**Challenge task-2**

**Use Case: Setting up Transit Gateway and VPC Endpoints for a Multi-VPC Architecture**

**Scenario:**

A large organization is migrating its on-premises infrastructure to the AWS cloud.

The organization's architecture involves multiple VPCs for different departments and applications, each requiring secure communication with centralized services and external resources.

The IT team needs to design and implement a scalable and efficient network architecture to accommodate the organization's growth and ensure robust connectivity between VPCs and external services.

1. **Design and deploy a scalable network architecture using AWS Transit Gateway to simplify network connectivity between multiple VPCs.**

**Step 1: Plan Your IP Addressing**

**non-overlapping CIDRs from different classes:**

**1.)VPC-A (N. Virginia, us-east-1)** → 10.0.0.0/16

2.) **VPC-B (OHIO -us-east-2)**  → 10.10.0.0/16

**3). VPC-C (California, us-west-1)** → 10.20.0.0/16

**4.)VPC-D (Mumbai, ap-south-1)**  → 10.30.0.0/16

**Step 2: Create 4 VPCs (Non-default)**

Create VPCs (in 4 different regions)

**Go to VPC Console → Your VPCs → Create VPC.**

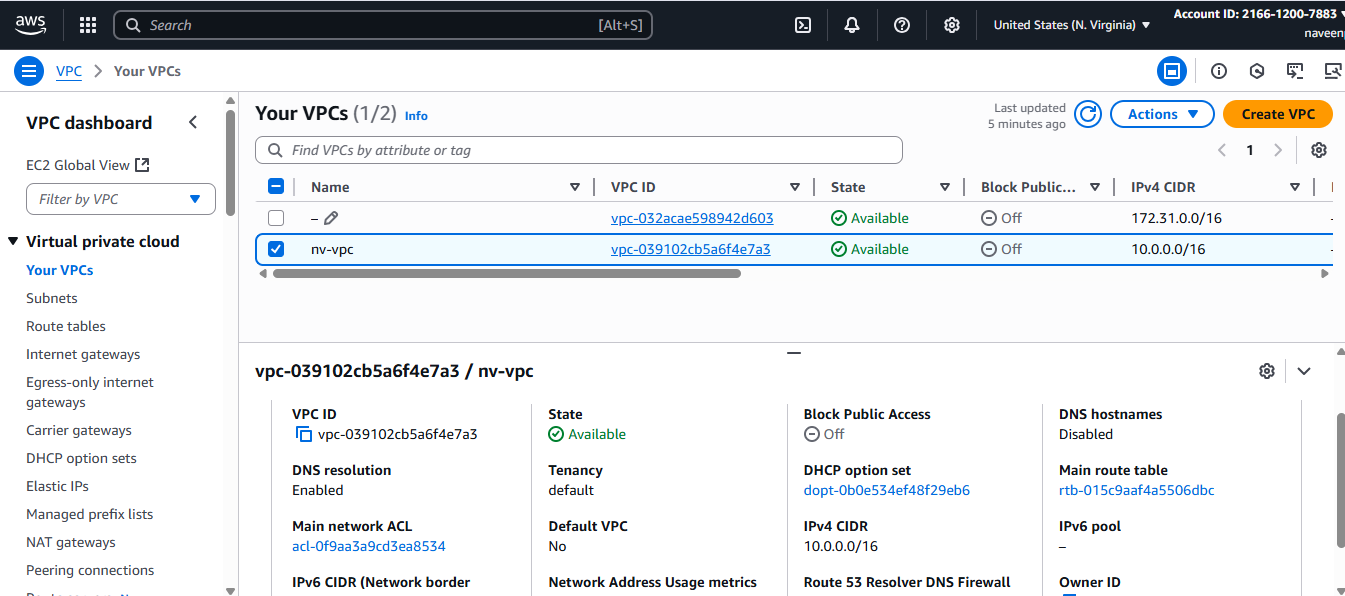
Create **1 private subnets** in each VPC (in different AZs).

No public subnets needed (since no IGW/NAT) except Virginia region.

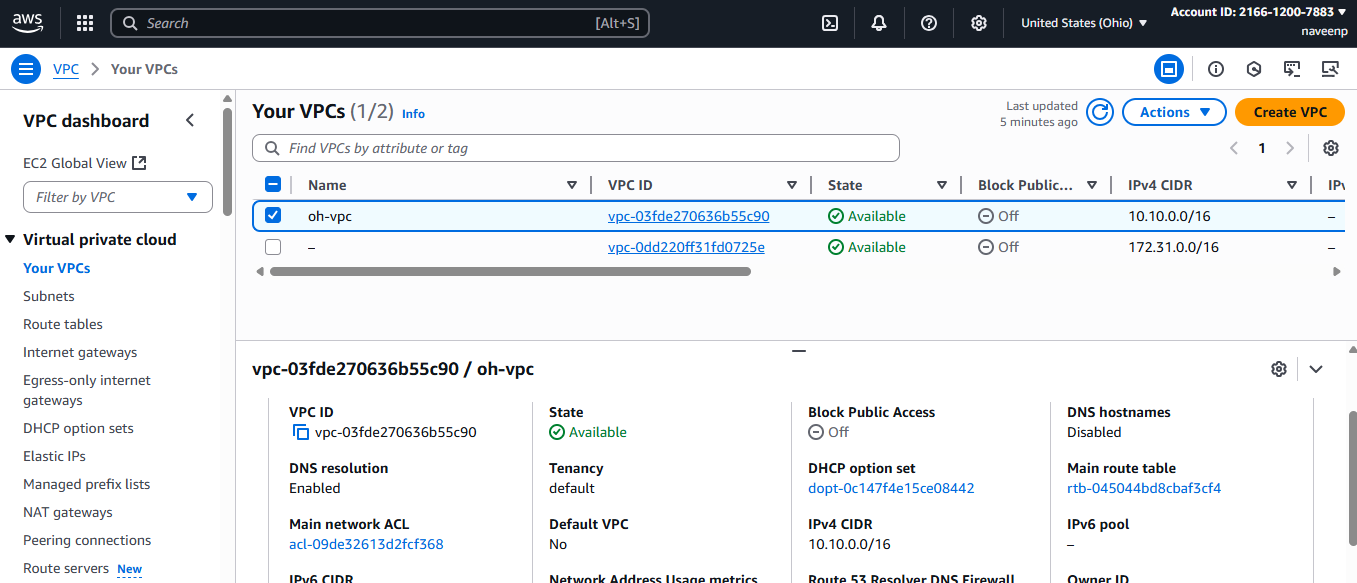
**Attach IGW TO only oneVPC to**  (**VPC-A (N. Virginia, us-east-1) → 10.0.0.0/16**)

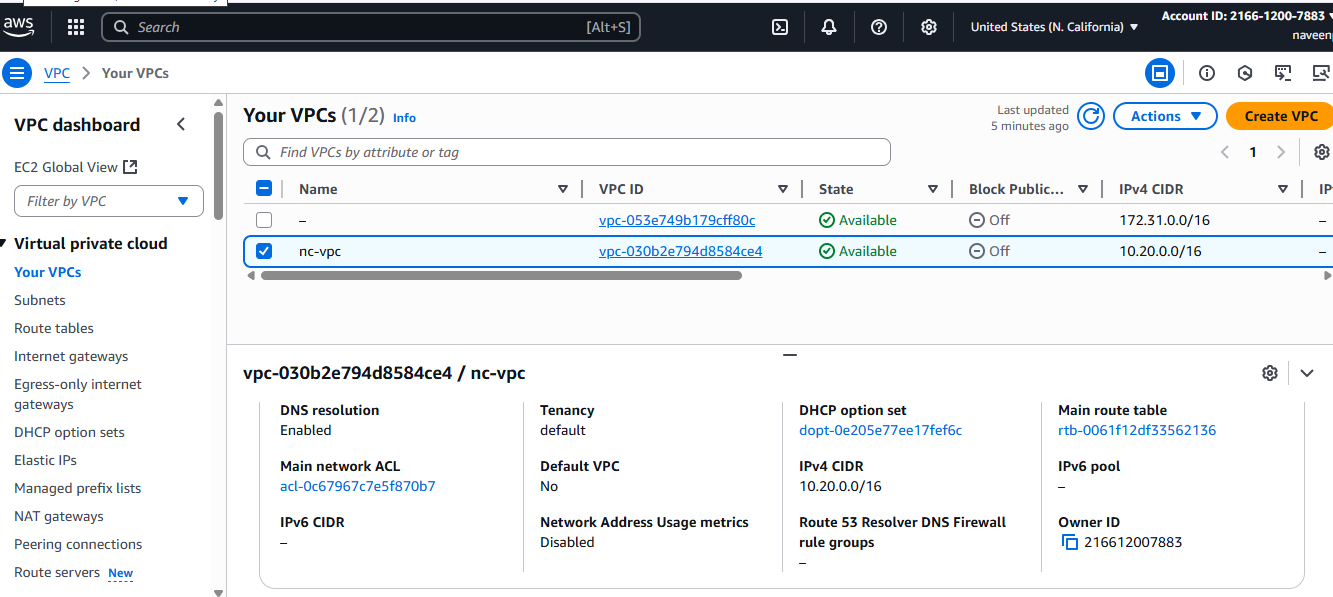
**Create a Route Table for each set of subnets and associate them.**

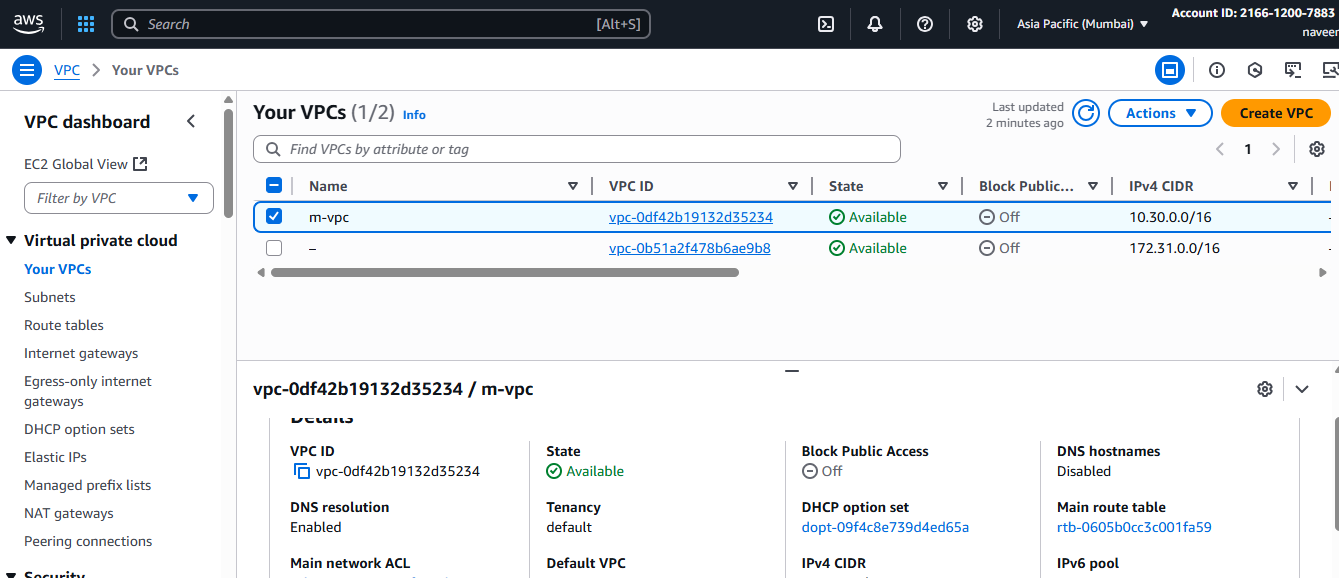
**Repeat for all 4 regions.**



Now Simlarly create Vpcs subnets and RouteTables in other vpc2,vpc3 and vpc4  
i.e in 4 different regions







**Step 3: Create Transit Gateways**

## Create Transit Gateways (1 per region)

Repeat per region.

