Multi-Region Transit Routing from AWS to OCI

Author: Shadab Mohammad, Master Principal Cloud Architect

Executive Summary

This whitepaper presents a scalable, high-performance solution for establishing multi-region transit routing between AWS and OCI using Transit Gateway (TGW) and Dynamic Routing Gateway (DRG). The architecture enables secure and optimized connectivity across multiple AWS and OCI regions via IPSec VPN. The solution is designed to facilitate seamless communication across AWS VPCs and OCI VCNs using a single transit region in both clouds, with the capability to extend into additional regions as needed.

Important Note : The setup documented in this Whitepaper can also be used to create a Transit Routing network topology for DB@AWS. The only requirement is to ensure the ODB Network Peering route is added to the Route tables for the VPC and then extended to connect to other VPCs in one or more regions using Transit Gateway and Transit Gateway peering on AWS.

Introduction

Organizations increasingly operate in multi-cloud environments to leverage the unique capabilities of various cloud providers. This solution integrates AWS Transit Gateway (TGW) and OCI Dynamic Routing Gateway (DRG) to create a highly available, secure, and scalable transit routing framework, facilitating cross-region and cross-cloud connectivity.

Key Objectives:

- Establish a multi-region transit architecture between AWS and OCI.
- Utilize AWS Transit Gateway and OCI DRG to enable secure connectivity.
- Ensure seamless routing and high availability across multiple cloud regions.
- Implement a scalable, extendable mesh network architecture.

Architecture Overview

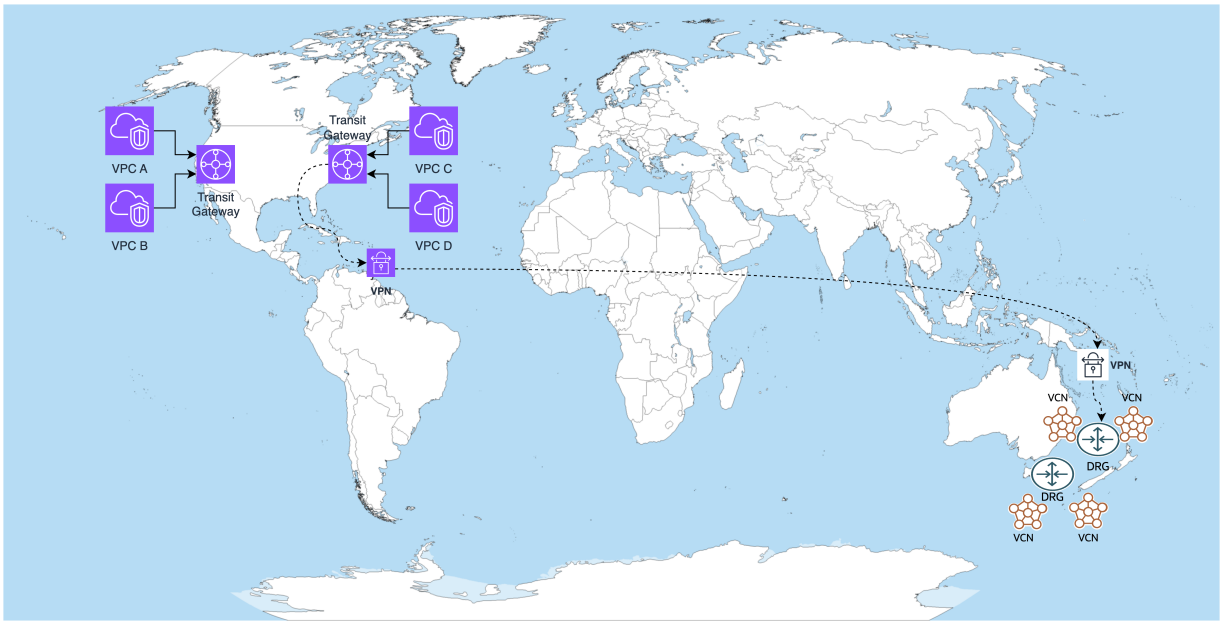
The architecture consists of the following key components:

AWS Components:

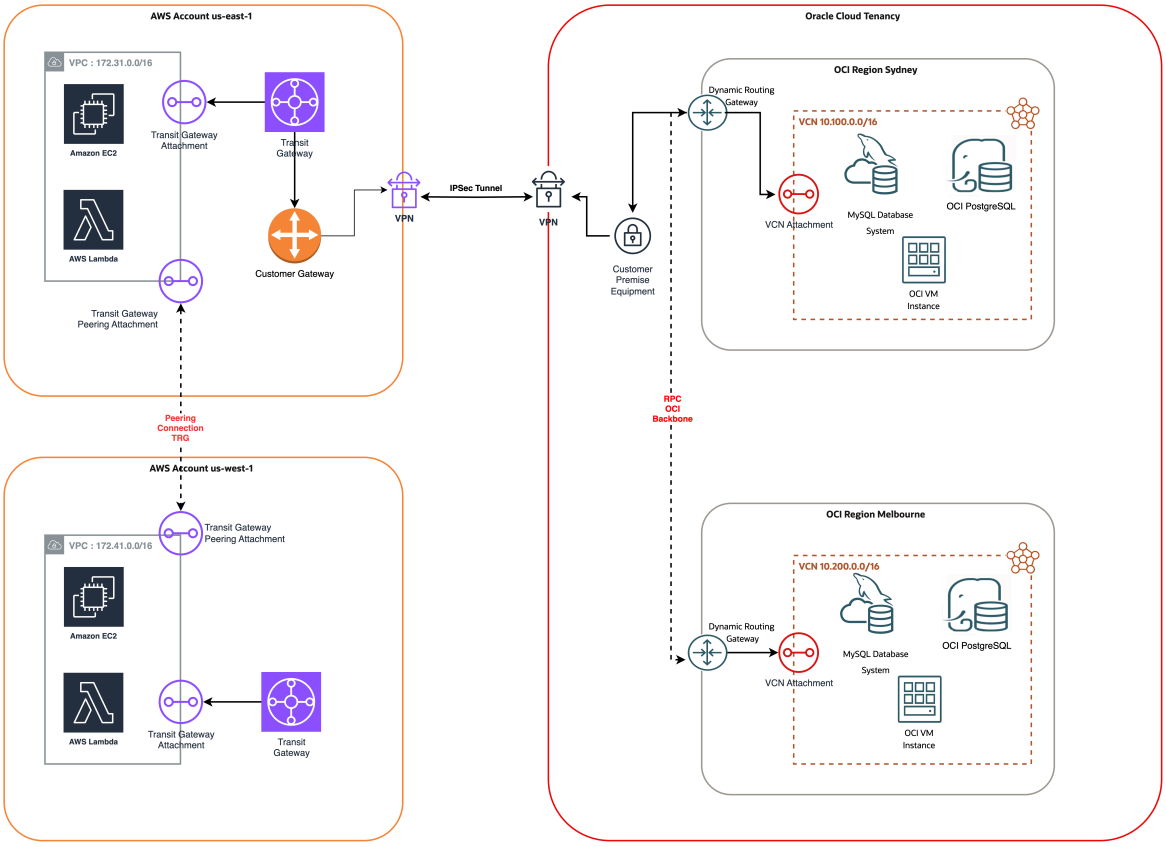
- **AWS Transit Gateway (TGW):** Centralized hub for routing between AWS VPCs and OCI.
- **AWS Customer Gateway:** Used to establish an IPSec VPN connection.
- **AWS Site-to-Site VPN:** Provides secure communication with OCI.
- **AWS VPC Route Tables:** Configured to facilitate cross-region connectivity.

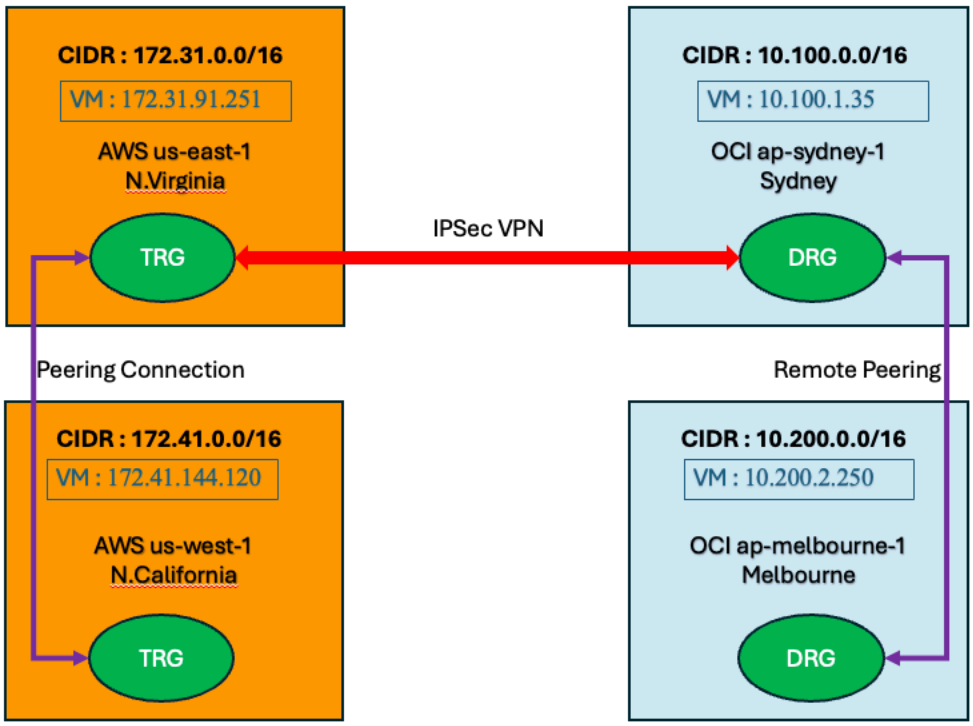
OCI Components:

- **Dynamic Routing Gateway (DRG):** Acts as a central routing hub for OCI VCNs.
- **Customer Premises Equipment (CPE):** Represents AWS as a remote VPN peer.
- **OCI Site-to-Site VPN:** Enables connectivity between OCI and AWS.
- **Remote Peering Connection (RPC):** Extends connectivity across multiple OCI regions.



Architecture





Site-to-Site VPN OCI ap-sydney-1 BGP Routes received:

BGP routes received

IP route prefix	Route age	Is best path	AS path length	AS path
172.31.0.0/16	29337 Seconds	No	2	65515...Show all (2)
172.41.0.0/16	8306 Seconds	No	2	65515...Show all (2)

Showing 2 items Page 1

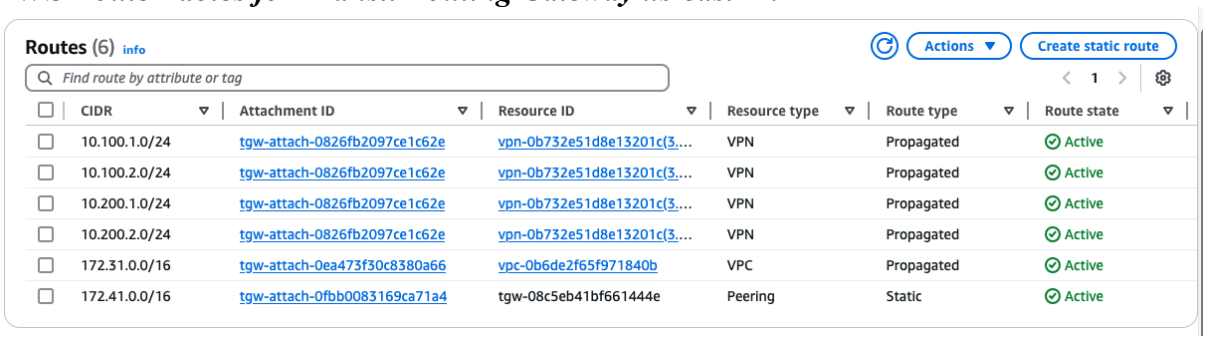
Site-to-Site VPN OCI ap-sydney-1 BGP Routes advertised:

BGP routes advertised

IP route prefix	Route age	Is best path	AS path length	AS path
10.100.1.0/24	899721 Seconds	No	0	-
10.100.2.0/24	899721 Seconds	No	0	-
10.200.1.0/24	807321 Seconds	No	0	-
10.200.2.0/24	807321 Seconds	No	0	-

Showing 4 items Page 1

AWS Route Tables for Transit Routing Gateway us-east-1 :



