**Second Challenge**

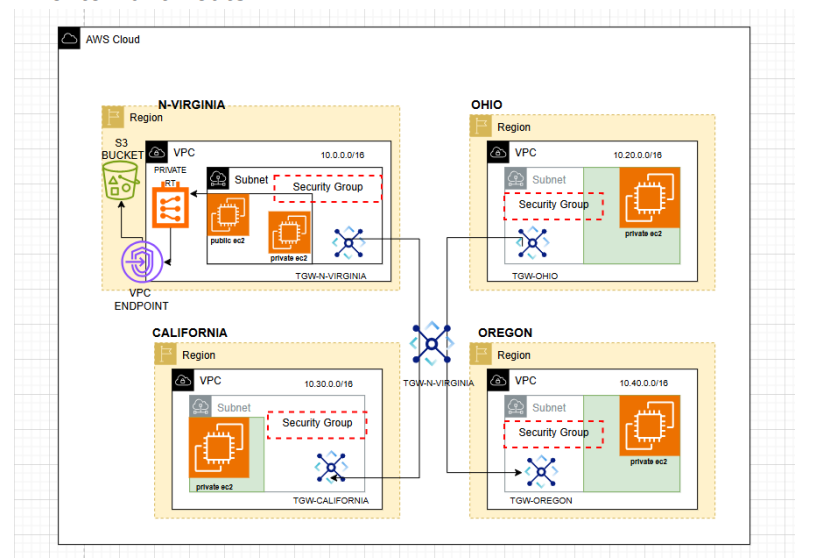
**Multi-Region AWS Transit Gateway and VPC Endpoints Setup for Secure Multi-VPC Networking**

**Scenario Overview**

A large enterprise is transitioning its on-premises infrastructure to AWS and requires a secure, scalable network design for multiple VPCs distributed across different AWS regions. The architecture must enable efficient inter-VPC communication and private access to AWS services without exposing traffic to the public internet.

**Objectives**

* Design a hub-and-spoke network topology using AWS Transit Gateway to enable seamless communication across multiple VPCs in various regions.
* Configure VPC endpoints within each VPC to securely access AWS services (such as S3, SSM, ECR) privately, eliminating the need for internet gateways or NAT instances.



* Avoid using default VPCs to maintain strict control over network resources.

**Step 1: Create Custom VPCs in Four AWS Regions**

Create one VPC in each of the following regions, ensuring unique CIDR blocks and no default VPC usage:

* **US East 1 (N. Virginia) - Hub:** 10.0.0.0/16
* **US East 2 (OHIO) - Spoke 1:** 10.1.0.0/16
* **US WEST 1 (N. CALIFORNIA) - Spoke 2:** 10.2.0.0/16
* **US WEST 1 (OREGON) - Spoke 3:** 10.3.0.0/16

For each VPC:

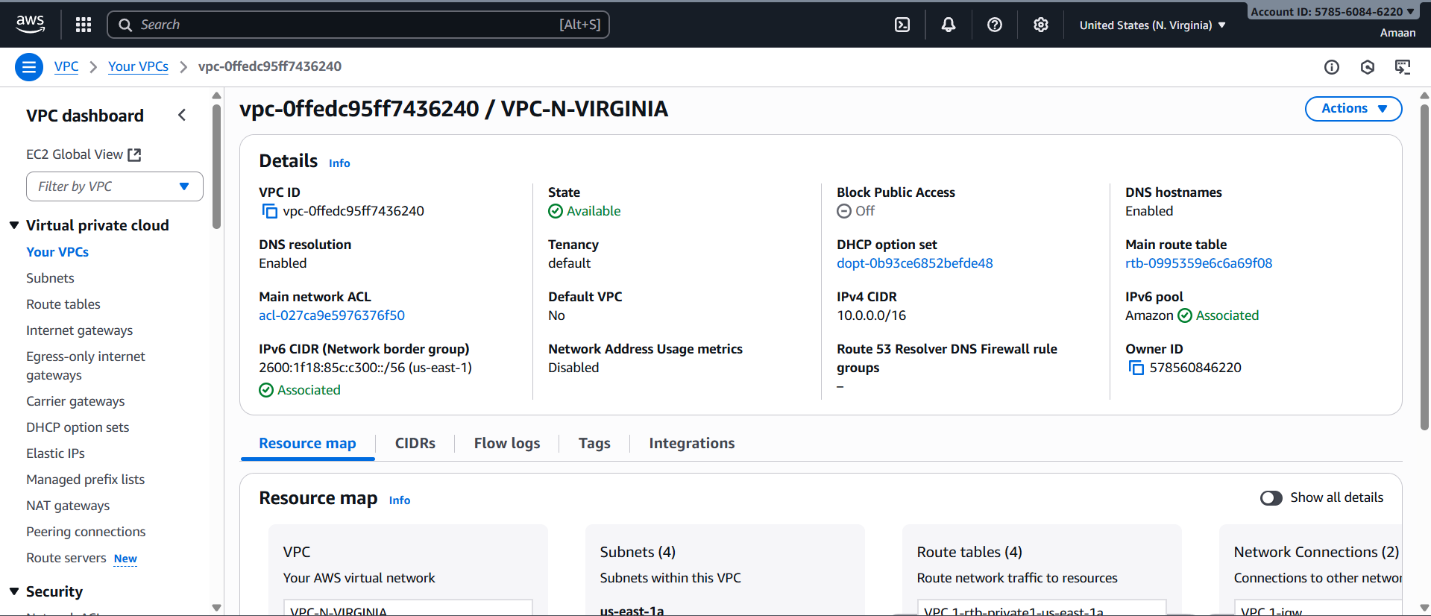
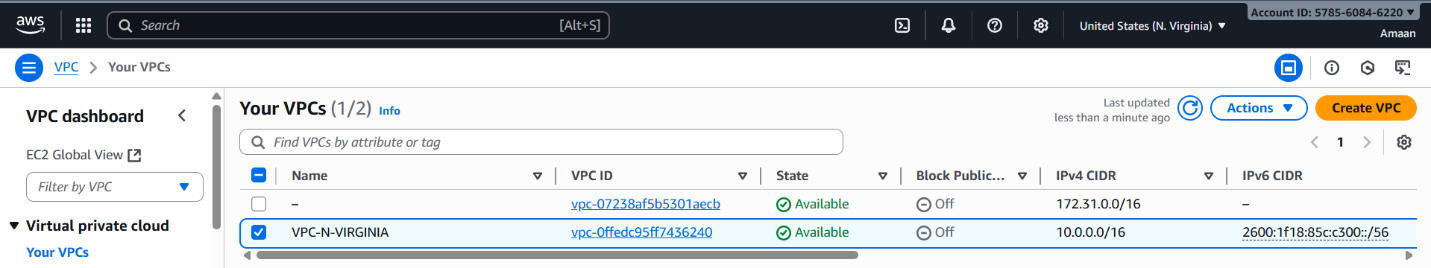
* Create at least two private subnets in different availability zones.
* Do not attach an Internet Gateway or NAT Gateway; subnets remain private.

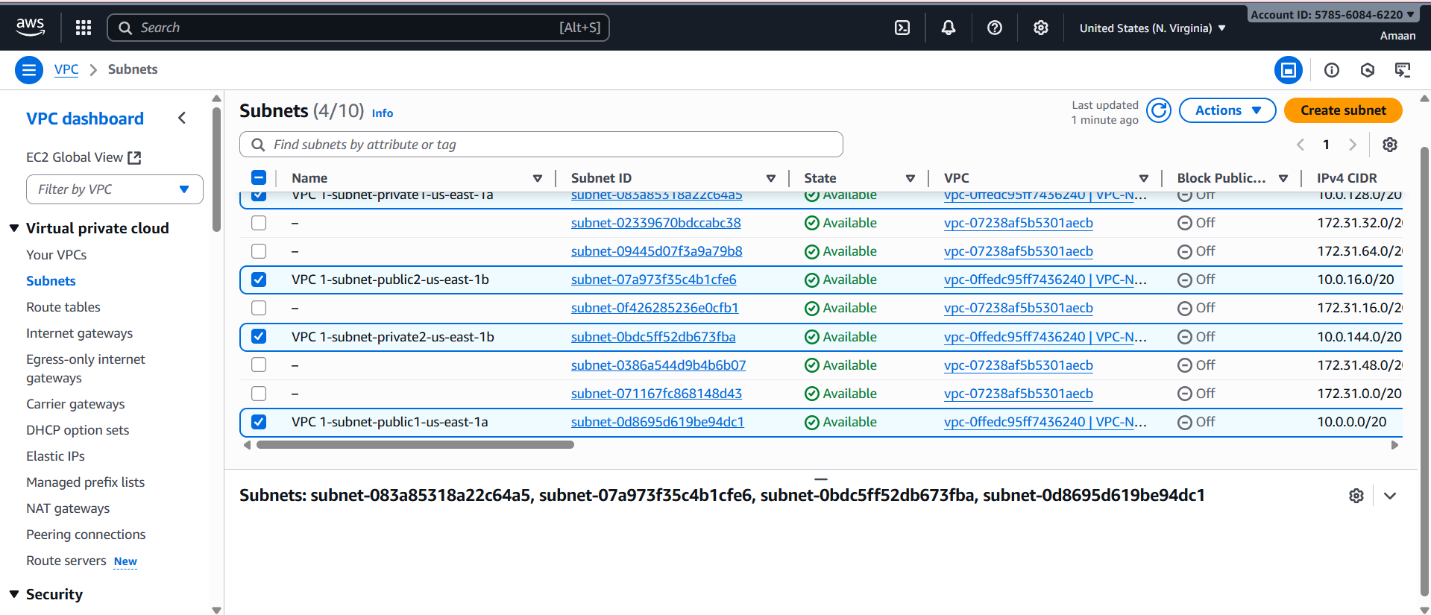
To create Custom VPCs in four AWS regions, the general steps are:

1. Open the Amazon VPC console.
2. Select the desired AWS region from the top-right corner of the console.
3. Click "Create VPC" or "Launch VPC Wizard" on the VPC dashboard.
4. Choose "VPC and more" or a configuration like "VPC with Public and Private Subnets."
5. Enter a name tag for the VPC.
6. Specify an IPv4 CIDR block (e.g., 10.0.0.0/16).
7. (Optional) Enable IPv6 CIDR block.
8. Choose tenancy (default or dedicated).
9. Customize the number of subnets and availability zones if needed.
10. Create the VPC.
11. Repeat for each additional region by switching the region in the console and repeating the steps.

