www.linkedin.com/in/arun-kumar-akula

AWS EBS – Elastic Block Storage

EBS (Elastic Block Store) is a block-level storage service provided by AWS that you can use with EC2 instances. Think of it as a virtual hard disk you can attach and detach to your server (EC2) in the cloud.

- **Persistent Storage**: Your data remains even if the instance is stopped or terminated.
- **Attach/Detach**: You can attach EBS to any EC2 in the same Availability Zone.
- **Snapshot Support**: You can take backups (snapshots) and restore anytime.
- **Resize and Modify**: Change size, performance, and volume type even while in use.
- **Encryption**: Supports encryption for secure data.
- Pricing is based on storage you use.

Example Use Case:

You launch an EC2 instance but the default root disk**(8GB)** isn't enough. You create an EBS volume, attach it, format it, and now you have additional storage space available on your server.

Types of AWS EBS Volumes:

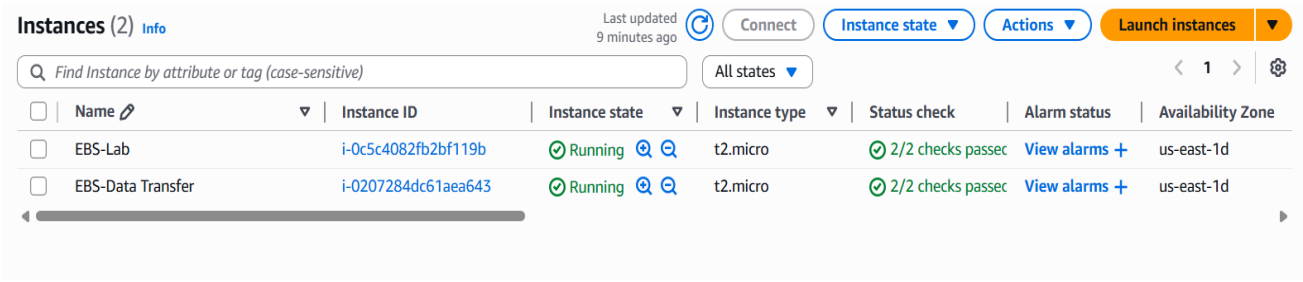
- **gp3** – General Purpose SSD (default, for most use cases like Web servers, dev/test, general use)
- **io2/io1** – High-performance SSD (for databases, high IOPS)
- **st1** – Throughput-optimized HDD (for big data, logs)
- **sc1** – Cold HDD (Lowest cost, for infrequently accessed data)

Now let's see how to practically attach an EBS volume to an EC2 instance and mount it

Step 1: Launch 3- EC2 Instances (2 Instances in same Availability Zones and 1 in different Az)

- Go to AWS Console → EC2 → **Launch Instance**
- Select AMI: Ubuntu or Amazon Linux
- Instance Type: t2.micro

- Key Pair: Create or use existing one
- Network default
- Add additional storage of 15 GiB to only one server.
- Click on create instance



Step2 : Connect to your EC2 instance and check if the additional EBS volume has been attached.

```
ubuntu@ip-172-31-87-171:~$ lsblk
NAME     MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0      7:0    0 27.2M  1 loop /snap/amazon-ssm-agent/11320
loop1      7:1    0 73.9M  1 loop /snap/core22/1981
loop2      7:2    0 50.9M  1 loop /snap/snapd/24505
xvda     202:0    0    8G  0 disk
├─xvda1  202:1    0    7G  0 part /
├─xvda14 202:14   0    4M  0 part
├─xvda15 202:15   0  106M  0 part /boot/efi
└─xvda16 259:0    0  913M  0 part /boot
xvdb     202:16   0   15G  0 disk
```

Step3 : Format the volume with ext4 filesystem (We format the EBS volume with ext4 so the operating system can store and manage files on it.)