Routes (6) info

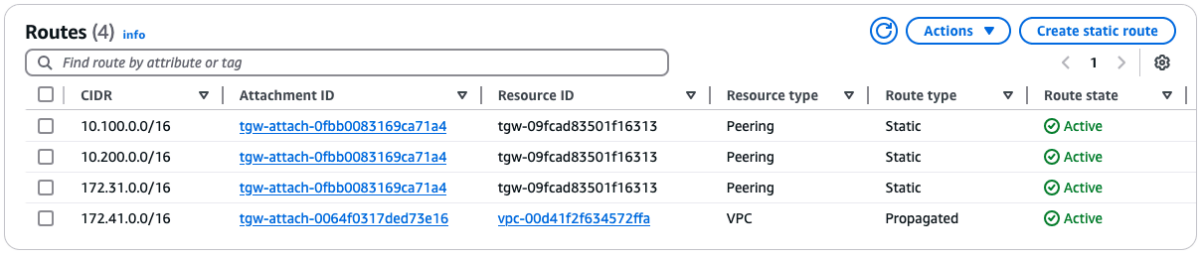
Actions Create static route

Find route by attribute or tag

1

CIDR	Attachment ID	Resource ID	Resource type	Route type	Route state
10.100.1.0/24	tgw-attach-0826fb2097ce1c62e	vpn-0b732e51d8e13201c(3,...	VPN	Propagated	Active
10.100.2.0/24	tgw-attach-0826fb2097ce1c62e	vpn-0b732e51d8e13201c(3,...	VPN	Propagated	Active
10.200.1.0/24	tgw-attach-0826fb2097ce1c62e	vpn-0b732e51d8e13201c(3,...	VPN	Propagated	Active
10.200.2.0/24	tgw-attach-0826fb2097ce1c62e	vpn-0b732e51d8e13201c(3,...	VPN	Propagated	Active
172.31.0.0/16	tgw-attach-0ea473f30c8380a66	vpc-0b6de2f65f971840b	VPC	Propagated	Active
172.41.0.0/16	tgw-attach-0fbb0083169ca71a4	tgw-08c5eb41bf661444e	Peering	Static	Active

AWS Route Tables for Transit Routing Gateway us-west-1 :



Routes (4) info

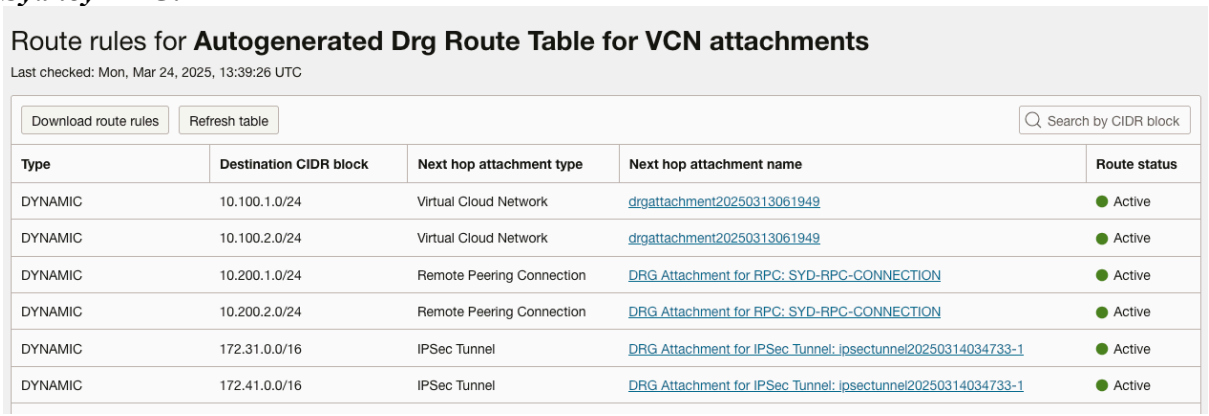
Actions Create static route

Find route by attribute or tag

1

CIDR	Attachment ID	Resource ID	Resource type	Route type	Route state
10.100.0.0/16	tgw-attach-0fbb0083169ca71a4	tgw-09fcad83501f16313	Peering	Static	Active
10.200.0.0/16	tgw-attach-0fbb0083169ca71a4	tgw-09fcad83501f16313	Peering	Static	Active
172.31.0.0/16	tgw-attach-0fbb0083169ca71a4	tgw-09fcad83501f16313	Peering	Static	Active
172.41.0.0/16	tgw-attach-0064f0317ded73e16	vpc-00d41f2f634572ffa	VPC	Propagated	Active

Sydney DRG:

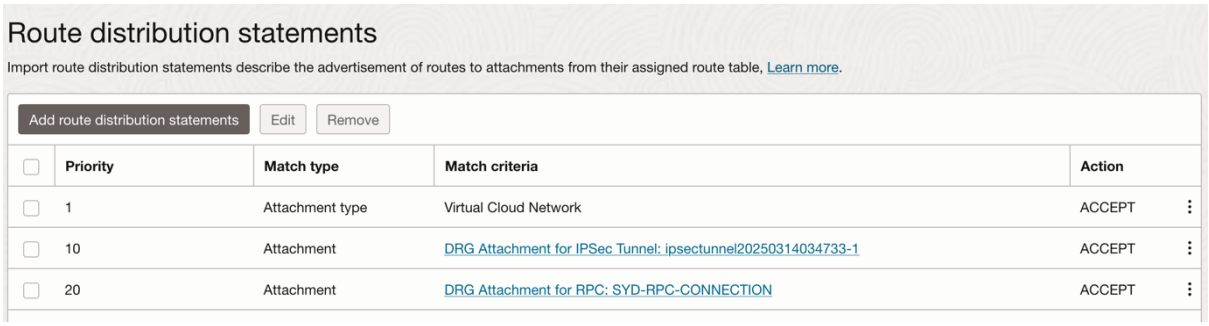


Route rules for **Autogenerated Drg Route Table for VCN attachments**

Last checked: Mon, Mar 24, 2025, 13:39:26 UTC

Download route rules | Refresh table | Search by CIDR block

Type	Destination CIDR block	Next hop attachment type	Next hop attachment name	Route status
DYNAMIC	10.100.1.0/24	Virtual Cloud Network	drgattachment20250313061949	Active
DYNAMIC	10.100.2.0/24	Virtual Cloud Network	drgattachment20250313061949	Active
DYNAMIC	10.200.1.0/24	Remote Peering Connection	DRG Attachment for RPC: SYD-RPC-CONNECTION	Active
DYNAMIC	10.200.2.0/24	Remote Peering Connection	DRG Attachment for RPC: SYD-RPC-CONNECTION	Active
DYNAMIC	172.31.0.0/16	IPSec Tunnel	DRG Attachment for IPSec Tunnel: ipsectunnel20250314034733-1	Active
DYNAMIC	172.41.0.0/16	IPSec Tunnel	DRG Attachment for IPSec Tunnel: ipsectunnel20250314034733-1	Active



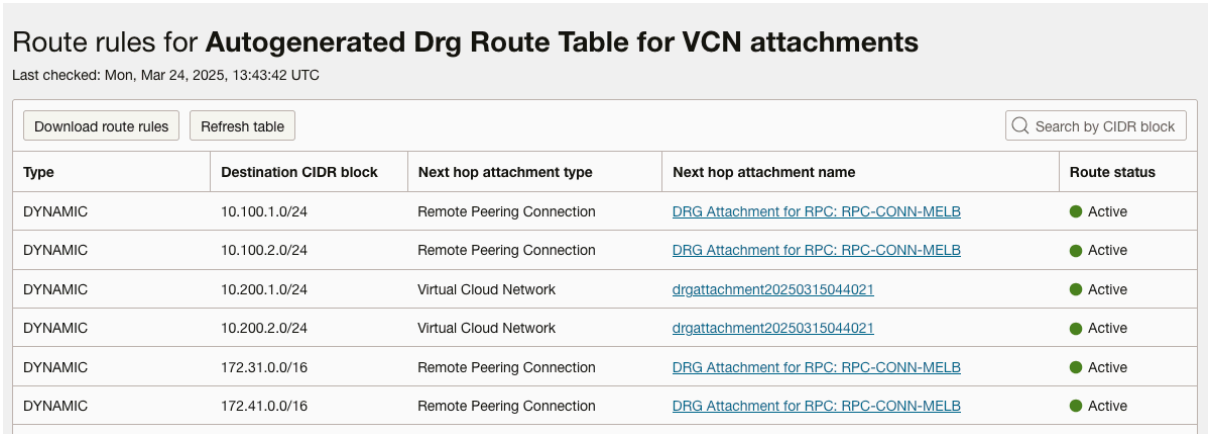
Route distribution statements

Import route distribution statements describe the advertisement of routes to attachments from their assigned route table, Learn more.

Add route distribution statements | Edit | Remove

Priority	Match type	Match criteria	Action
1	Attachment type	Virtual Cloud Network	ACCEPT
10	Attachment	DRG Attachment for IPSec Tunnel: ipsectunnel20250314034733-1	ACCEPT
20	Attachment	DRG Attachment for RPC: SYD-RPC-CONNECTION	ACCEPT

Melbourne DRG:



Route rules for **Autogenerated Drg Route Table for VCN attachments**

Last checked: Mon, Mar 24, 2025, 13:43:42 UTC

Download route rules | Refresh table | Search by CIDR block

Type	Destination CIDR block	Next hop attachment type	Next hop attachment name	Route status
DYNAMIC	10.100.1.0/24	Remote Peering Connection	DRG Attachment for RPC: RPC-CONN-MELB	Active
DYNAMIC	10.100.2.0/24	Remote Peering Connection	DRG Attachment for RPC: RPC-CONN-MELB	Active
DYNAMIC	10.200.1.0/24	Virtual Cloud Network	drgattachment20250315044021	Active
DYNAMIC	10.200.2.0/24	Virtual Cloud Network	drgattachment20250315044021	Active
DYNAMIC	172.31.0.0/16	Remote Peering Connection	DRG Attachment for RPC: RPC-CONN-MELB	Active
DYNAMIC	172.41.0.0/16	Remote Peering Connection	DRG Attachment for RPC: RPC-CONN-MELB	Active

High-Level Implementation Steps

1. Configure AWS Transit Gateway

1. **Create AWS Transit Gateway**
   - Navigate to AWS Console > VPC > Transit Gateway > Create Transit Gateway.
   - Attach AWS VPCs to TGW.
2. **Create AWS Customer Gateway**
   - Use a temporary mock IP initially.
   - Assign Oracle's BGP ASN (31898 for most regions).
3. **Establish AWS Site-to-Site VPN**
   - Navigate to AWS Console > VPC > Virtual Private Network > Site-to-Site VPN > Create VPN Connection.
   - Configure Tunnel 1 only.
   - Download the VPN configuration.
4. **Modify AWS Route Tables**
   - Add OCI VCN CIDR (e.g., `10.100.0.0/16`) to AWS VPC Subnet Route Table.

2. Configure OCI Dynamic Routing Gateway

1. **Create and Attach DRG to OCI VCN**
   - Attach the AWS VPC CIDR (`172.31.0.0/16`) to the DRG.
2. **Create Customer Premises Equipment (CPE) in OCI**
   - Use AWS VPN Tunnel 1 Public IP.
3. **Create Site-to-Site VPN in OCI**
   - Match AWS VPN settings and shared secret key.
   - Configure ISAKMP settings for Tunnel 1.

3. Establish Cross-Region Peering in OCI

1. **Create OCI VCN in Secondary Region (e.g., Melbourne)**
2. **Deploy a DRG in Melbourne and Attach VCN**
3. **Create Remote Peering Connection (RPC) Between Regions**
   - Establish an RPC in Sydney DRG and Melbourne DRG.
   - Exchange OCIDs and verify connectivity.

4. Extend AWS Transit Gateway Across Regions

1. **Deploy Another AWS Transit Gateway in a Different Region (e.g., us-west-1).**
2. **Create Peering Connection Between AWS TGWs**
   - Accept the peering request in the target region.
   - Update route tables for cross-region connectivity.

5. Verify Connectivity

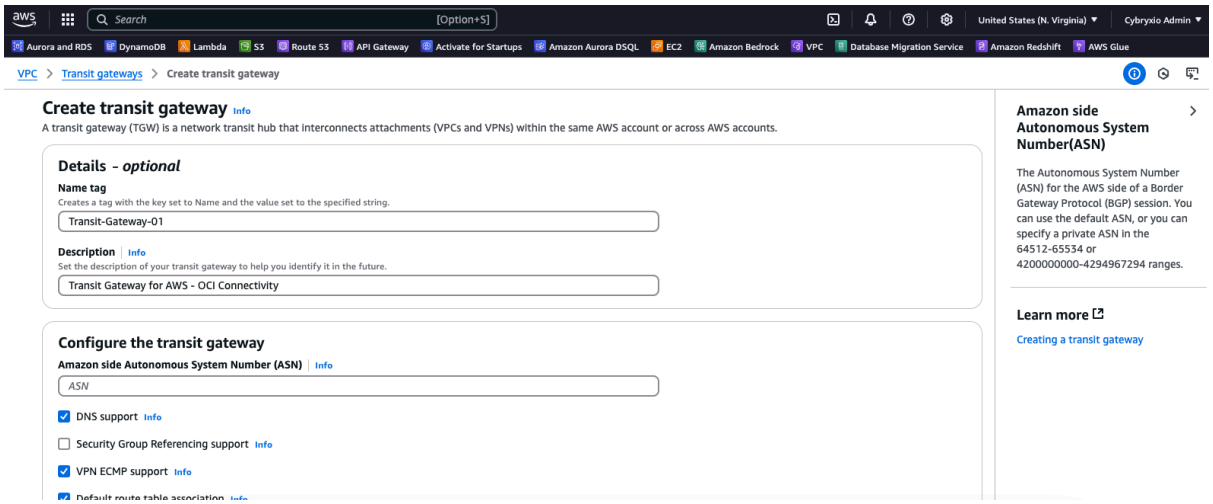
- Ensure AWS VPC Route Tables include OCI VCN routes.
- Check BGP Route Propagation in OCI.