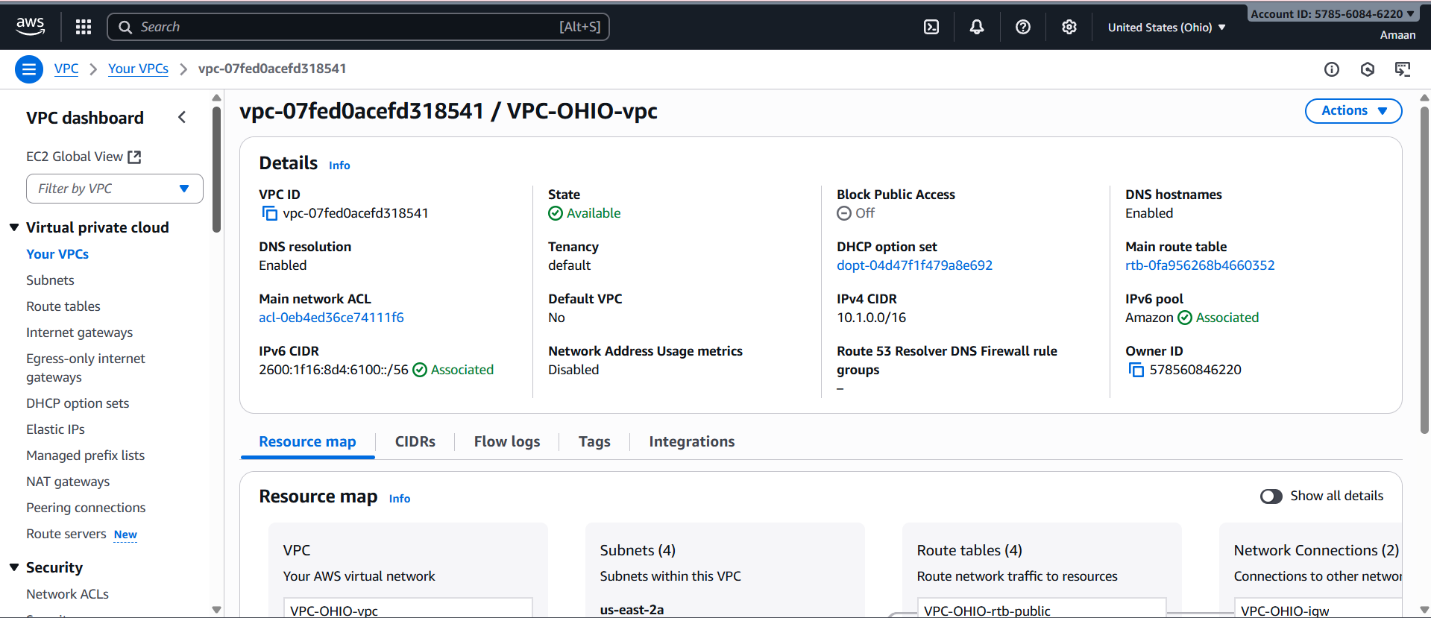
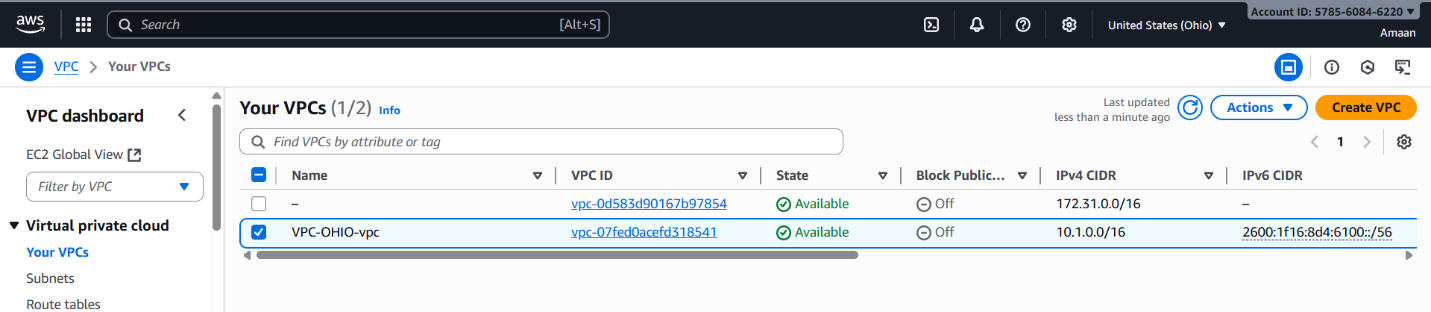
This procedure allows creating distinct custom VPCs in multiple regions.