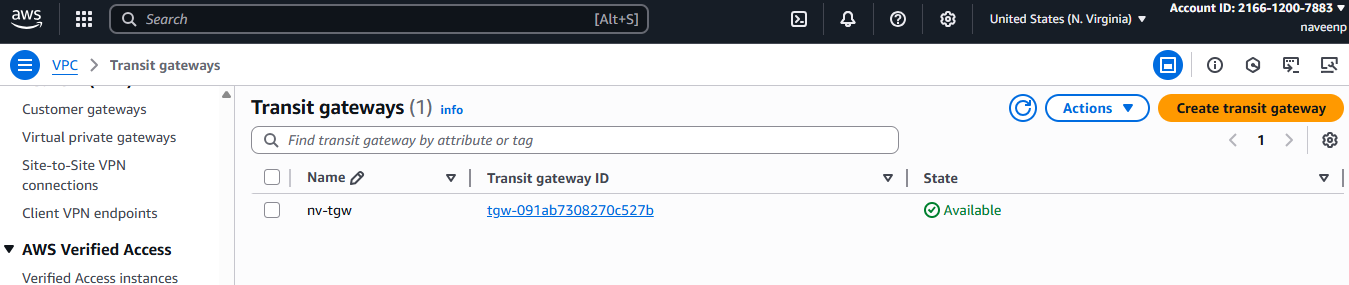
1. **VPC Console → Transit Gateways → Create transit gateway**

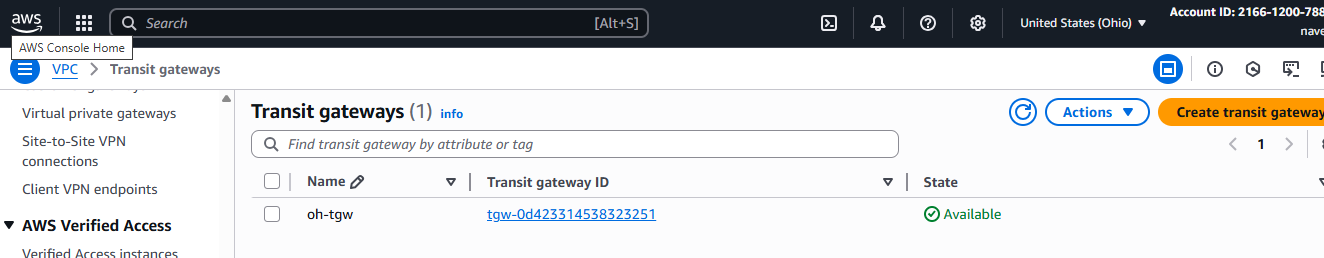
Attach each VPC to its regional TGW

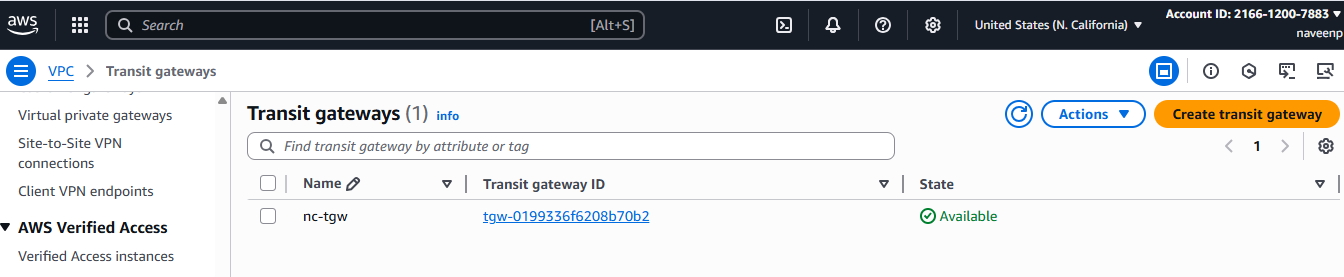
**TGW VPC attachment** with the **private subnets**.

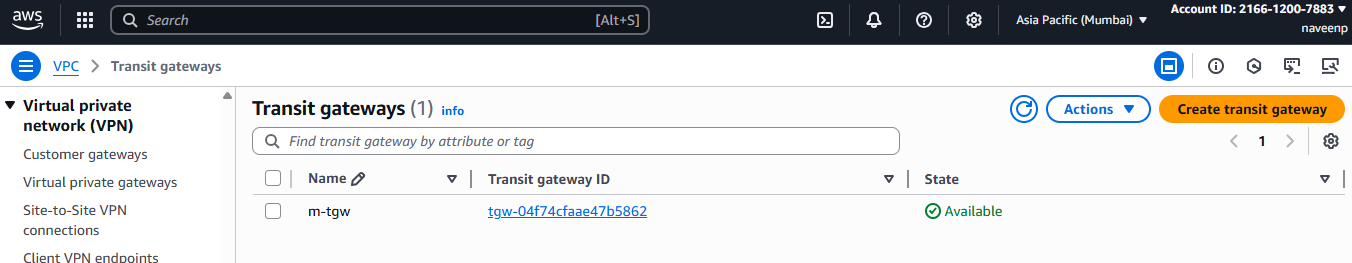
**Associate** the attachment with the **TGW route table** in that region.