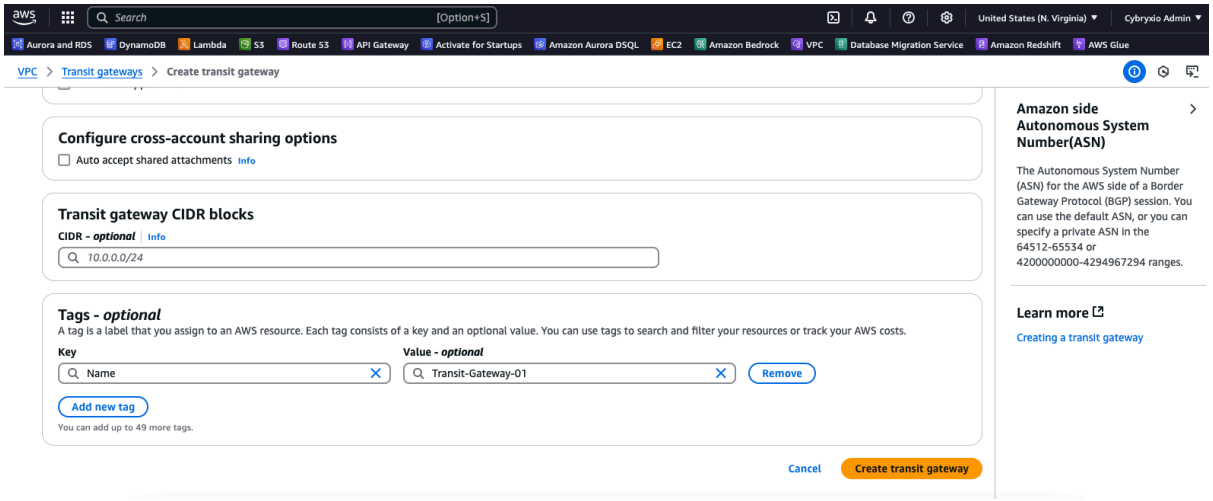
Implementation Steps

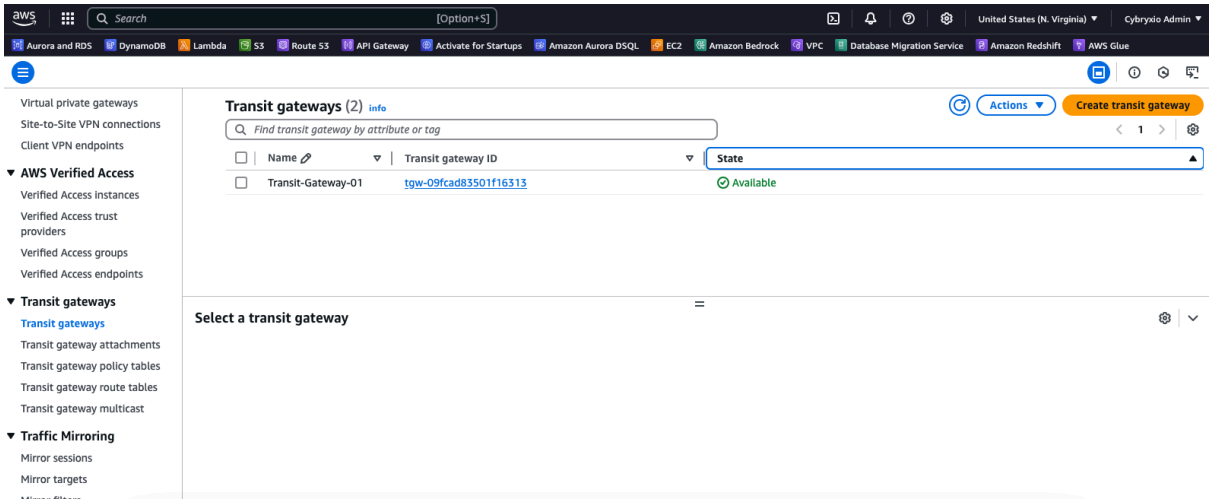
1. *Create AWS Transit Gateway and Attach to VPC*

   Go to AWS Console

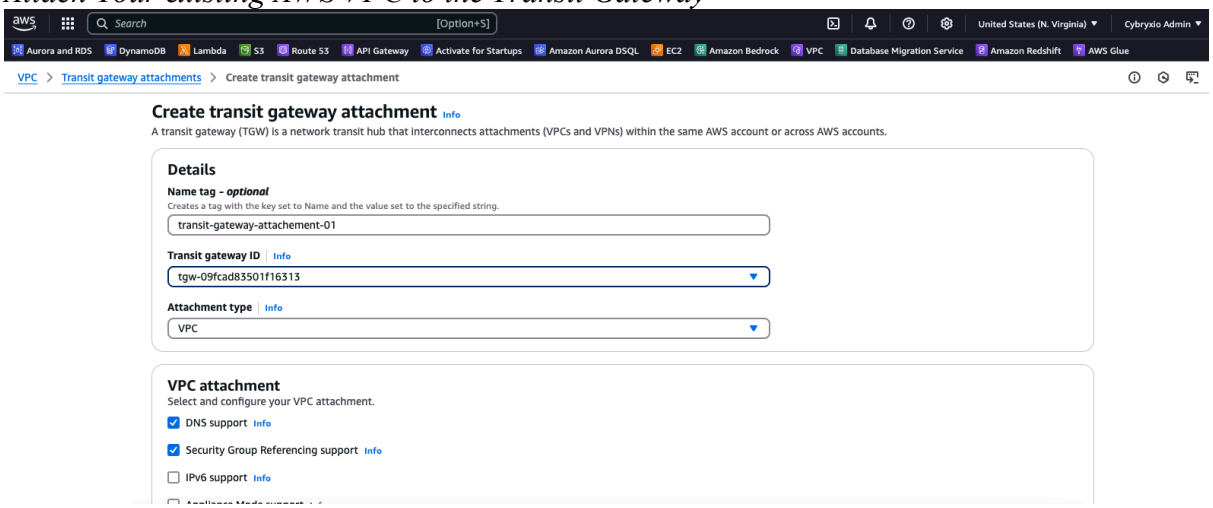
   VPC > Transit Gateway > Create Transit Gateway