Command to format : sudo mkfs -t ext4 /dev/xvdb

```
ubuntu@ip-172-31-87-171:~$ sudo mkfs -t ext4 /dev/xvdb
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 3932160 4k blocks and 983040 inodes
Filesystem UUID: 0c73c689-b595-45c3-b517-9494fcf720ea
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

Step4 : Create a mount directory (We create a mount directory to provide a location in the file system where the EBS volume can be accessed)

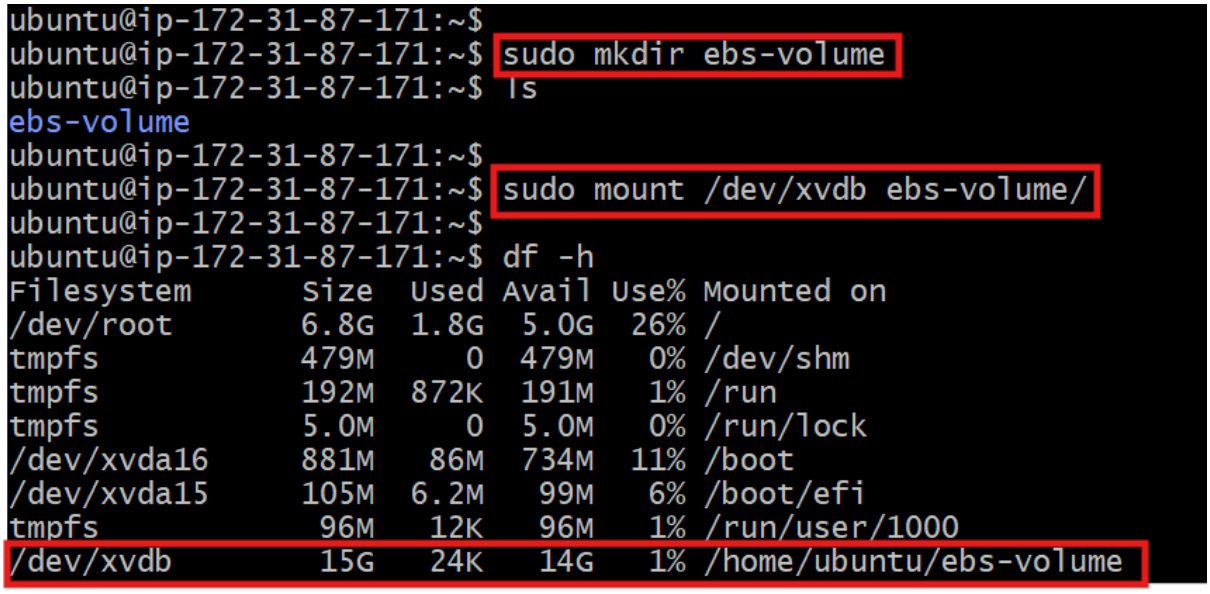
Command to create directory : sudo mkdir ebs-

Step5 : Mount the volume (We mount the volume to link the formatted EBS volume to the mount directory so we can read and write data to it)

Command to mount: sudo mount /dev/xvdb ebs-volume/

- Check whether the additional EBS volume is attached and properly mounted to the directory

Command to Check : df -h



Step 6 : Create a file to test **(**We create a file to test in order to verify that the EBS volume is writable and mounted correctly)

Command to create file : vi my-volume (add some data into the file)

- Check the contents of the file

Command to check : cat my-volume

```
ubuntu@ip-172-31-87-171:~/ebs-volume$
ubuntu@ip-172-31-87-171:~/ebs-volume$ vi My-volume
ubuntu@ip-172-31-87-171:~/ebs-volume$ sudo vi my-volume
ubuntu@ip-172-31-87-171:~/ebs-volume$
ubuntu@ip-172-31-87-171:~/ebs-volume$ cat my-volume
Hello I'm Arun Kumar Akula
working as a DevOps Engineer.!
```

As of now, we have created the instance, formatted, and mounted the storage. Now, we will proceed to reattach the existing EBS volume to another EC2 instance.

How to Reattach an Existing EBS Volume to Another EC2 Instance

we need to unmount from the old server before mounting to the new one because EBS volumes can only be attached to one EC2 instance at a time in read/write mode.

- If you don't unmount and detach properly AWS **won't allow** attaching the volume to another EC2 until it's detached and You **risk data corruption**.

Step 1: Unmount the volume.

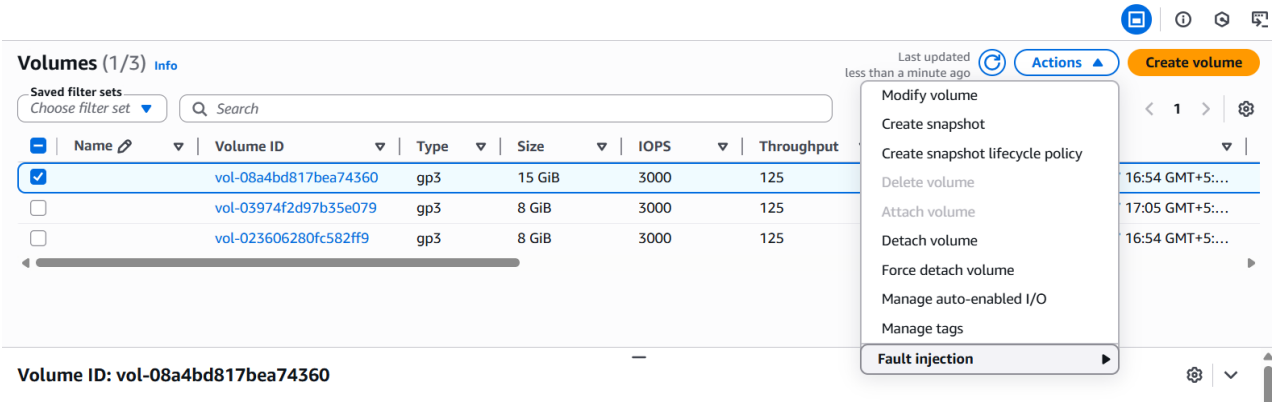
```
sudo umount  /dev/xvdb
```

