Tasks 3: AWS EC2

1) EC2 is in us-east-1a AZ and EBS is in us-east-1b AZ. How do you attach?

Detailed Instructions for Moving EBS Volumes Across Availability Zones

---

1. Create a Snapshot and Restore It in the Target AZ

Step 1: Create a Snapshot of the Existing EBS Volume
Identify the volume ID of the EBS volume in the source AZ:

```
aws ec2 describe-volumes --filters Name=availability-zone,Values=us-east-1b
```

1. Replace `us-east-1b` with your source AZ.

Create a snapshot of the volume:

```
aws ec2 create-snapshot --volume-id vol-xxxxxxx --description "Snapshot for
moving EBS volume"
```

2.
   - Replace `vol-xxxxxxx` with the ID of your EBS volume.
   - Note the `SnapshotId` from the output.

Step 2: Create a New Volume from the Snapshot in the Target AZ
Create a new EBS volume in the target AZ (e.g., `us-east-1a`):
bash
Copy code

```
aws ec2 create-volume --availability-zone us-east-1a --snapshot-id snap-xxxxxxx
```

1.
   - Replace `snap-xxxxxxx` with the snapshot ID created in Step 1.
   - Replace `us-east-1a` with the AZ of your target EC2 instance.

Wait for the new volume to be created. Check the status:
bash
Copy code

```
aws ec2 describe-volumes --volume-ids vol-yyyyyyy
```

2. Replace `vol-yyyyyyy` with the new volume ID.

Step 3: Attach the New Volume to the Target EC2 Instance
Attach the new volume to the EC2 instance in the target AZ:
bash

Copy code
```
aws ec2 attach-volume --volume-id vol-yyyyyyy --instance-id i-zzzzzzzz --device
/dev/xvdf
```

1.
    - Replace `vol-yyyyyyy` with the new volume ID.
    - Replace `i-zzzzzzzz` with the target EC2 instance ID.

SSH into the instance and verify the volume:
bash
Copy code
```
lsblk
```

2.

---

2) Use Cross-Zone Data Transfer

This approach requires manual data copying and is useful when you cannot use snapshots.

Step 1: Prepare the Source and Target EC2 Instances

1. Launch an EC2 instance in the same AZ as the original EBS volume (e.g., `us-east-1b`).

Attach the original EBS volume to this instance:
bash
Copy code
```
aws ec2 attach-volume --volume-id vol-xxxxxxx --instance-id i-aaaaaaa --device
/dev/xvdf
```

2.

Mount the volume:
bash
Copy code
```
sudo mkdir /mnt/old-volume
sudo mount /dev/xvdf /mnt/old-volume
```

3.
4. Launch another EC2 instance in the target AZ (e.g., `us-east-1a`) and attach a new EBS volume to it. Repeat the above steps to mount it.

Step 2: Transfer Data Directly

1. SSH into the source EC2 instance.

Use `rsync` to copy data directly from the source instance to the target instance:
bash