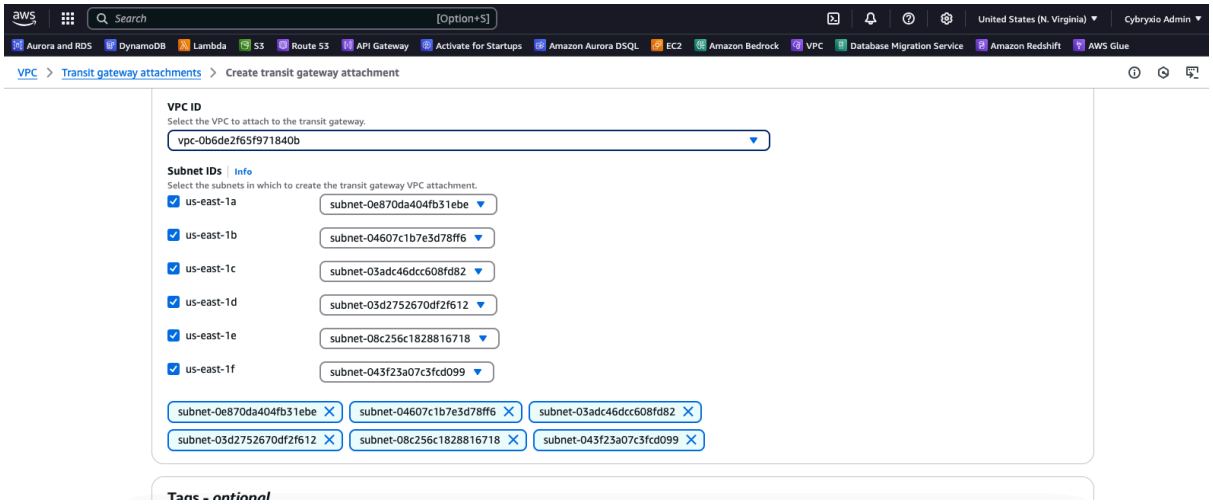


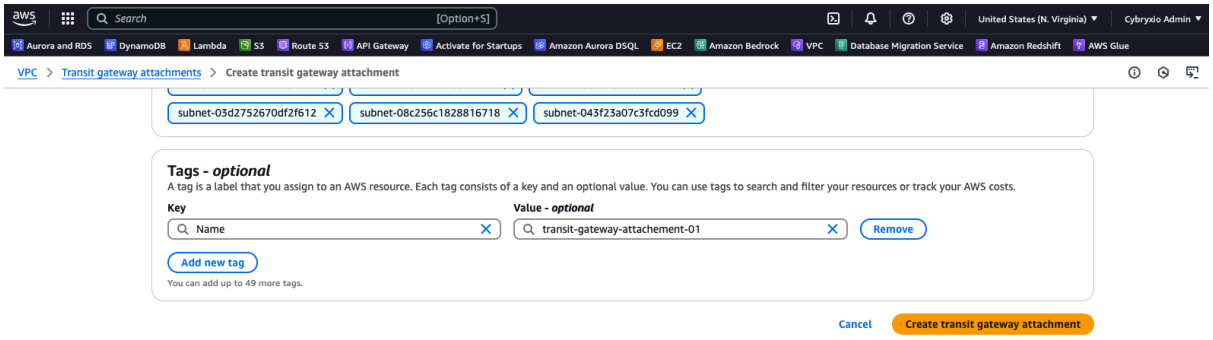


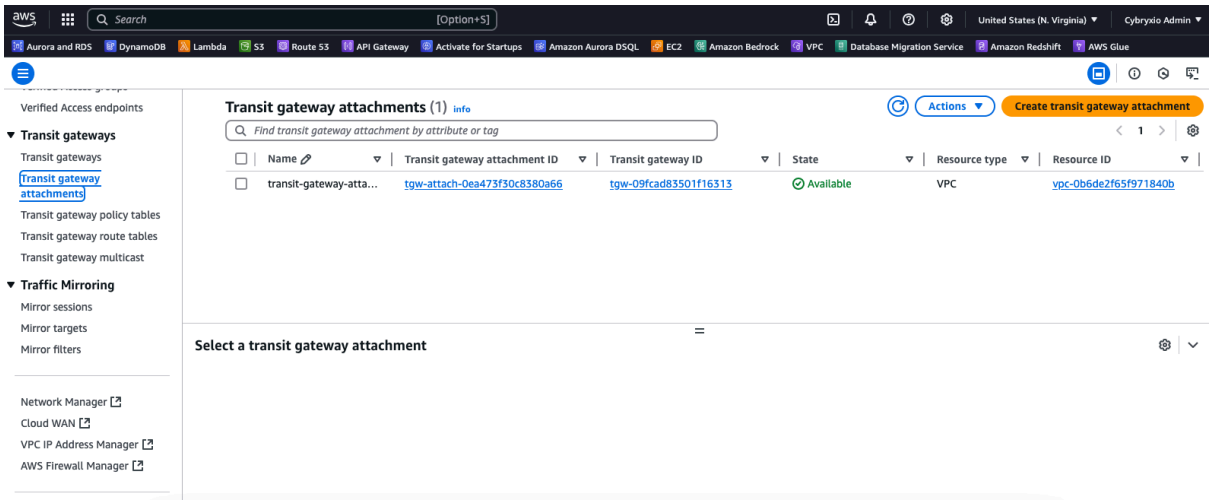


Attach Your existing AWS VPC to the Transit Gateway





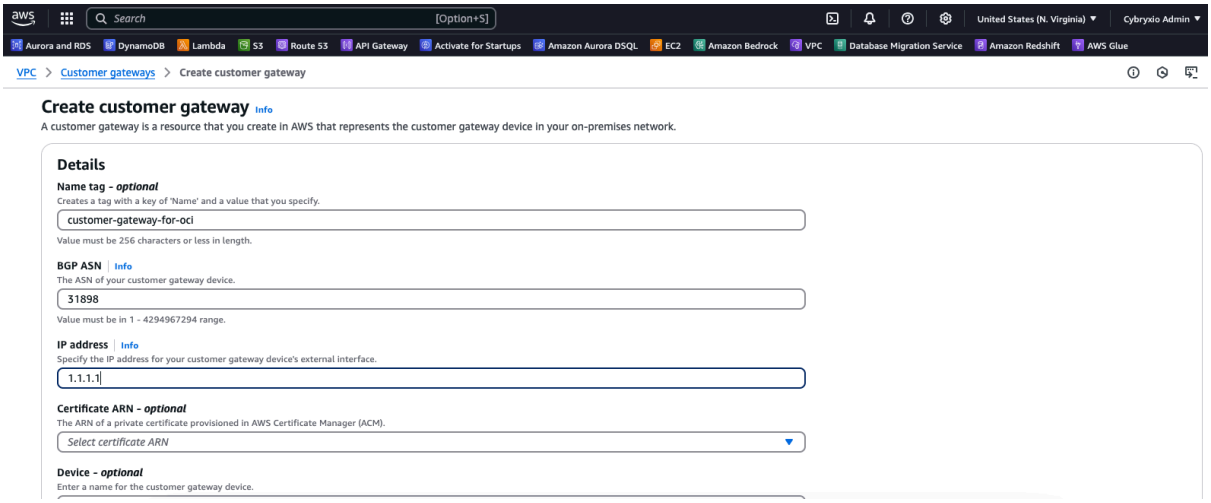


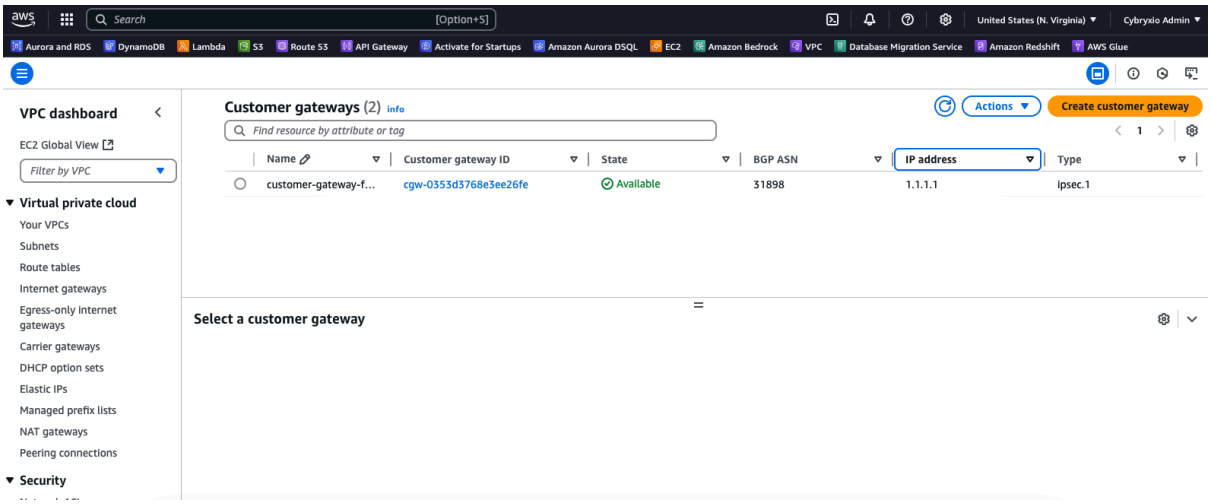


2. *Create AWS Customer Gateway*

   Create a temporary AWS customer gateway with a mock ip 1.1.1.1, this will eventually be replaced with a permanent CustomerGateway once we know the Public IP of OCI VPN Tunnel

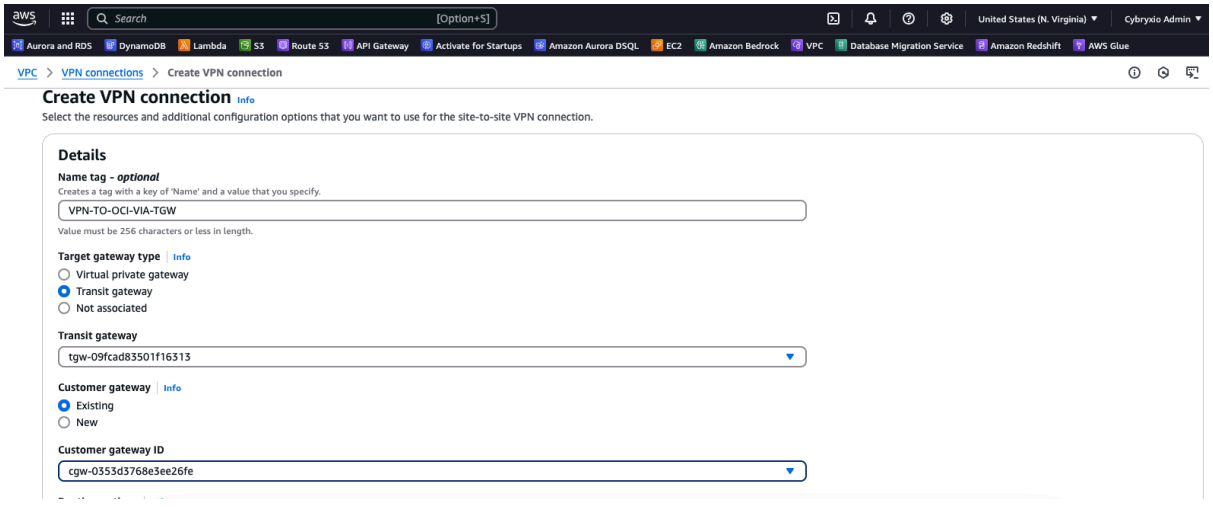
   ***BGP ASN:** Enter the OCI BGP ASN. Oracle's BGP ASN for the commercial cloud is 31898, except the Serbia Central (Jovanovac) region which is 14544.*

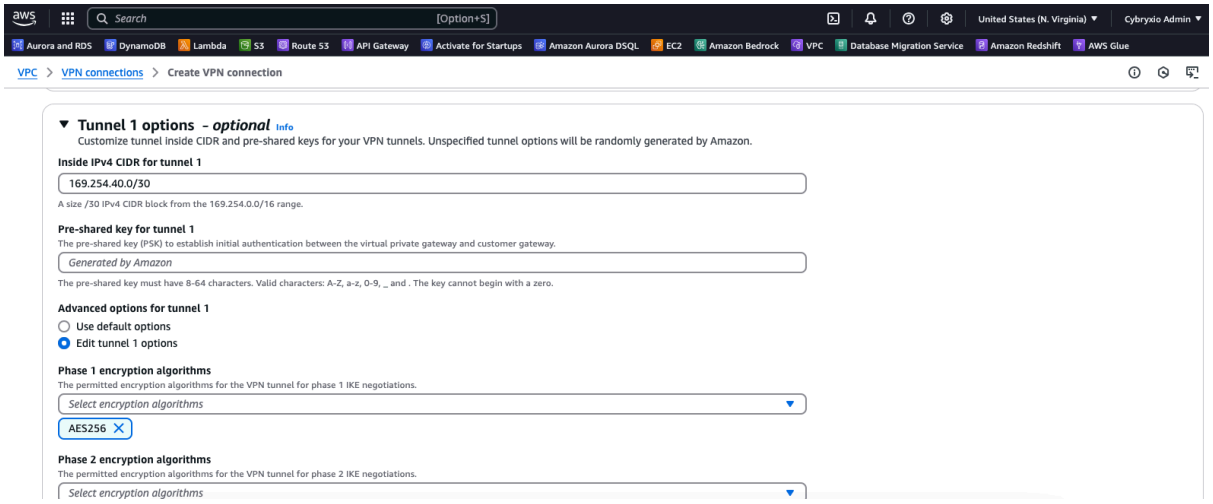




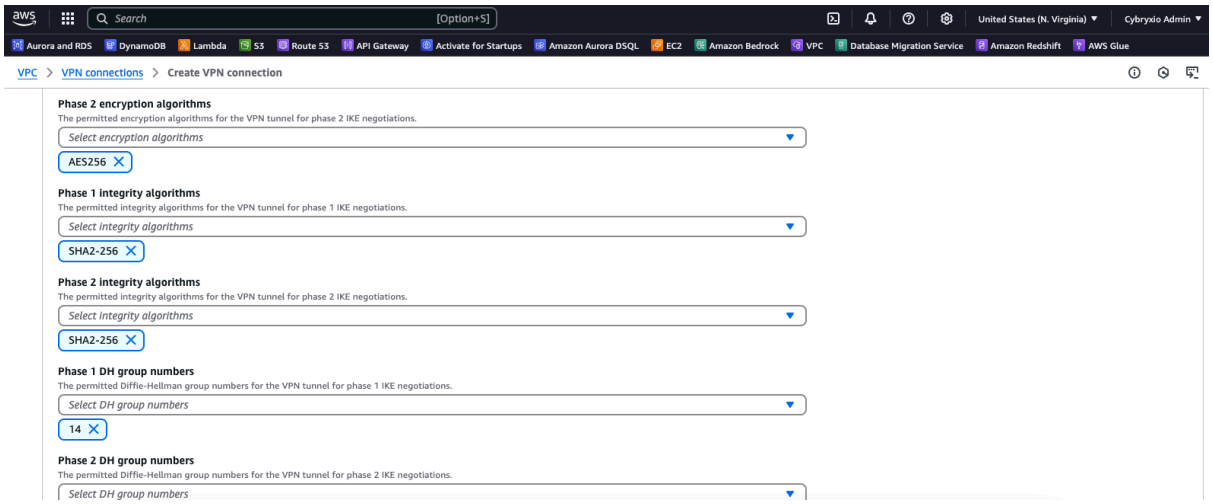
3. *Create Site-to-Site VPN from AWS Console*

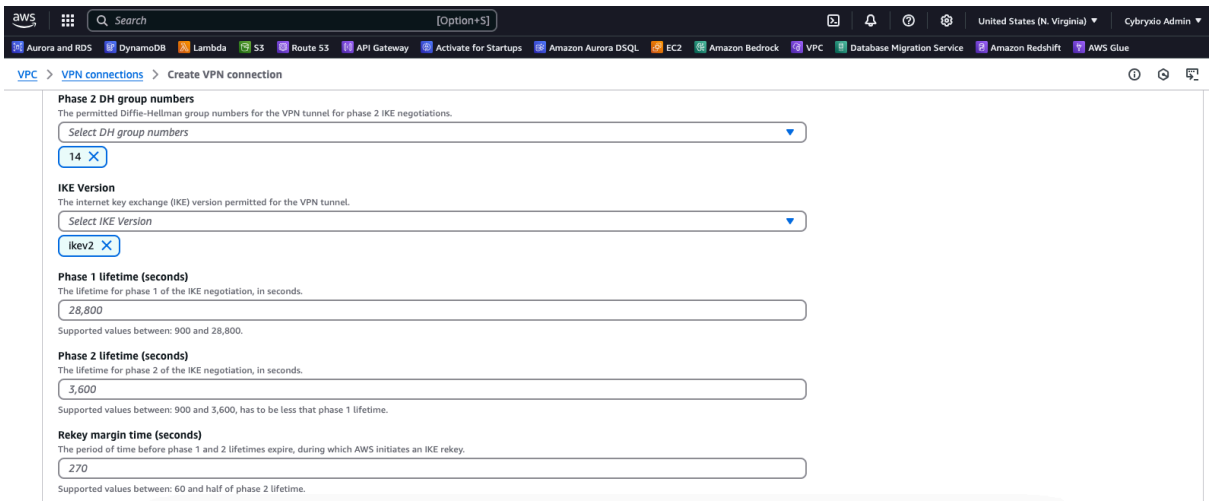
   AWS Console > VPC > Virtual Private Network > Site-to-Site VPN > Create VPN Connection

