some key point of EBS

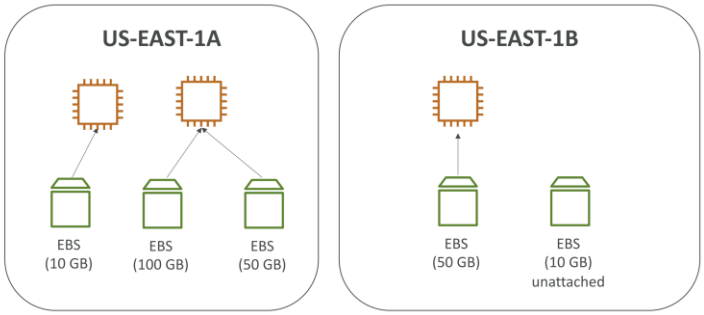
- An EBS (Elastic Block Store) Volume is a network drive you can attach to your instances while they run
- It allows your instances to persist data, even after their termination
- **They can only be mounted to one instance at a time** (at the CCP level)
- They are bound to **a specific availability zone**

- Analogy: Think of them as a "network USB stick"
- Free tier: 30 GB of free EBS storage of type General Purpose (SSD) or Magnetic per month
- It's a network drive (i.e. not a physical drive)
  - It uses the network to communicate the instance, which means there might be a bit of latency
  - It can be detached from an EC2 instance and attached to another one quickly

- It's locked to an Availability Zone (AZ)
  - An EBS Volume in us-east-1a cannot be attached to us-east-1b
  - To move a volume across, you first need to snapshot it

- Have a provisioned capacity (size in GBs, and IOPS)
  - You get billed for all the provisioned capacity
  - You can increase the capacity of the drive over time

Show EBS volume by an example image:



Ec2- Instance Store

- EBS volumes are **network drives** with good but "limited" performance
- <u>If you need a high-performance hardware disk, use EC2 Instance Store</u>

- Better I/O performance
- EC2 Instance Store lose their storage if they're stopped (ephemeral)
- Good for buffer / cache / scratch data / temporary content
- Risk of data loss if hardware fails
- Backups and Replication are your responsibility

Local Ec2 Instance Store

Very high IOPS

Instance Size	100% Random Read IOPS	Write IOPS
i3.large *	100,125	35,000
i3.xlarge *	206,250	70,000
i3.2xlarge	412,500	180,000
i3.4xlarge	825,000	360,000
i3.8xlarge	1.65 million	720,000
i3.16xlarge	3.3 million	1.4 million
i3.metal	3.3 million	1.4 million
i3en.large *	42,500	32,500
i3en.xlarge *	85,000	65,000
i3en.2xlarge *	170,000	130,000
i3en.3xlarge	250,000	200,000
i3en.6xlarge	500,000	400,000
i3en.12xlarge	1 million	800,000
i3en.24xlarge	2 million	1.6 million
i3en.metal	2 million	1.6 million

EBS – Delete on Termination attributes

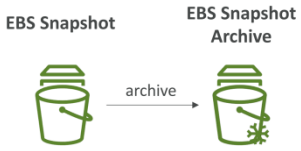
- Controls the EBS behaviour when an EC2 instance terminates
  - By default, the root EBS volume is deleted (attribute enabled)
  - By default, any other attached EBS volume is not deleted (attribute disabled)
- This can be controlled by the AWS console / AWS CLI
- Use case: preserve root volume when instance is terminated

EBS Snapshots

- Make a backup (snapshot) of your EBS volume at a point in time
- Not necessary to detach volume to do snapshot, but recommended
- Can copy snapshots across AZ or Region

EBS Snapshots Features

- EBS Snapshot Archive
  - Move a Snapshot to an "archive tier" that is 75% cheaper
  - Takes within 24 to 72 hours for restoring the archive



- Recycle Bin for EBS Snapshots
  - Setup rules to retain deleted snapshots so you can recover them after an accidental deletion
  - Specify retention (from 1 day to 1 year)