```
ubuntu@ip-172-31-87-171:~$
ubuntu@ip-172-31-87-171:~$ sudo umount /dev/xvdb
ubuntu@ip-172-31-87-171:~$
ubuntu@ip-172-31-87-171:~$ lsblk
NAME     MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0      7:0    0 27.2M  1 loop /snap/amazon-ssm-agent/11320
loop1      7:1    0 73.9M  1 loop /snap/core22/1981
loop2      7:2    0 50.9M  1 loop /snap/snapd/24505
xvda     202:0    0    8G  0 disk
├─xvda1  202:1    0    7G  0 part /
├─xvda14 202:14   0    4M  0 part
├─xvda15 202:15   0  106M  0 part /boot/efi
└─xvda16 259:0    0  913M  0 part /boot
xvdb     202:16   0   15G  0 disk
```

Step 2: Detach the volume from Server 1

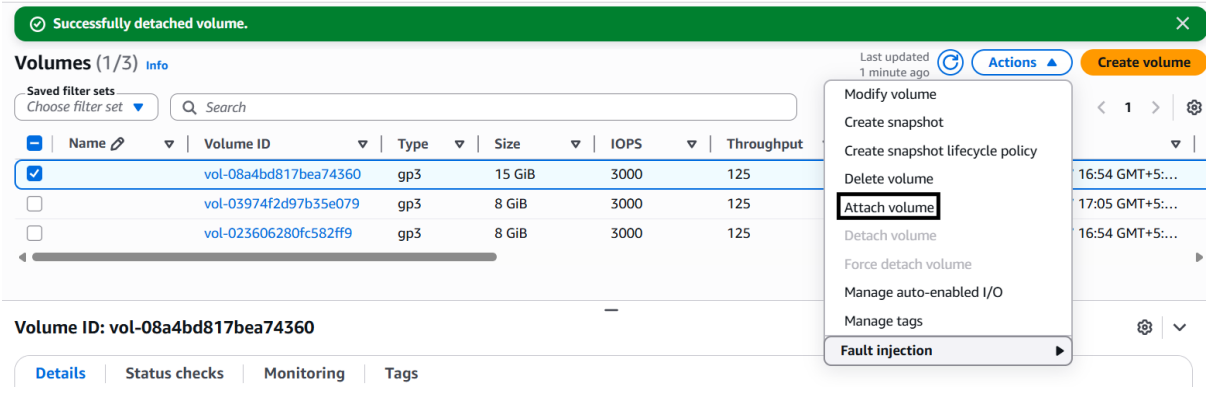
- Go to AWS Console → EC2 → Volumes
- Select the attached volume
- Click Actions → Detach Volume

- Wait until the volume state becomes "available"

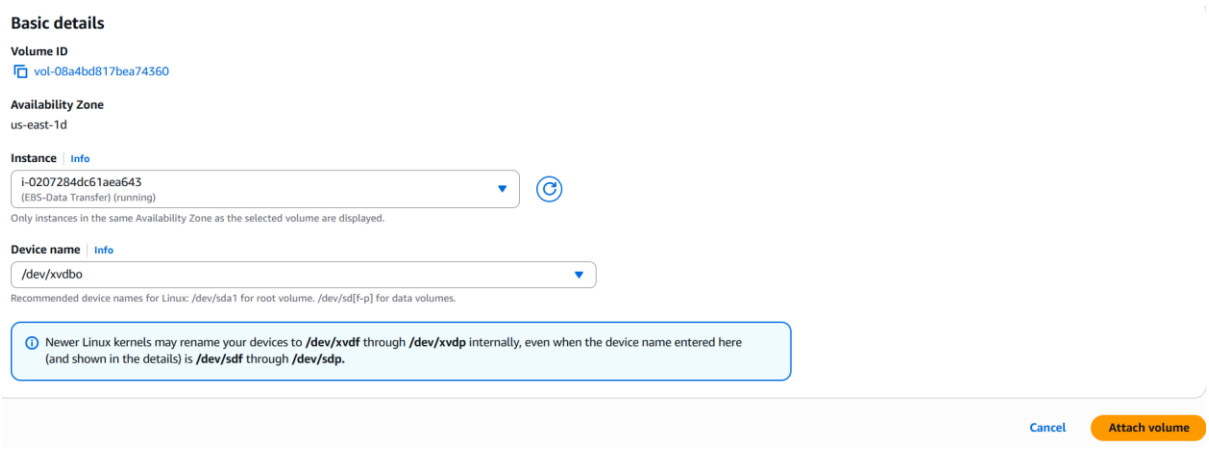


Step : Attach the Volume to Server 2

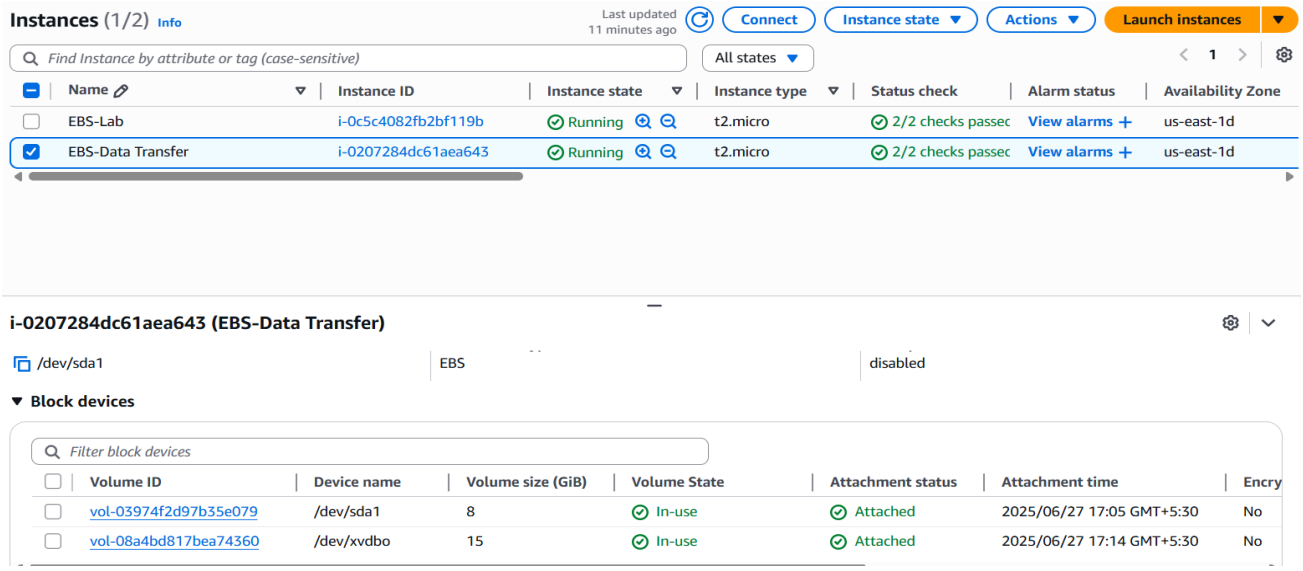
- Select the available volume
- Click Actions → Attach Volume



- Choose Server 2 from the instance list
- Set device name (e.g., /dev/xvdbo)



- Now you can see that the volume is attached to your EC2 instance.



Step 5: Connect to Server 2 using SSH

- Create mount directory