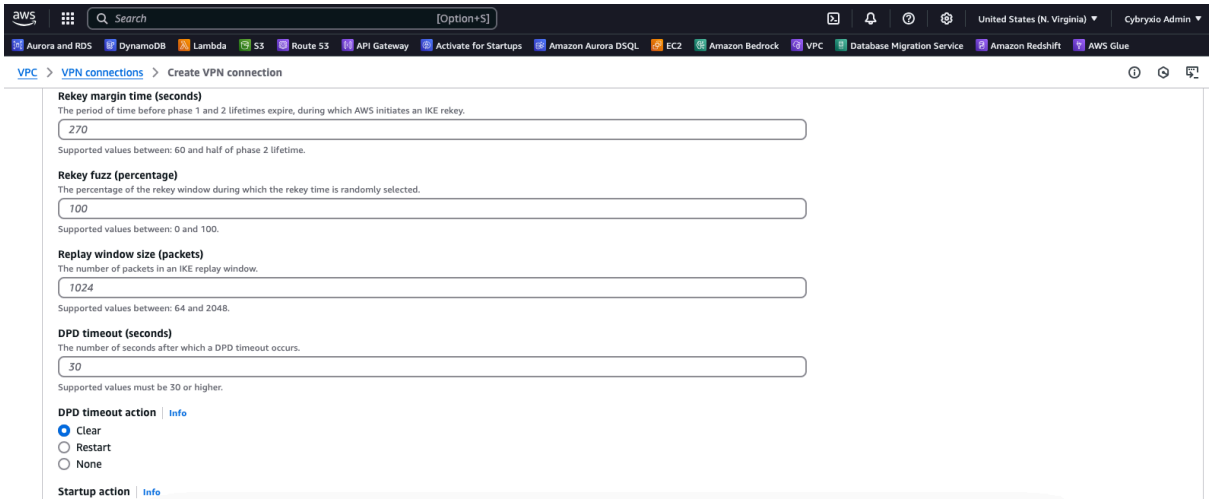




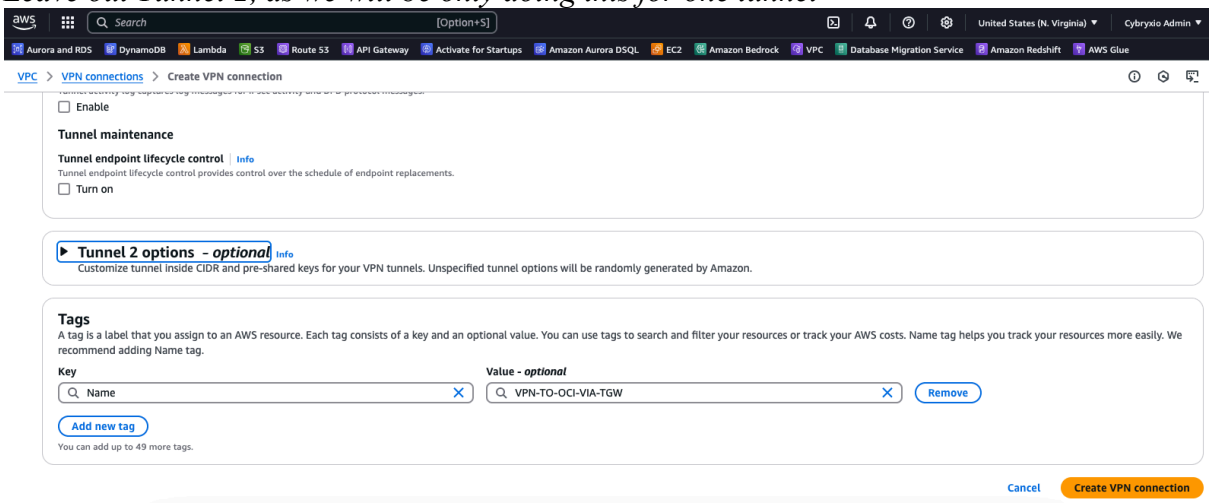
Use the Options in the Screenshot and remove the other ones as per below



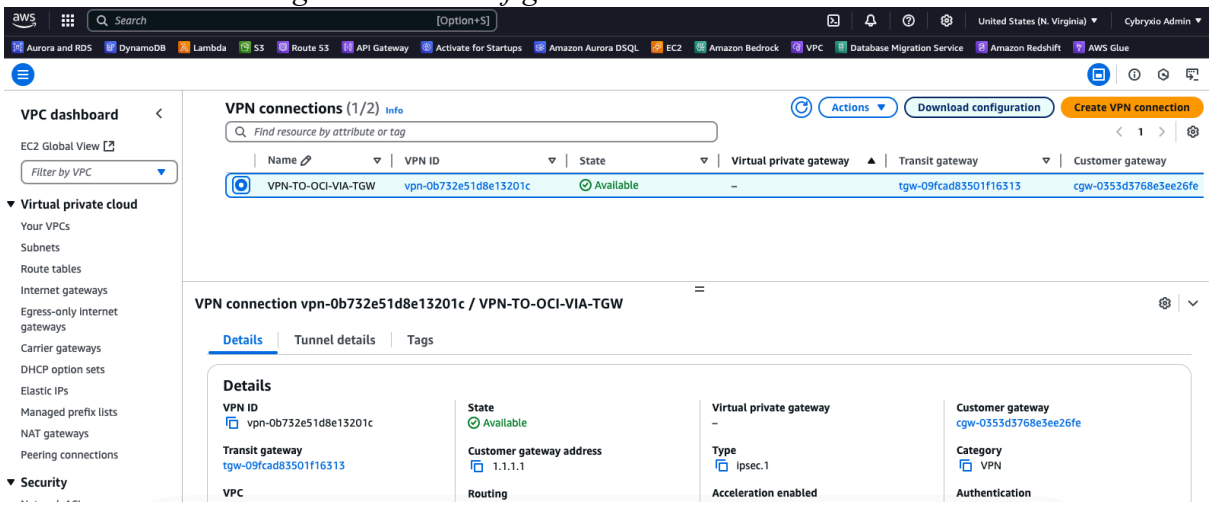




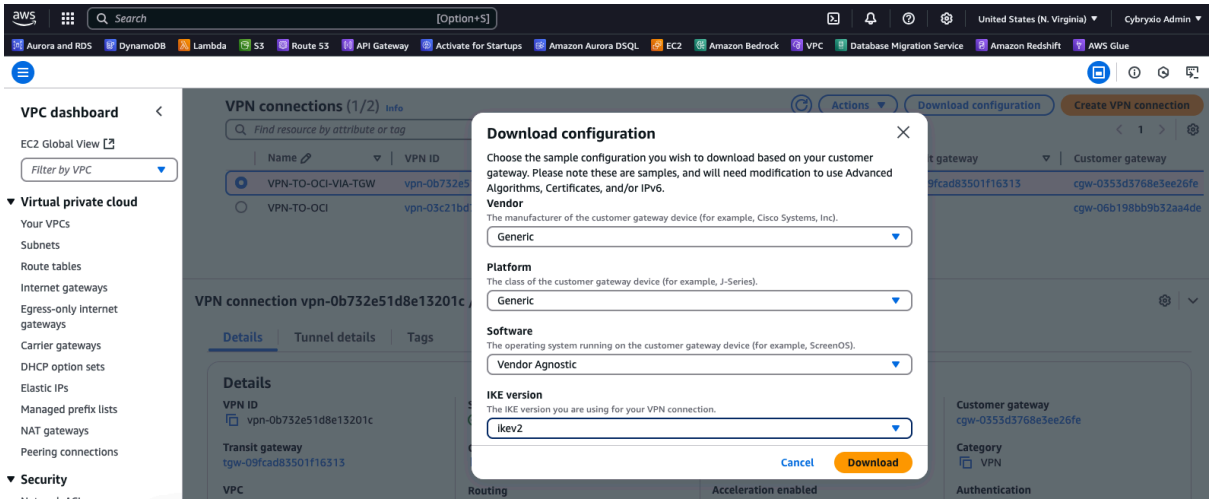
Leave out Tunnel 2, as we will be only doing this for one tunnel



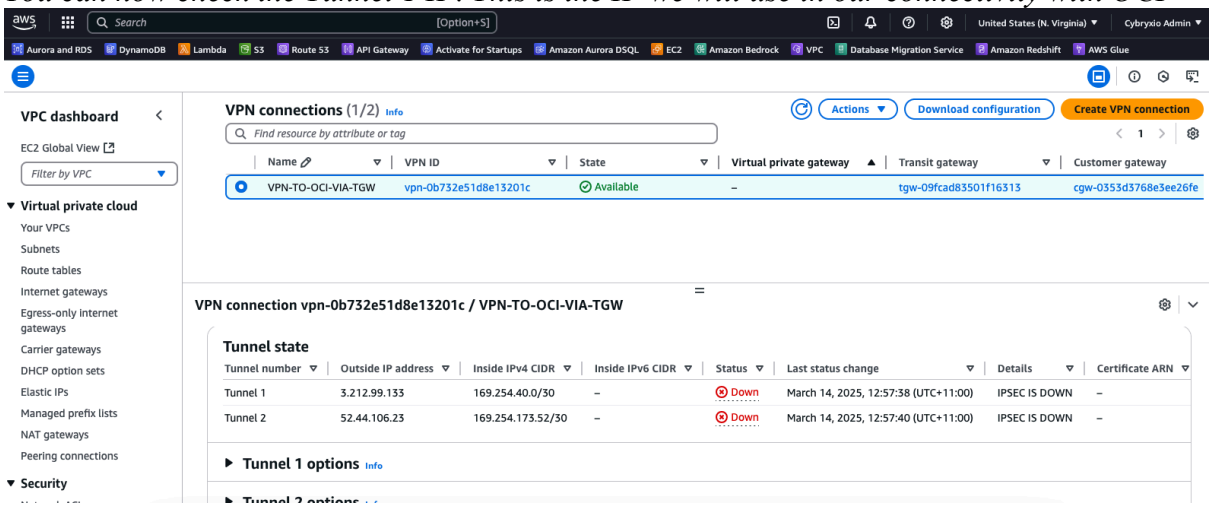
Once the VPN connection is completed, download the profile, by selecting the VPN connection and clicking Download configuration



Download Configuration

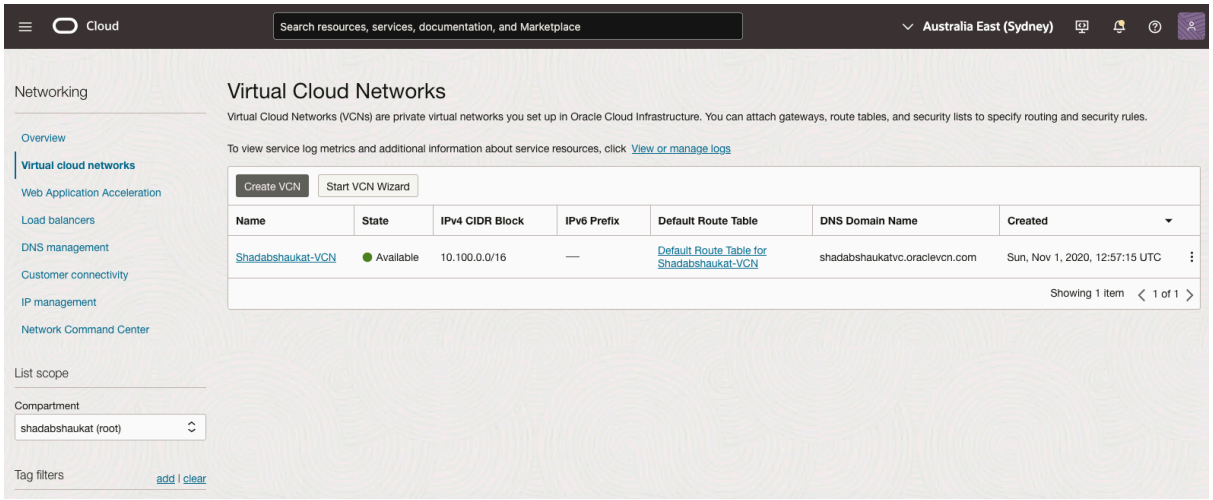


You can now check the Tunnel 1 IP. This is the IP we will use in our connectivity with OCI