- Fast Snapshot Restore (FSR)
  - Force full initialization of snapshot to have no latency on the first use ($$$)

EBS volume Types

- EBS Volumes come in 6 types
  - gp2 / gp3 (SSD): General purpose SSD volume that balances price and performance for a wide variety of workloads
  - io1 / io2 (SSD): Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads
  - st1 (HDD): Low cost HDD volume designed for frequently accessed, throughput-intensive workloads
  - sc1 (HDD): Lowest cost HDD volume designed for less frequently accessed workloads

- EBS Volumes are characterized in Size | Throughput | IOPS (I/O Ops Per Sec)
- When in doubt always consult the AWS documentation – it's good!
- Only gp2/gp3 and io1/io2 can be used as boot volumes

EBS volume types Usage Case-General purpose SSD

- Cost effective storage, low-latency
- System boot volumes, Virtual desktops, Development and test environments
- 1 GiB - 16 TiB
- gp3:
  - Baseline of 3,000 IOPS and throughput of 125 MiB/s
  - Can increase IOPS up to 16,000 and throughput up to 1000 MiB/s independently
- gp2:
  - Small gp2 volumes can burst IOPS to 3,000
  - Size of the volume and IOPS are linked, max IOPS is 16,000
  - 3 IOPS per GB, means at 5,334 GB we are at the max IOPS

EBS volume types Usage Case-Provisioned (PIOPS) SSD

- Critical business applications with sustained IOPS performance
- Or applications that need more than 16,000 IOPS
- Great for **databases workloads** (sensitive to storage perf and consistency)
- io1/io2 (4 GiB - 16 TiB):
  - Max PIOPS: 64,000 for Nitro EC2 instances & 32,000 for other
  - Can increase PIOPS independently from storage size
  - io2 have more durability and more IOPS per GiB (at the same price as io1)
- io2 Block Express (4 GiB – 64 TiB):
  - Sub-millisecond latency
  - Max PIOPS: 256,000 with an IOPS:GiB ratio of 1,000:1
- Supports EBS Multi-attach

EBS volume types Usage Case-Hard Disk Drive (HDD)

- Cannot be a boot volume
- 125 GiB to 16 TiB
- Throughput Optimized HDD (st1)
  - Big Data, Data Warehouses, Log Processing
  - **Max throughput** 500 MiB/s – max IOPS 500
- Cold HDD (sc1):
  - For data that is infrequently accessed
  - Scenarios where lowest cost is important
  - **Max throughput** 250 MiB/s – max IOPS 250

EBS -Volume Types Summary

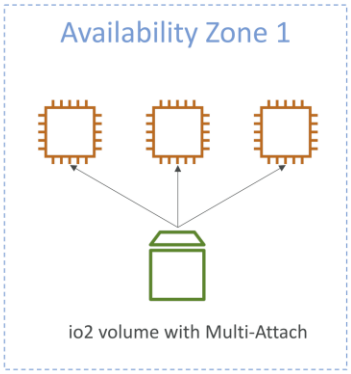
	General Purpose SSD		Provisioned IOPS SSD		
Volume type	gp3	gp2	io2 Block Express ‡	io2	io1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.999% durability (0.001% annual failure rate)		99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	• Low-latency interactive apps • Development and test environments		Workloads that require sub-millisecond latency, and sustained IOPS performance or more than 64,000 IOPS or 1,000 MiB/s of throughput	• Workloads that require sustained IOPS performance or more than 16,000 IOPS • I/O-intensive database workloads	
Volume size	1 GiB - 16 TiB		4 GiB - 64 TiB	4 GiB - 16 TiB	
Max IOPS per volume (16 KiB I/O)	16,000		256,000	64,000 †	