```
Mkdir ebs-volume-2
```

Step 6 : Mount the volume

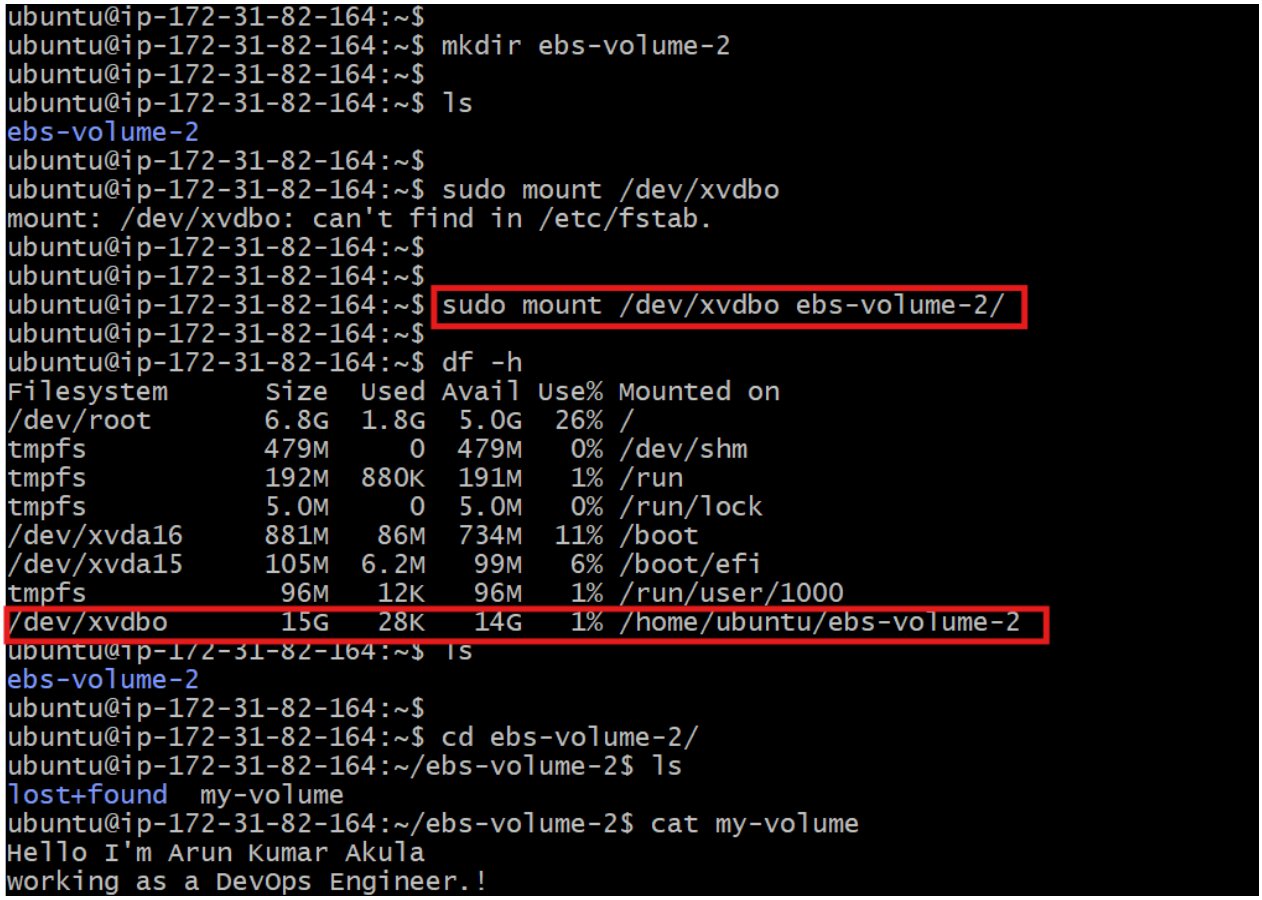
```
sudo mount /dev/xvdbo ebs-volume-2/
```

- Check whether the additional EBS volume is attached and properly mounted to the directory and after mounting, check the data created on Server 1 it should now be available on Server 2.

Command to check storage : lsblk or df -h

Command to data : cd ebs-volume-2

Command the data into the directory : cat my-volume



Now the volume is successfully mounted to Server 2. Next, we'll move the data to another Availability Zone. so far, all operations were within the same zone.