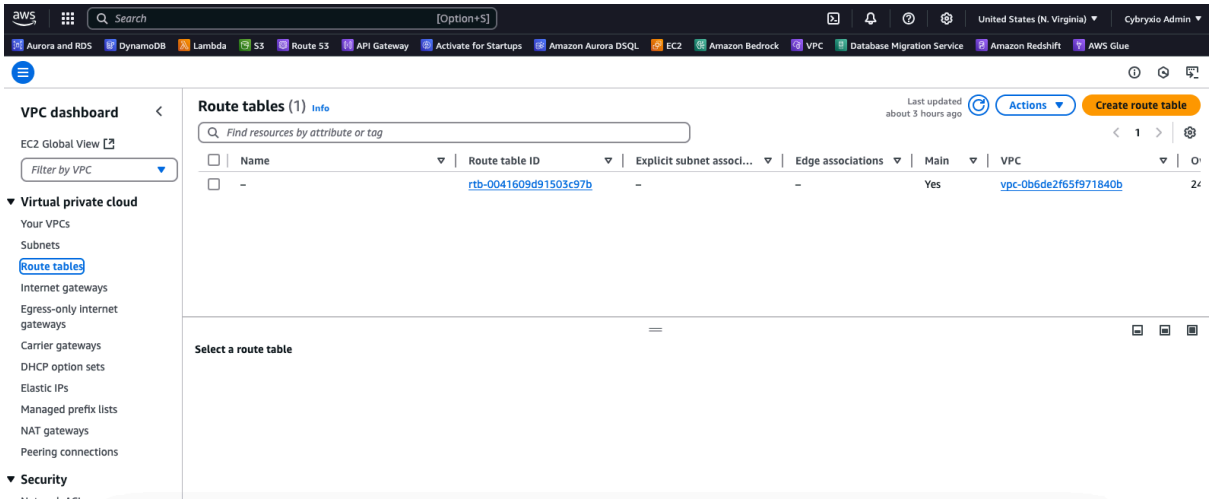


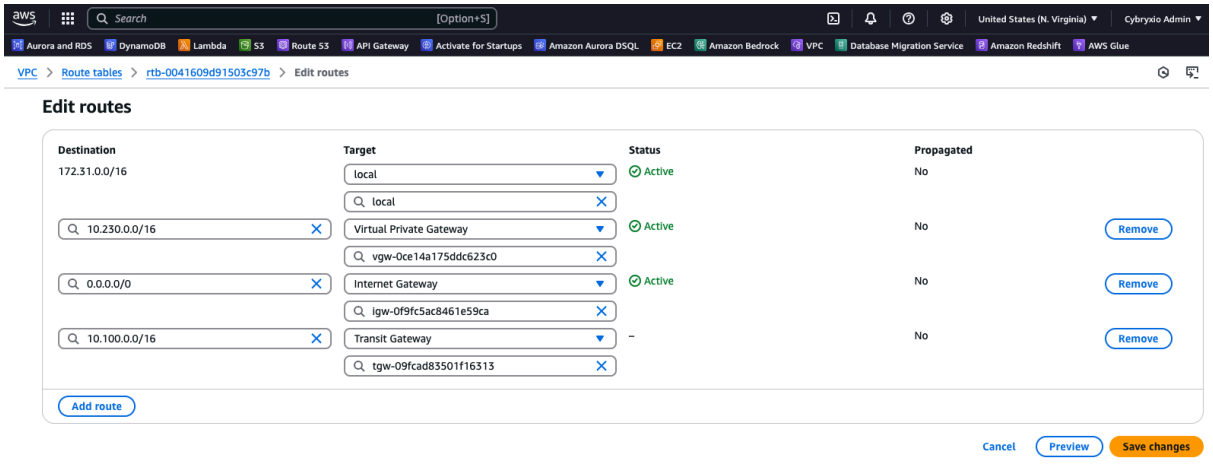
4. *Add your OCI VCN Routes to the Subnet in your AWS VPC*

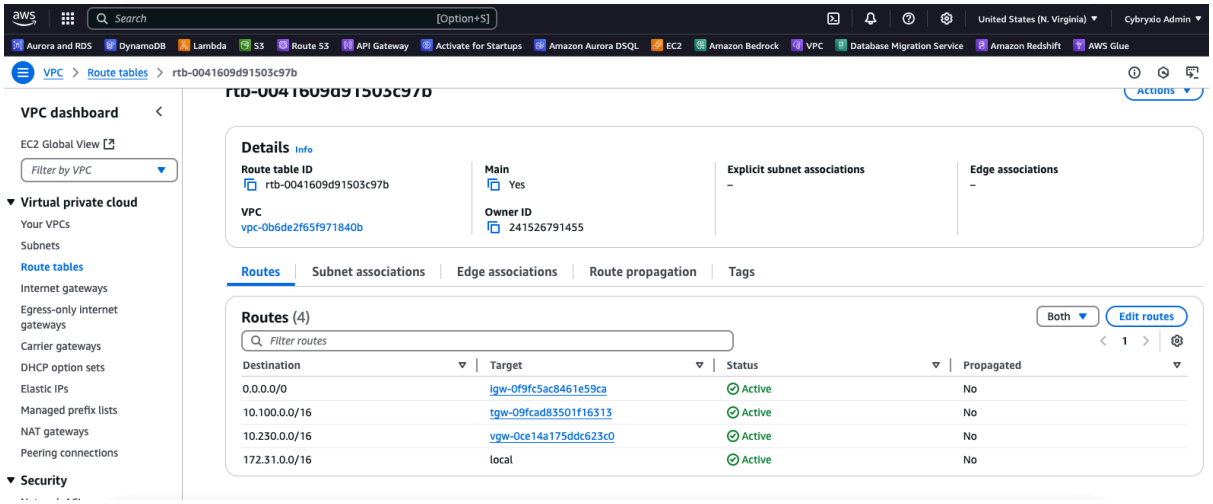
OCI VCN CIDR is : 10.100.0.0/16



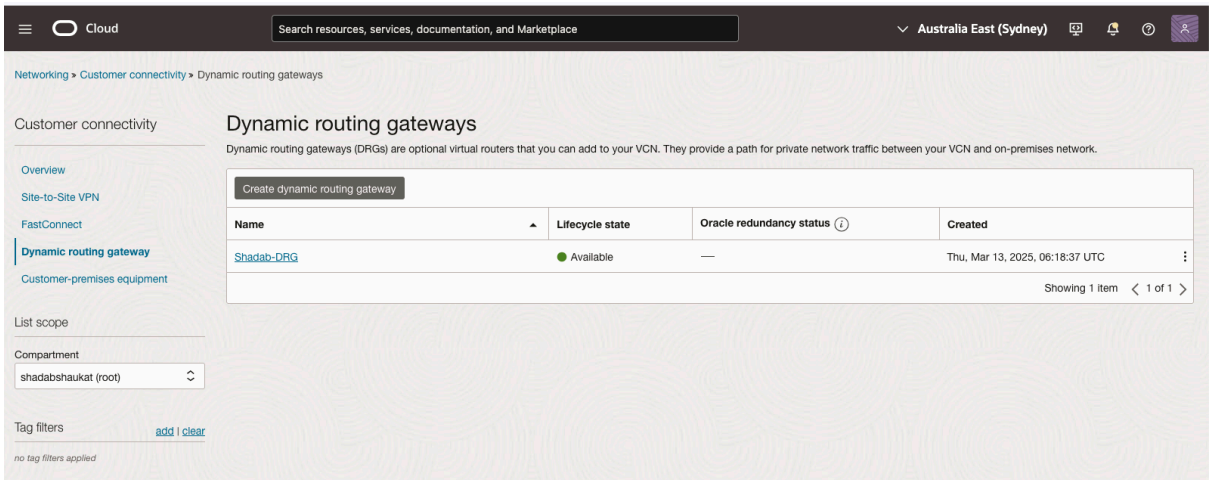
Add 10.100.0.0/16 to AWS VPC Subnet's ***Route Table***

