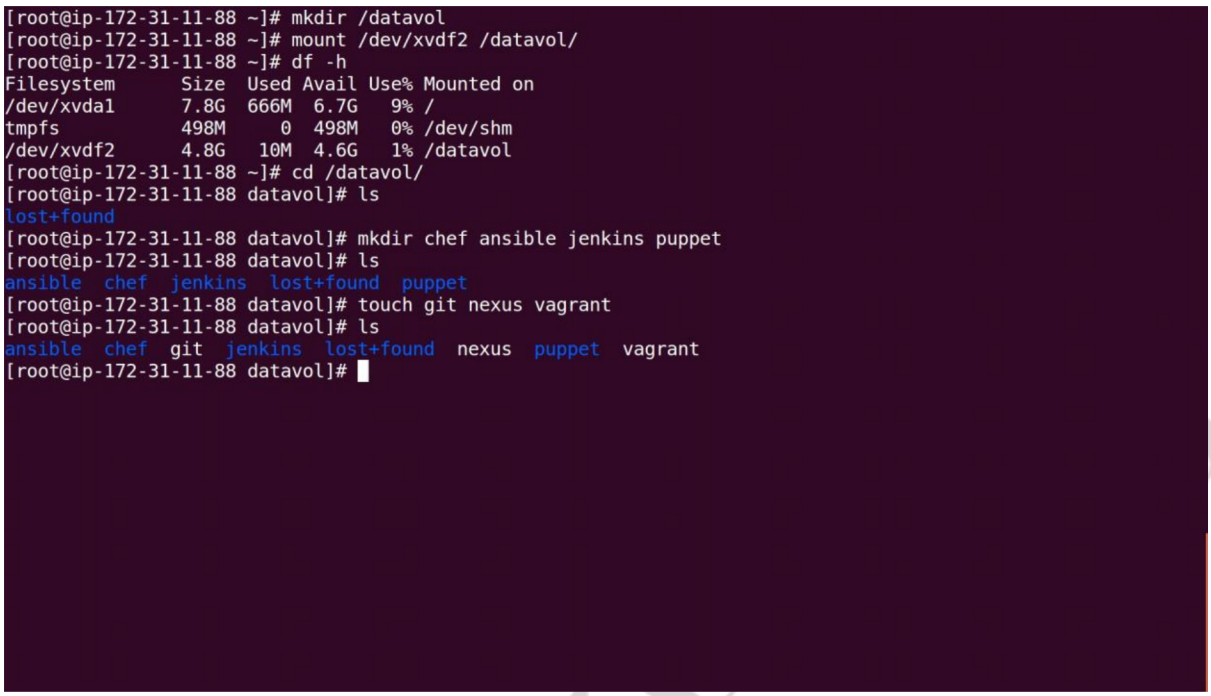
```
[root@ip-172-31-11-88 ~]# fdisk /dev/xvdf
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0xc5bae138.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4):
Value out of range.
Partition number (1-4): 2
First cylinder (1-652, default 1):
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-652, default 652):
Using default value 652

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@ip-172-31-11-88 ~]#
```

- After creating the partition, we have to format it with a filesystem of your choice. Here I have chosen ext4 filesystem.

```
[root@ip-172-31-11-88 ~]# fdisk -l

Disk /dev/xvda: 8589 MB, 8589934592 bytes
255 heads, 63 sectors/track, 1044 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00057cbb

    Device Boot      Start         End      Blocks   Id  System
/dev/xvda1   *           1        1045     8387584   83  Linux

Disk /dev/xvdf: 5368 MB, 5368709120 bytes
255 heads, 63 sectors/track, 652 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xc5bae138

    Device Boot      Start         End      Blocks   Id  System
/dev/xvdf2               1         652     5237158+  83  Linux
[root@ip-172-31-11-88 ~]# mkfs.ext4 /dev/xvdf2
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
327680 inodes, 1309289 blocks
65464 blocks (5.00%) reserved for the super user
First data block=0
```

- Once formatting is done, we can mount that partition to a directory. Create a directory and mount it to a volume as shown below:



Create some directories and files in that directory which is mounted to the volume.

Backup & Restore

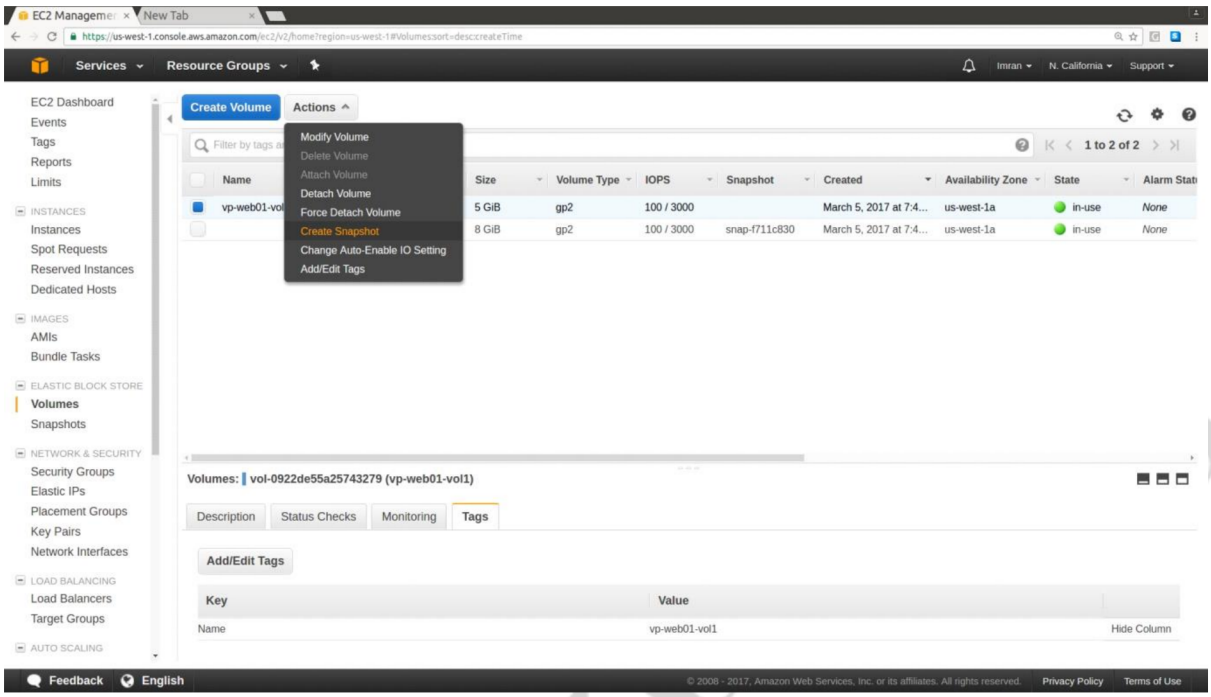
We will create a situation where we need to restore the lost data.
We will delete few files from the mount point after taking the backup and then restore the deleted data.