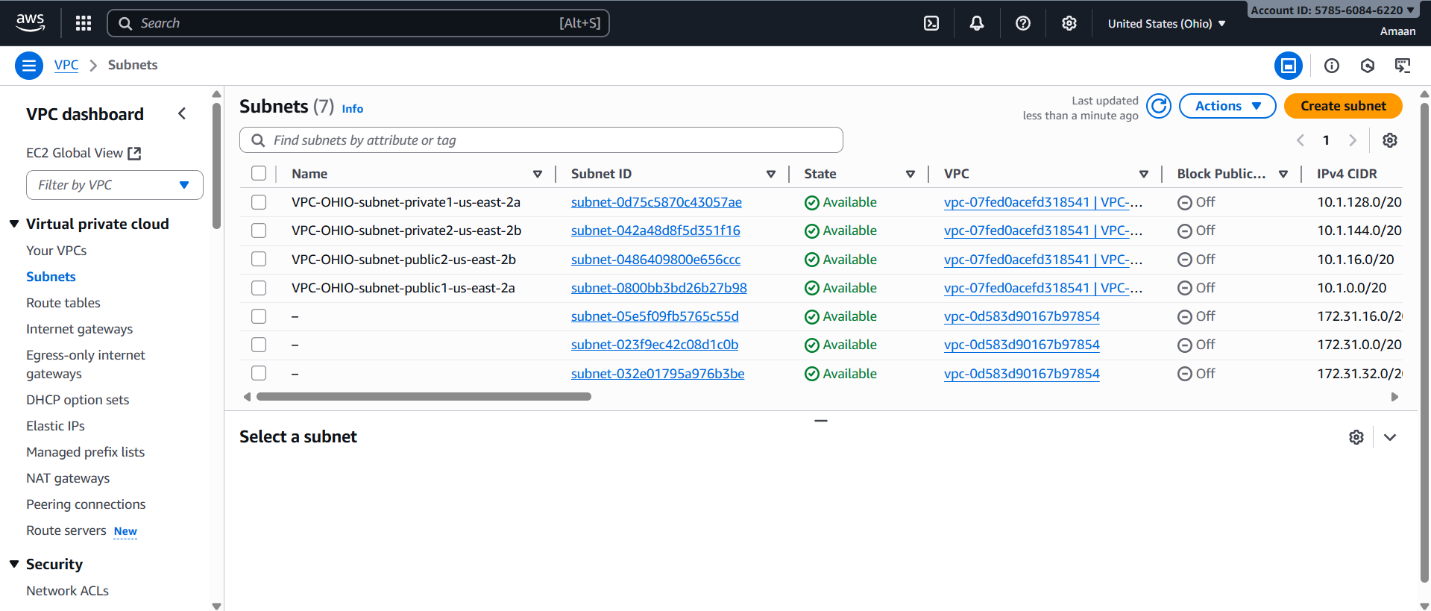
VPC-N-VIRGINIA



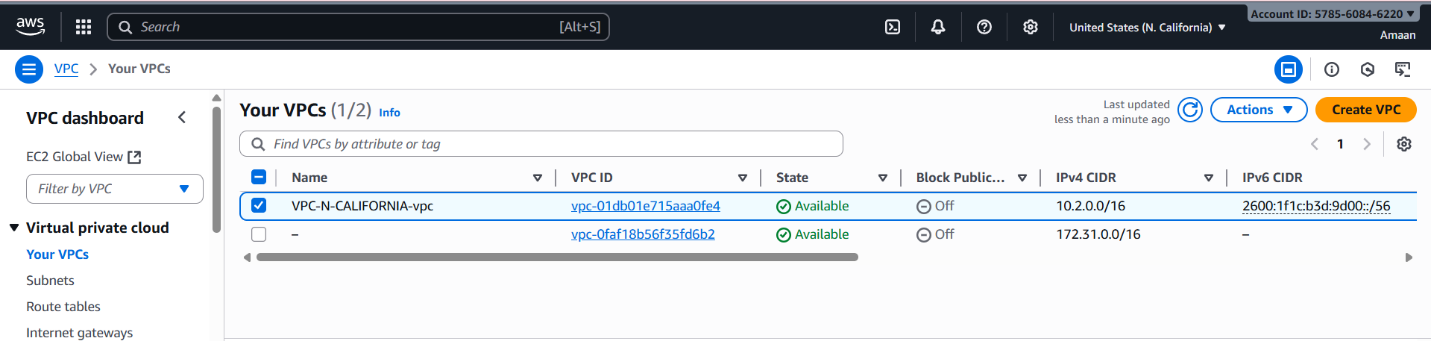


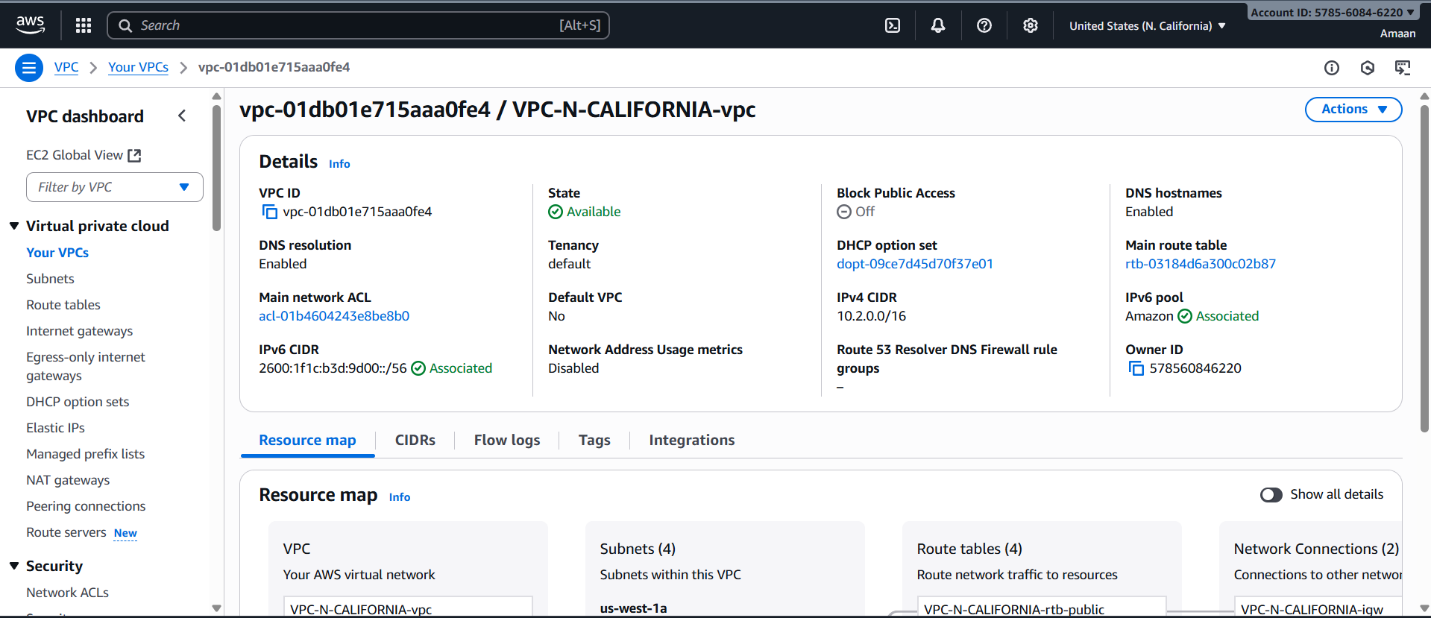
VPC-OHIO

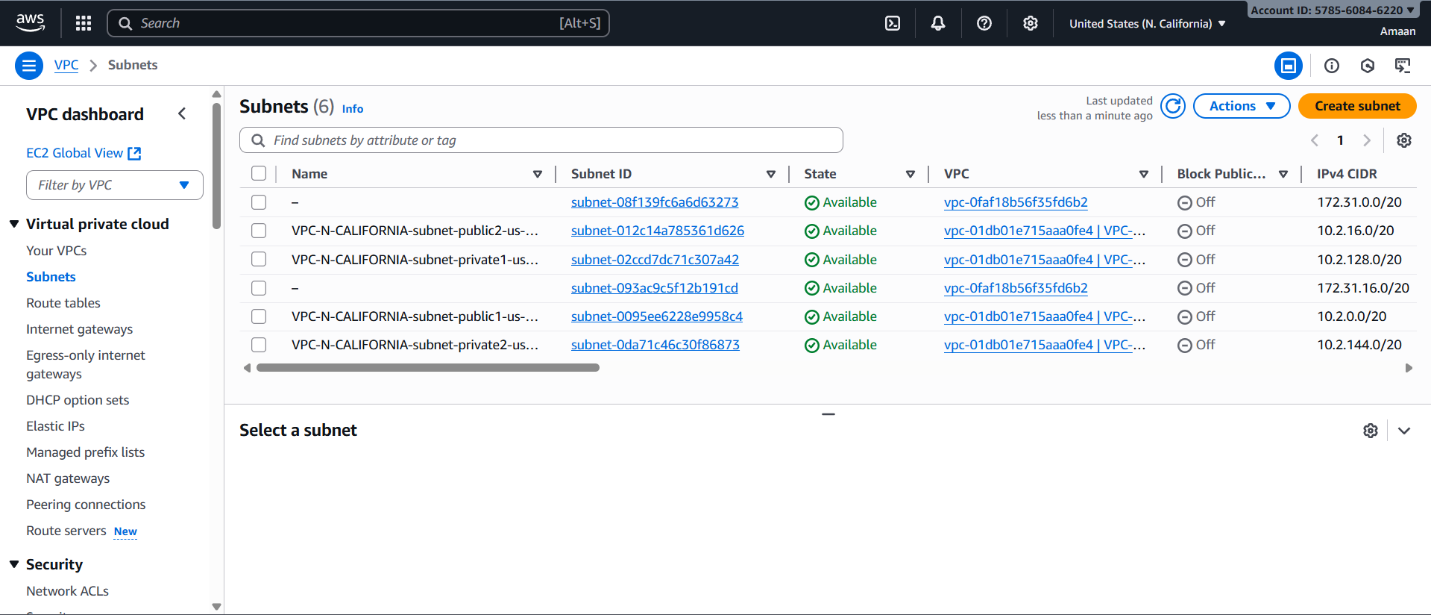




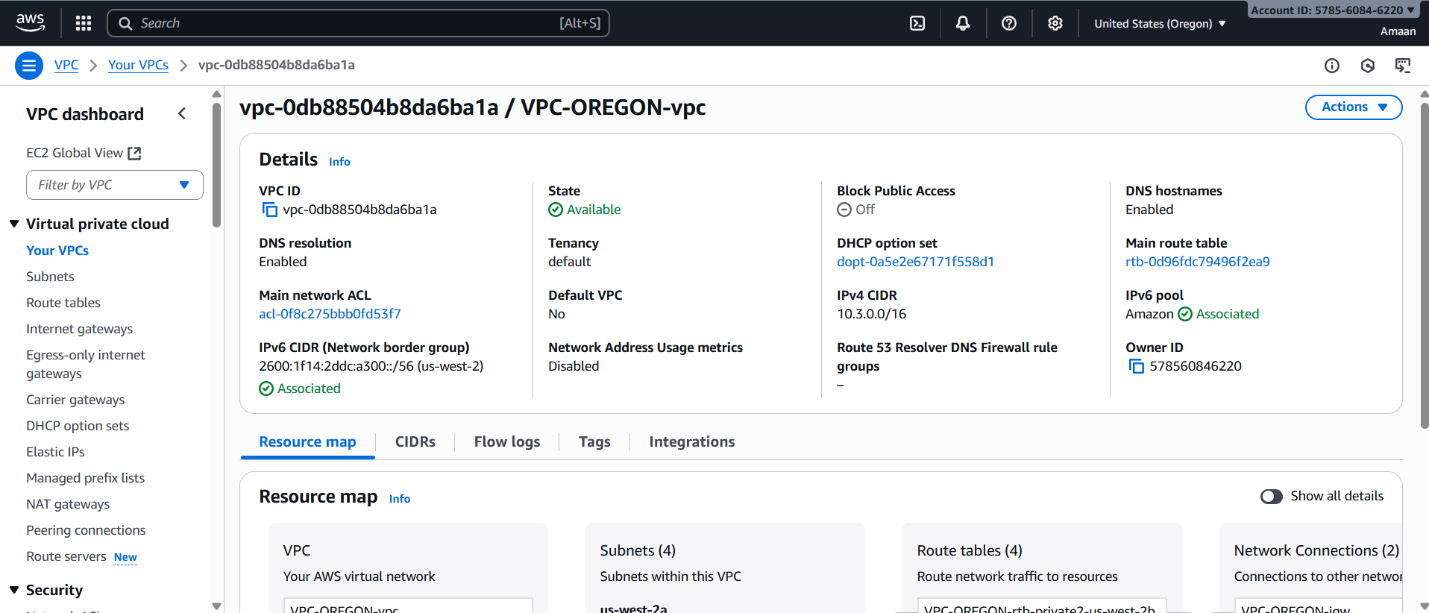
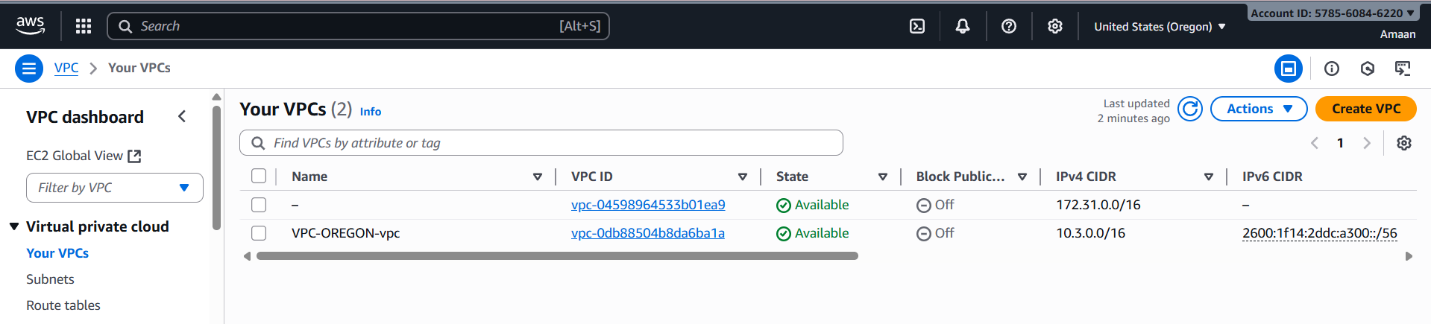
VPC-N-CALIFORNIA

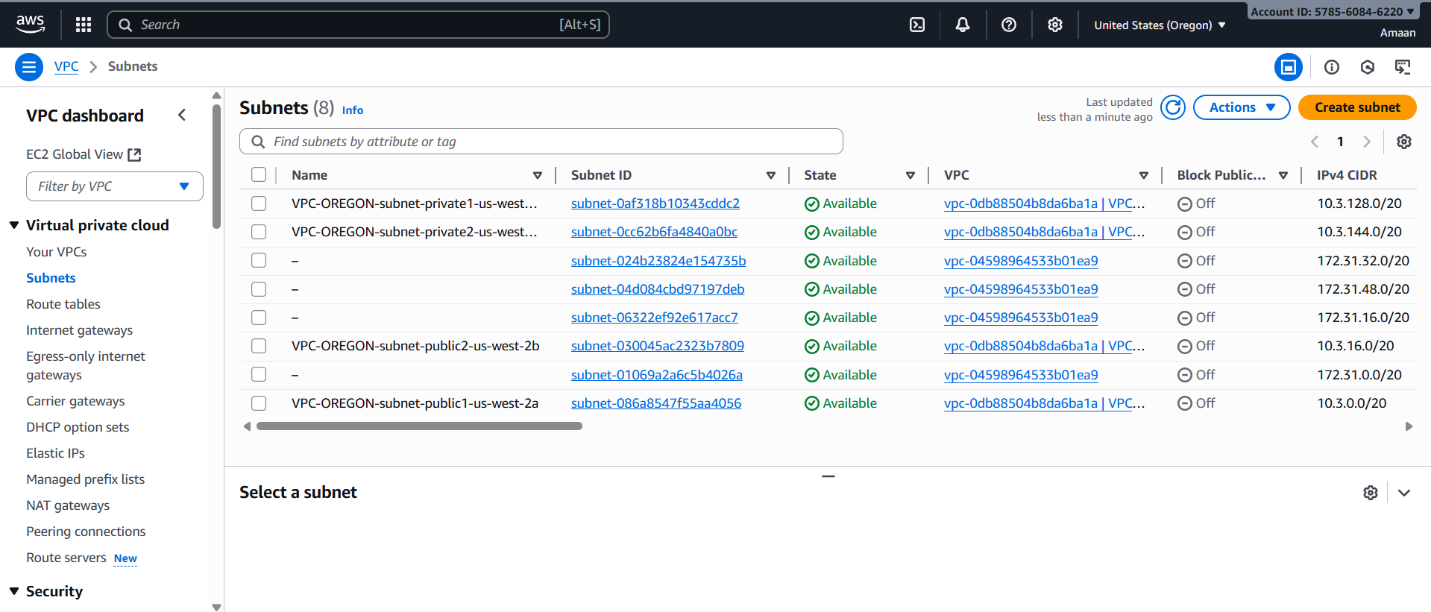






VPC-OREGON





**Step 2: Deploy a Transit Gateway in Each Region**

For each region's VPC:

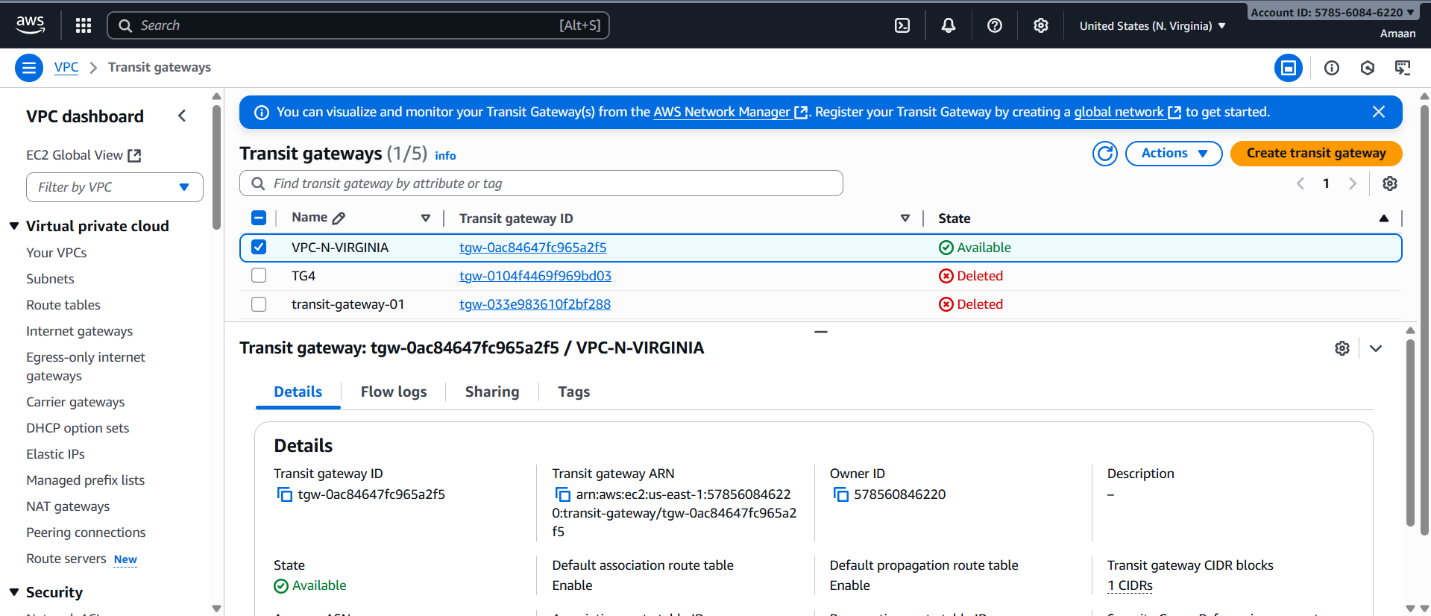
* Navigate to the Transit Gateway console and create a new Transit Gateway.
* Use default ASN unless specific requirements exist.
* Enable DNS support; disable multicast if not needed.
* Record the Transit Gateway ID immediately after creation.