	Throughput Optimized HDD	Cold HDD
Volume type	st1	sc1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	• Big data • Data warehouses • Log processing	• Throughput-oriented storage for data that is infrequently accessed • Scenarios where the lowest storage cost is important
Volume size	125 GiB - 16 TiB	125 GiB - 16 TiB
Max IOPS per volume (1 MiB I/O)	500	250
Max throughput per volume	500 MiB/s	250 MiB/s
Amazon EBS Multi-attach	Not supported	Not supported
Boot volume	Not supported	Not supported

For more details see the blow URL
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html#solid-state-drives

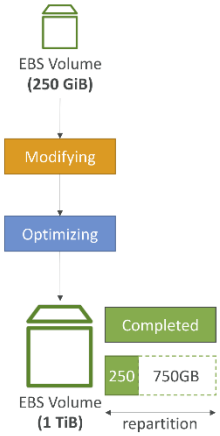
EBS Multi Attach – io I/io2 family:

- Attach the same EBS volume to multiple EC2 instances in the same AZ
- Each instance has full read & write permissions to the high-performance volume
- Use case:
  - Achieve **higher application availability** in clustered Linux applications (ex: Teradata)
  - Applications must manage concurrent write operations
- Up to 16 EC2 Instances at a time
- Must use a file system that's cluster-aware (not XFS, EXT4, etc…)



EBS Volume Resizing

- You can only increase the EBS volumes:
  - Size (any volume type)
  - IOPS (only in IO1)
- After resizing an EBS volume, you need to repartition your drive
- After increasing the size, it's possible for the volume to be in a long time in the "optimisation" phase. The volume is still usable
- You can't decrease the size of your EBS volume (create another smaller volume then migrate data)



EBS Encryption

- When you create an encrypted EBS volume, you get the following:
  - Data at rest is encrypted inside the volume
  - All the data in flight moving between the instance and the volume is encrypted
  - All snapshots are encrypted
  - All volumes created from the snapshot
- Encryption and decryption are handled transparently (you have nothing to do)
- Encryption has a minimal impact on latency
- EBS Encryption leverages keys from KMS (AES-256)
- Copying an unencrypted snapshot allows encryption
- Snapshots of encrypted volumes are encrypted

Encryption: encrypt an unencrypted EBS volume

- Create an EBS snapshot of the volume
- Encrypt the EBS snapshot (using copy)
- Create new ebs volume from the snapshot (the volume will also be encrypted)
- Now you can attach the encrypted volume to the original instance

EBS volume Types Usages

	Solid State Drives (SSD)		Hard Disk Drives (HDD)		
Volume Type	General Purpose	Provisioned IOPS SSD	Throughput Optimized HDD	Cold HDD	EBS Magnetic
API Names	**gp2**	**io1**	**st1**	**sc1**	**standard**
Description	Balances price and performance	Highest SSD performance for Mission-critical low latency or high throughput	Low-cost. Designed for frequently accessed, throughput intensive workloads	Lowest HDD cost. Less frequently used workloads	
Use Cases	Most Workloads	Large Databases IOPS greater than 16,000 or Throughput greater than 250 MiB	Data Warehouses Big Data Log Processing	File Storage	Archival Storage
Volume Size	1GiB - 16TiB	4GB - 16 TiB	500GiB - 15TiB	500GiB - 15TiB	500GiB - 15TiB
Max IOPS	16,000	64,000	500	250	40-200

1. **General Purpose (SSD)** (gp2) for general usage without specific requirements
2. **Provisioned IOPS (SSD)** (io1) when you require really fast input & output
3. **Throughput Optimized HDD** (st1) magnetic drive optimised for quick throughput
4. **Cold HDD** (sc1) Lowest cost HDD volume for infrequently access workloads
5. **EBS Magnetic** (standard) previous generation HDD

LAB:

Create an ec2 instance.