Backup the EBS volume by taking its snapshot:

Amazon EBS provides the ability to save point-in-time snapshots of your volumes to Amazon S3. Amazon EBS Snapshots are stored incrementally: only the blocks that have changed after your last snapshot are saved, and you are billed only for the changed blocks. If you have a device with 100 GB of data but only 5 GB has changed after your last snapshot, a subsequent snapshot consumes only 5 additional GB and you are billed only for the additional 5 GB of snapshot storage, even though both the earlier and later snapshots appear complete.
When you delete a snapshot, you remove only the data not needed by any other snapshot. All active snapshots contain all the information needed to restore the volume to the instant at which that snapshot was taken. The time to restore changed data to the working volume is the same for all snapshots.
Snapshots can be used to instantiate multiple new volumes, expand the size of a volume, or move volumes across Availability Zones. When a new volume is created, you may choose to create it based on an existing Amazon EBS snapshot. In that scenario, the new volume begins as an exact replica of the snapshot.
The following are key features of Amazon EBS Snapshots:

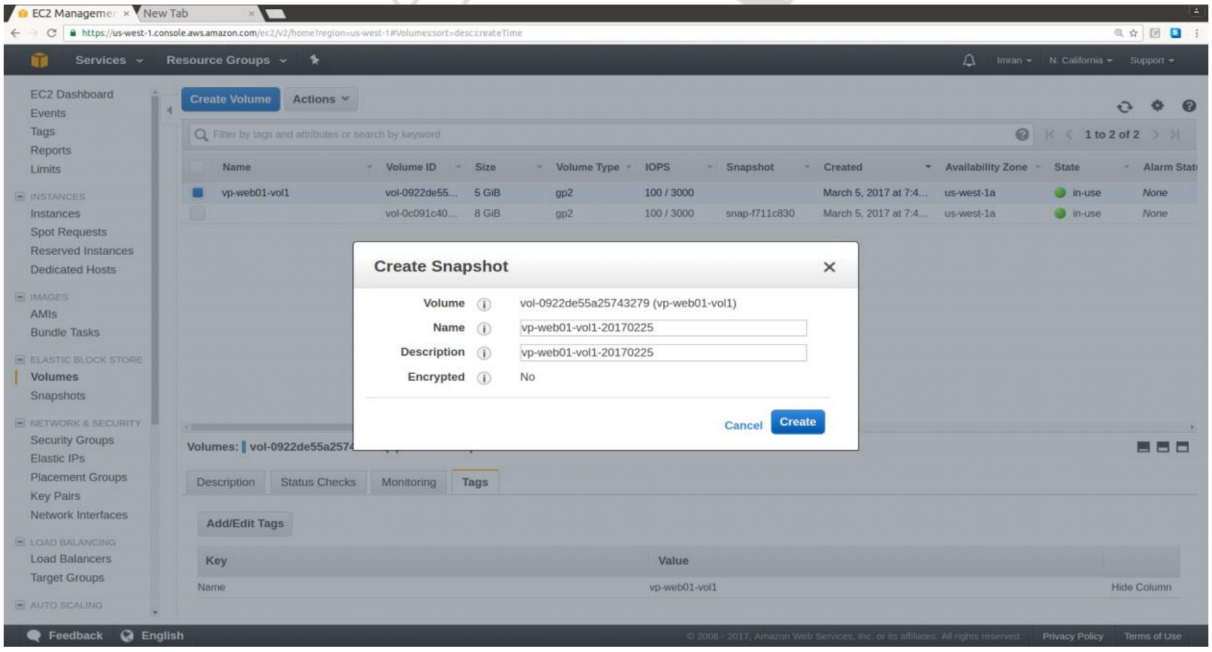
- Immediate access to Amazon EBS volume data
- Resizing Amazon EBS volumes
- Sharing Amazon EBS Snapshots
- Copying Amazon EBS Snapshots across AWS regions

To create a snapshot from the AWS Management Console, go to the Volume Management Dashboard, Select the Volume, Click on Actions and Click on Create Snapshot.