**Enable route propagation** from that attachment into the TGW route table (or add static routes).





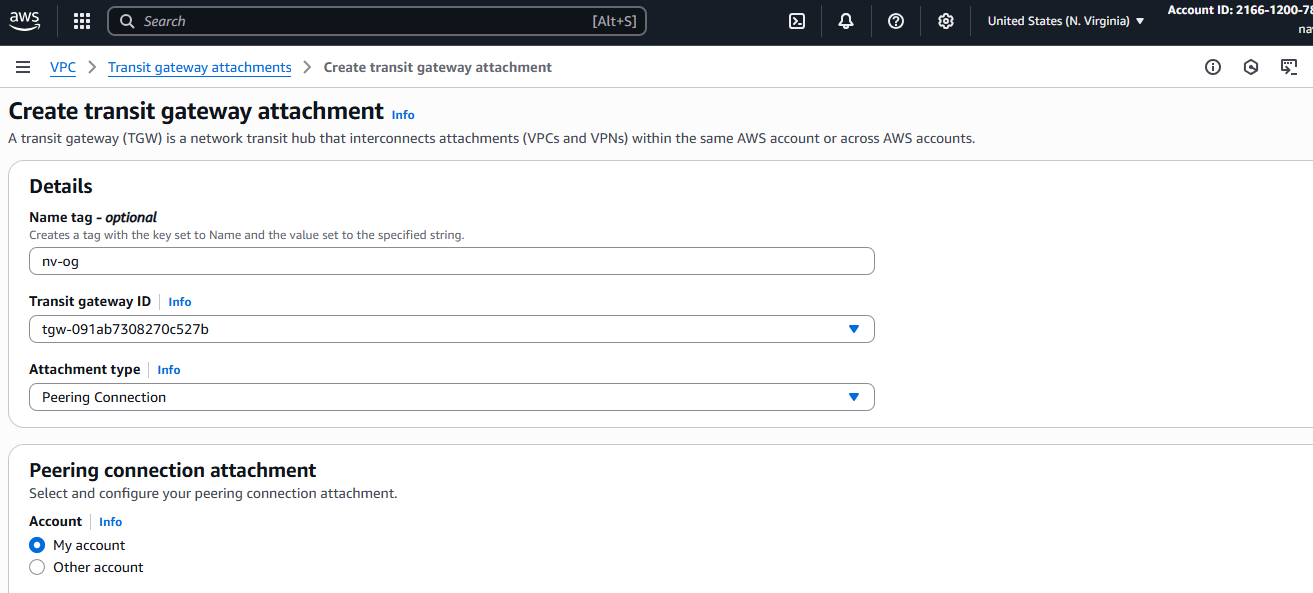


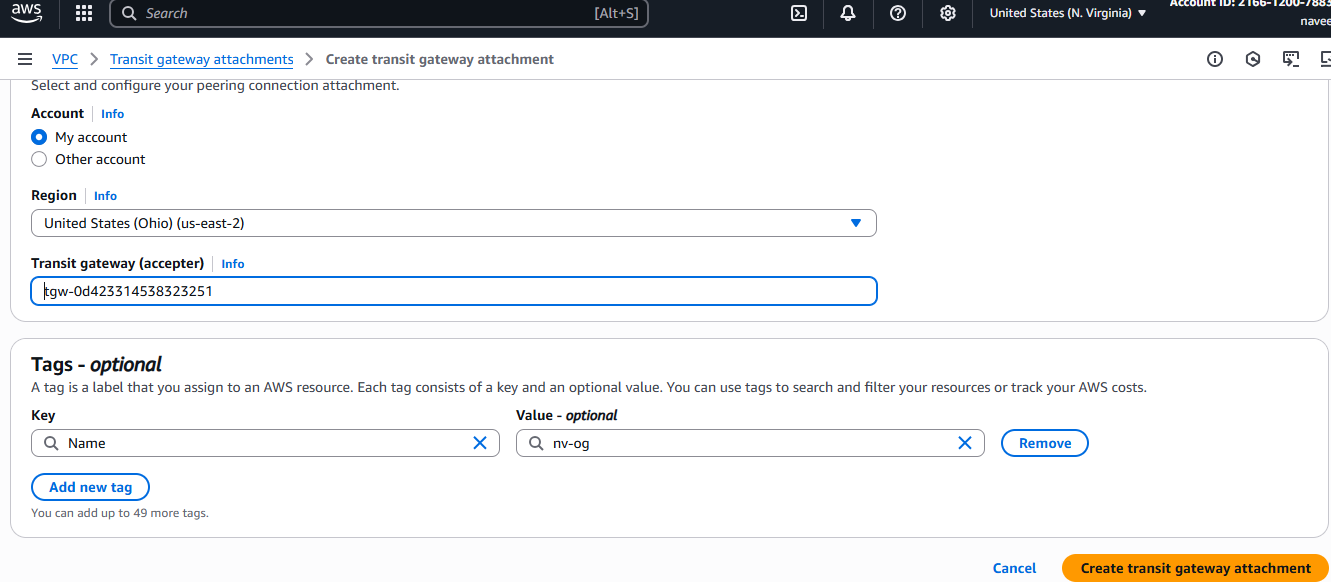


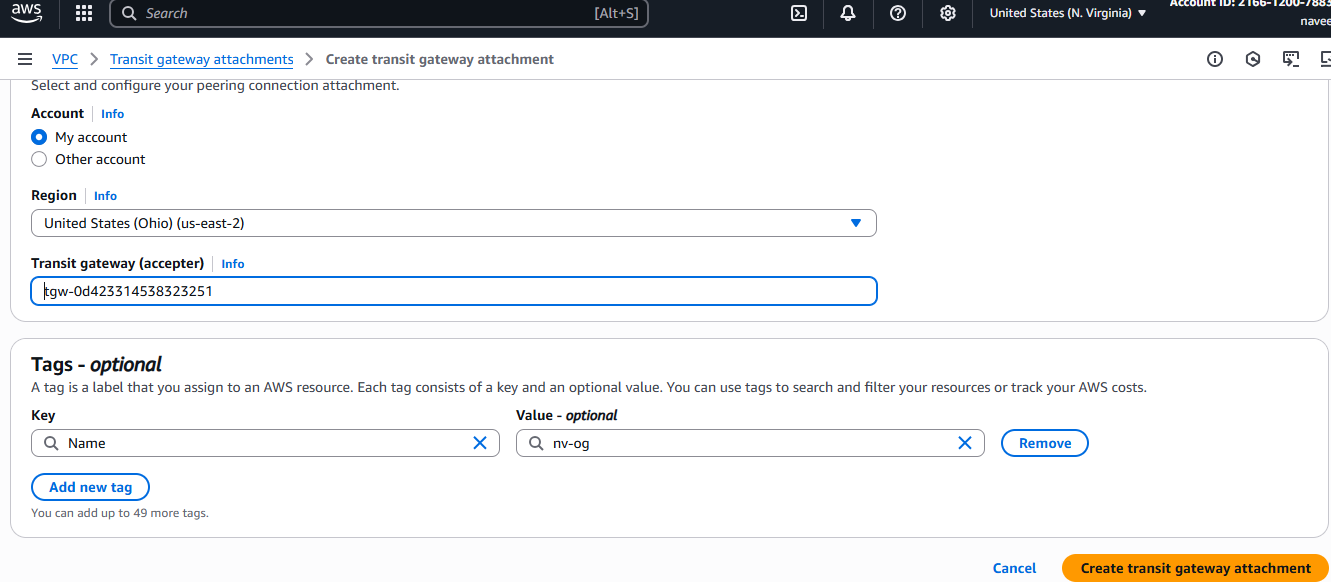
## Step-4 Attach Each VPC to its Regional TGW

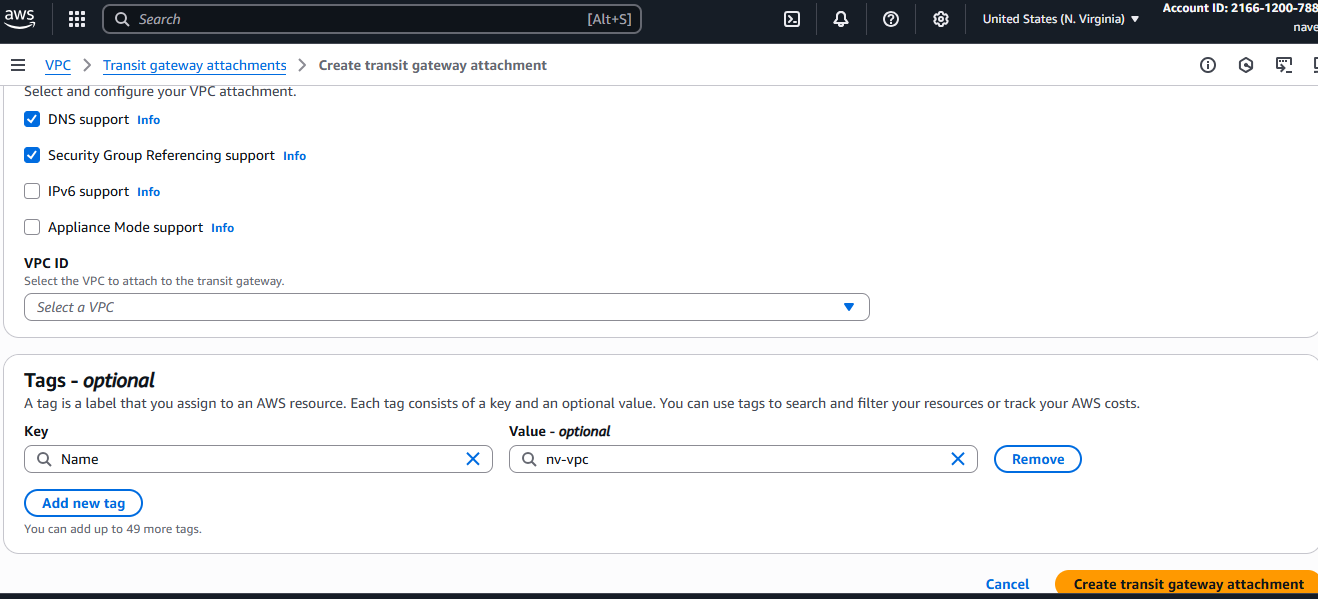
Repeat per region.

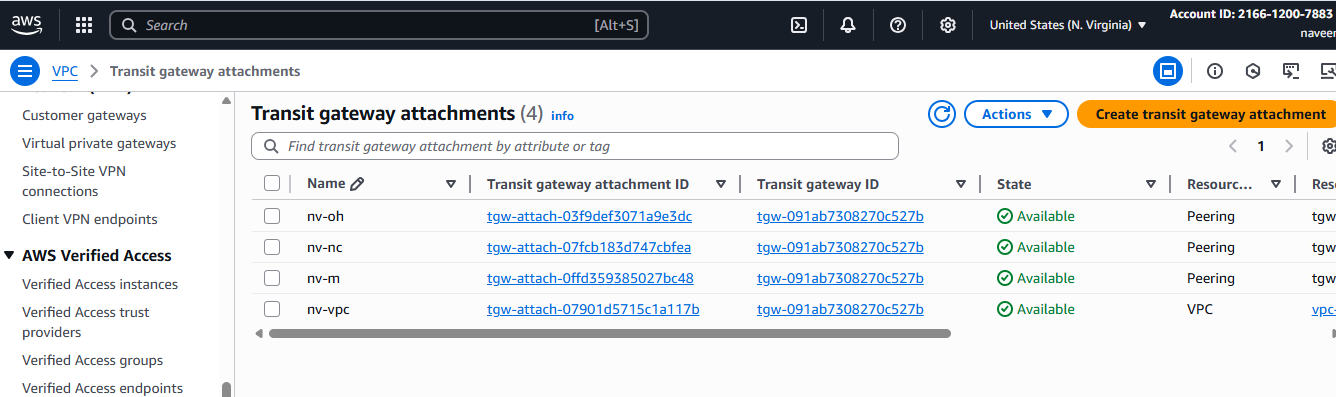
1. **Transit Gateway Attachments → Create**



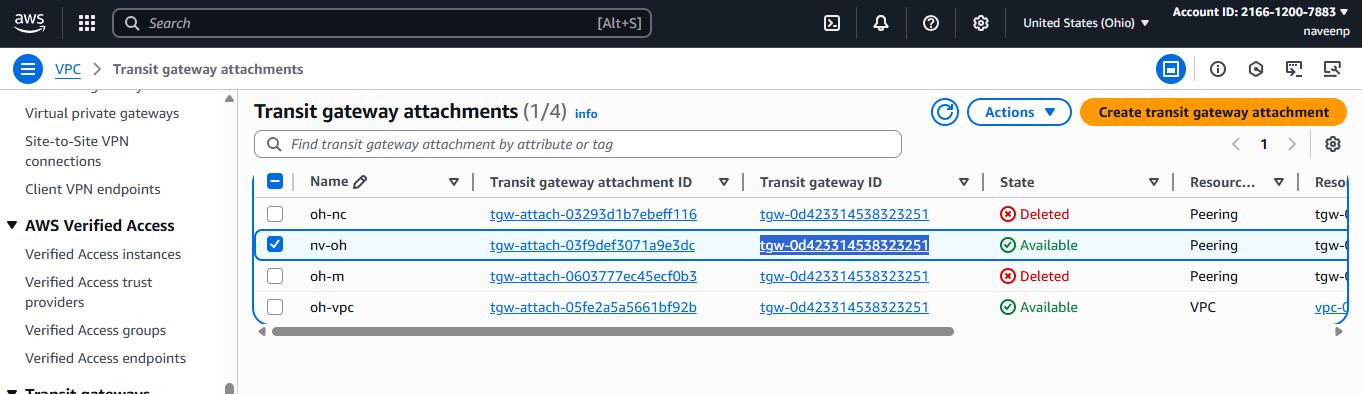


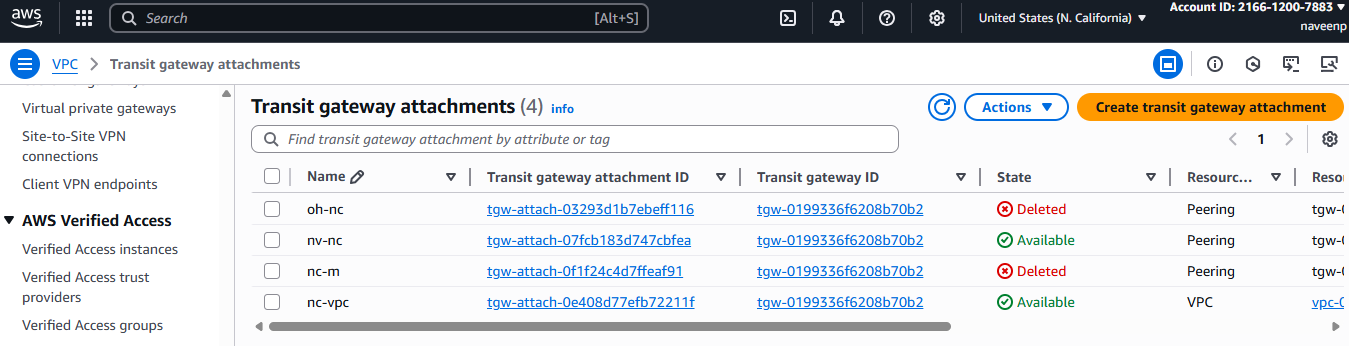


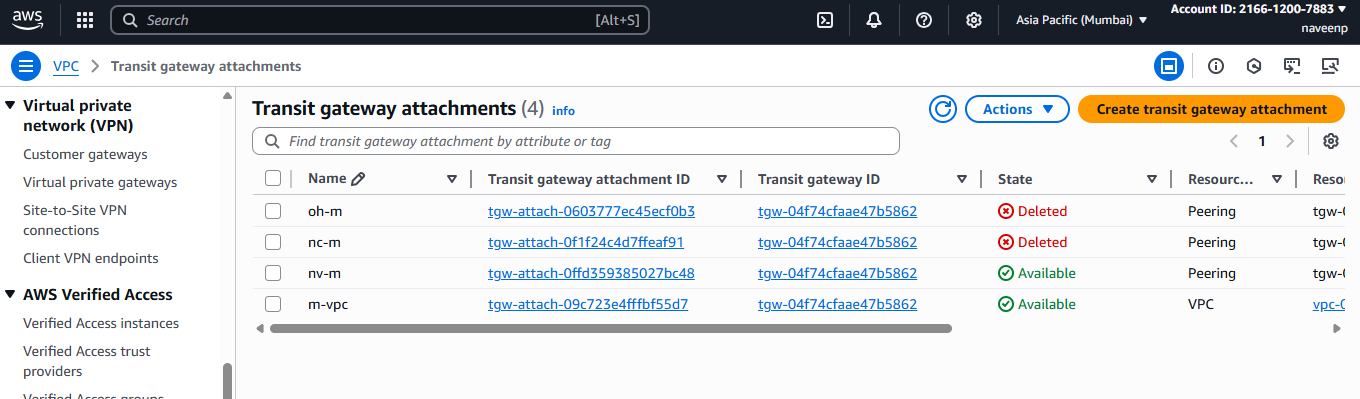




Now create Transit Gateway in vpc2,vpc3 and vpc4  
 in 4 different regions and attach with the Attach Each VPC to its Regional TGW