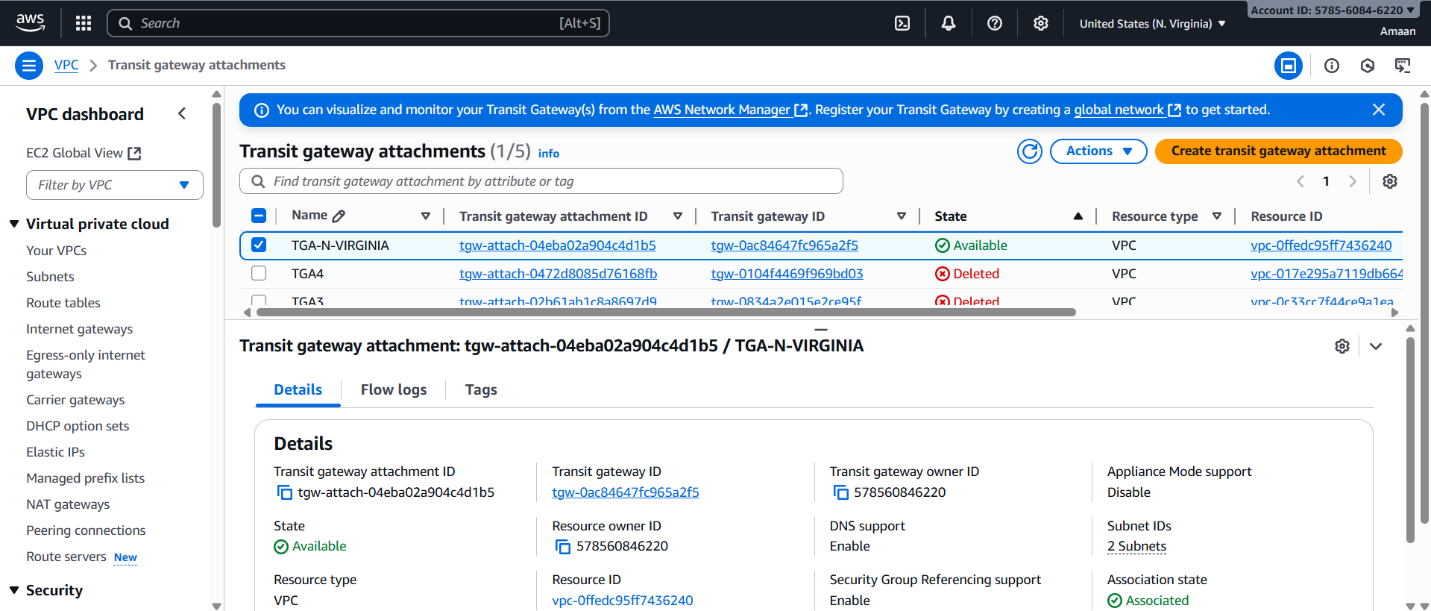
To deploy a Transit Gateway in each AWS region, here is a summary of the key steps from the AWS Console:

1. Open the Amazon VPC console.
2. Select the region in which you want to create the Transit Gateway.
3. In the navigation pane, choose "Transit Gateways."
4. Click "Create transit gateway."
5. (Optional) Enter a Name tag and description for easy identification.
6. Configure the options:
   * Specify the Amazon side Autonomous System Number (ASN) for Border Gateway Protocol (BGP). Use a unique ASN per region if multi-region.
   * Optionally enable DNS support, VPN ECMP support, default route table association and propagation, and multicast support.
7. (Optional) Configure cross-account sharing and auto-accept attachments if needed.
8. (Optional) Add Transit Gateway CIDR blocks for routing.
9. Click "Create transit gateway."

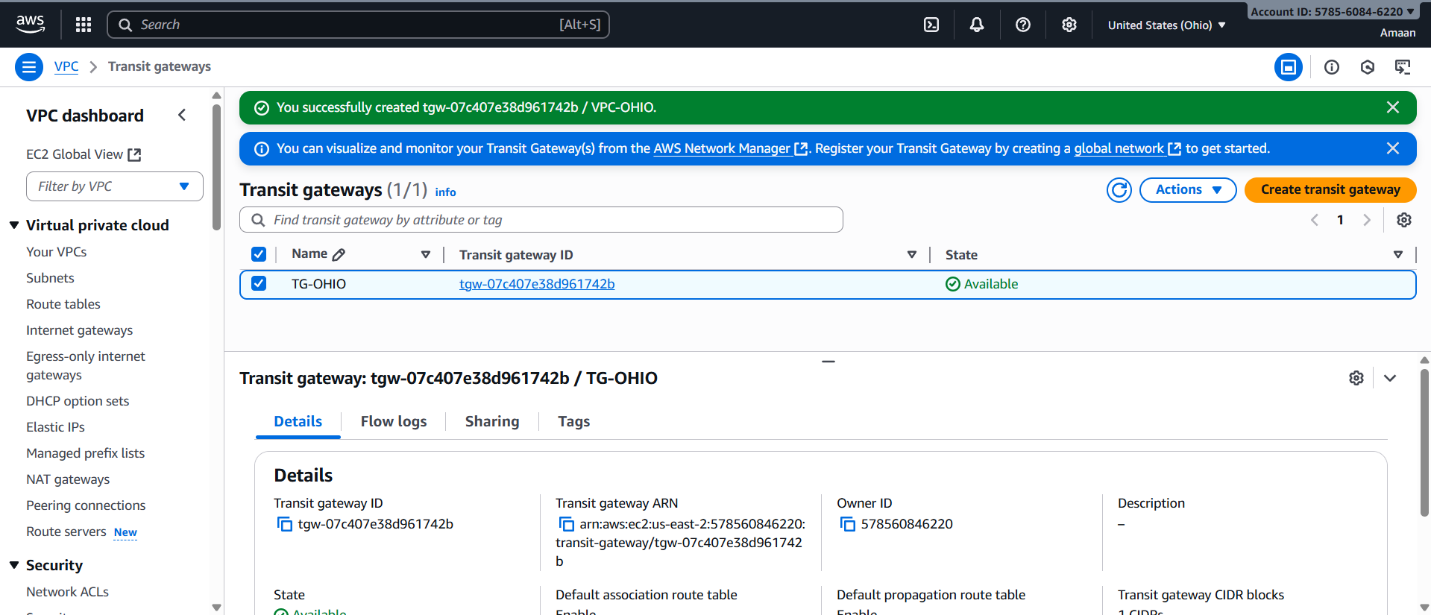
Each Transit Gateway is regional and can be used to connect VPCs within that region or across accounts.