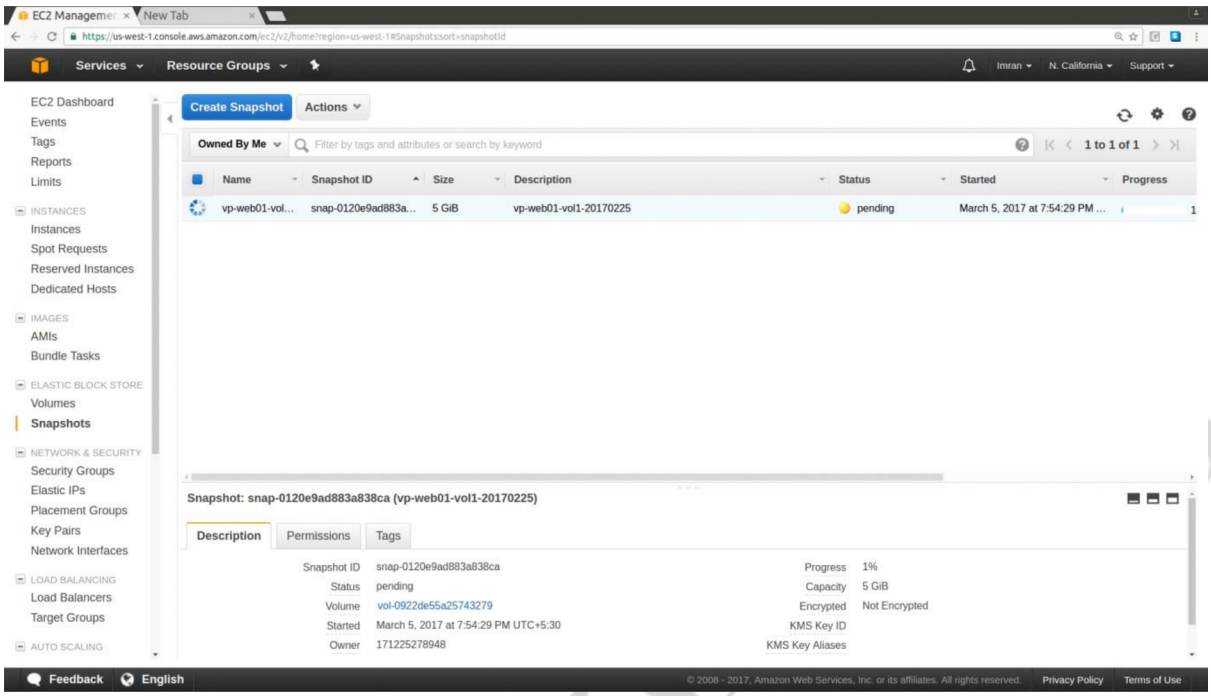


Note: It's a good practice to stop your instance before taking a snapshot if you are taking a snapshot of its root volume.

You will see Create Snapshot dialog box as shown in the following screenshot. Provide Suitable Name and Description for your new Snapshot. Here there is no Encryption for your snapshot because the Volume is not encrypted. Snapshots of encrypted volumes will be encrypted automatically.



- After filling the required details Click on Create to complete the snapshot process. The process will take few minutes to complete.



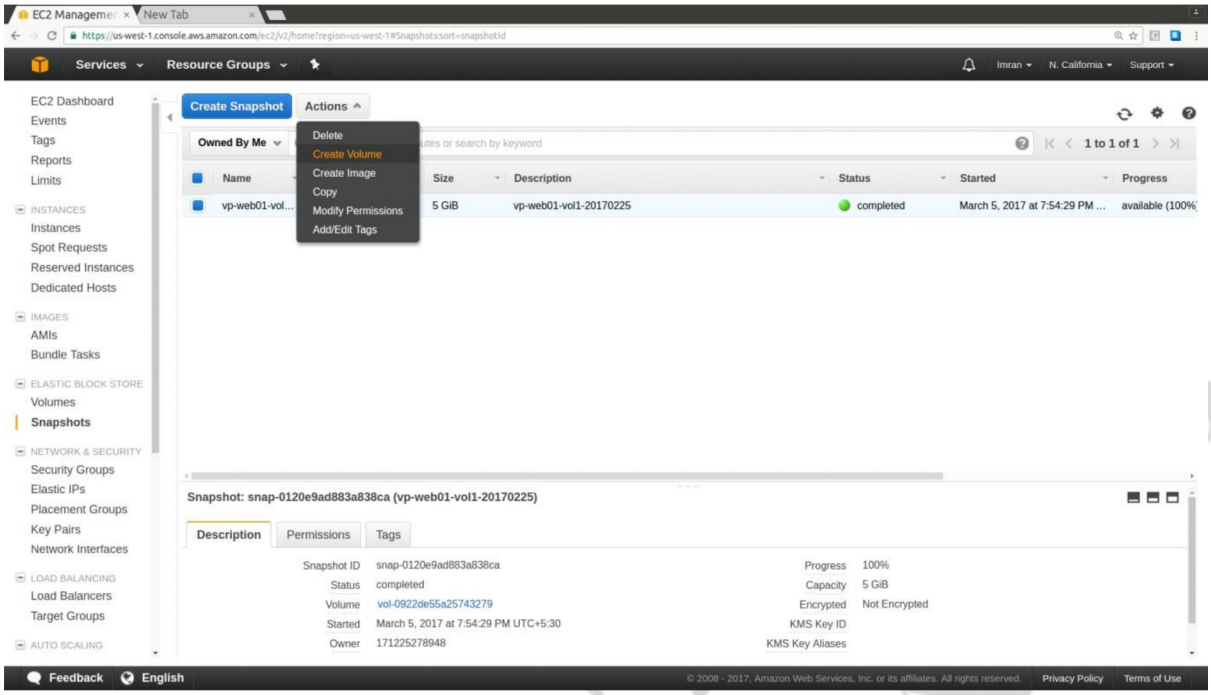
- After creating the snapshot, we will delete few files from the mount point.

```
[root@ip-172-31-11-88 ~]# cd /datavol/
[root@ip-172-31-11-88 datavol]# ls
ansible  chef  git  jenkins  lost+found  nexus  puppet  vagrant
[root@ip-172-31-11-88 datavol]# rm -rf ansible/ chef/ jenkins/ vagrant
[root@ip-172-31-11-88 datavol]# ls
git  lost+found  nexus  puppet
[root@ip-172-31-11-88 datavol]# cd
[root@ip-172-31-11-88 ~]# clear
```