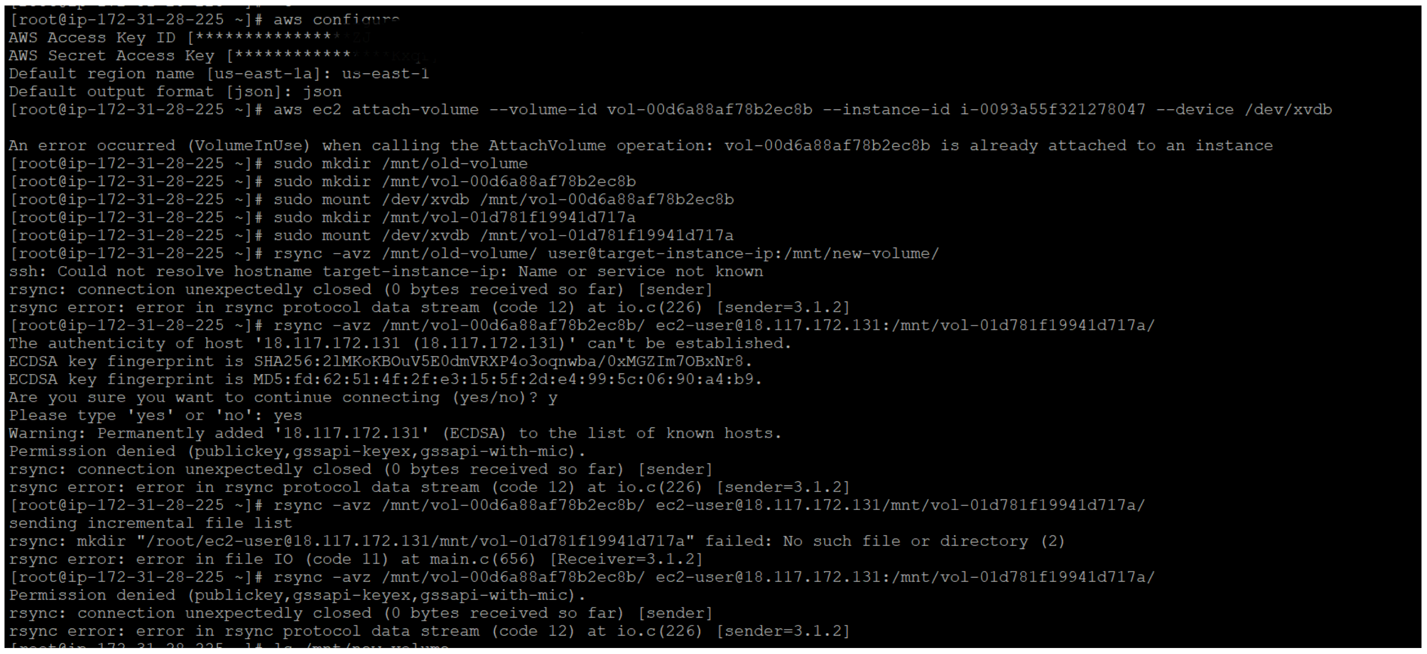
Copy code
```
rsync -avz /mnt/old-volume/ user@target-instance-ip:/mnt/new-volume/
```

2.
   - Replace `/mnt/old-volume/` with the mount path of the source volume.
   - Replace `user@target-instance-ip` with the user and IP address of the target EC2 instance.
   - Replace `/mnt/new-volume/` with the mount path of the new volume.

Step 3: Verify Data Transfer

1. SSH into the target instance.

Verify the data in the new volume:
bash
Copy code
```
ls /mnt/new-volume
```



```
[root@ip-172-31-28-225 ~]# aws configure
AWS Access Key ID [***************[redacted]
AWS Secret Access Key [************[redacted]
Default region name [us-east-1a]: us-east-1
Default output format [json]: json
[root@ip-172-31-28-225 ~]# aws ec2 attach-volume --volume-id vol-00d6a88af78b2ec8b --instance-id i-0093a55f321278047 --device /dev/xvdb

An error occurred (VolumeInUse) when calling the AttachVolume operation: vol-00d6a88af78b2ec8b is already attached to an instance
[root@ip-172-31-28-225 ~]# sudo mkdir /mnt/old-volume
[root@ip-172-31-28-225 ~]# sudo mkdir /mnt/vol-00d6a88af78b2ec8b
[root@ip-172-31-28-225 ~]# sudo mount /dev/xvdb /mnt/vol-00d6a88af78b2ec8b
[root@ip-172-31-28-225 ~]# sudo mkdir /mnt/vol-01d781f19941d717a
[root@ip-172-31-28-225 ~]# sudo mount /dev/xvdb /mnt/vol-01d781f19941d717a
[root@ip-172-31-28-225 ~]# rsync -avz /mnt/old-volume/ user@target-instance-ip:/mnt/new-volume/
ssh: Could not resolve hostname target-instance-ip: Name or service not known
rsync: connection unexpectedly closed (0 bytes received so far) [sender]
rsync error: error in rsync protocol data stream (code 12) at io.c(226) [sender=3.1.2]
[root@ip-172-31-28-225 ~]# rsync -avz /mnt/vol-00d6a88af78b2ec8b/ ec2-user@18.117.172.131:/mnt/vol-01d781f19941d717a/
The authenticity of host '18.117.172.131 (18.117.172.131)' can't be established.
ECDSA key fingerprint is SHA256:2lMKoKBOuV5E0dmVRXP4o3oqnwba/0xMGZIm7OBxNr8.
ECDSA key fingerprint is MD5:fd:62:51:4f:2f:e3:15:5f:2d:e4:99:5c:06:90:a4:b9.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '18.117.172.131' (ECDSA) to the list of known hosts.
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
rsync: connection unexpectedly closed (0 bytes received so far) [sender]
rsync error: error in rsync protocol data stream (code 12) at io.c(226) [sender=3.1.2]
[root@ip-172-31-28-225 ~]# rsync -avz /mnt/vol-00d6a88af78b2ec8b/ ec2-user@18.117.172.131/mnt/vol-01d781f19941d717a/
sending incremental file list
rsync: mkdir "/root/ec2-user@18.117.172.131/mnt/vol-01d781f19941d717a" failed: No such file or directory (2)
rsync error: error in file IO (code 11) at main.c(656) [Receiver=3.1.2]
[root@ip-172-31-28-225 ~]# rsync -avz /mnt/vol-00d6a88af78b2ec8b/ ec2-user@18.117.172.131:/mnt/vol-01d781f19941d717a/
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
rsync: connection unexpectedly closed (0 bytes received so far) [sender]
rsync error: error in rsync protocol data stream (code 12) at io.c(226) [sender=3.1.2]
```