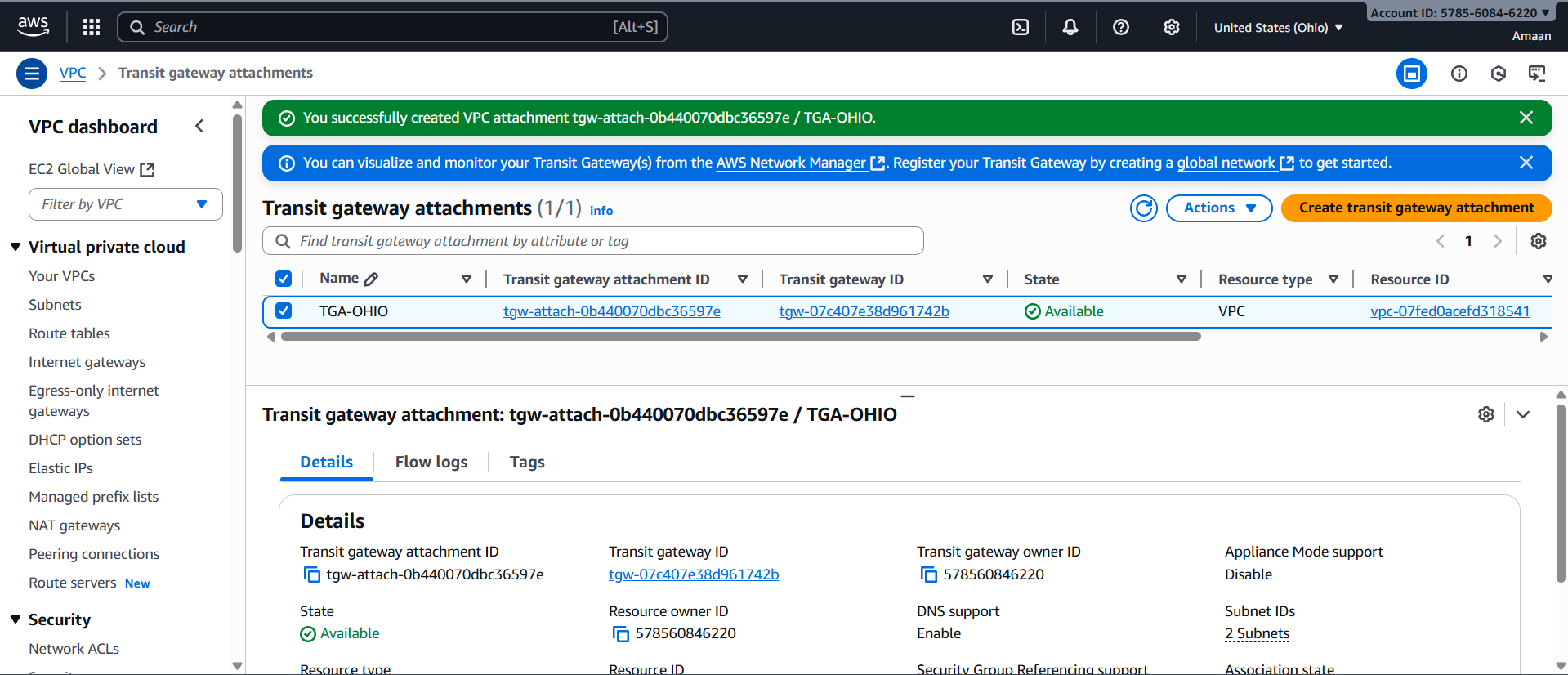
TGW-N-VIRGINIA



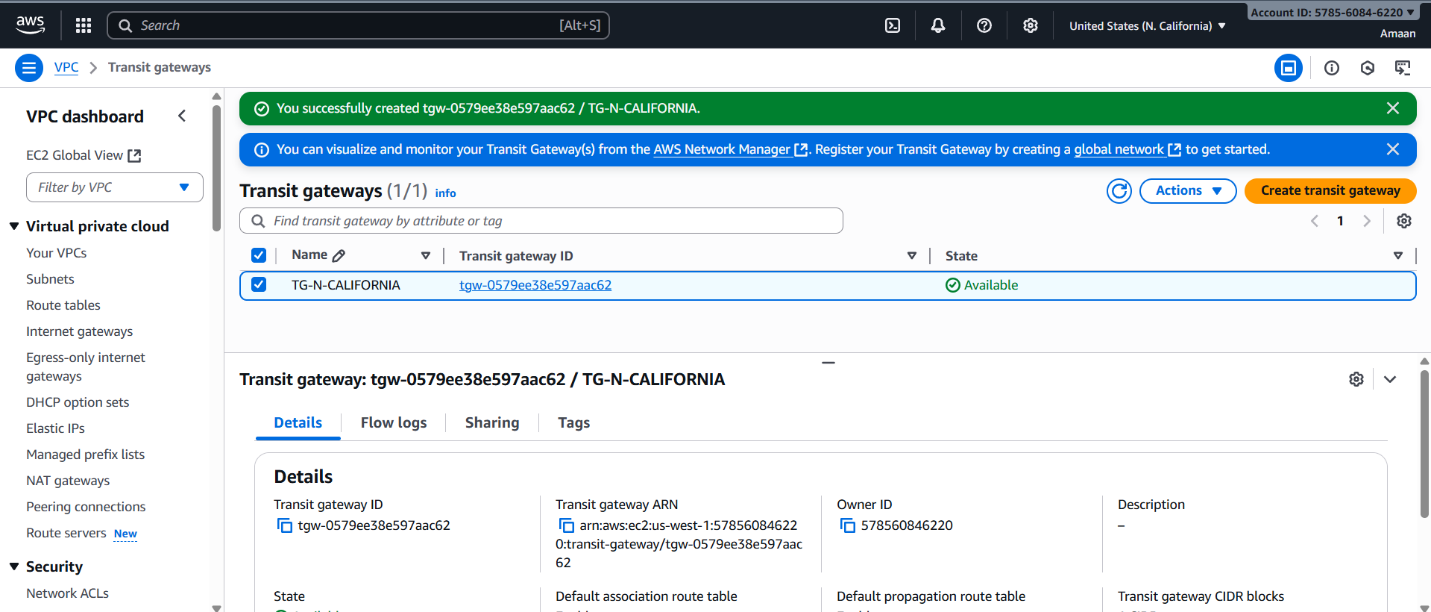


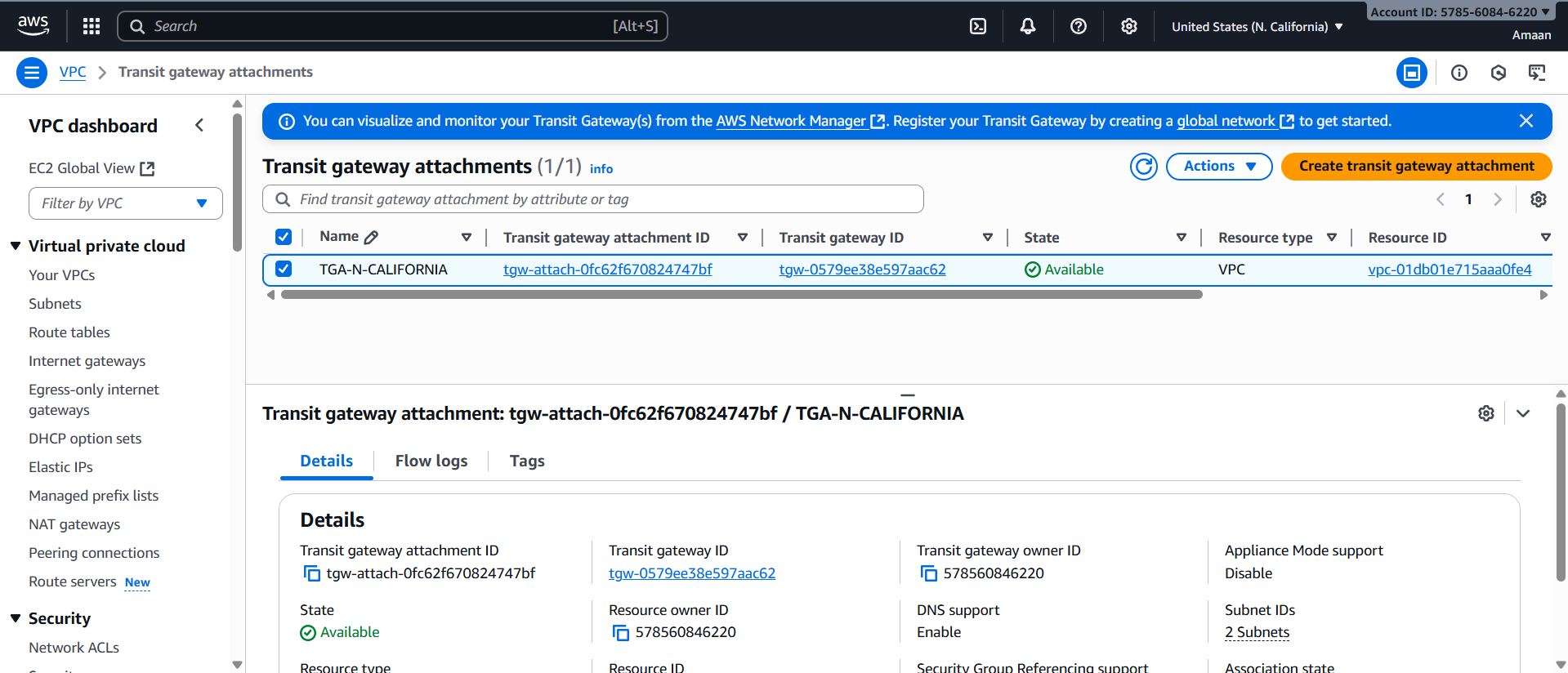
TGW-OHIO



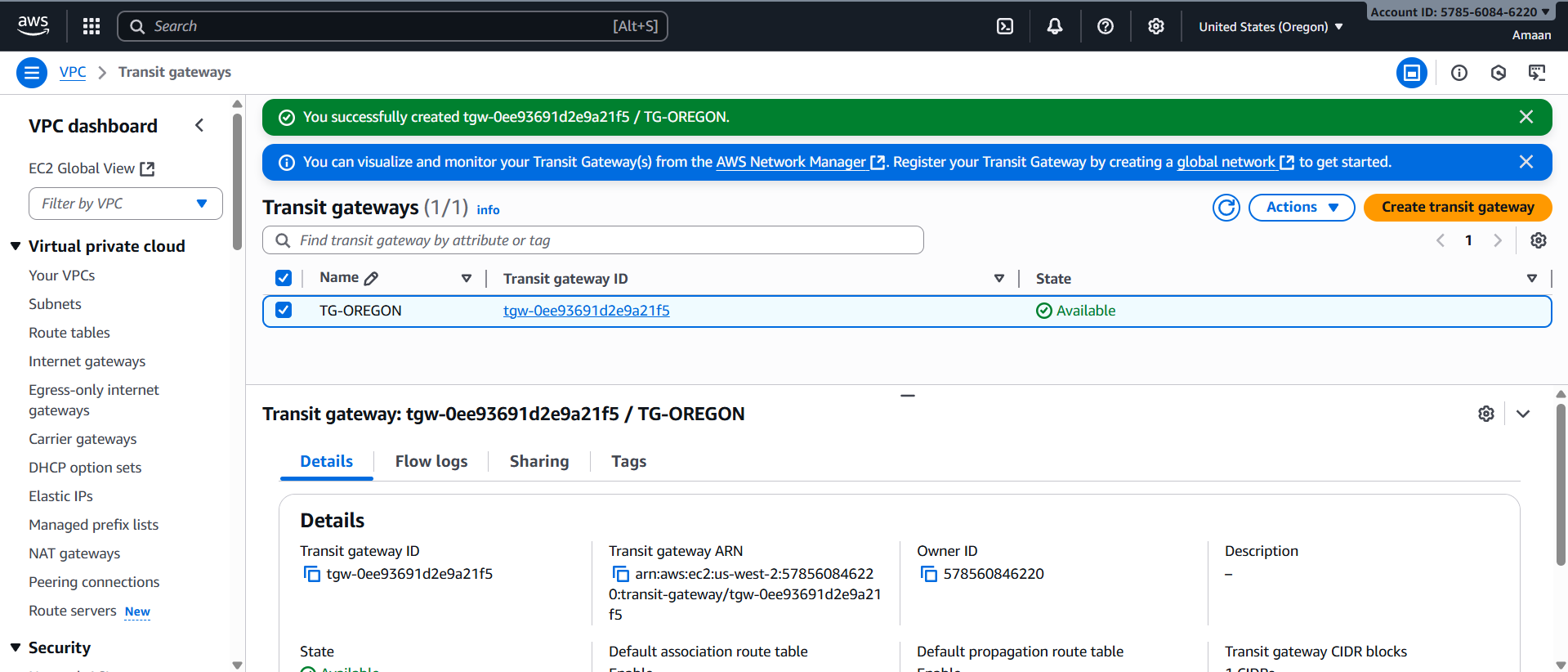


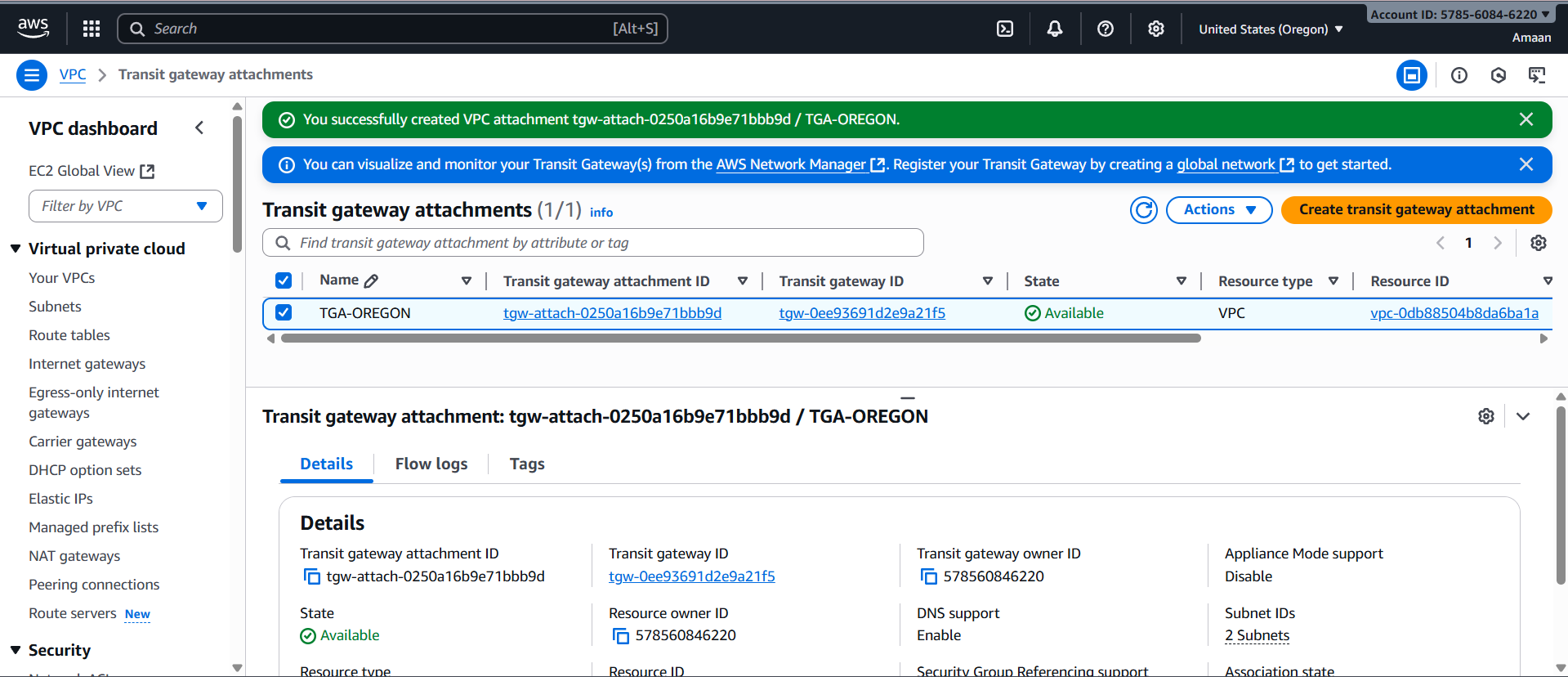
TGW-N-CALIFORNIA





TGW-Oregon





**Step 3: Attach VPCs to Their Regional Transit Gateways**

* For each VPC, create an attachment linking it to its regional Transit Gateway.
* When creating the attachment, select the private subnets for the attachment.
* Confirm the availability zone mappings are correct.

Step 3 to attach VPCs to their regional Transit Gateways involves the following process:

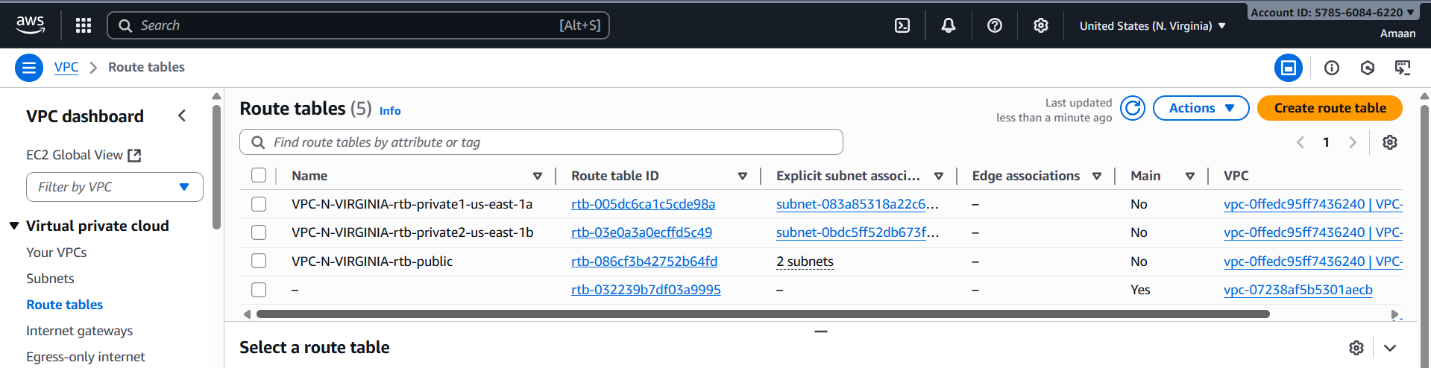
1. Open the Amazon VPC Console.
2. In the left navigation pane, select "Transit Gateway Attachments."
3. Click "Create transit gateway attachment."
4. Optionally, enter a Name tag for the attachment.
5. For "Transit gateway ID," select the appropriate regional transit gateway.
6. For "Attachment type," select "VPC."
7. Choose whether to enable DNS support, IPv6 support, and appliance mode support according to your needs.
8. For "VPC ID," select the VPC from the same region to attach.
9. Select at least one subnet from each Availability Zone associated with the VPC for routing.
10. Click "Create transit gateway attachment."

After the attachment is created, update the VPC route tables to send traffic to the Transit Gateway attachment for routing between VPCs.