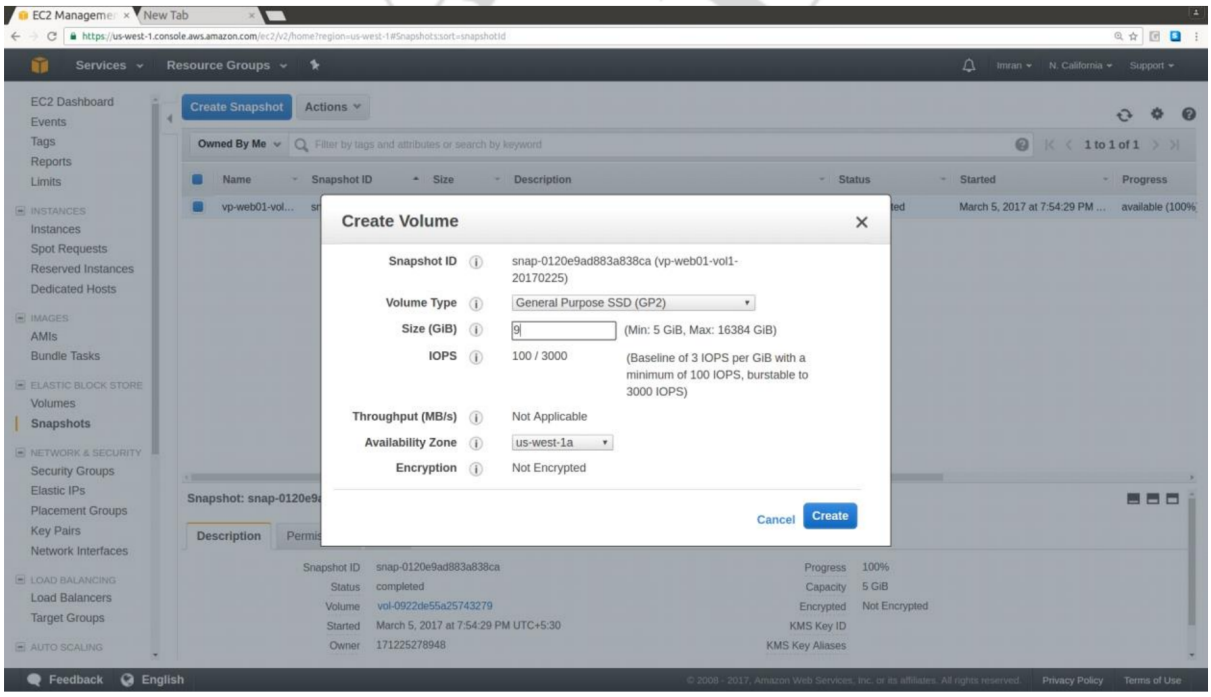
Create Volume from Snapshot & Resize:

Now if we want to recover the lost data, we have to create a new volume from snapshot & replace old volume with the new one. We will also increase the volume size.

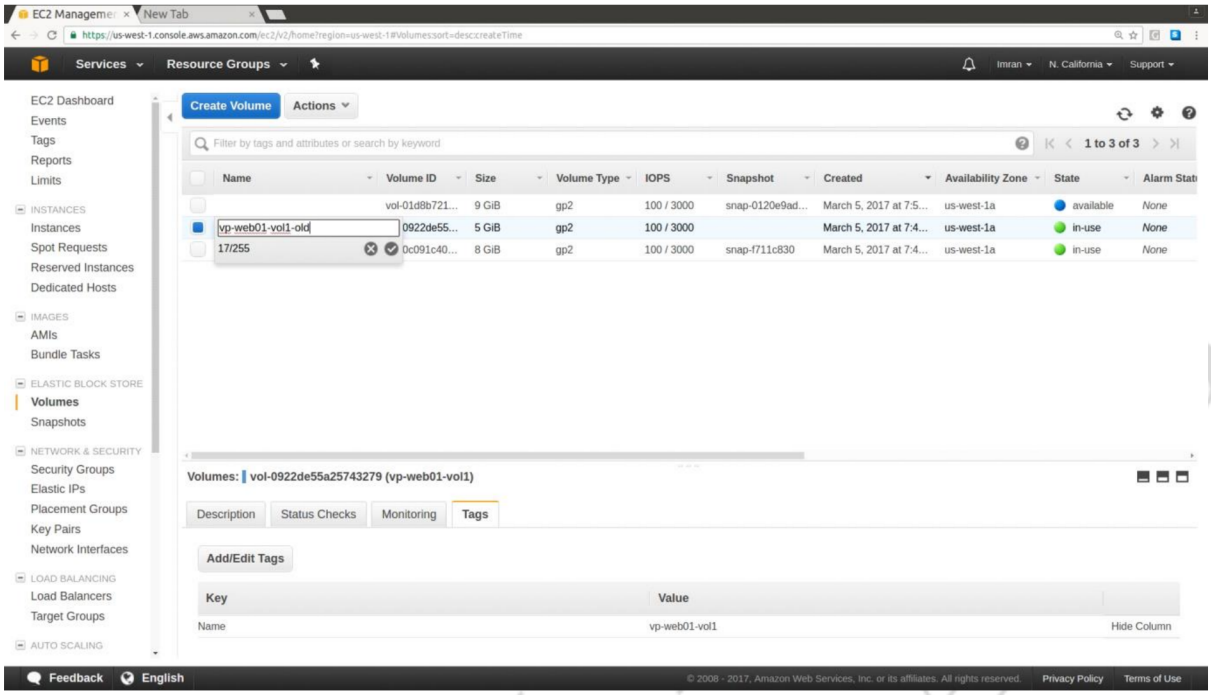
Click Snapshots in left pane of EC2 Dashboard, Select the snapshot from which new volume has to be created, Click on Actions and select Create volume, you will get a create Volume pop up dialog box as shown below:



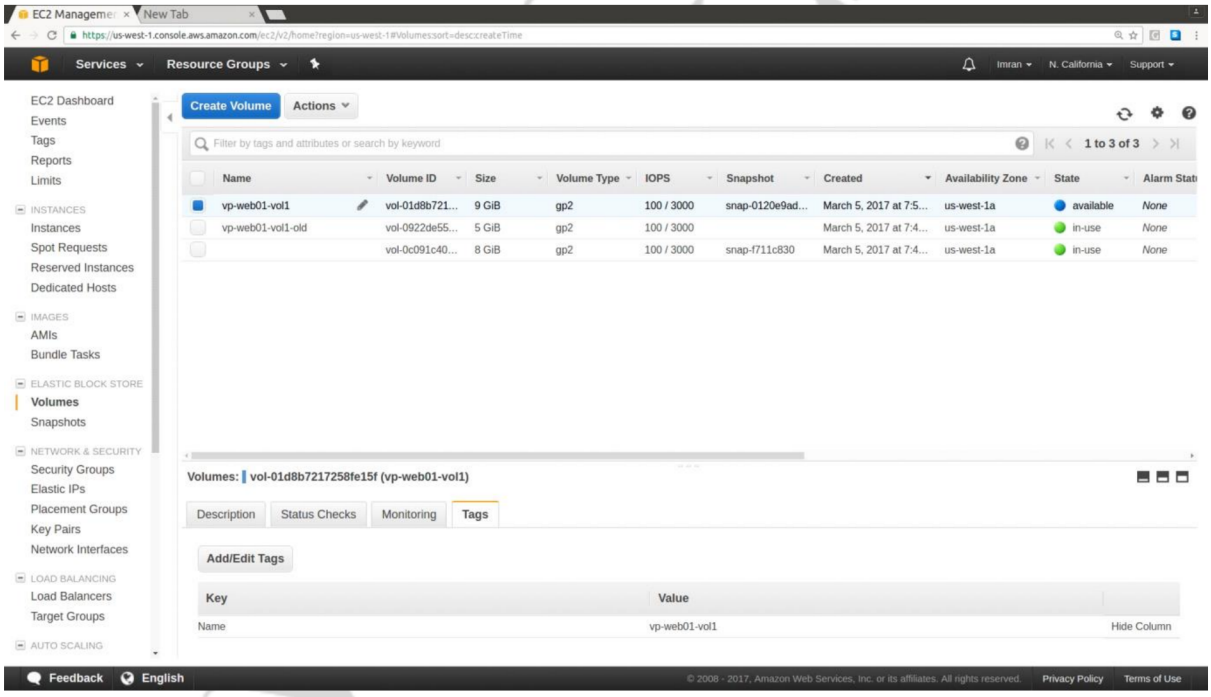
- Fill the details to create new volume. Here the new volume size must be greater than 5GB because we created this snapshot from 5GB volume which is attached to the instance. And the Availability Zone should be same where the instance is created. Once this is done Click on Create to complete the process.



- Tag the old volume with -old extension for identification and tag new volume with other name as shown below:

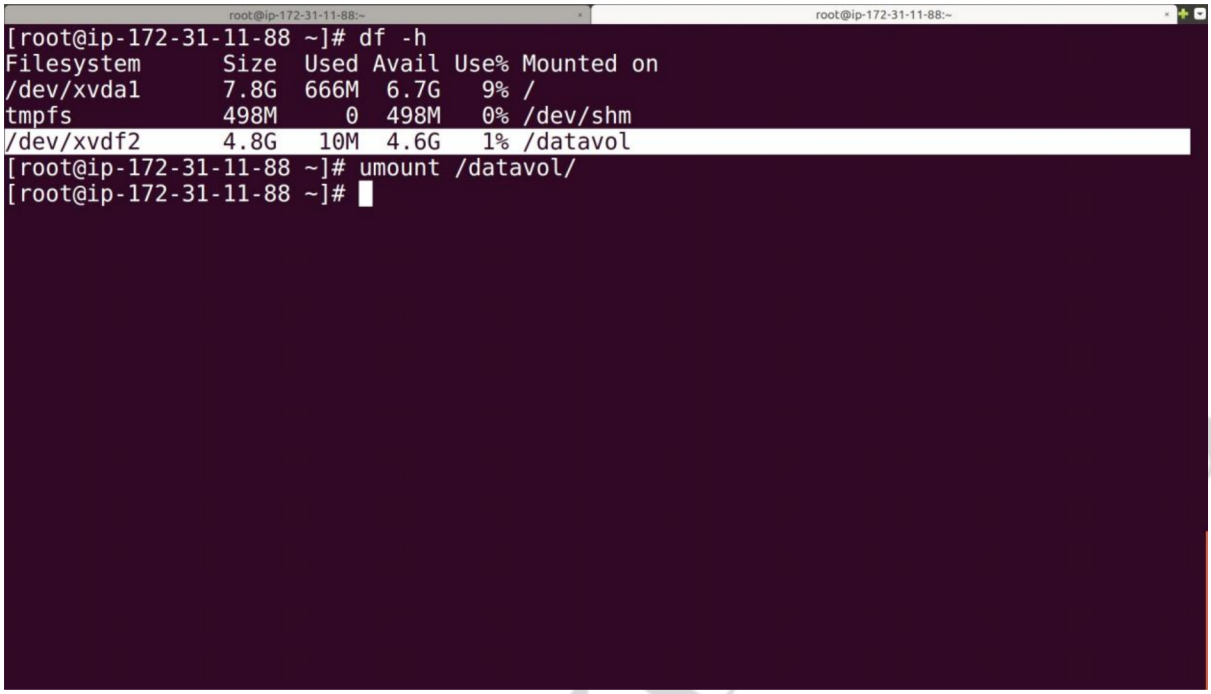


❖ Once the volume is created it is available for use.