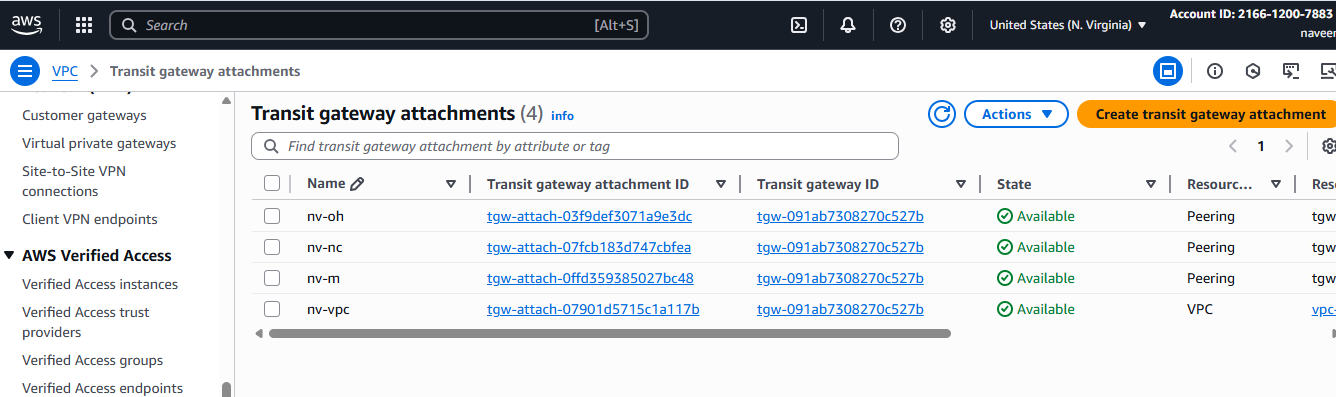
**STEP-5:**

## Establish Inter‑Region TGW Peering (Hub‑and‑Spoke)

### Create Peering Attachment (Requester Side)

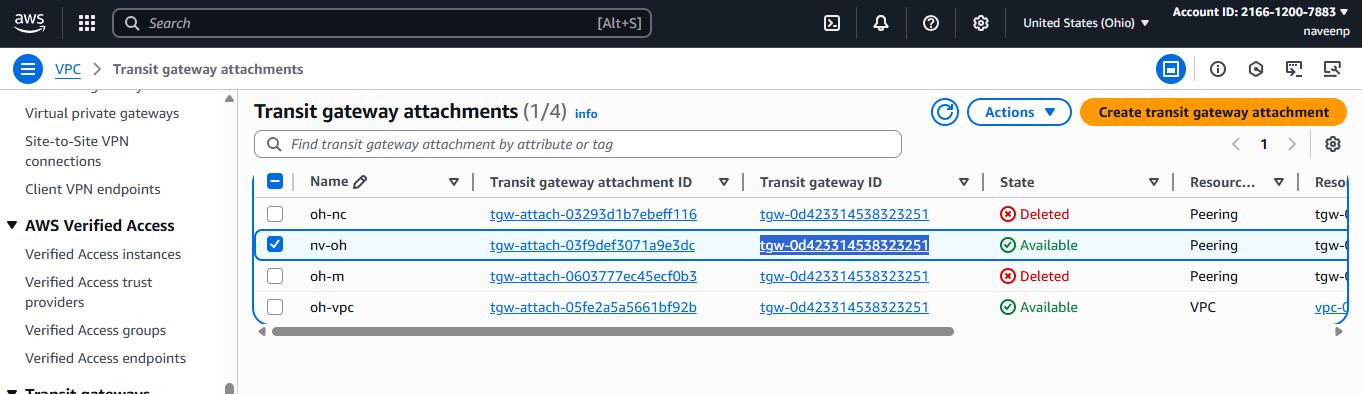
1. In the **AWS Console**, go to the **VPC Dashboard** in **Region A** (e.g., us-east-1).
2. On the left menu, choose **Transit Gateway Attachments → Create transit gateway attachment**.
3. Select:
   * **Attachment type** → Peering connection.
   * **Transit Gateway ID** → TGW-US-E1.-Virginia
   * **Region** → target region (e.g., us-west-2 ohio).
   * **Transit Gateway ID** of the peer (TGW-of-Ohio)

Click **Create transit gateway attachment**.

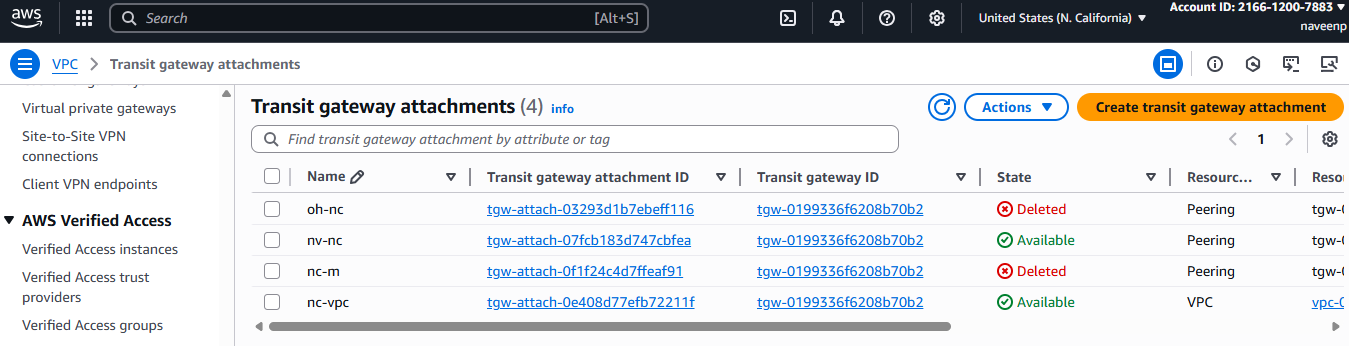


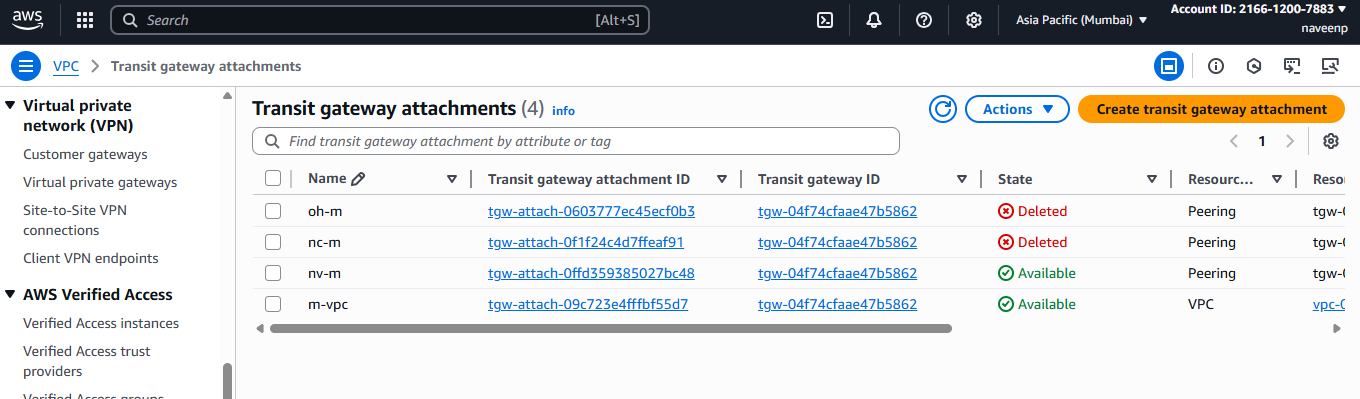
### .Accept Peering Attachment (Accepter Side)

1. **Switch to the peer region’s console (e.g.,Ohio-region).**
2. **Go to Transit Gateway Attachments.**
3. **You’ll see a pending peering request.**
4. **Select it → Actions → Accept.**



* Similarly configure with the vpc3 and vpc4 and
* also configure the same with all the Regions with one to all