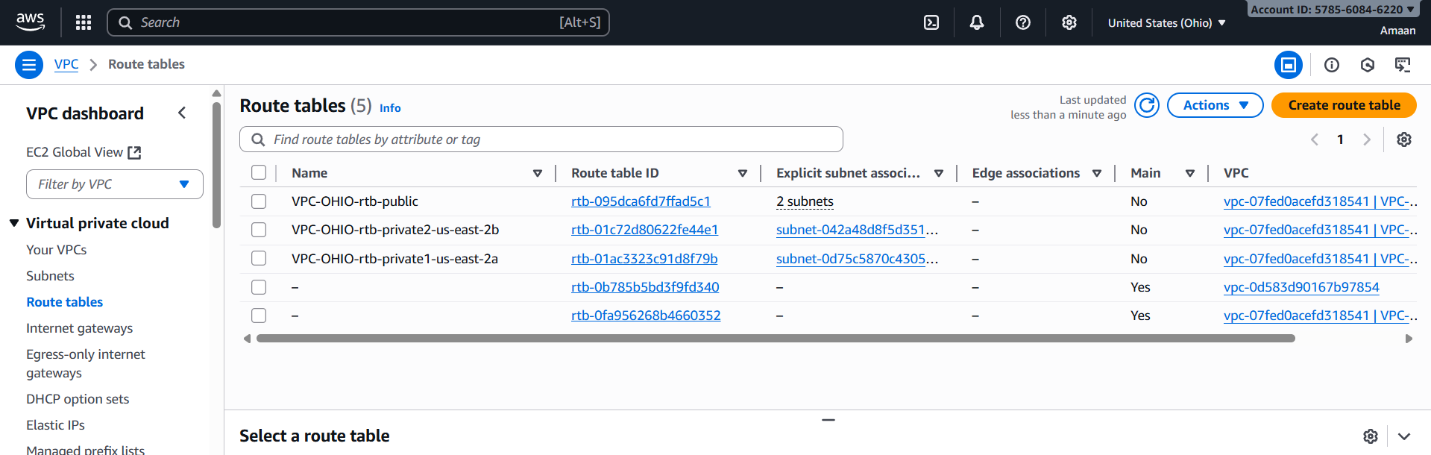
You will repeat these steps in each region for the respective VPC and Transit Gateway.

This will centrally enable routing between attached VPCs within each region via the Transit Gateway.

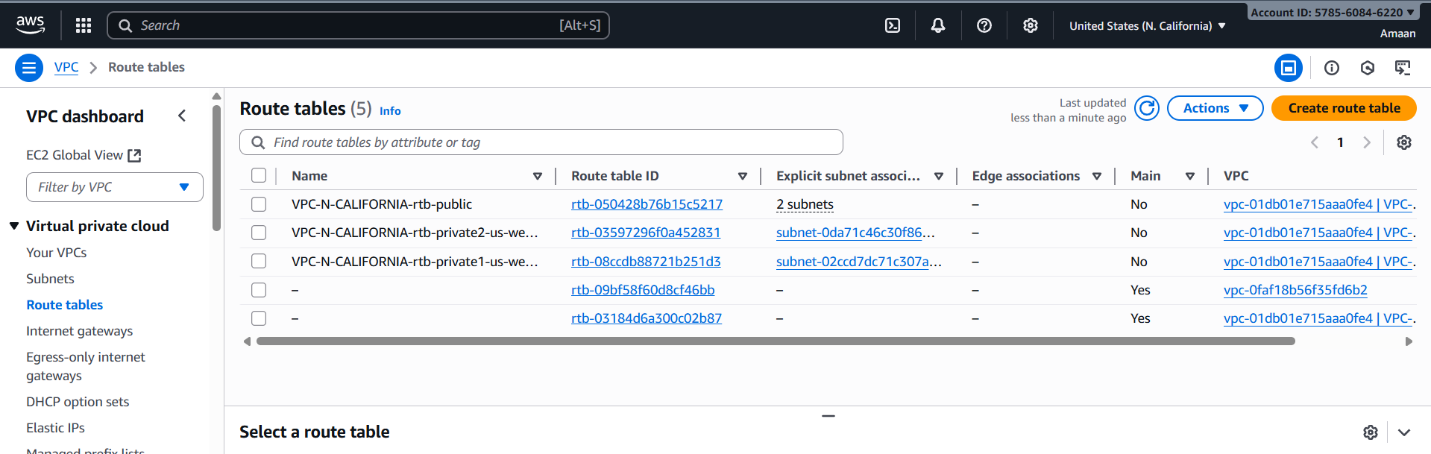
N-VIRGINIA ROUTE TABLE



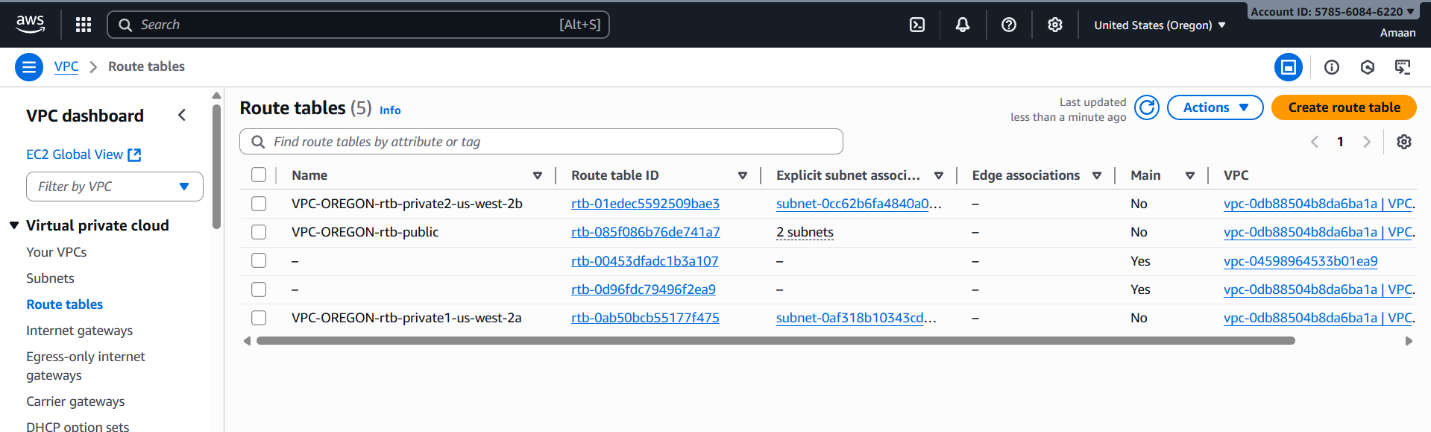
OHIO ROUTE TABLE



N-CALIFORNIA ROUTE TABLE



OREGON-ROUTE TABLE



**Step 4: Establish Inter-Region Peering Between Transit Gateways**

* From the hub region Transit Gateway, initiate peering connections to each spoke region Transit Gateway.
* Accept peering requests in each spoke region.
* Ensure peering connections are in an "available" state before proceeding.
* This peering setup enables cross-region communication between VPCs.

To connect AWS Transit Gateways across regions, establish an inter-Region Transit Gateway peering attachment that enables routing of traffic securely and efficiently between regions. The main steps are:

1. Open the Amazon VPC console.
2. In the navigation pane, select "Transit Gateway Attachments."
3. Click "Create transit gateway attachment."
4. For "Transit gateway ID," select the Transit Gateway in the source region.
5. For "Attachment type," select "Peering Connection."
6. Choose account and specify the peer Transit Gateway’s owner account if in a different AWS account.
7. Select the Region and Transit Gateway ID of the peer Transit Gateway in the target region.
8. Optionally assign a name tag to identify the peering connection.
9. Click "Create transit gateway attachment" to send the peering request.
10. The owner of the peer Transit Gateway accepts the request in their console.
11. After acceptance, update route tables in both Transit Gateways to route traffic via the peering attachment.
12. Optionally configure security and network ACLs to allow traffic flow between regions.

Traffic over inter-Region peering is encrypted by AES-256 at the network layer and physical layer when crossing outside AWS infrastructure, ensuring secure communication.

This linkage allows your Transit Gateways in different regions to exchange traffic, enabling seamless multi-region networking.



**Step 5: Configure Routing Tables for Connectivity**

**VPC Route Tables**

* In each VPC, update route tables attached to private subnets with routes for the CIDR blocks of other VPCs.
* Set the Transit Gateway attachment as the target for these routes.

**Transit Gateway Route Tables**

* In the hub Transit Gateway route table, add routes for all spoke VPC CIDR blocks, pointing to the respective Transit Gateway attachments.
* In each spoke Transit Gateway route table, add routes directing traffic to the hub VPC CIDR through the relevant peering attachment.

To enable full connectivity between VPCs attached to your Transit Gateways and across regions if peered, routing tables must be configured both on the Transit Gateway and inside each VPC.

**Transit Gateway Route Tables**

* Create or use existing Transit Gateway route tables.
* Associate each VPC attachment with the appropriate route table.
* Propagate routes from VPC attachments and peering attachments to the route tables.
* Add static routes as needed (e.g., for peering attachments).
* Ensure route tables have entries for all destination CIDR blocks of connected VPCs or on-prem networks.
* Routes should direct traffic to the correct Transit Gateway attachment as the next hop.

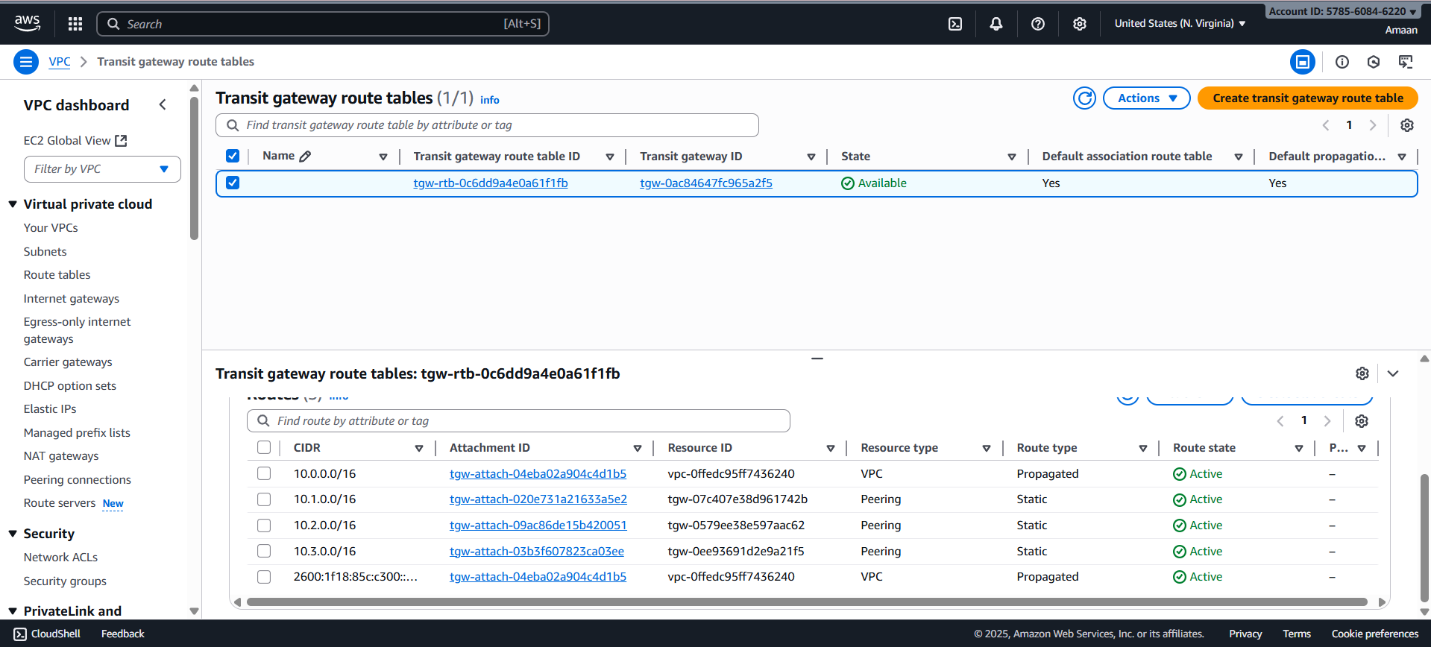
**VPC Route Tables**

* In each attached VPC, update the subnet route tables.
* Add routes for remote VPC CIDR blocks that point to the Transit Gateway attachment as the target.
* This ensures instances in VPC subnets send traffic destined for other VPCs or connected networks through the Transit Gateway.