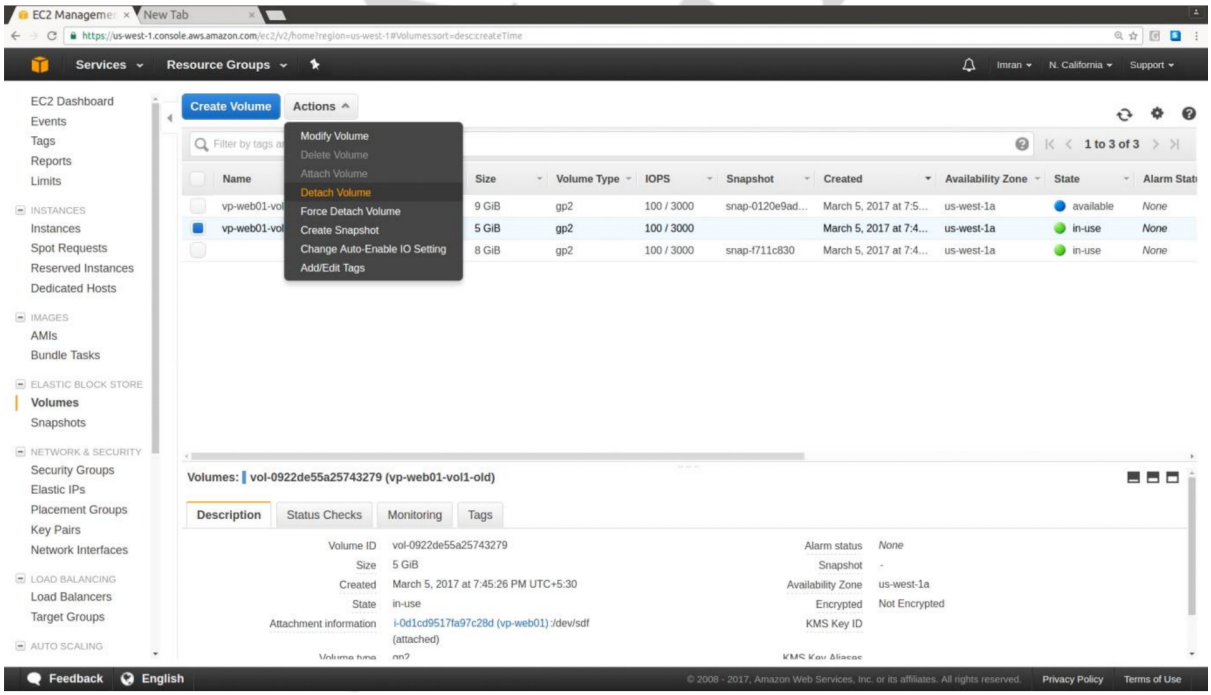


Unmount & detach Old Volume:

To unmount and detach old volume from the instance first check the mount points in the created partition and then unmount the mounted directory as shown below:

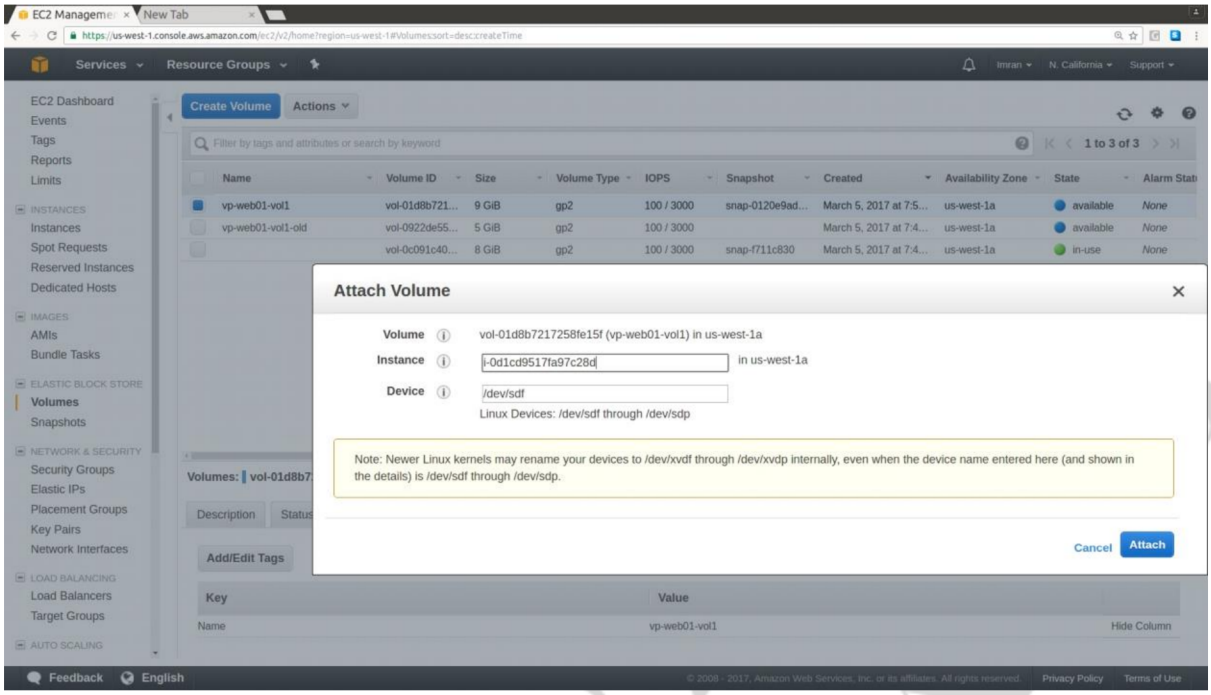


- Once we are done with unmounting then detach the old volume from the instance. Select the volume to be detached, click on actions and select detach volume as shown below:

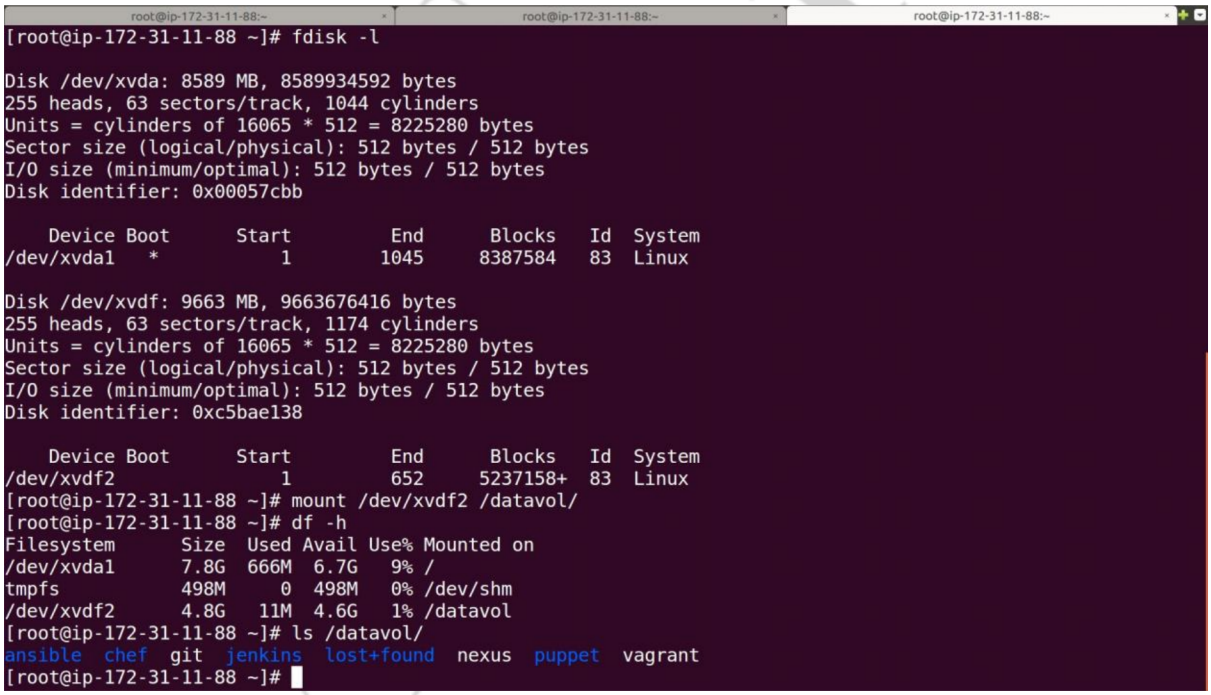


Attach and mount new volume.

Finally attach the new volume to the instance by selecting the new volume, Click on Actions and select Attach Volume. We will get Attach Volume pop up dialog box as shown below:
Provide the instance ID device partition.



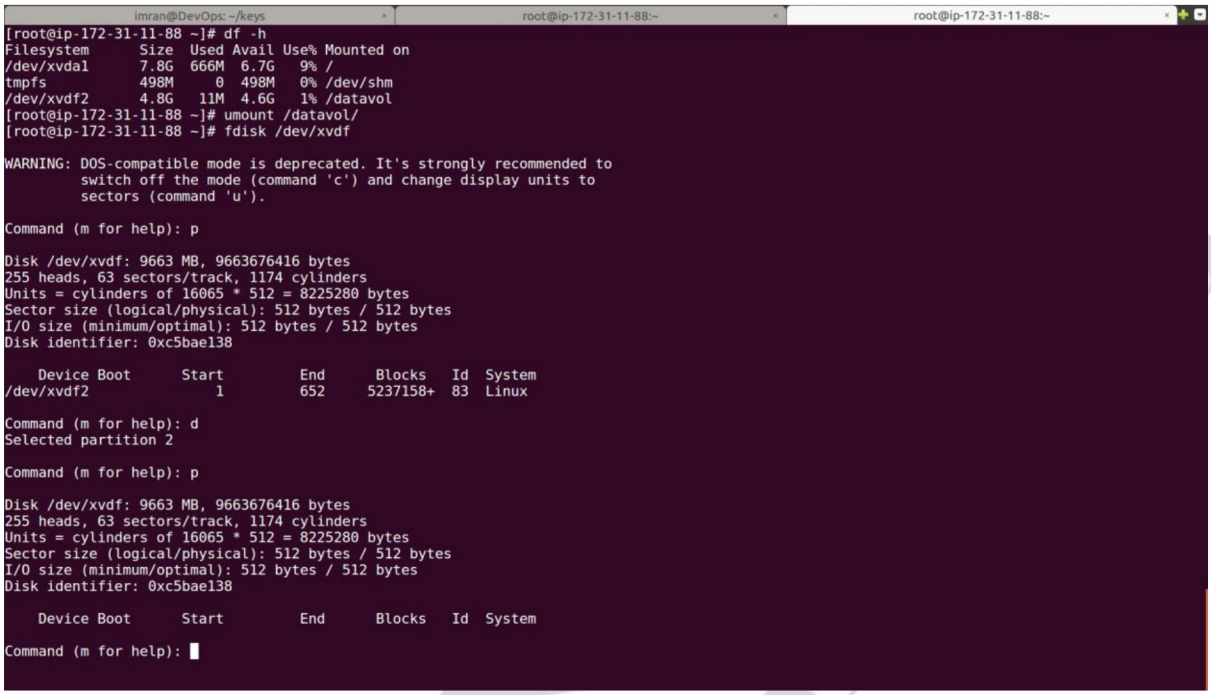
❖ After attaching the volume mount, it to the instance as we done before.