****

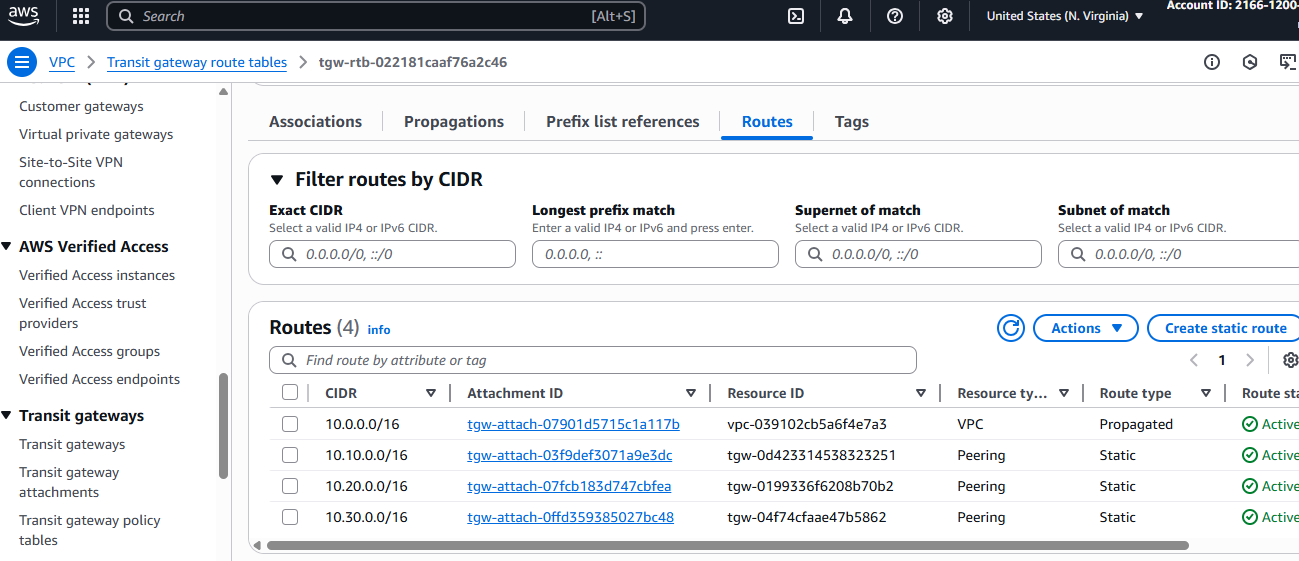
****

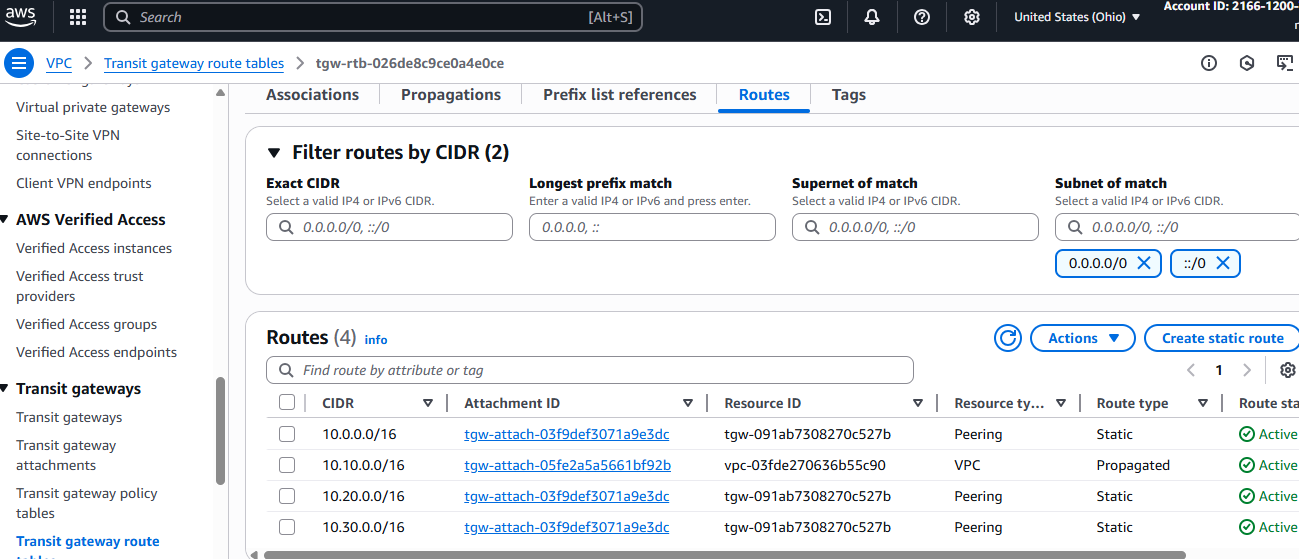
**Step-6:**

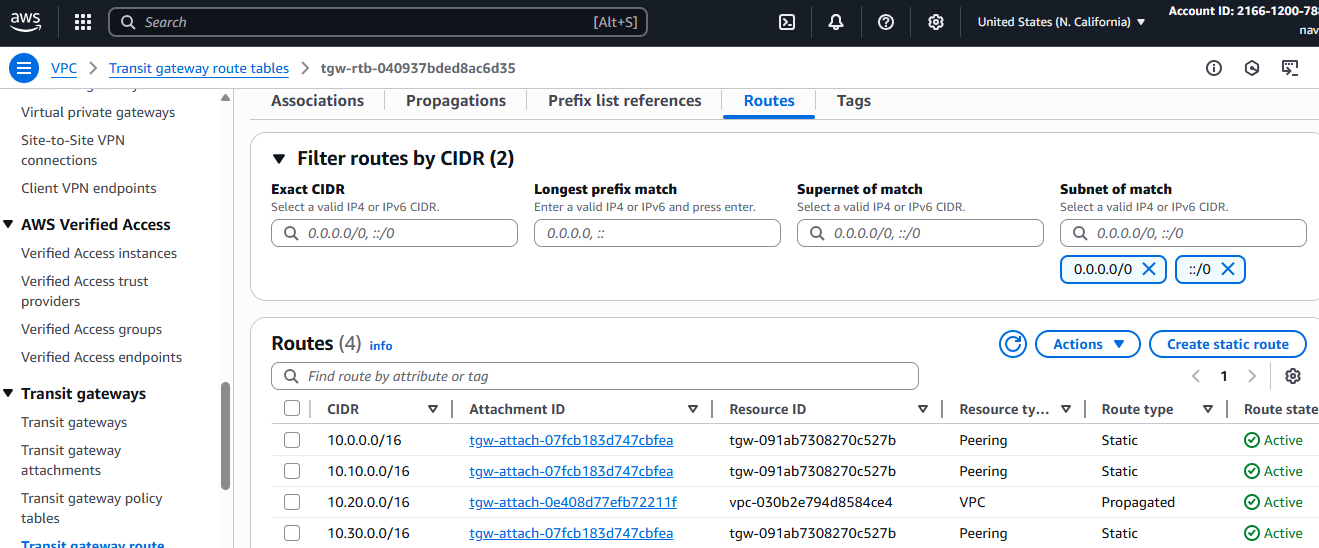
**Configure TGW Route Tables**

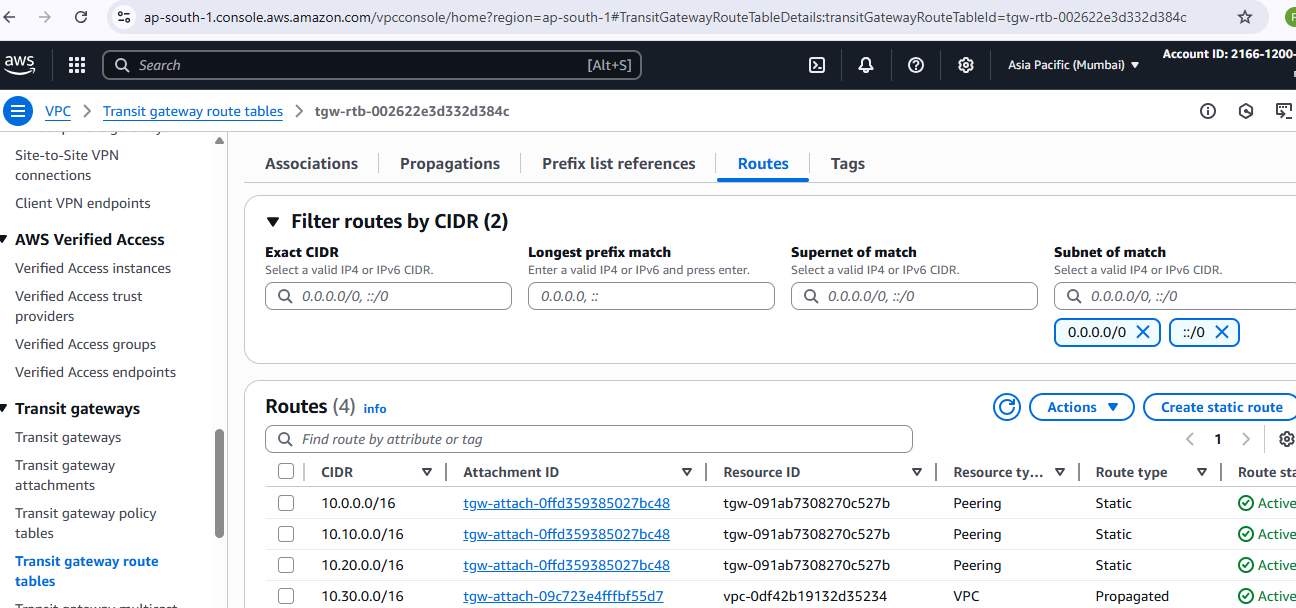
* After the peering is active, you must **update TGW route tables** so traffic can flow.