---

3) Additional Option: S3 as an Intermediary

If direct transfer isn't feasible, use Amazon S3 as a middleman:

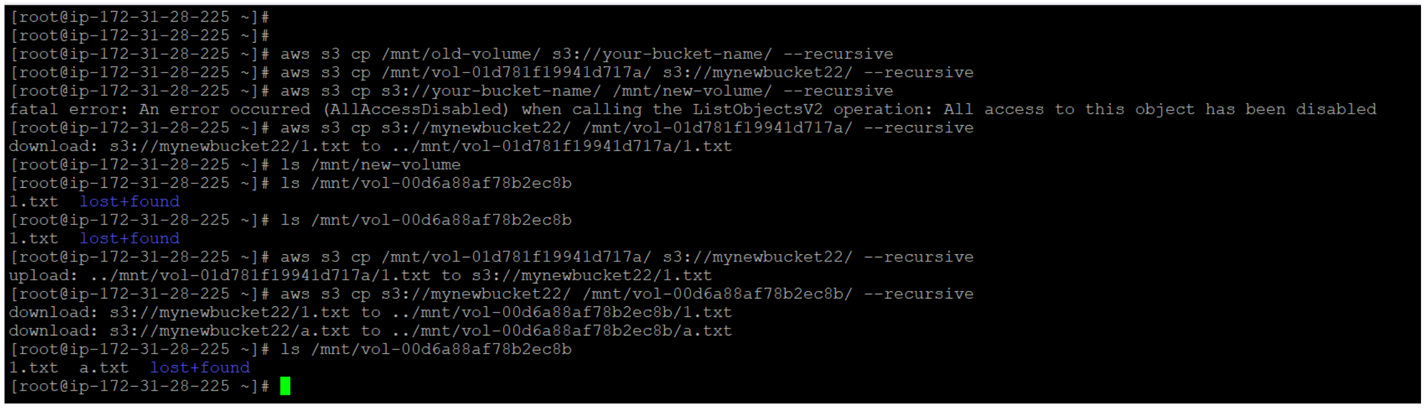
1. **From the Source AZ:**

Upload data to S3:
bash
Copy code
```
aws s3 cp /mnt/old-volume/ s3://your-bucket-name/ --recursive
```