Move EBS Volume Data Across Availability Zones Using Snapshot

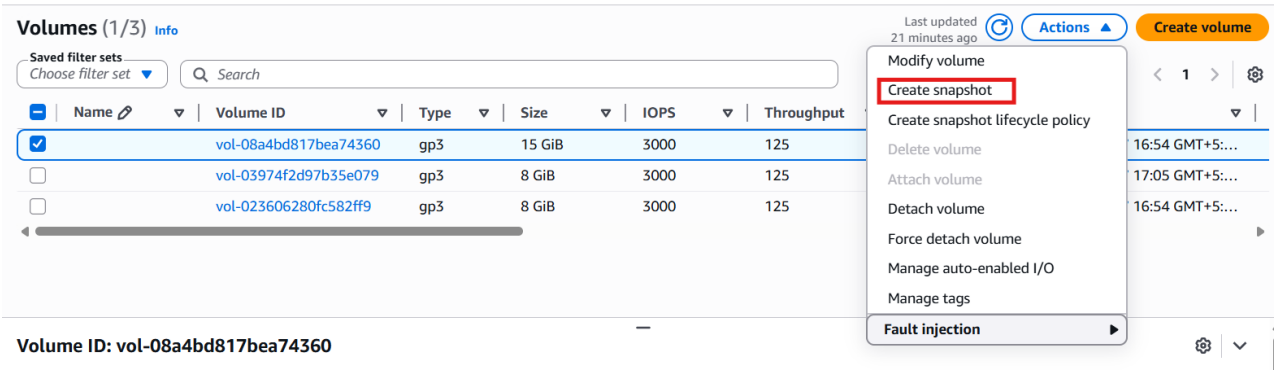
What is a Snapshot:

A **snapshot** is a backup of an EBS volume that you can use to create a new volume with the same data, even in a different Availability Zone.

- **Backup:** To save a copy of your EBS volume.
- **Recovery:** Restore data if something goes wrong.
- **Cross-AZ transfer:** Create volumes in a different Availability Zone.

Step 1: Create a Volume from Snapshot

- Go to AWS EC2 Console
- In the left menu, click **“Volumes”** under **Elastic Block Store**.