1. In each TGW’s **Route Table**, add routes for the remote VPC CIDRs.







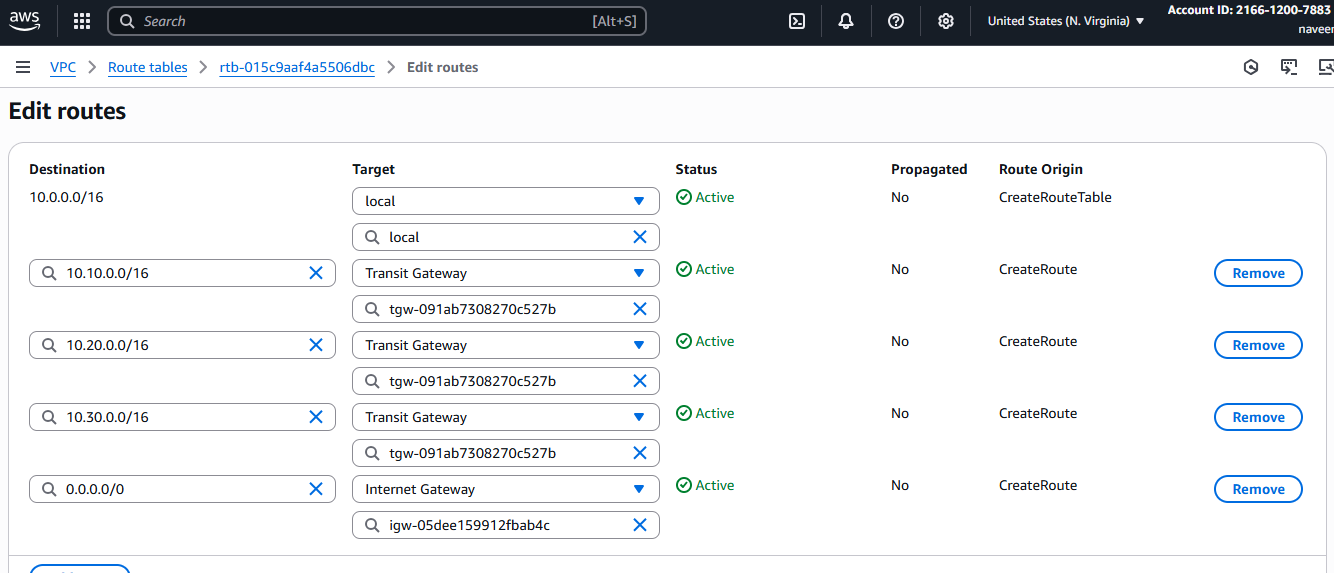


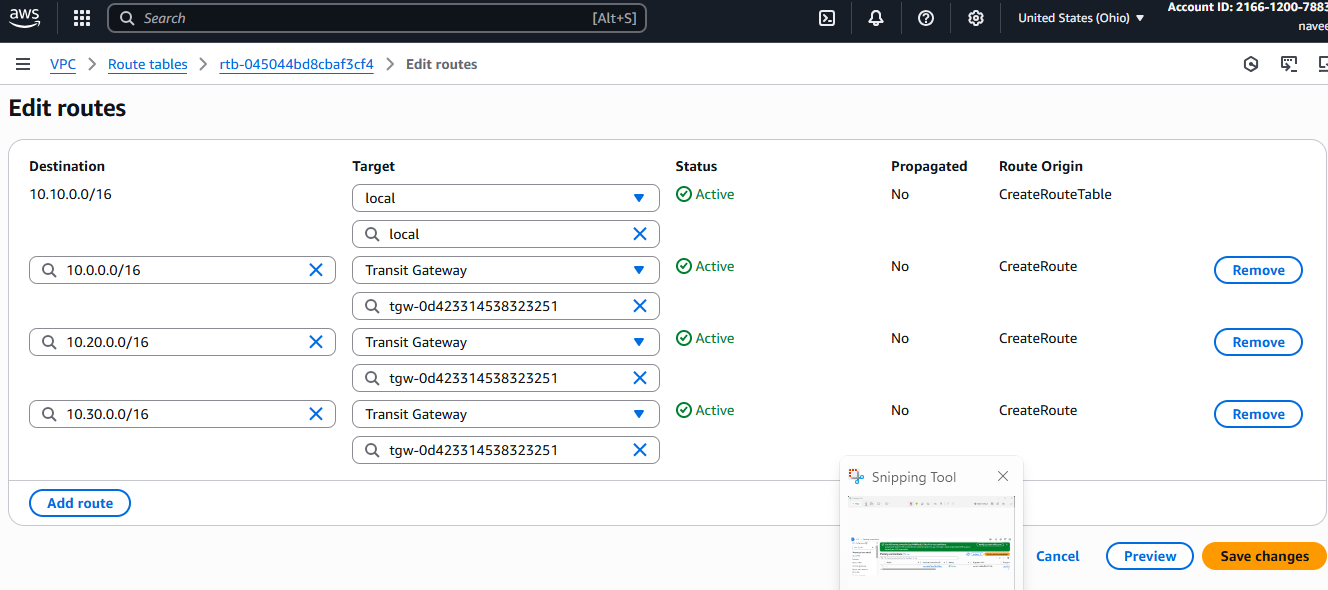
**Step -7 Update VPC Route Tables in all regions**

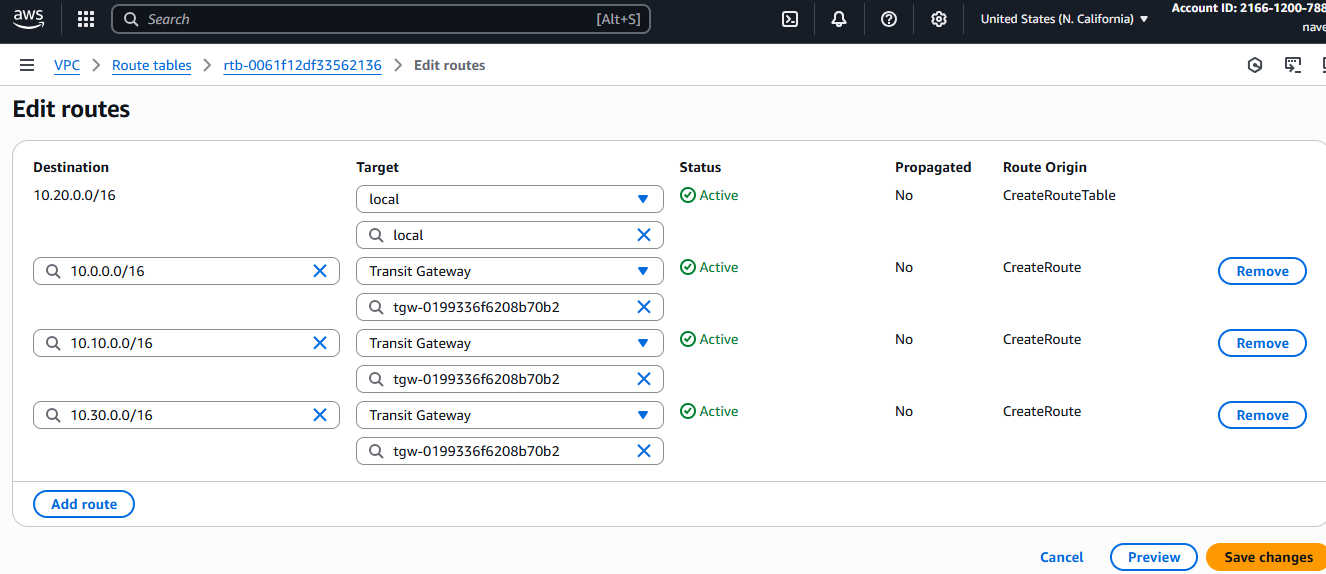
For each **VPC’s private subnet RTB**:

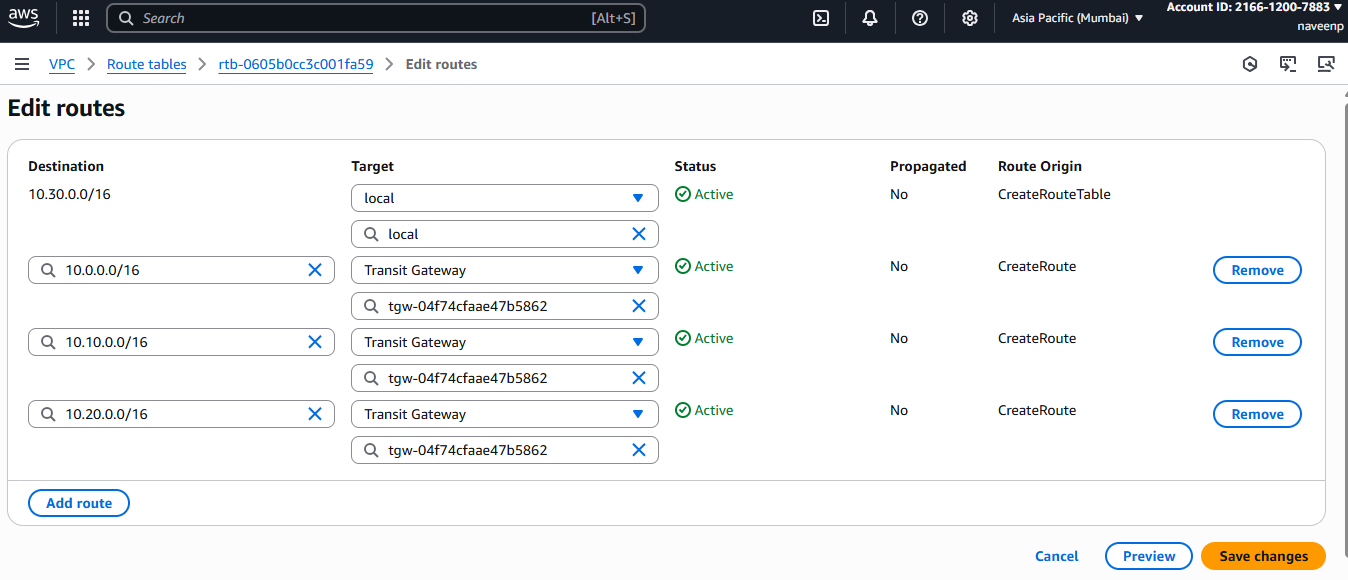
Add a route to the **other VPC CIDRs via TGW attachment**.