   -

2. **In the Target AZ:**

Download data from S3:
bash
Copy code
```
aws s3 cp s3://your-bucket-name/ /mnt/new-volume/ --recursive
```



```
[root@ip-172-31-28-225 ~]#
[root@ip-172-31-28-225 ~]#
[root@ip-172-31-28-225 ~]# aws s3 cp /mnt/old-volume/ s3://your-bucket-name/ --recursive
[root@ip-172-31-28-225 ~]# aws s3 cp /mnt/vol-01d781f19941d717a/ s3://mynewbucket22/ --recursive
[root@ip-172-31-28-225 ~]# aws s3 cp s3://your-bucket-name/ /mnt/new-volume/ --recursive
fatal error: An error occurred (AllAccessDisabled) when calling the ListObjectsV2 operation: All access to this object has been disabled
[root@ip-172-31-28-225 ~]# aws s3 cp s3://mynewbucket22/ /mnt/vol-01d781f19941d717a/ --recursive
download: s3://mynewbucket22/1.txt to ../mnt/vol-01d781f19941d717a/1.txt
[root@ip-172-31-28-225 ~]# ls /mnt/new-volume
[root@ip-172-31-28-225 ~]# ls /mnt/vol-00d6a88af78b2ec8b
1.txt  lost+found
[root@ip-172-31-28-225 ~]# ls /mnt/vol-00d6a88af78b2ec8b
1.txt  lost+found
[root@ip-172-31-28-225 ~]# aws s3 cp /mnt/vol-01d781f19941d717a/ s3://mynewbucket22/ --recursive
upload: ../mnt/vol-01d781f19941d717a/1.txt to s3://mynewbucket22/1.txt
[root@ip-172-31-28-225 ~]# aws s3 cp s3://mynewbucket22/ /mnt/vol-00d6a88af78b2ec8b/ --recursive
download: s3://mynewbucket22/1.txt to ../mnt/vol-00d6a88af78b2ec8b/1.txt
download: s3://mynewbucket22/a.txt to ../mnt/vol-00d6a88af78b2ec8b/a.txt
[root@ip-172-31-28-225 ~]# ls /mnt/vol-00d6a88af78b2ec8b
1.txt  a.txt  lost+found
[root@ip-172-31-28-225 ~]#
```