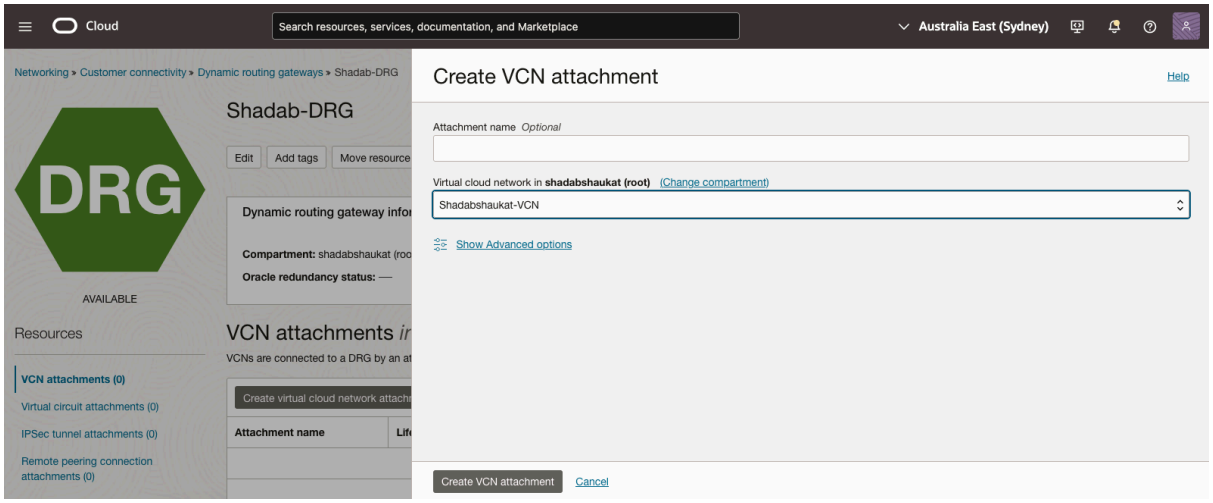


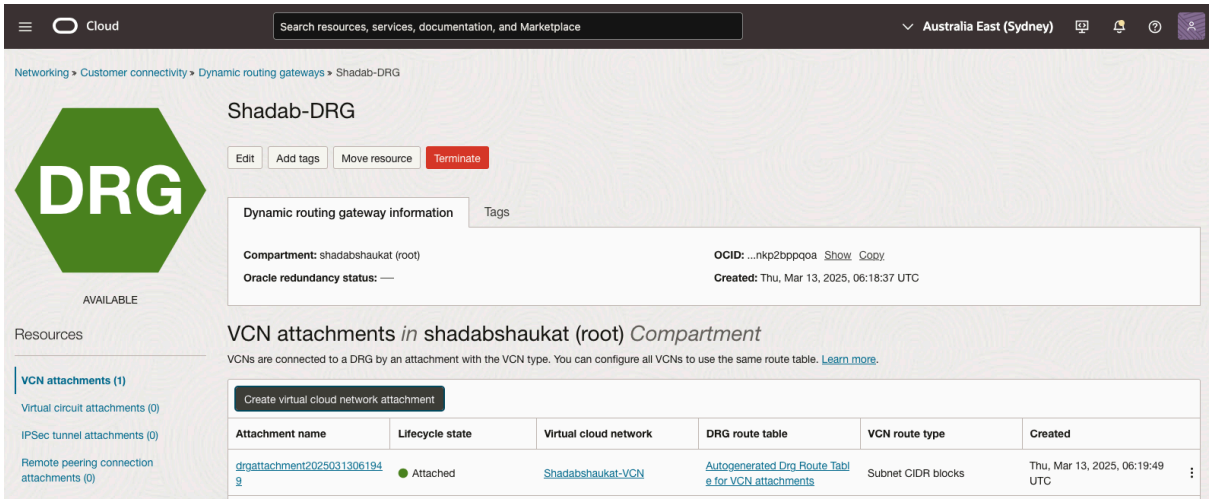




5. *Add a DRG on OCI and Attach your VCN to it*

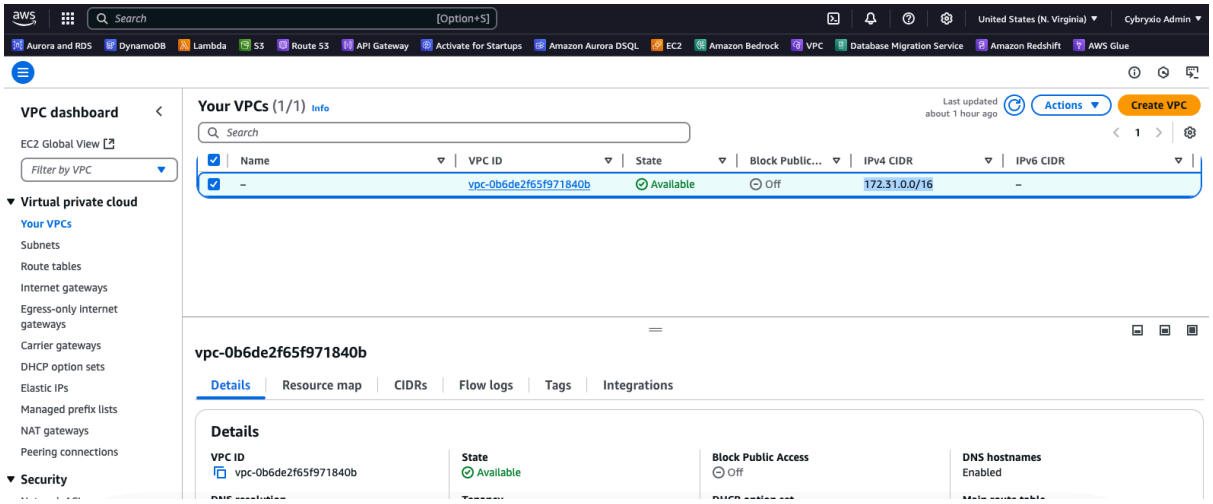




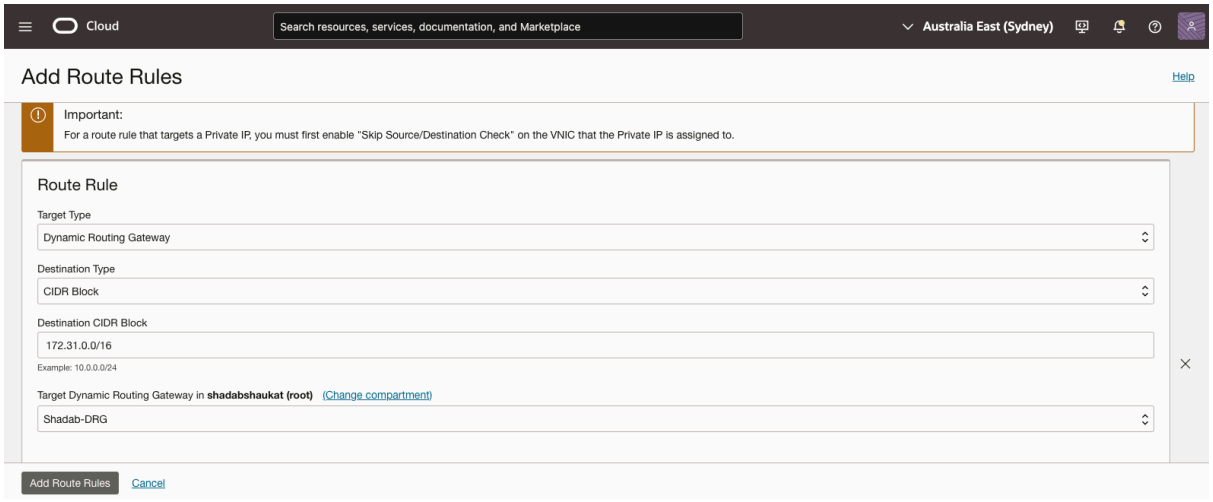


6. *Add your AWS VPC CIDR to Route tables on OCI*

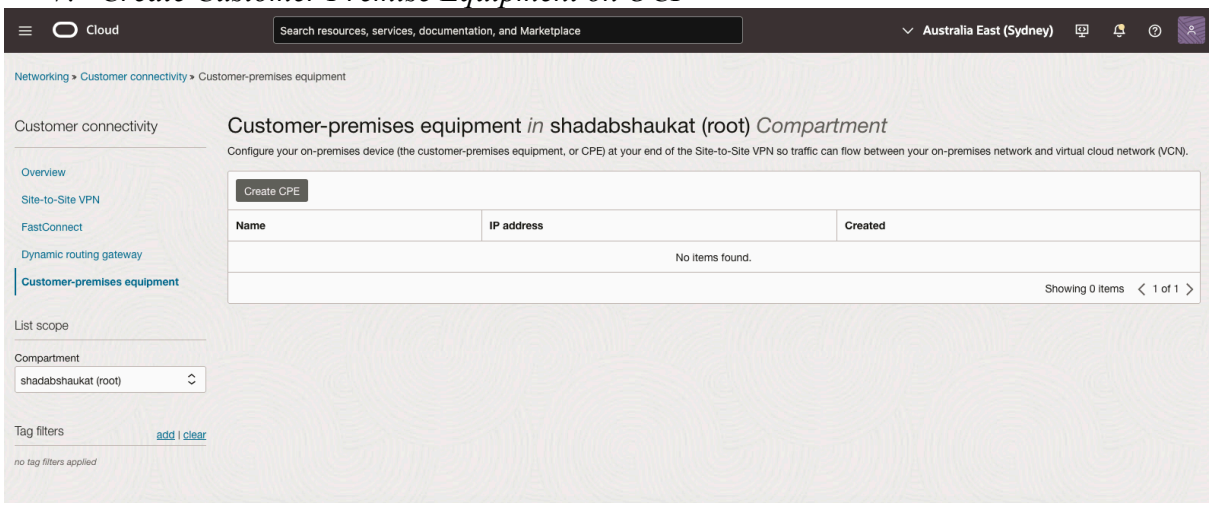
AWS VPC CIDR is 172.31.0.0/16



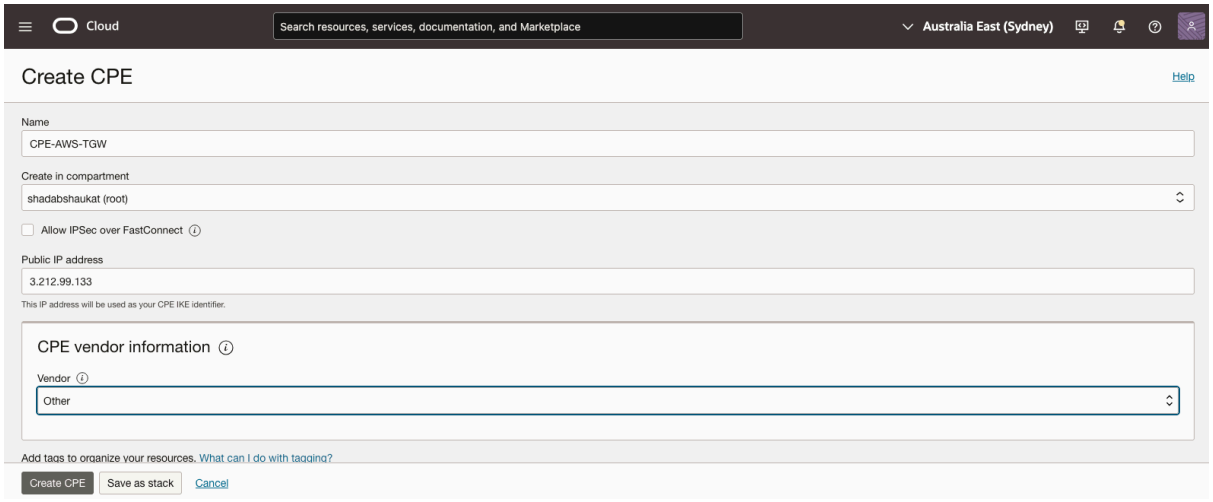
Add Route Table Rule on OCI VCN Subnet



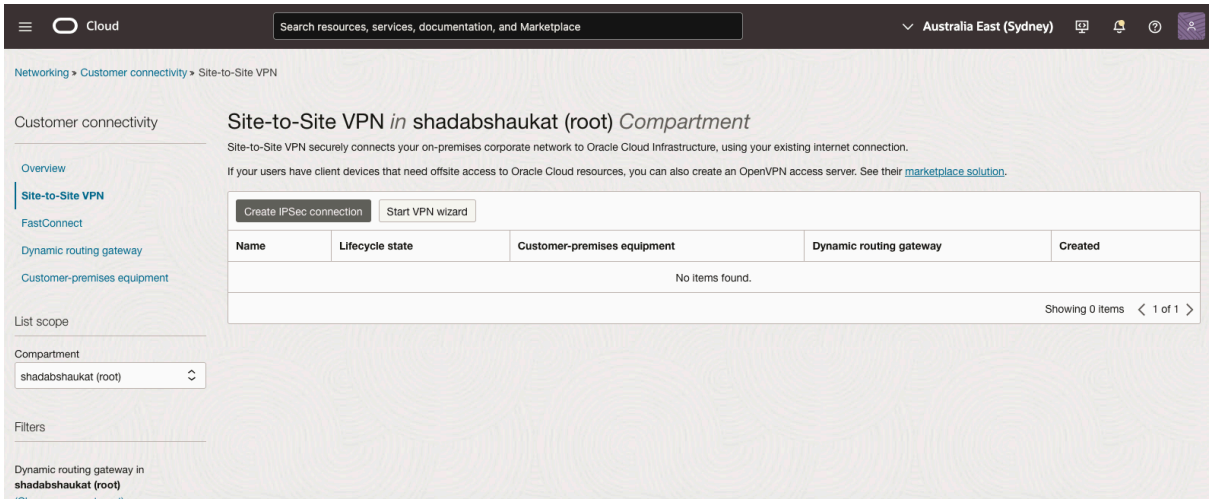
7. *Create Customer Premise Equipment on OCI*

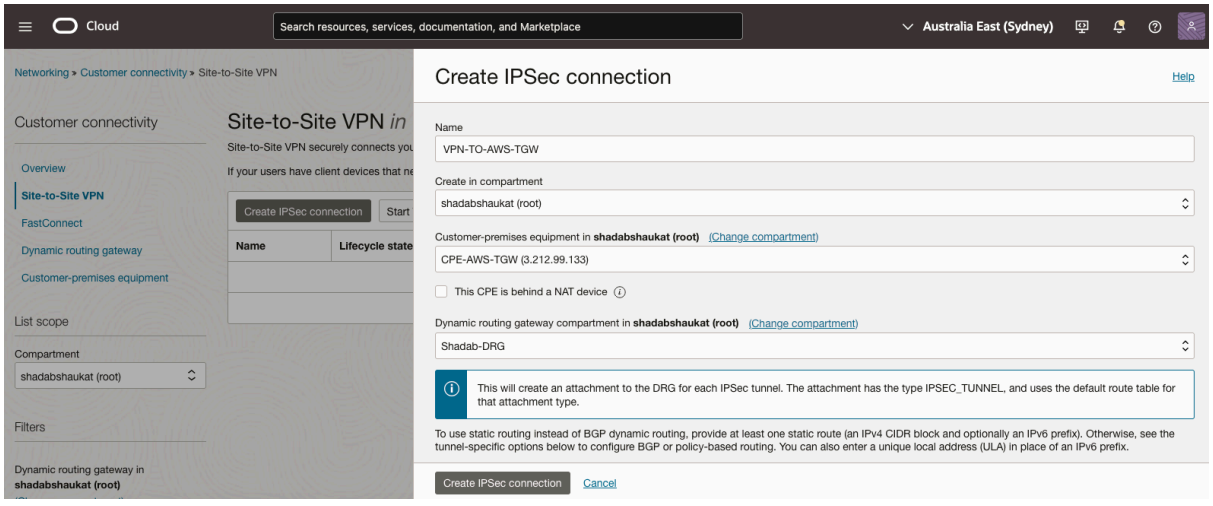


Use the Public IP address of your Tunnel 1 which we provisioned earlier 3.212.99.133