Repeat in all the other Regions

****

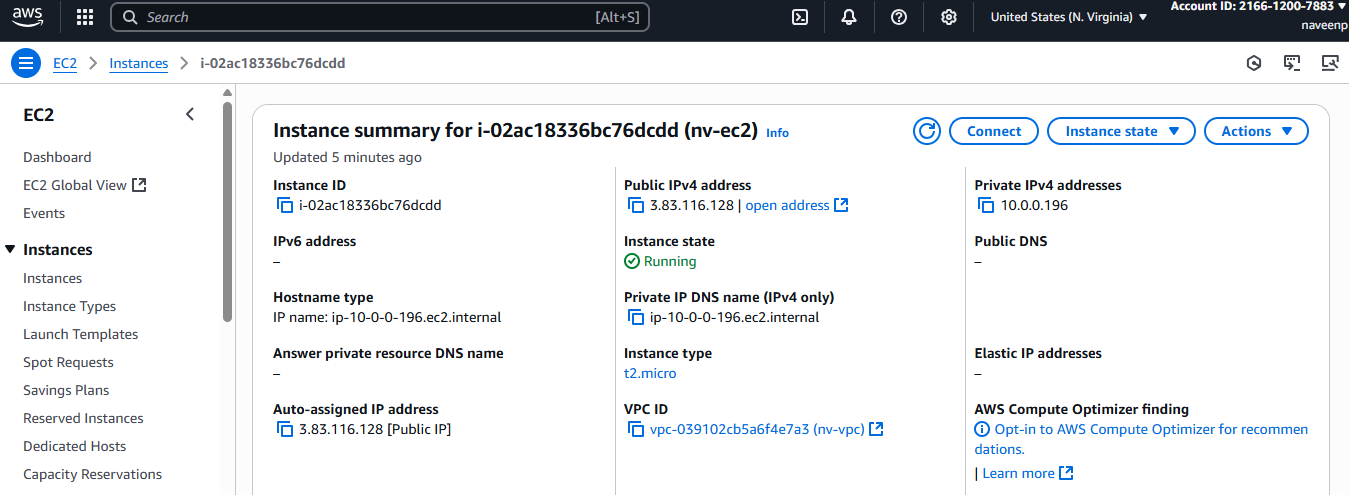
****

****

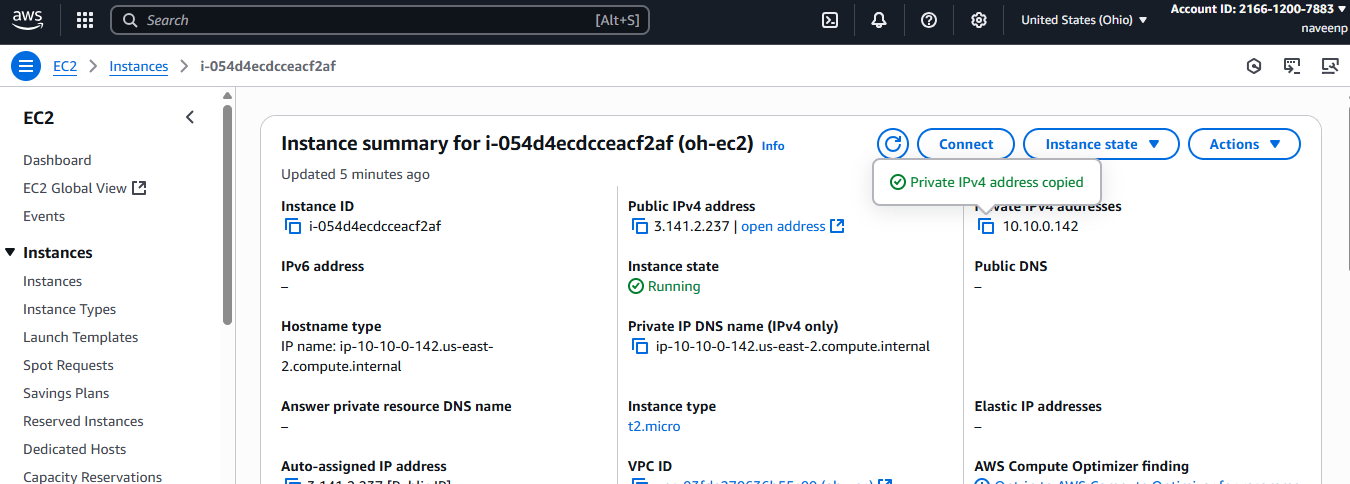
****

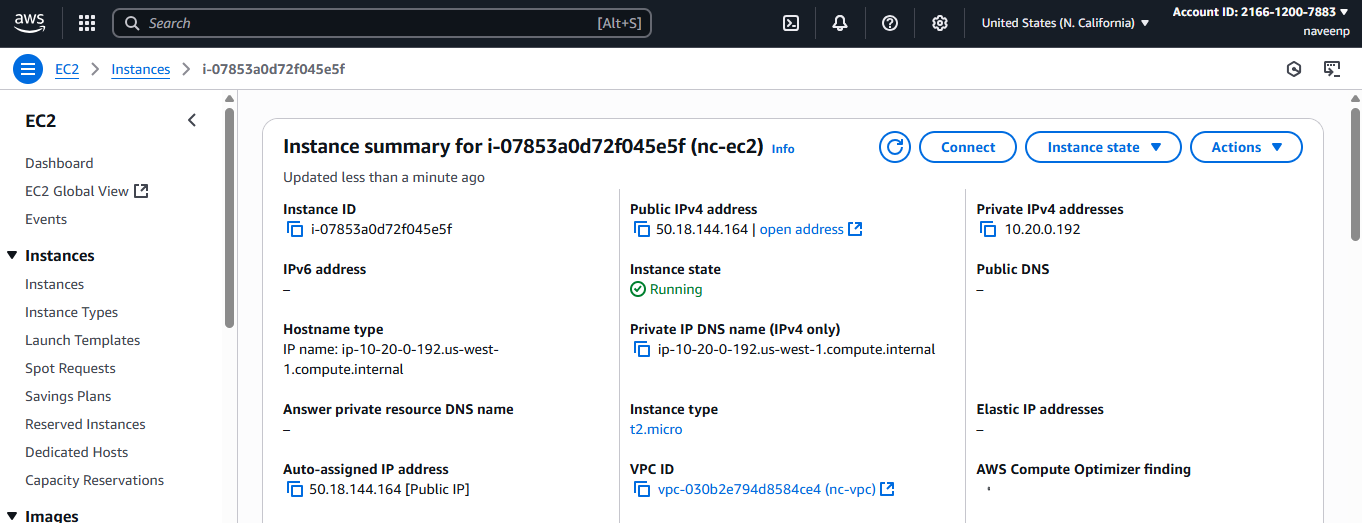
**Step-8 :Verification**

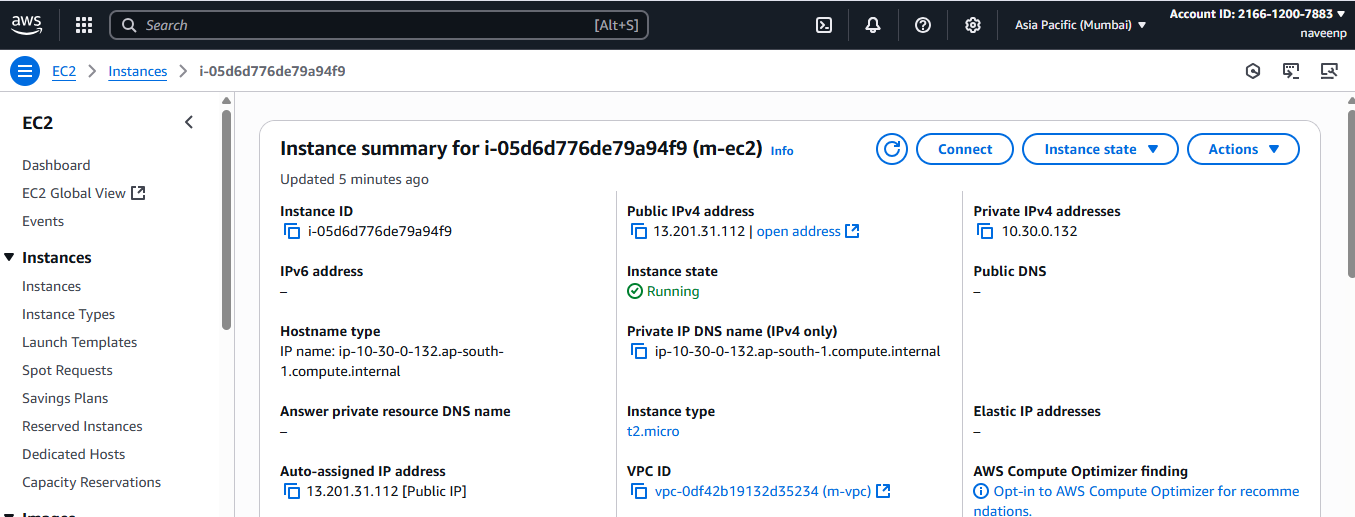
**EC2-to-EC2 connectivity:**

* **Launch 1 EC2 in each VPC in public subnet- - Vpc-1-( N.Virginia)**
* ****

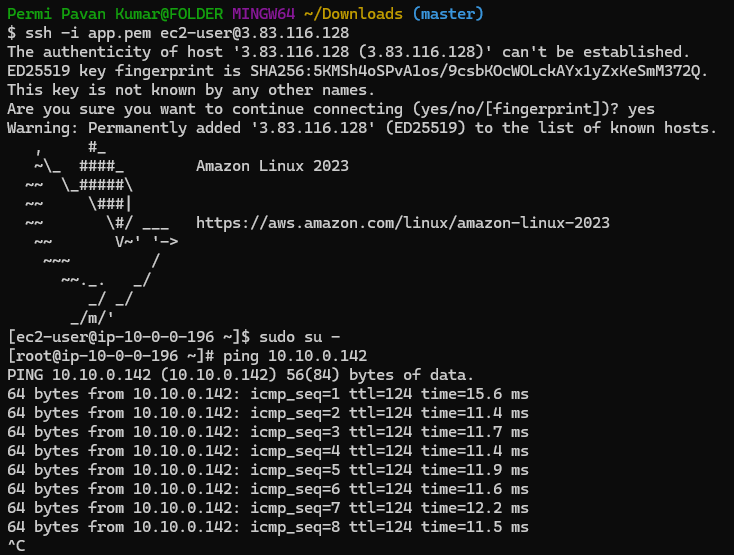
**Remaining all Launch 1 EC2 in each VPC private subnet.**

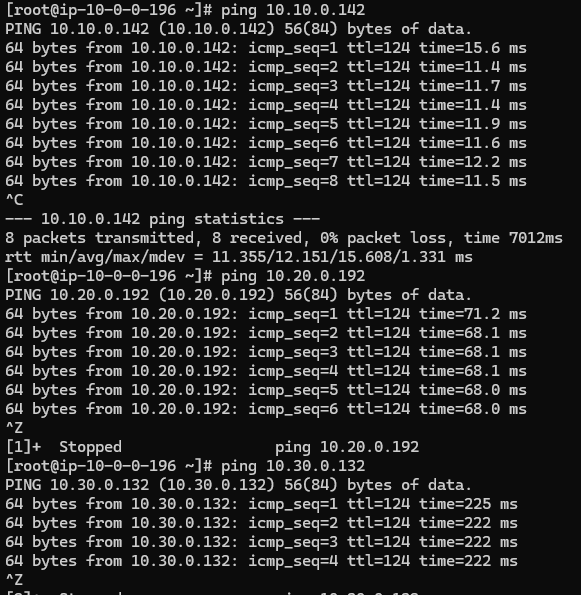
****

****

****

* **From Virginia EC2 → ping Ohio EC2’s private IP.**
* **From Virginia EC2 → ping California EC2’s private IP.**
* **From Virginia EC2 → ping Mumbai EC2’s private IP.**







1. **Configure VPC endpoints to securely access AWS services without internet gateways or NAT gateways, ensuring data privacy and minimizing exposure to external threats.**

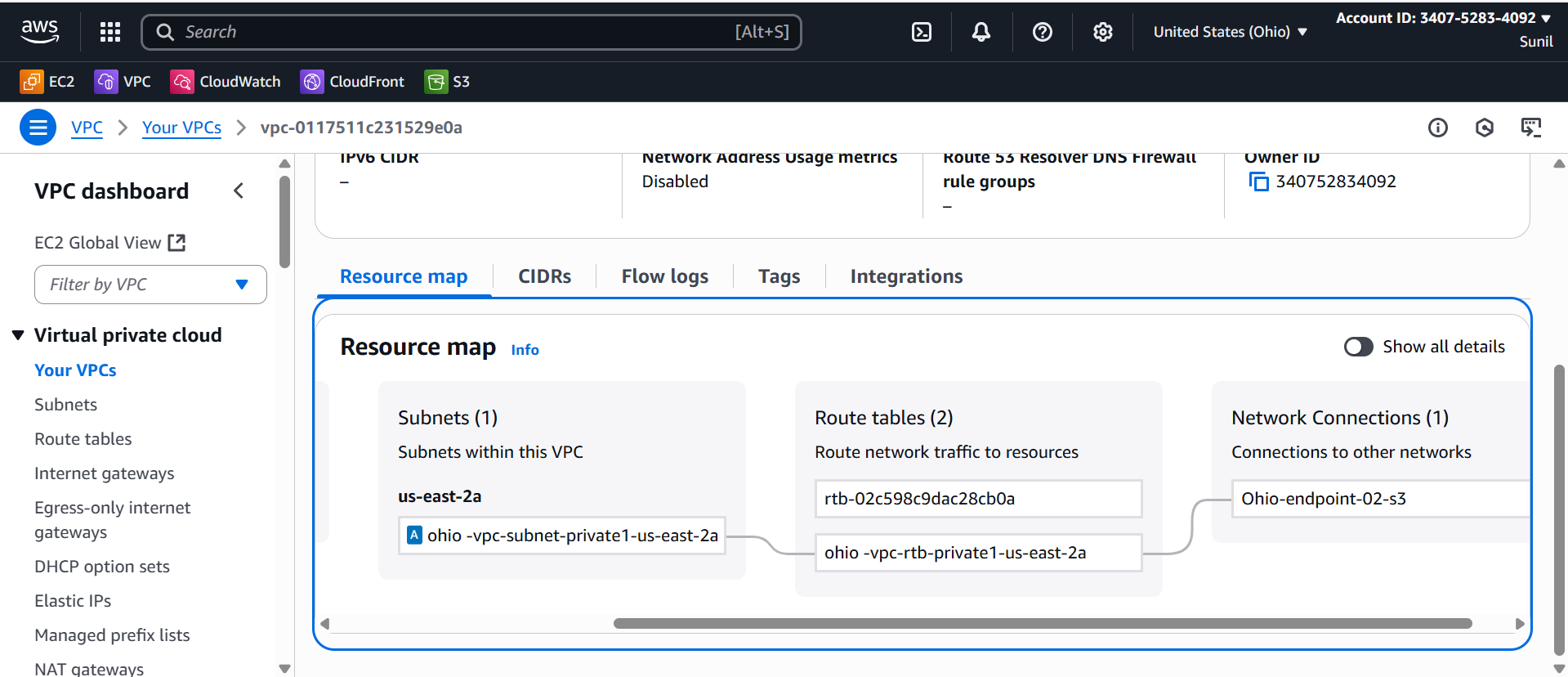
**Configure VPC Endpoints (Private access to AWS services)**

For **each VPC** → Go to **Endpoints → Create Endpoint**.