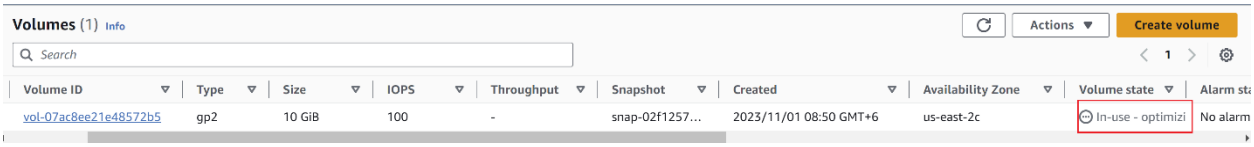
Login that instance and check the hard disk size and partition

#lsblk
#df -hT

```
[ec2-user@ip-172-31-47-57 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
[ec2-user@ip-172-31-47-57 ~]$ df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  468M     0  468M   0% /dev
tmpfs          tmpfs     477M     0  477M   0% /dev/shm
tmpfs          tmpfs     477M  472K  476M   1% /run
tmpfs          tmpfs     477M     0  477M   0% /sys/fs/cgroup
/dev/xvda1     xfs       8.0G  1.7G  6.4G  21% /
tmpfs          tmpfs      96M     0   96M   0% /run/user/1000
```

disk size
partition size

Now Increate the hard disk size from 8 to 10 from aws website and after done this it will take some time and the screen as shown as like below



Now we can see the hard disk has increased but the partition size is not increased.

```
[ec2-user@ip-172-31-47-57 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  10G  0 disk
└─xvda1 202:1    0   8G  0 part /
[ec2-user@ip-172-31-47-57 ~]$ df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  468M     0  468M   0% /dev
tmpfs          tmpfs     477M     0  477M   0% /dev/shm
tmpfs          tmpfs     477M  472K  476M   1% /run
tmpfs          tmpfs     477M     0  477M   0% /sys/fs/cgroup
/dev/xvda1     xfs       8.0G  1.7G  6.4G  21% /
tmpfs          tmpfs      96M     0   96M   0% /run/user/1000
[ec2-user@ip-172-31-47-57 ~]$
```

If we restart the server then it will be resized automatically.

Or

we do the following steps by without restarting the server.

First increate the partition size

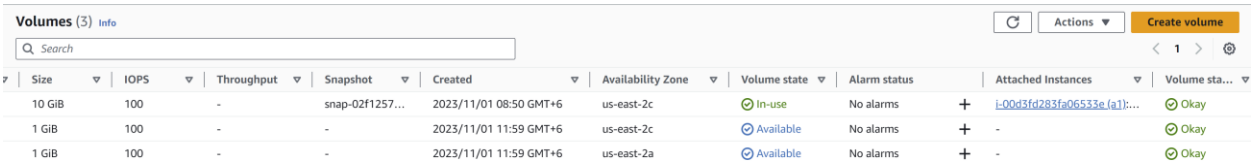
growpart /dev/xvda 1 =if the file system is xfs

Second resize the file system

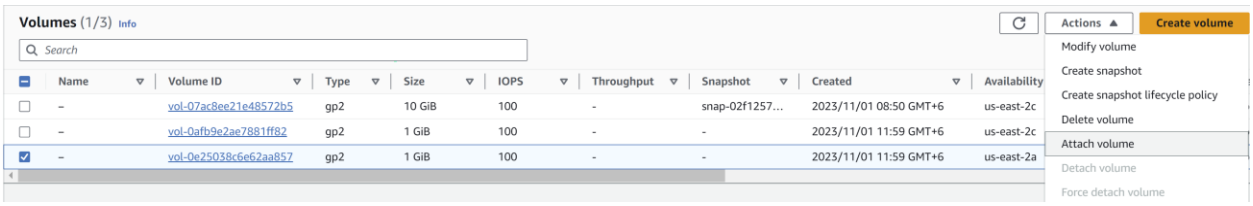
xfs_growfs /dev/xvda1

```
[root@ip-172-31-47-57 ec2-user]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  468M     0  468M   0% /dev
tmpfs          tmpfs     477M     0  477M   0% /dev/shm
tmpfs          tmpfs     477M  536K  476M   1% /run
tmpfs          tmpfs     477M     0  477M   0% /sys/fs/cgroup
/dev/xvda1     xfs       8.0G  1.7G  6.4G  21% /
tmpfs          tmpfs      96M     0   96M   0% /run/user/1000
[root@ip-172-31-47-57 ec2-user]# xfs_growfs /dev/xvda1
meta-data=/dev/xvda1             isize=512    agcount=4, agsize=524159 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=0, rmapbt=0
         =                       reflink=0    bigtime=0 inobtcount=0
data     =                       bsize=4096   blocks=2096635, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
data blocks changed from 2096635 to 2620923
[root@ip-172-31-47-57 ec2-user]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  468M     0  468M   0% /dev
tmpfs          tmpfs     477M     0  477M   0% /dev/shm
tmpfs          tmpfs     477M  536K  476M   1% /run
tmpfs          tmpfs     477M     0  477M   0% /sys/fs/cgroup
/dev/xvda1     xfs        10G  1.7G  8.4G  17% /
tmpfs          tmpfs      96M     0   96M   0% /run/user/1000
```