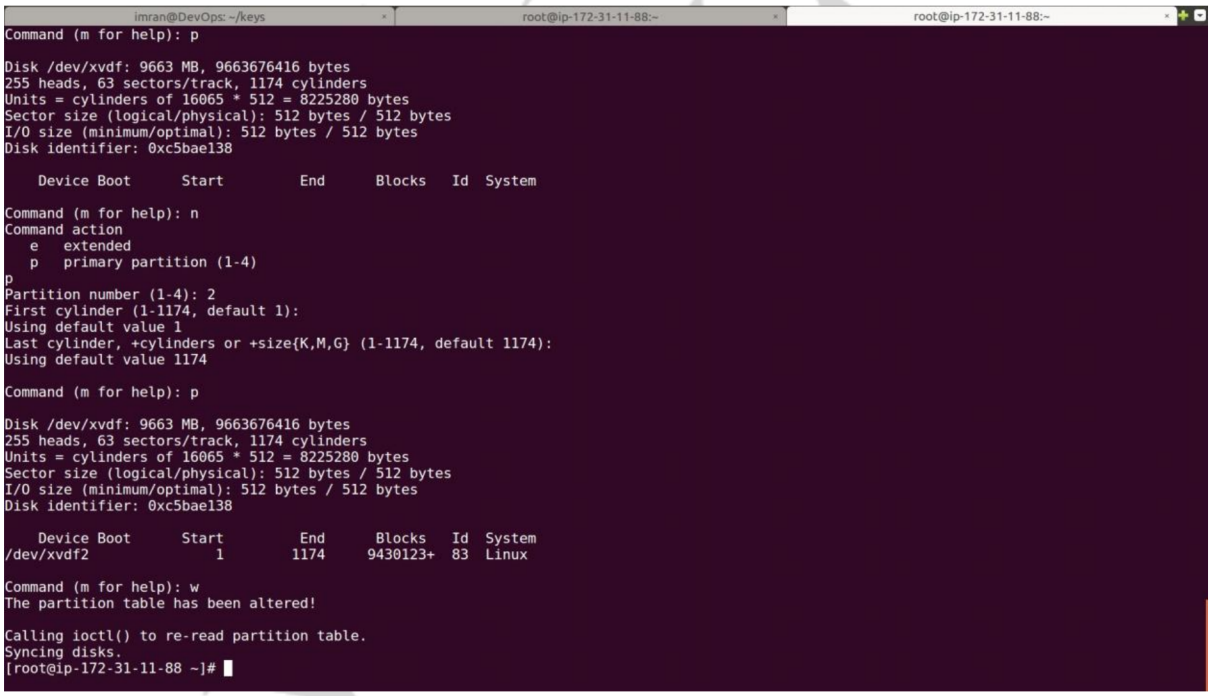


In the above screenshot, we can see that the data is restored.
But the mount point size is still around 5 GB even though we changed the EBS volume size to 9 GB.
This is because the rest of the volume is still not resized and we need to resize it.

Resizing the increased volume:

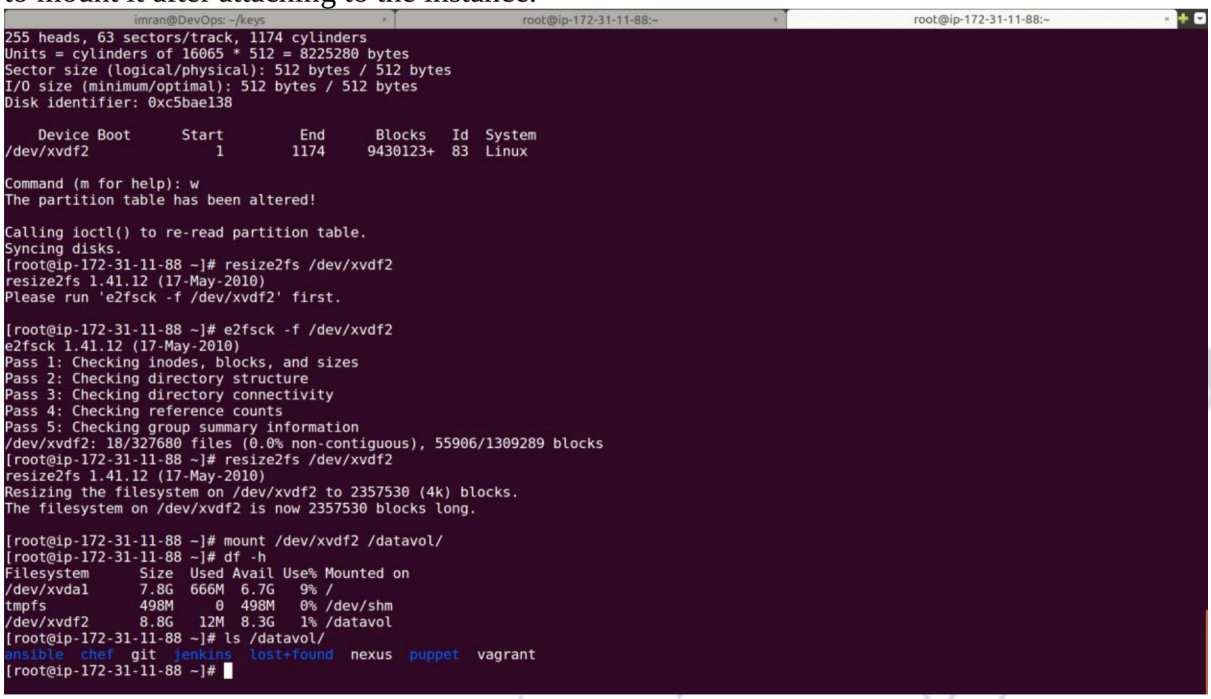
We have already mounted the new volume but to resize it we need to unmount, delete partition, recreate it with new size, resize it and mount it. Follow the same steps to unmount, delete and create new partition as shown below:





Resize the Volume: In order to resize the volume, we need to run the command
resize2fs /dev/xvdf2
(If we run this command it gives an error to run another command because new volume is greater than old volume)
Follow the steps as shown in the below screenshot to resize the volume.

Note: If the new volume is also of same size as old volume then no need of resizing it, just we have to mount it after attaching to the instance.



Now the Volume size is increased and data is also restored.

10. AWS VPC

Virtual private cloud (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can configure your VPC; you can select its IP address range, create subnets, and configure route tables, network gateways, and security setting.

Subnet is a range of IP addresses in your VPC. You can launch AWS resources into a subnet that you select. Use a public subnet for resources that must be connected to the Internet, and a private subnet for resources that won't be connected to the Internet.

Private Subnet: A private subnet sets that route to a NAT instance. Private subnet instances only need a private IP and internet traffic is routed through the NAT in the public subnet. You could also have no route to 0.0.0.0/0 to make it a truly private subnet with no internet access in or out.

Public Subnet: A public subnet routes 0.0.0.0/0 through an Internet Gateway (igw). Instances in a public subnet require public IPs to talk to the internet.

Network Address Translation (NAT) gateway is used to enable instances in a private subnet to connect to the Internet or other AWS services, but prevent the Internet from initiating a connection with those instances.