**Important Notes**

* Traffic is bidirectional, so routes in both directions are required for communication.
* Route propagation simplifies management by dynamically adding routes from attachments.
* For inter-region peering, only static routes are supported on Transit Gateway route tables.
* Security group and network ACL rules need to allow traffic based on your connection design.

This routing setup enables seamless connectivity between your multi-region VPCs via Transit Gateways.



**Step 6: Create VPC Endpoints for Secure AWS Service Access**

Within each VPC:

* Create a **Gateway Endpoint** for Amazon S3 and associate it with the VPC route tables.
* Create **Interface Endpoints** for critical AWS services such as:
  + Systems Manager (SSM)
  + EC2
  + Elastic Container Registry (ECR) - API and Docker
  + CloudWatch Logs
* Enable private DNS for interface endpoints to allow transparent service access.
* This enables services' private connectivity without traffic traversing the internet.

**Steps to Create a VPC Endpoint**

1. **Open the VPC Console:**
   * Go to the AWS Management Console and navigate to **VPC**.
2. **Go to Endpoints:**
   * In the navigation pane, select **Endpoints**.
3. **Create Endpoint:**
   * Click **Create endpoint**.
4. **Select Service:**
   * For **Service category**, choose **AWS services**.
   * Search for the desired AWS service (e.g., S3, DynamoDB, or services supporting PrivateLink). For S3, select the "Gateway" type. For other AWS services, it might be "Interface".
5. **Configure Endpoint:**
   * Select your VPC.
   * For gateway endpoints (S3, DynamoDB), select route tables.
   * For interface endpoints, select subnets and security groups.
6. **Set Policies:**
   * Define an endpoint policy to restrict which resources or actions are allowed via this endpoint.
7. **Review and Create:**
   * Review configuration and click **Create endpoint**.

**Optional Security Hardening**

* For S3, enhance security by adding a bucket policy to only allow traffic from your VPC Endpoint using the aws:SourceVpce condition.
* For interface endpoints, ensure only necessary subnets and security groups are allowed.

**Benefit**

VPC Endpoints ensure your traffic never leaves the Amazon private network, providing secure, high-bandwidth access to AWS services from within your VPC.

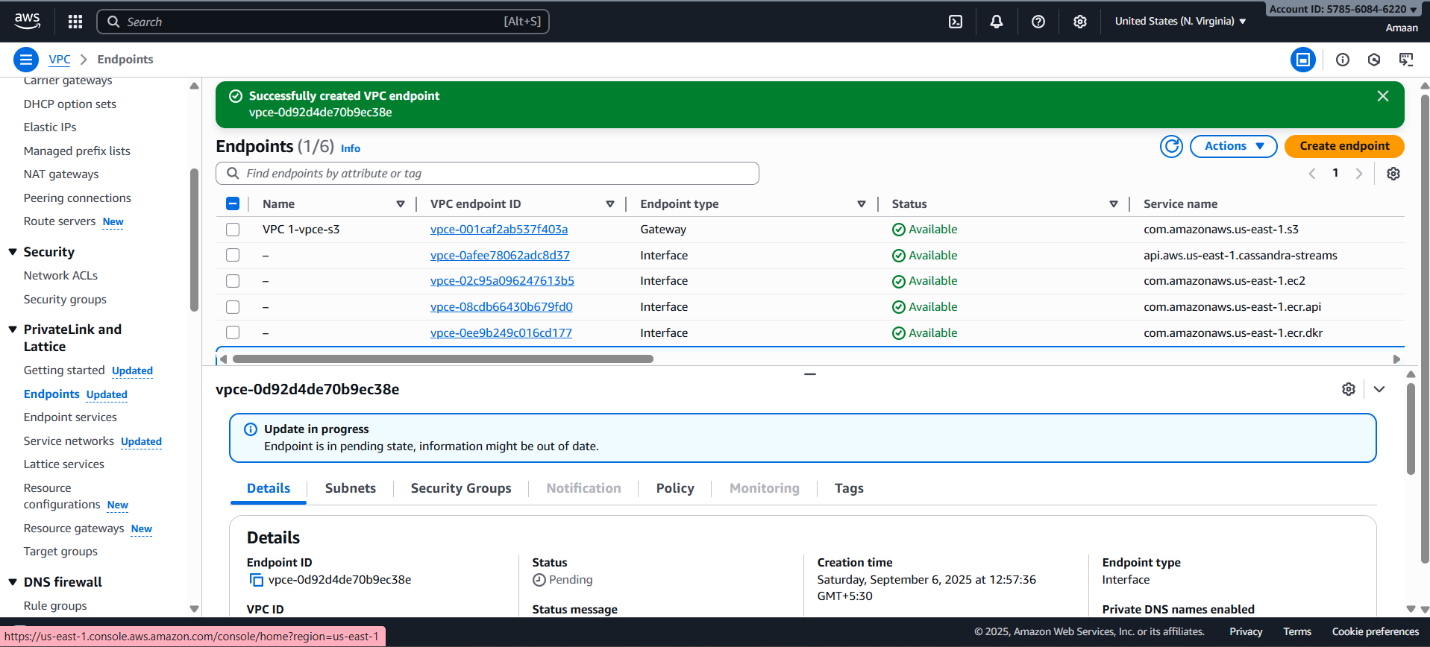
**Step 1: Create Interface Endpoints for Critical AWS Services**

1. **Open AWS VPC Console**
   * Navigate to **VPC** > **Endpoints** in your AWS Console.
2. **Create Endpoint**
   * Click **Create endpoint**.
3. **Select Service**
   * For **Service category**, choose **AWS services**.
   * Search and select the following services (interface endpoint type):
     + **com.amazonaws.<region>.ssm** (Systems Manager)
     + **com.amazonaws.<region>.ec2** (EC2 API)
     + **com.amazonaws.<region>.ecr.api** (ECR API)
     + **com.amazonaws.<region>.ecr.dkr** (ECR Docker)
     + **com.amazonaws.<region>.logs** (CloudWatch Logs)
   * Replace <region> with your AWS region code.
4. **Configure Endpoint Settings**
   * Choose your **VPC**.
   * Select subnets (usually private subnets) where you want endpoint network interfaces created.
   * Select or create appropriate **security groups** that allow inbound traffic on the required ports.
5. **Enable Private DNS**
   * Ensure **Enable Private DNS Name** checkbox is selected for each endpoint to enable transparent DNS routing to the service inside your VPC.
6. **Policy Configuration**
   * Optionally define an endpoint policy to restrict or allow access.
7. **Create Endpoint**
   * Click **Create endpoint**.

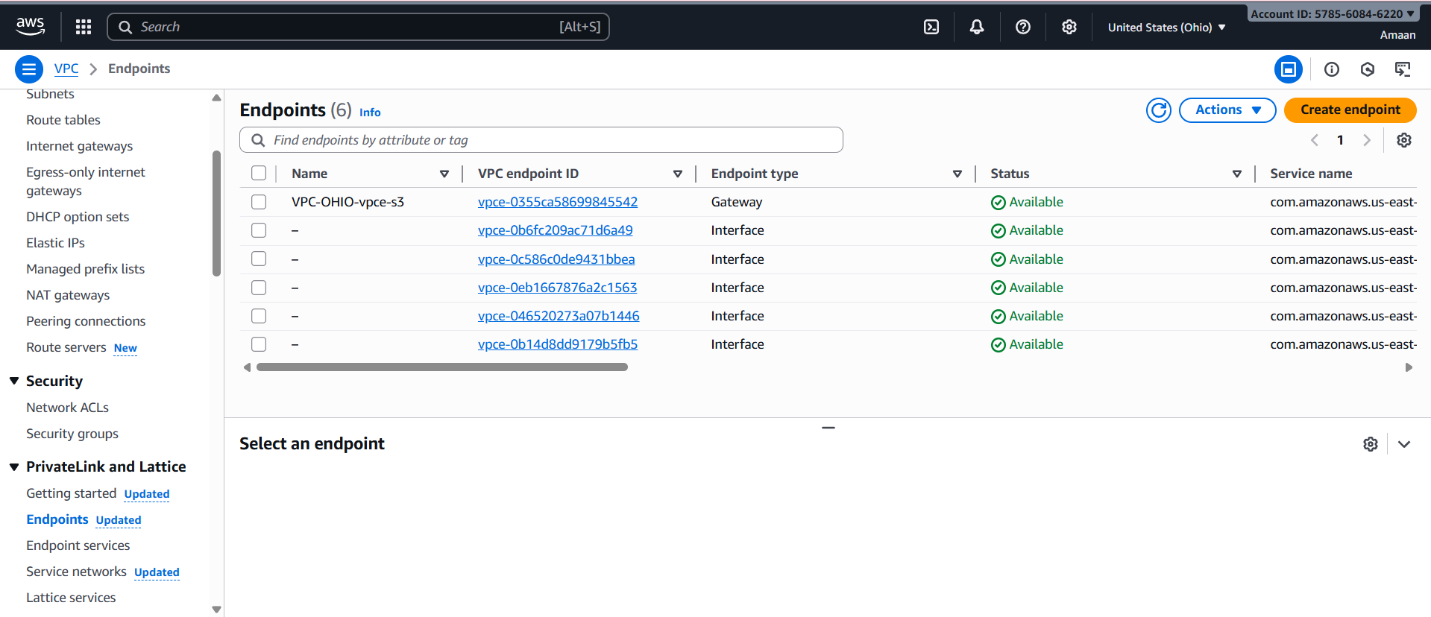
**Step 2: Verify and Test the Endpoints**

* Confirm the status changes to **Available**.
* Use an instance in your private subnet to test API connectivity to the services through the endpoint without needing internet access.
* Use DNS lookup (e.g., nslookup ssm.<region>.amazonaws.com) inside the VPC to verify private DNS is resolving to the interface endpoint.