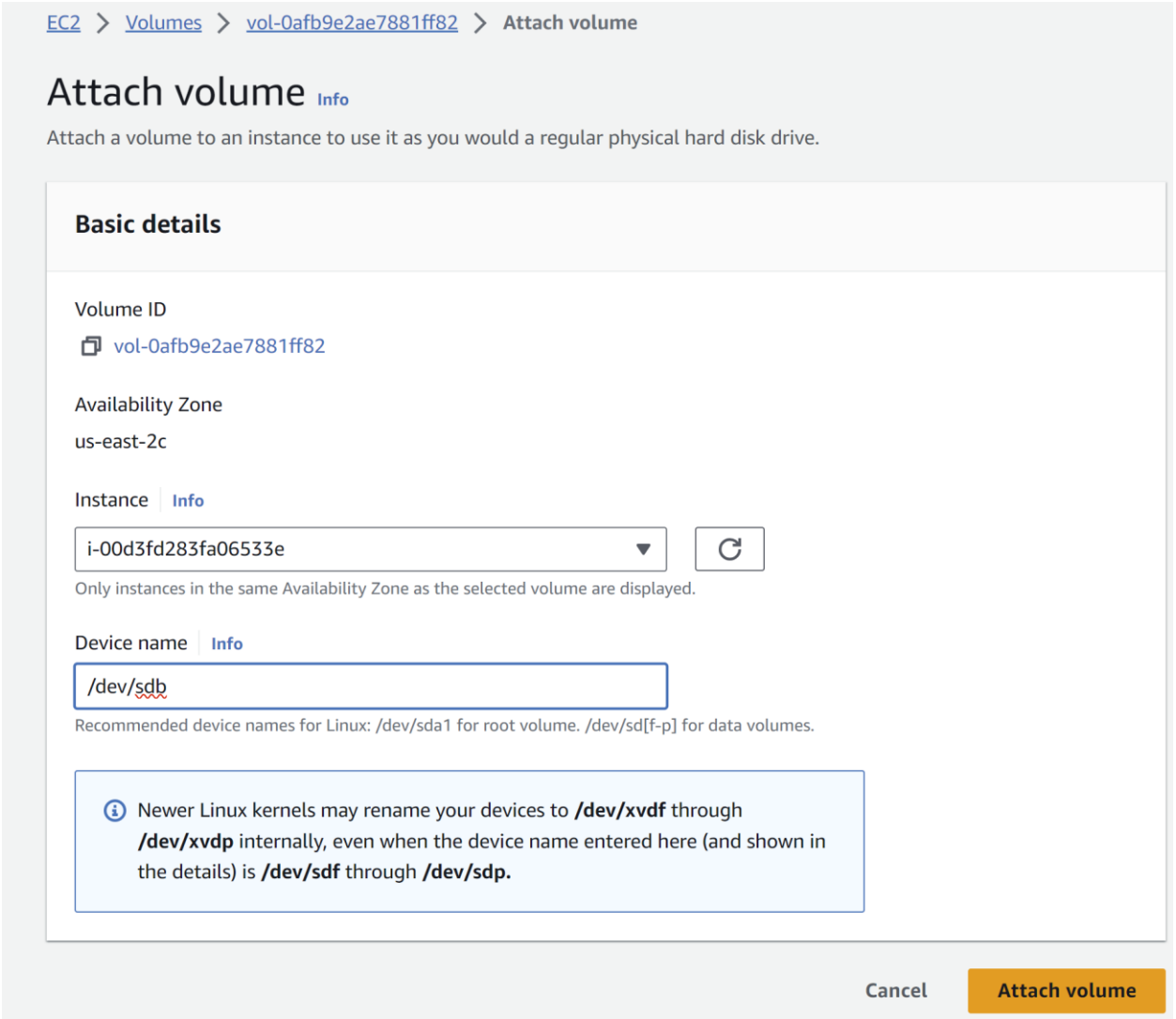
Now create a another volume and attached it to our Ec2 make sure we create the volume in same availability zone where our Ec2 is available.