- Find and select the volume you want to snapshot
- Click **“Actions” → “Create Snapshot”**.

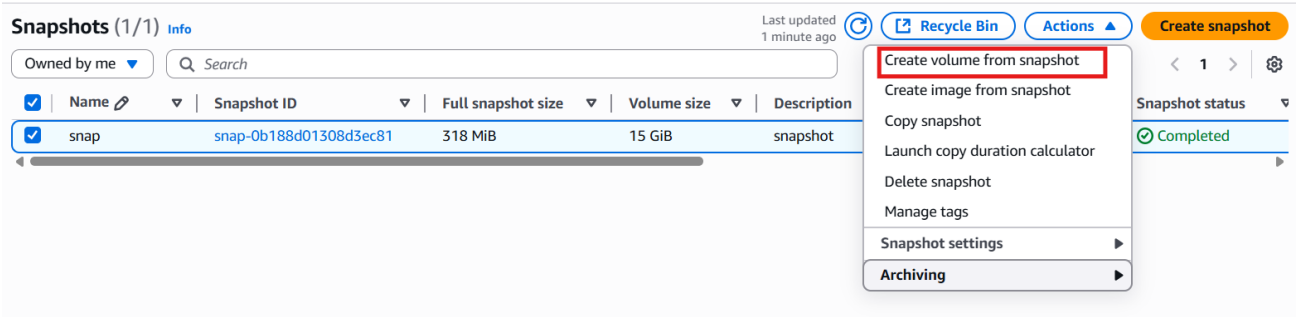


In the popup:

- **Name** (optional): my-server1-backup
- **Description**: Snapshot from Server 1 EBS volume

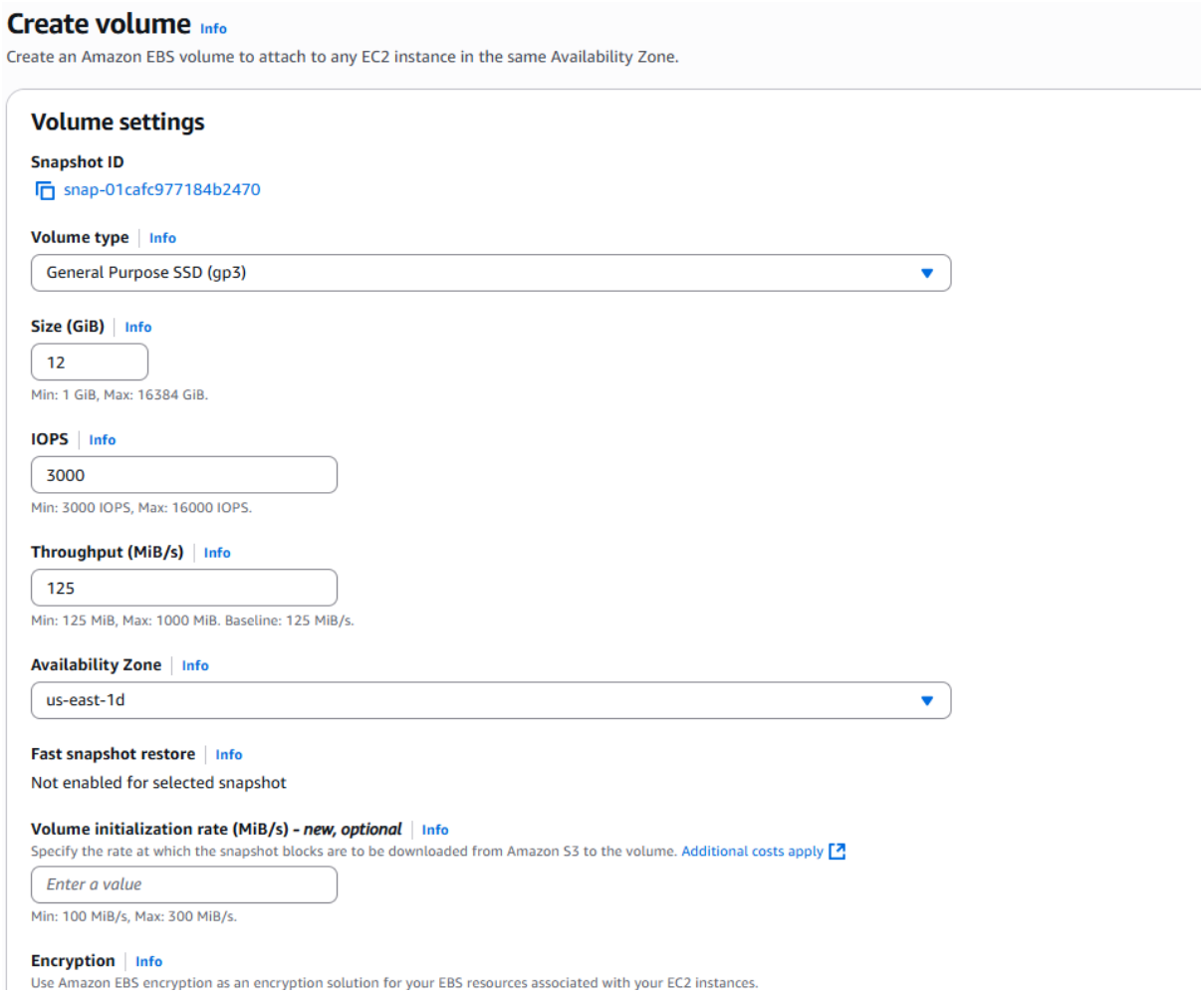
Step 2 : Create Volume from Snapshot

- Go to the **AWS EC2 Console**
- In the left menu, click **"Snapshots"** under **Elastic Block Store**.
- Select the snapshot you created earlier.
- Click **"Actions" → "Create Volume"**.



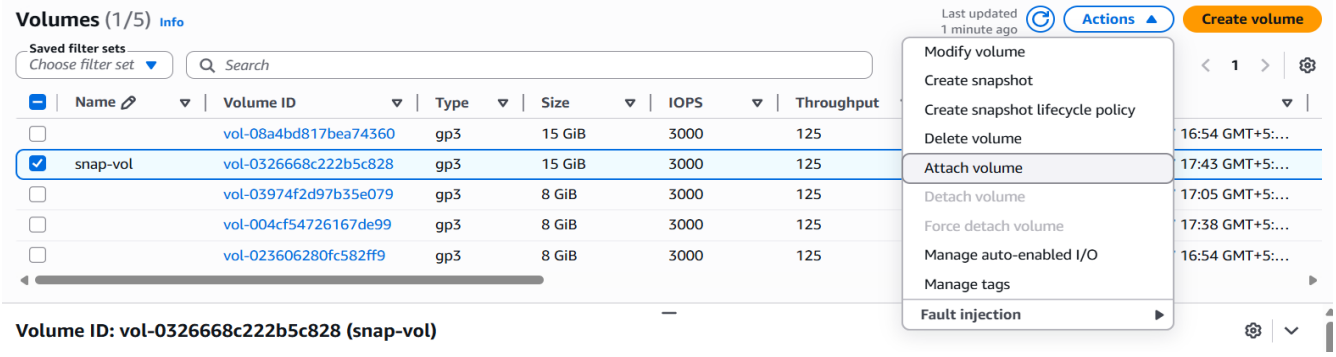
Fill in the volume details:

- Select the volume type (general purpose SSD)