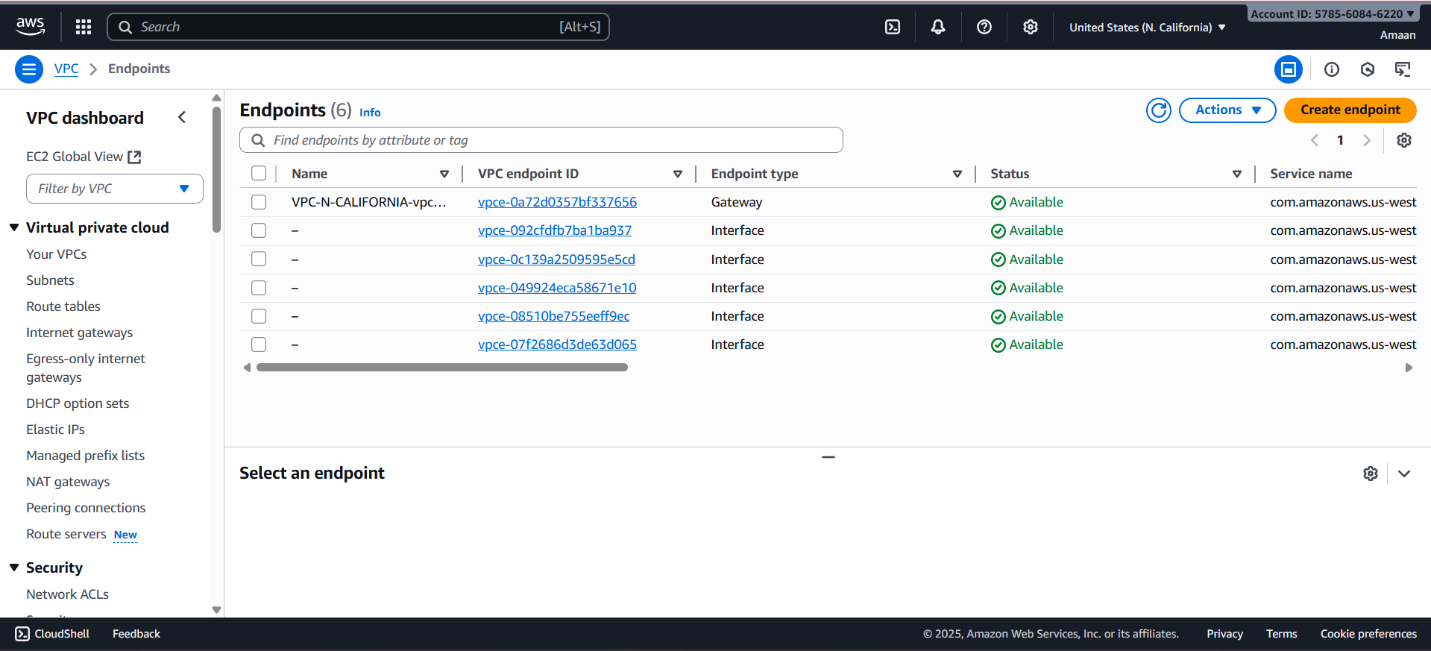
N-VIRGINIA ENDPOINT



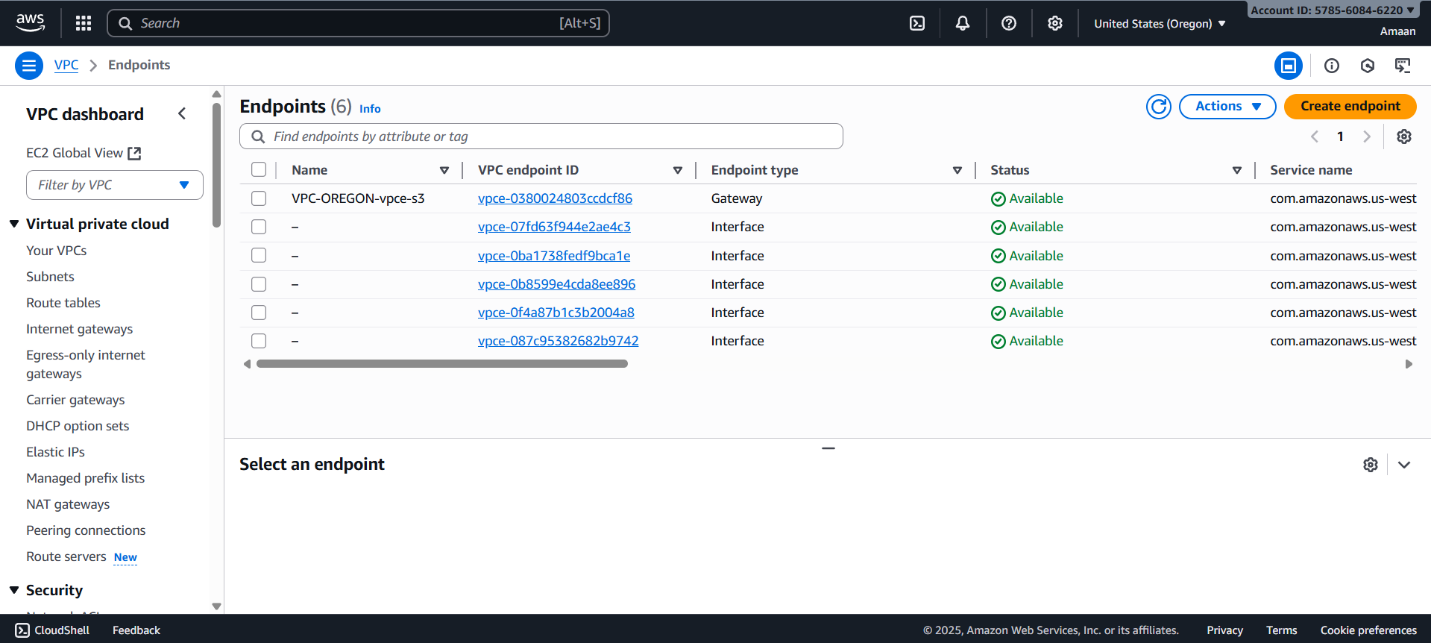
OHIO ENDPOINT



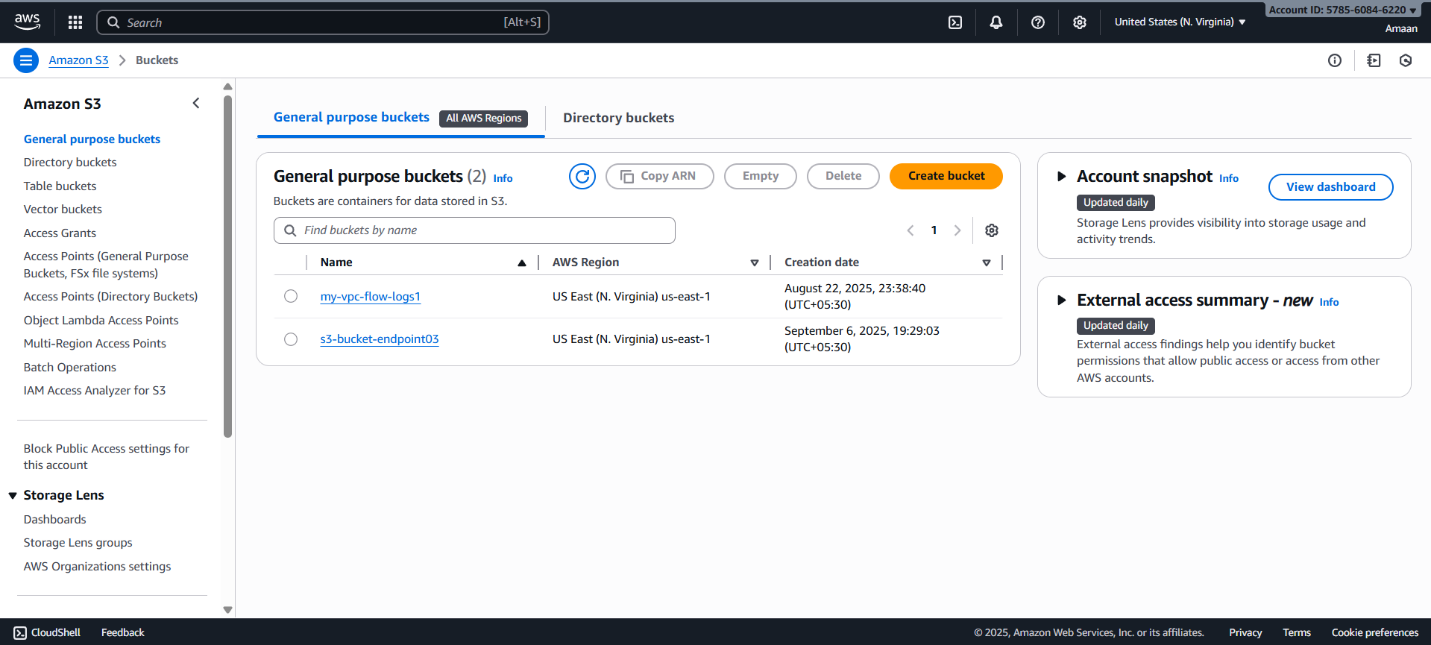
N-CALIFORNIA ENDPOINT



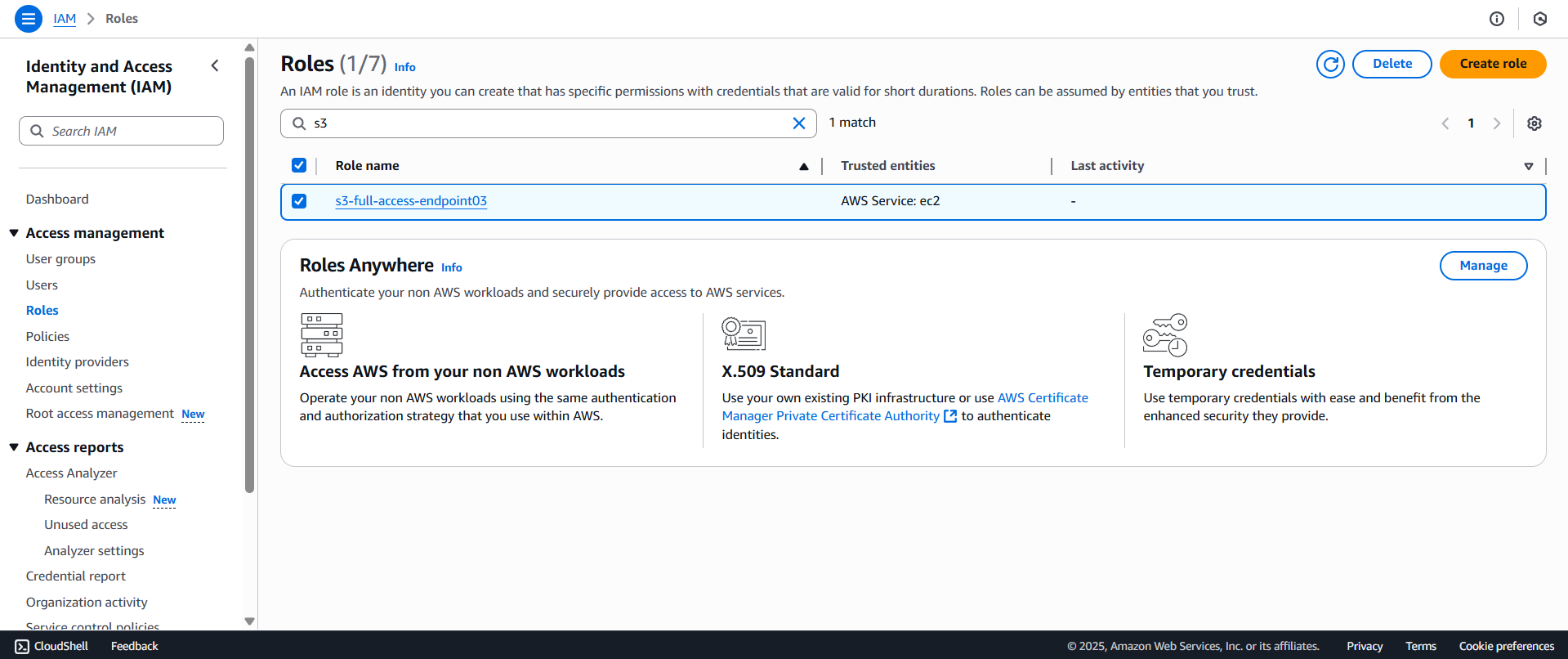
OREGON ENDPOINT



Create an s3 bucket



Create an Iam role and give permissions to s3 bucket



**Step 7: Validate the Setup**

* Launch EC2 instances in private subnets using AWS Systems Manager-enabled AMIs.
* Use Systems Manager Session Manager to connect to EC2 instances without opening SSH ports.
* Test connectivity:
  + Ping EC2 instances across VPCs via Transit Gateway.
  + Execute AWS CLI commands (e.g., aws s3 ls) to ensure access through VPC endpoints works properly.
* Confirm all network traffic remains private and secure.

**1. Launch EC2 Instances in Private Subnets**

* Use AWS Systems Manager (SSM)-enabled AMIs (such as Amazon Linux 2 with SSM agent pre-installed).
* Assign an IAM role with **AmazonSSMManagedInstanceCore** permissions to the instances for SSM access.

**2. Connect Using Systems Manager Session Manager**

* Confirm a VPC SSM interface endpoint exists. This allows SSM traffic to stay private.
* In the EC2 or SSM console, choose **Session Manager** to start a session with instances in private subnets.
* **No SSH port** or public IP needed.

**3. Test Transit Gateway Connectivity**

* Ensure your EC2 security groups allow ICMP (ping) and required traffic.
* From an EC2 in one VPC, ping the private IP of an EC2 instance in another VPC.
* Successful replies confirm Transit Gateway routing is working.

**4. Test VPC Endpoint Service Access**

* On EC2, run:

text

aws s3 ls

This should return S3 bucket contents if the S3 Gateway endpoint and IAM policy are correctly set up.

* For other services, use AWS CLI or SDK commands (e.g., describe-instances for EC2, access ECR, CloudWatch Logs).
* If your interface endpoints and private DNS are configured, all traffic is routed privately via AWS.

**5. Confirm Traffic Is Private**

* Instances should NOT require a public IP or NAT Gateway for AWS API/service access.
* Traffic to S3, EC2 API, ECR, SSM, and CloudWatch stays within the AWS network via VPC endpoints.