Select the volume>Actions>Attach volume



Now select your ec2 instance and give a device name here I use /dev/sdb



Now go to the command line inter face

#lsblk

```
[root@ip-172-31-47-57 ec2-user]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  10G  0 disk
└─xvda1 202:1    0  10G  0 part /
xvdb    202:16   0   1G  0 disk
[root@ip-172-31-47-57 ec2-user]#
```

now we do the partition by using the following command

#fdisk /dev/xvdb

n>> p >> enter >> enter >> enter >> w

```
[root@ip-172-31-47-57 ec2-user]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  10G  0 disk
└─xvda1 202:1    0  10G  0 part /
xvdb    202:16   0   1G  0 disk
[root@ip-172-31-47-57 ec2-user]# fdisk /dev/xvdb

Welcome to fdisk (util-linux 2.30.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x9abc9446.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-2097151, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-2097151, default 2097151):

Created a new partition 1 of type 'Linux' and of size 1023 MiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

[root@ip-172-31-47-57 ec2-user]# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   10G  0 disk
└─xvda1 202:1    0   10G  0 part /
xvdb    202:16   0    1G  0 disk
└─xvdb1 202:17   0 1023M  0 part
[root@ip-172-31-47-57 ec2-user]#
```

now format this partition
#mkfs.ext4 /dev/xvdb1

check how many way we can mount
#blkid

Create a folder where we mount the new harddisk
#mkdir /oracle

Mount the drive after restart also keep mounted
vi /etc/fstab

write the line in fstab file
/dev/xvdb1 /oracle ext4 defaults 0 0

now run the command for mount
#mount -av =fstab e entry ase mount kora nai sai gula mount hobe

Check the harddisk
#df -hT

Now create a file in oracle directory and write something
#sudo vi /oracle/a.txt

Now increase the hard disk size form web and make it 2
#resize2fs /dev/xvdb1 =if the file system is ext4
#df -hT

Lab: 2

We create a snapshot from the newly created volume
copy the volume to another AZ
crate a volume from that snapshot
crate a ec2 to the differenct az
now attach the volume to the ec2
mount the volume and now use manual command
check the old file is available or not
create a file to this volume and make sure everything is working fine
after done the lab delete all resources we used

Some few command we use for this lab
#growpart /dev/xvdb 1 =extend the partition size
#lsblk =check the hard disk partition
#df -hT =Check file system and use and free space
#df -hT /oracle =check the /orcale file system
#resize2fs /dev/xvdb1 = resize the file system if file system is ext4
#mkdir /oracle =create a directory

#cd /oracle	=get in to the directory
#cat data.txt	=view the file content
#xfs_growfs if the file system is xfs	=resize the file system if file system is xfs
#mount /dev/xvdb1 /oracle	=mount a hdd to /oracle directory
#umount /dev/xvdb1	=unmount the /dev/xvdb1