- **Size**: Must be equal to or greater than snapshot size (eg : 12)
- **Availability Zone**: Choose the **same AZ as your target EC2 instance** (e.g., ap-south-1b)
- Remining will be leave as the default
- Click on create volume

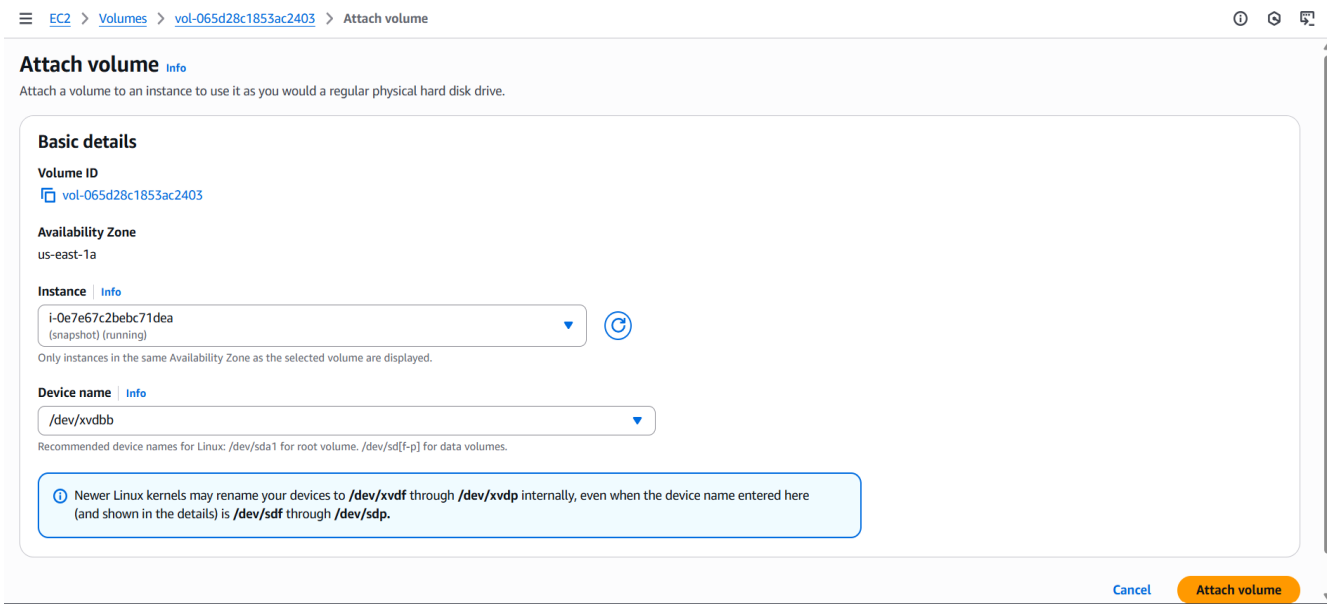


Step 3: After Volume is Created

- Go to Volumes in the sidebar.
- Select the new volume → Click “Actions → Attach Volume”.



- Select your EC2 instance (e.g., Server 3), and use a device name like /dev/xvdbd
- Click on Attach volume.



Step 4: Connect your server-3 and mount

Commands to dierctory: sudo mkdir snapshot