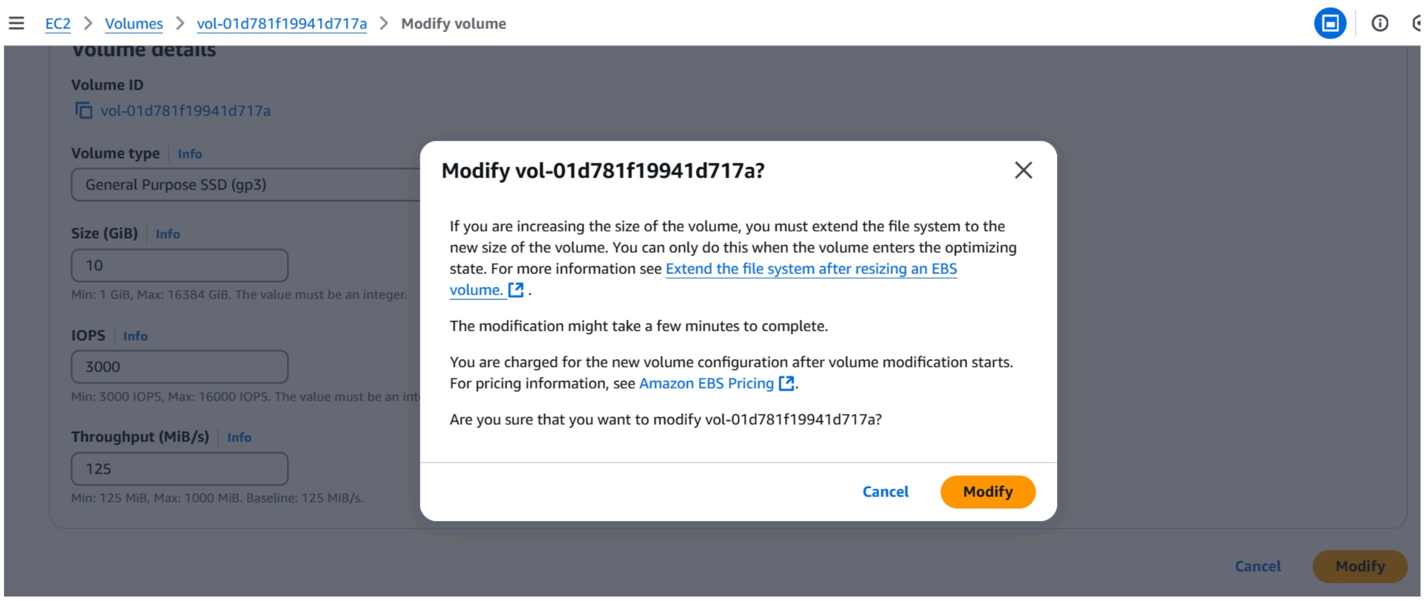
---

2) EBS - 10 GB, how can we increase it? And how to decrease it?

To modify the size of an **EBS (Elastic Block Store)** volume in AWS, you can follow these steps:

Increasing the Size of an EBS Volume:

1. **Stop the Instance (Optional):**
   - It's generally safe to resize the EBS volume while the instance is running. However, it's good practice to stop the instance if you're working on critical data or need to ensure no I/O operations happen during resizing.
2. **Modify the EBS Volume:**
   - Go to the **AWS Management Console** > **EC2 Dashboard**.
   - In the **Volumes** section under **Elastic Block Store**, select the volume you want to resize.
   - Choose **Actions** > **Modify Volume**.
   - Increase the size of the volume to your desired size (e.g., from 10 GB to 20 GB).
   - Click **Modify** and confirm the change.


EC2 > Volumes > vol-01d781f19941d717a > Modify volume
Volume details
Volume ID
vol-01d781f19941d717a
Volume type | Info
General Purpose SSD (gp3)
Size (GiB) | Info
10
Min: 1 GiB, Max: 16384 GiB. The value must be an integer.
IOPS | Info
3000
Min: 3000 IOPS, Max: 16000 IOPS. The value must be an int
Throughput (MiB/s) | Info
125
Min: 125 MiB, Max: 1000 MiB. Baseline: 125 MiB/s.
Cancel
Modify

Modify vol-01d781f19941d717a?

If you are increasing the size of the volume, you must extend the file system to the new size of the volume. You can only do this when the volume enters the optimizing state. For more information see Extend the file system after resizing an EBS volume.

The modification might take a few minutes to complete.

You are charged for the new volume configuration after volume modification starts. For pricing information, see Amazon EBS Pricing.

Are you sure that you want to modify vol-01d781f19941d717a?

Cancel | Modify

3. **Expand the File System:** After increasing the volume size, the file system on the instance needs to be resized to use the newly allocated space. You can do this from the instance:
   - For **Linux** instances:
     - SSH into the instance.

Run the following commands:

```
sudo resize2fs /dev/xvdf
```

   -
     1. Replace `/dev/xvdf` with your actual device name if it's different.
   - For **Windows** instances:
     1. Use the **Disk Management** tool to extend the volume and use the additional space.

Decreasing the Size of an EBS Volume:

AWS doesn't allow you to **directly** decrease the size of an EBS volume. However, you can work around this by creating a new smaller volume and transferring the data:

1. **Create a Snapshot of the Volume:**
   - Go to **Volumes** in the **EC2 Dashboard** and select the volume you want to shrink.
   - Choose **Actions** > **Create Snapshot** to back up the current state.
2. **Create a New Smaller Volume:**
   - Create a new EBS volume with the desired smaller size.
3. **Attach the New Volume to the Instance:**
   - Detach the original volume (optional if you're working with multiple volumes).
   - Attach the new, smaller volume to the instance.
4. **Transfer Data:**
   - Mount both volumes on the instance.
   - Copy the data from the old (larger) volume to the new (smaller) one.
5. **Update the File System (If Needed):**
   - Resize the file system on the new volume to ensure it is recognized correctly.
6. **Detach the Old Volume and Clean Up:**

- Detach the old volume after successfully migrating the data.
- Delete the old volume and snapshot if no longer needed.

These steps allow you to work around the restriction of not being able to directly shrink a volume.

3) ec2 - t2.micro (1cpu and 1gb ram). I need 2gb ram , how to upgrade it?