Choose services:

* Gateway Endpoint: **S3**

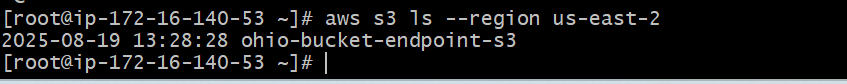
Create an S3 endpoint in Ohio



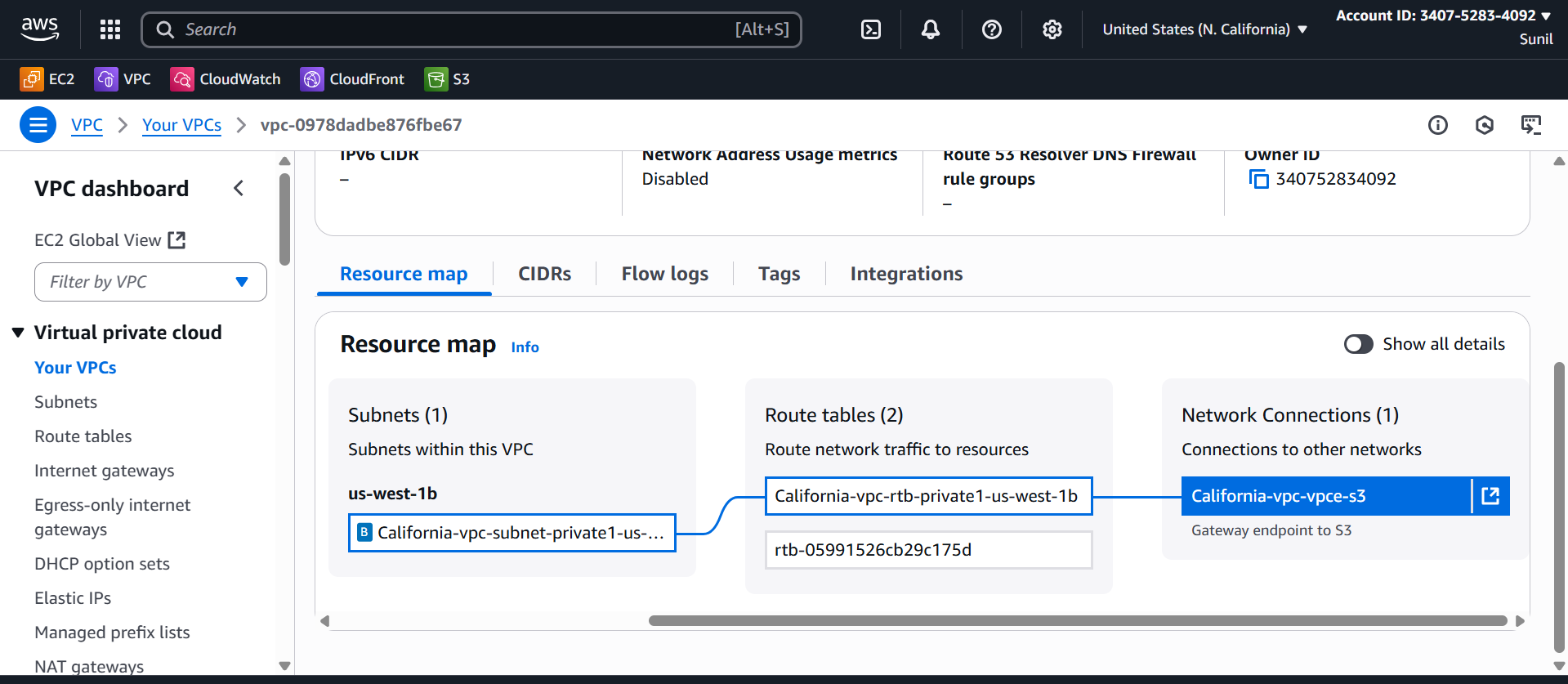
From N.virgina Ec2­--🡪ssh to Ohio private ip

Login to ohio ec2 and configure aws   
and check the access to resources like s3

aws s3 ls --region us-east-2



Create an S3 endpoint California and connect with Ssh private ip in ohio



From Ohio -🡪jump into California ec2 private ip

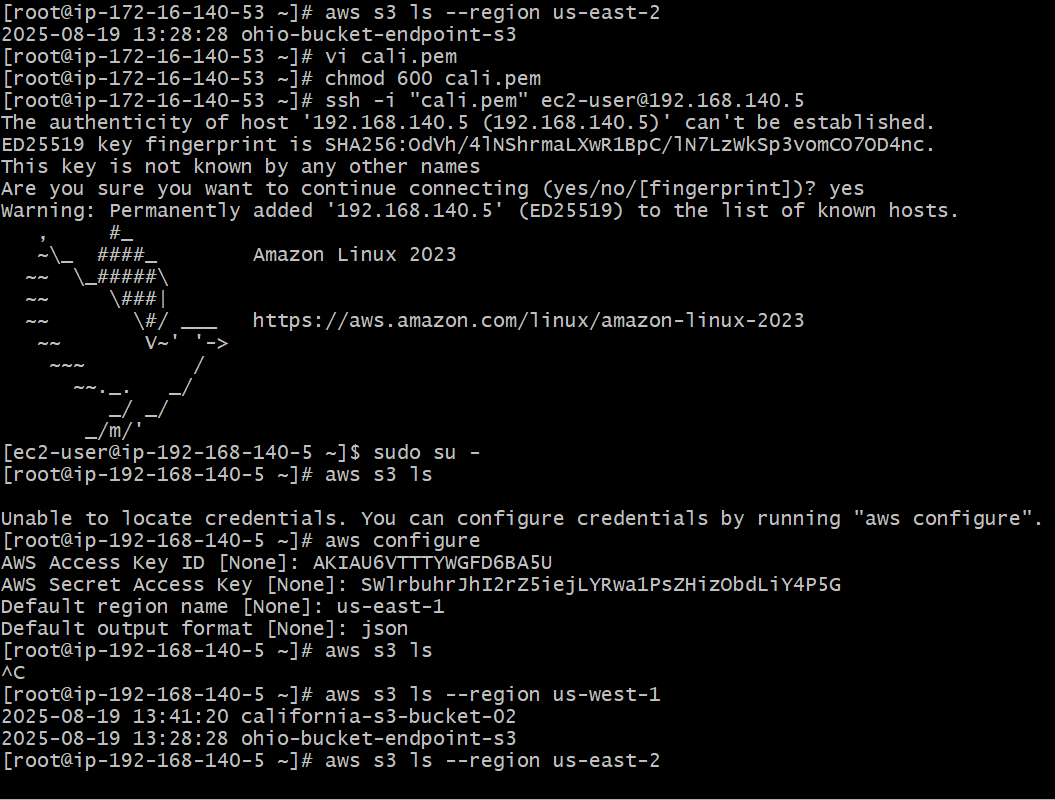
And give permissions

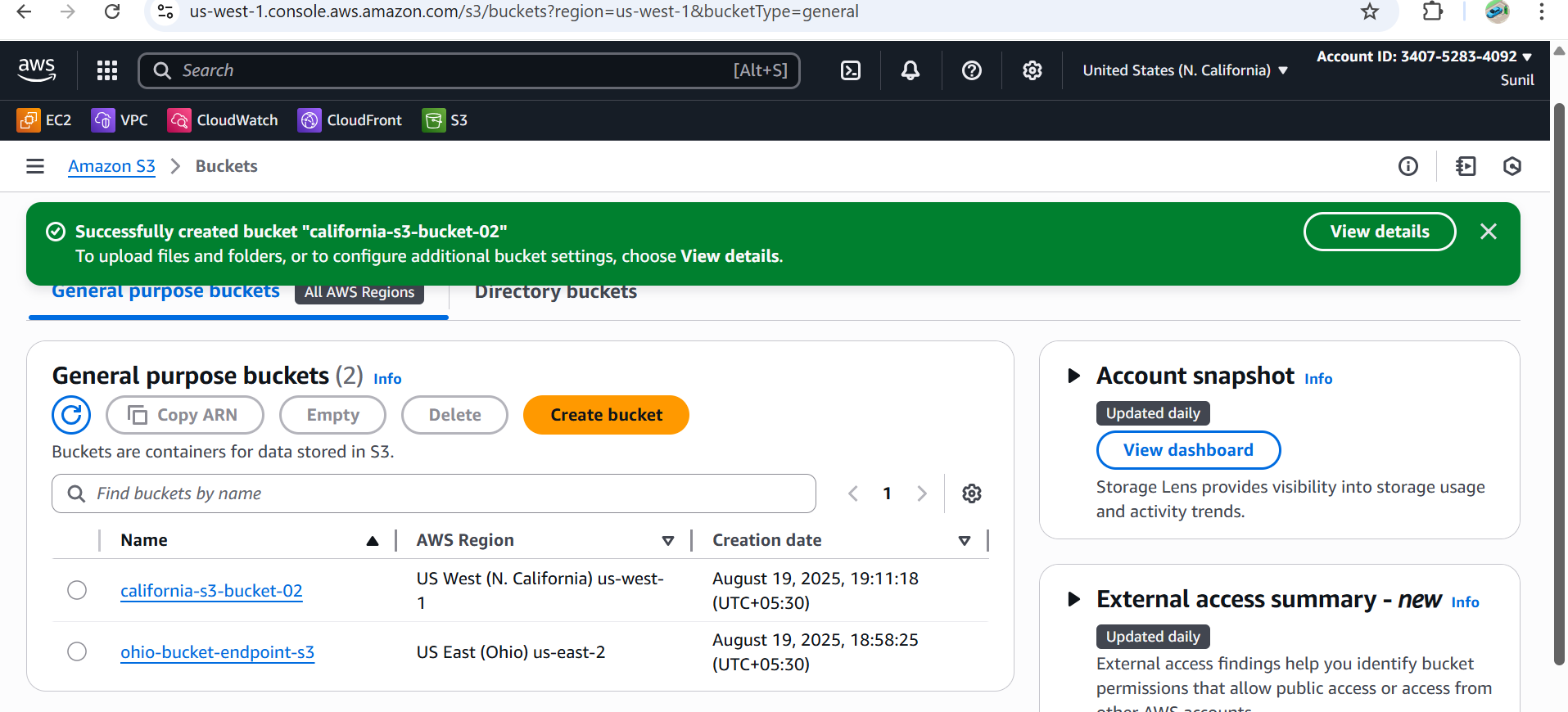
And configure aws in this region

Create a s3 bucket in California region

Verify using

aws s3 ls --region us-west-1





Simlarly Configure in Other regions and Verify