```
ubuntu@ip-172-31-28-42:~$ lsblk
NAME      MAJ:MIN   RM  SIZE RO TYPE MOUNTPOINTS
loop0       7:0      0 27.2M  1 loop /snap/amazon-ssm-agent/11320
loop1       7:1      0 73.9M  1 loop /snap/core22/1981
loop2       7:2      0 50.9M  1 loop /snap/snapd/24505
xvda      202:0      0    8G  0 disk
├─xvda1   202:1      0    7G  0 part /
├─xvda14  202:14     0    4M  0 part
├─xvda15  202:15     0  106M  0 part /boot/efi
└─xvda16  259:0      0  913M  0 part /boot
xvdbb     202:13568  0   15G  0 disk
ubuntu@ip-172-31-28-42:~$
ubuntu@ip-172-31-28-42:~$ mkdir snapshot
```

Command to mount : sudo mount /dev/xvdbd snapshot/

```
ubuntu@ip-172-31-28-42:~$ sudo mount /dev/xvdbb snapshot/
ubuntu@ip-172-31-28-42:~$
ubuntu@ip-172-31-28-42:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       6.8G  1.8G  5.0G  26% /
tmpfs           479M     0  479M   0% /dev/shm
tmpfs           192M  872K  191M   1% /run
tmpfs           5.0M     0  5.0M   0% /run/lock
/dev/xvda16     881M   86M  734M  11% /boot
/dev/xvda15     105M  6.2M   99M   6% /boot/efi
tmpfs            96M   12K   96M   1% /run/user/1000
/dev/xvdbb       15G   28K   14G   1% /home/ubuntu/snapshot
ubuntu@ip-172-31-28-42:~$
ubuntu@ip-172-31-28-42:~$ cd snapshot/
ubuntu@ip-172-31-28-42:~/snapshot$ ls
lost+found  my-volume
ubuntu@ip-172-31-28-42:~/snapshot$ cat my-volume
Hello I'm Arun Kumar Akula
working as a DevOps Engineer.!
ubuntu@ip-172-31-28-42:~/snapshot$
```