To upgrade the RAM for an EC2 instance, you need to change the instance type to one with more memory. Here's how you can do it:

1. **Stop the Instance:**
   - In the **AWS Management Console**, go to the EC2 Dashboard.
   - In the **Instances** section, select the instance you want to upgrade.
   - Click **Instance State** and then **Stop** (Note: stopping the instance will cause downtime).
2. **Change the Instance Type:**
   - Once the instance is stopped, select the instance again.
   - Click the **Actions** button, go to **Instance Settings**, and select **Change Instance Type**.
   - In the **Instance Type** dropdown, select an instance type with 2GB or more of RAM. For example, you can choose the `t3.micro` (2 vCPUs and 1GB RAM) or `t3.small` (2 vCPUs and 2GB RAM), depending on your requirements.
   - Click **Apply**.
3. **Start the Instance:**
   - After the instance type is changed, click **Instance State** and then **Start** to bring the instance back online.

Make sure to verify your instance's performance and pricing after making the change.

4) How to setup hostname temporary?

Temporary hostname: Only affects the current session until the system is rebooted.
Permanent hostname: Requires updating /etc/hostname and /etc/hosts as described in previous answers for persistent changes.

Setup Hostname Temporary

```
Command Hostname dev
Sudo -i
```

But if we reboot again IP will be shown

5) How to setup Permanent hostname?

Setup Hostname Permanently Using hostnamectl (for Systemd-based Systems)

Most modern Linux distributions (e.g., CentOS 7+, Ubuntu 16.04+, Debian 8+, etc.) use systemd, and hostnamectl is the recommended tool for managing hostnames.

Step 1: Set the Hostname

To change the hostname, run:

`sudo hostnamectl set-hostname new-hostname` (Replace `new-hostname` with the desired hostname.)

Step 2: Verify the Change

You can verify that the hostname has been set correctly using:

`Hostnamectl` (This will show the new hostname in the output.)

Step 3: Update `/etc/hosts` (Optional but Recommended)

After changing the hostname, you should also update the `/etc/hosts` file to associate the new hostname with the loopback address (`127.0.0.1`).

Open the `/etc/hosts` file with a text editor:

`sudo nano /etc/hosts` (Update the line that corresponds to your local host, replacing old-hostname with the new one)

Locate the existing lines: You'll see something like this in the file:

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4

::1         localhost6 localhost6.localdomain6
```

Add a new line with your new hostname: Just below the `127.0.0.1` line, add your new hostname:

```
127.0.1.1    new-hostname
```

Save and exit the editor:

- Press CTRL + O to save the file, then press Enter.
- Press CTRL + X to exit the editor.

Step 4: Reboot Your Instance

Finally, reboot your instance to apply the changes:

Rebooting Without Terminating

When rebooting an EC2 instance, ensure that you're just restarting the instance and not stopping it. You can do this either from the AWS Console or using the AWS CLI:

AWS Console:

- Select the instance and click **Actions > Instance State > Reboot**.

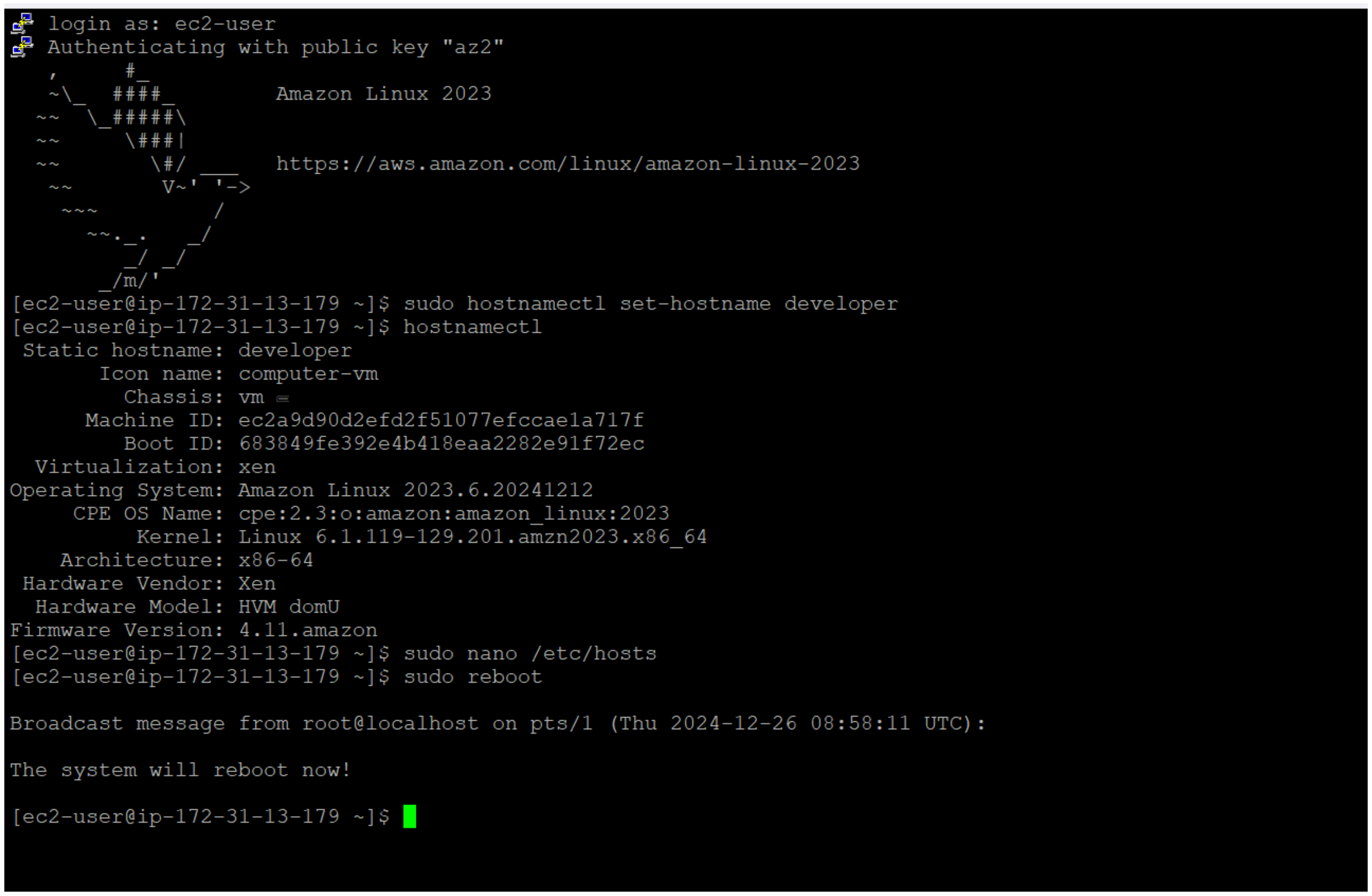
AWS CLI:

- Run the following command to reboot the instance

```
aws ec2 reboot-instances --instance-ids i-xxxxxxxxxxxxxxxxx
```

After the reboot, verify that the hostname is updated by running:

```
hostname
```

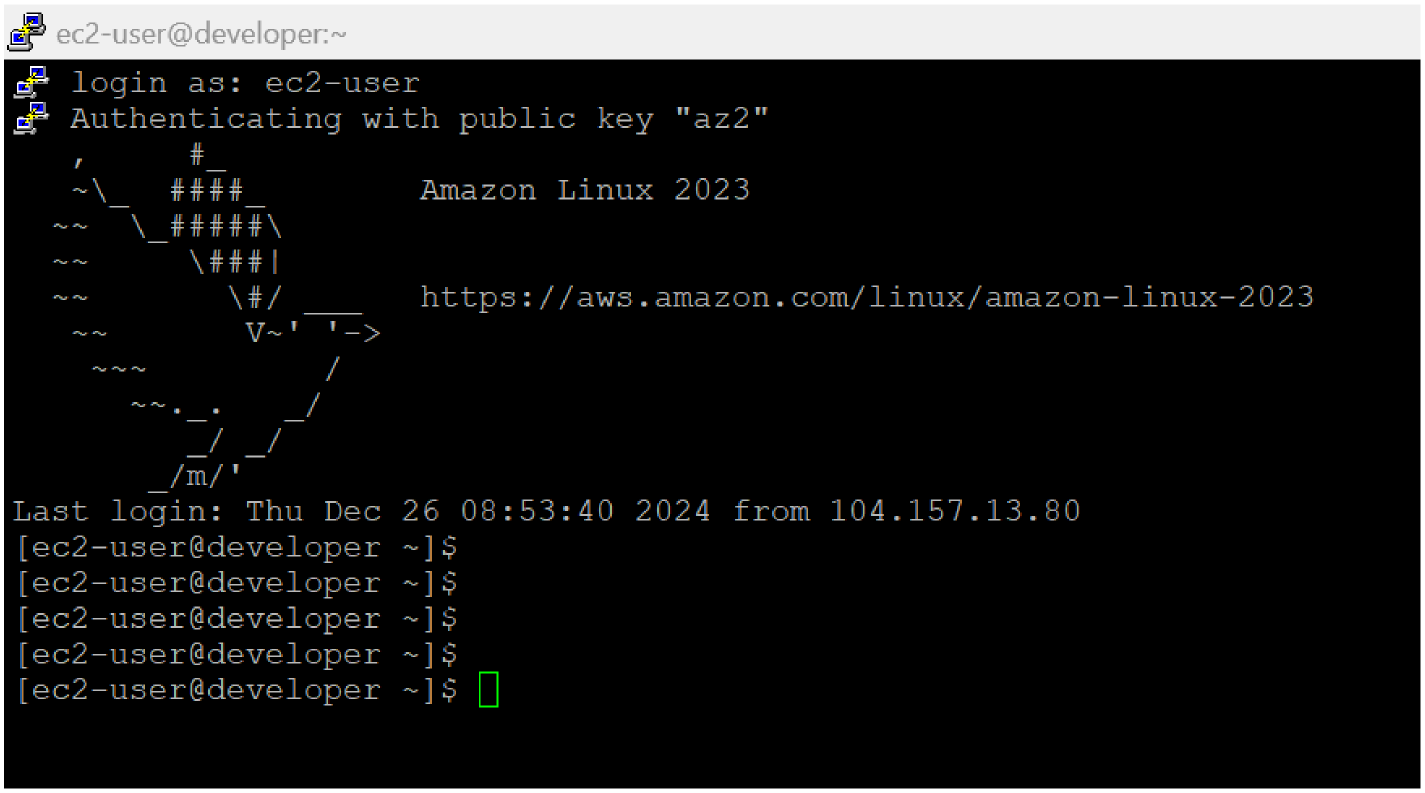


```
login as: ec2-user
Authenticating with public key "az2"
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-13-179 ~]$ sudo hostnamectl set-hostname developer
[ec2-user@ip-172-31-13-179 ~]$ hostnamectl
 Static hostname: developer
       Icon name: computer-vm
         Chassis: vm
      Machine ID: ec2a9d90d2efd2f51077efccae1a717f
         Boot ID: 683849fe392e4b418eaa2282e91f72ec
  Virtualization: xen
Operating System: Amazon Linux 2023.6.20241212
     CPE OS Name: cpe:2.3:o:amazon:amazon_linux:2023
          Kernel: Linux 6.1.119-129.201.amzn2023.x86_64
    Architecture: x86-64
 Hardware Vendor: Xen
  Hardware Model: HVM domU
Firmware Version: 4.11.amazon
[ec2-user@ip-172-31-13-179 ~]$ sudo nano /etc/hosts
[ec2-user@ip-172-31-13-179 ~]$ sudo reboot

Broadcast message from root@localhost on pts/1 (Thu 2024-12-26 08:58:11 UTC):

The system will reboot now!

[ec2-user@ip-172-31-13-179 ~]$
```



```
login as: ec2-user
Authenticating with public key "az2"
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Thu Dec 26 08:53:40 2024 from 104.157.13.80
[ec2-user@developer ~]$
[ec2-user@developer ~]$
[ec2-user@developer ~]$
[ec2-user@developer ~]$
[ec2-user@developer ~]$
```