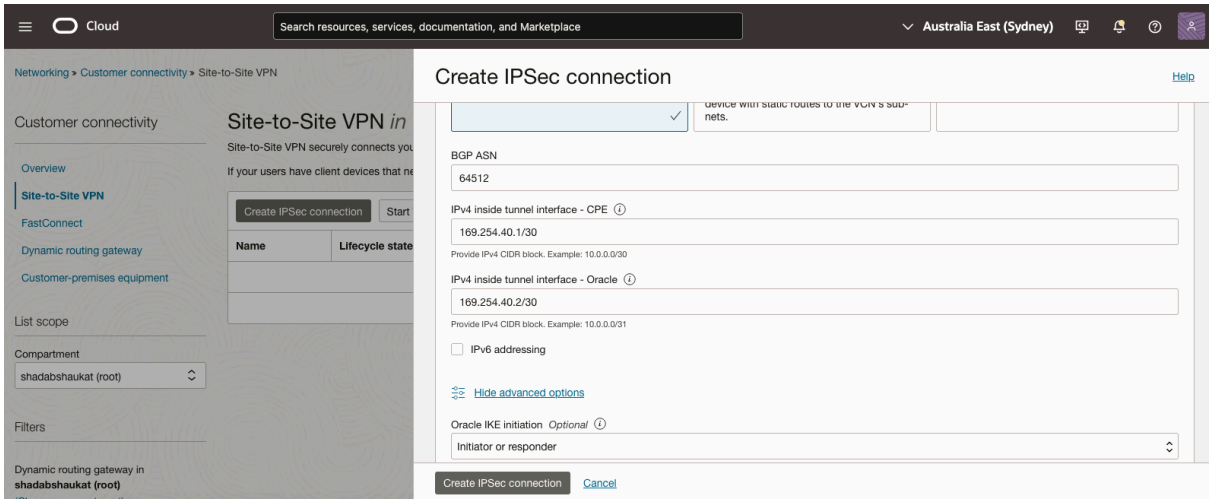


8. *Create Site-to-Site VPN on OCI*



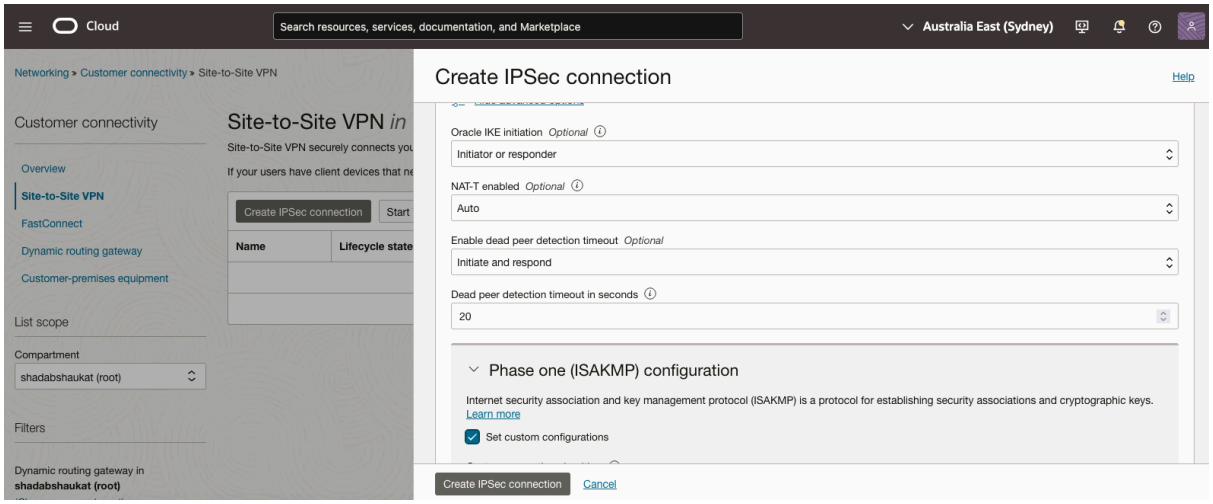


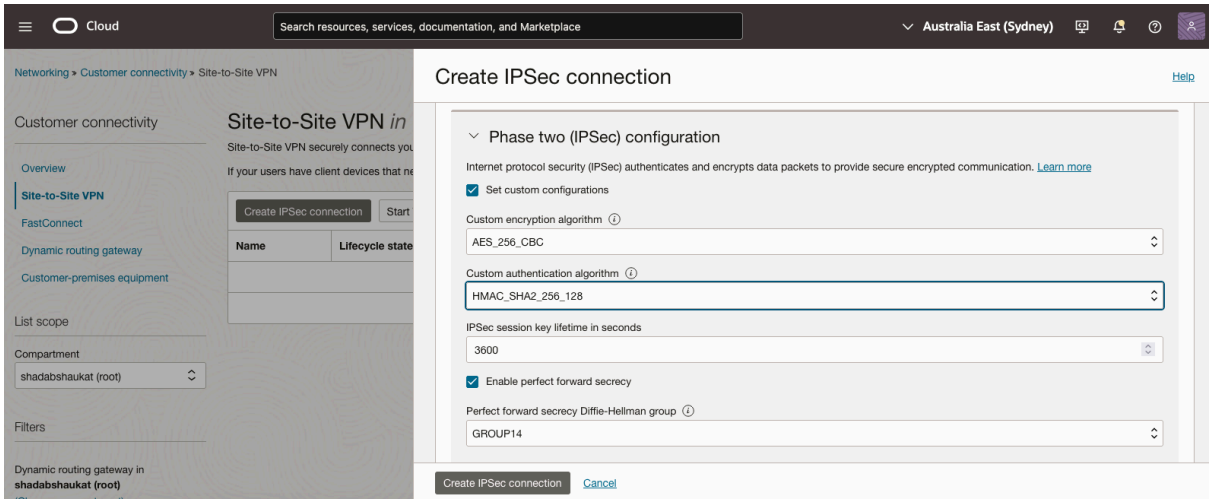
AWS BGP ASN is 64512.



Show advanced option

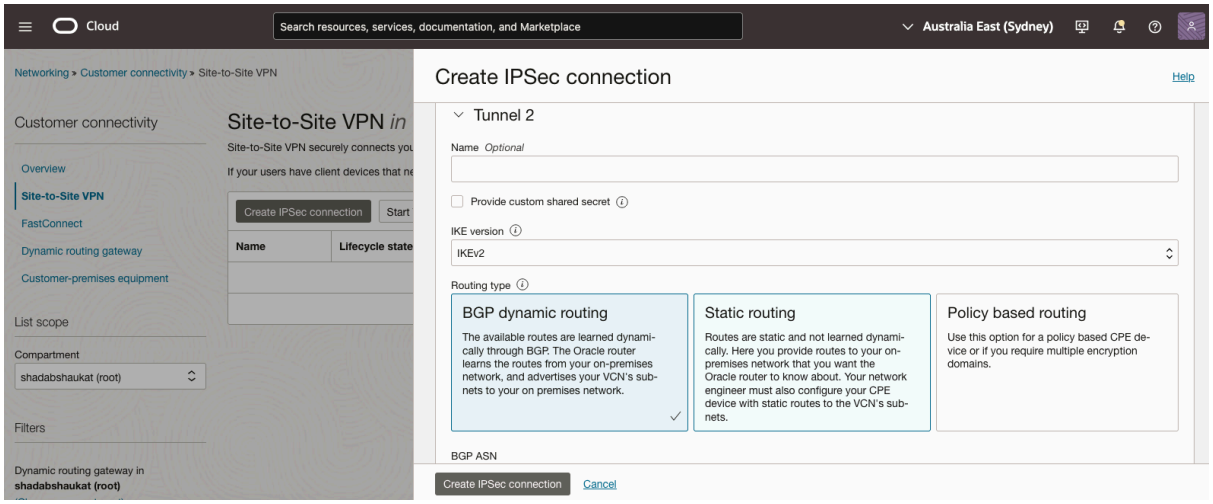
Go-to 'Phase one (ISAKMP) configuration'

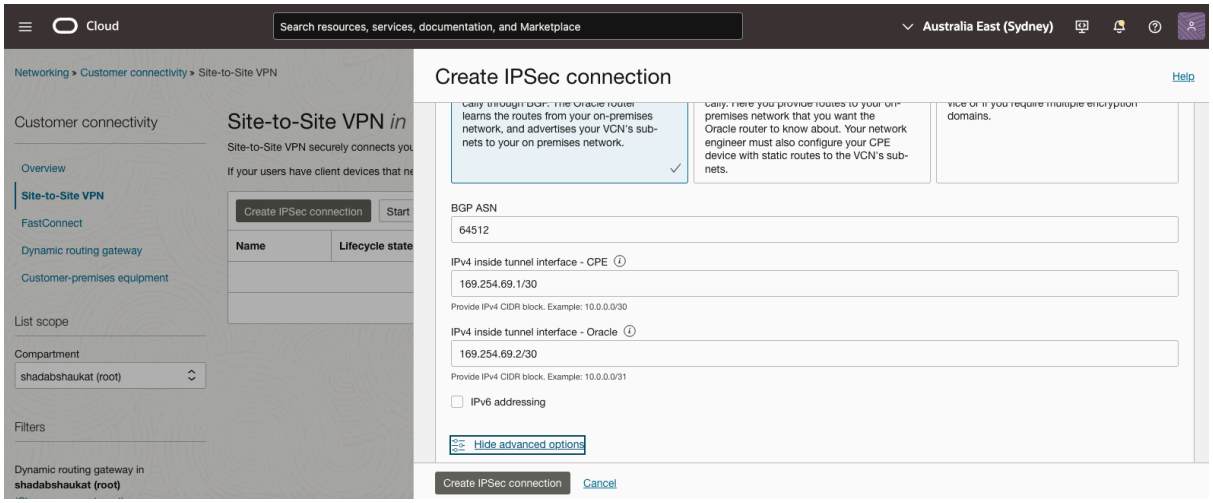


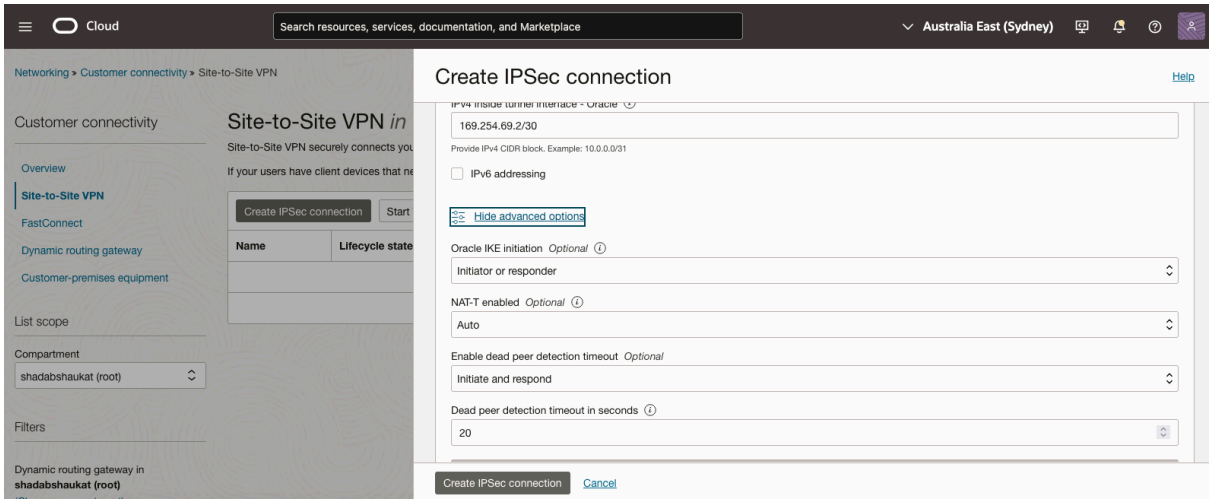


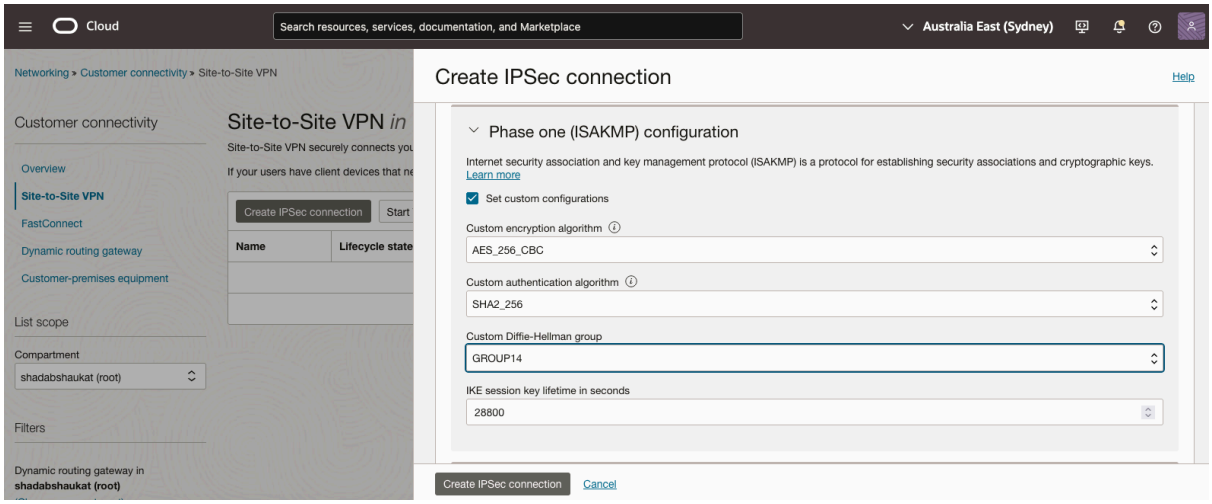
Do the same for Tunnel 2 - Phase one and two (ISAKMP) configuration.

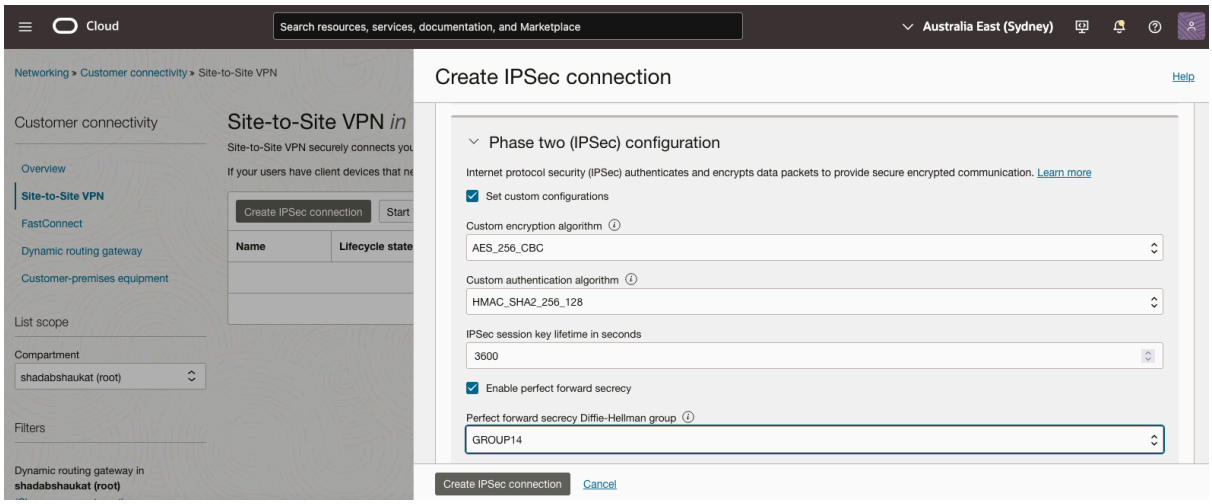
Important Note: We're only going to configure this for Tunnel 1 on AWS to Tunnel 2 on OCI. For redundancy you need to create a second Site to Site VPN on AWS and map to Tunnel 2 on OCI.



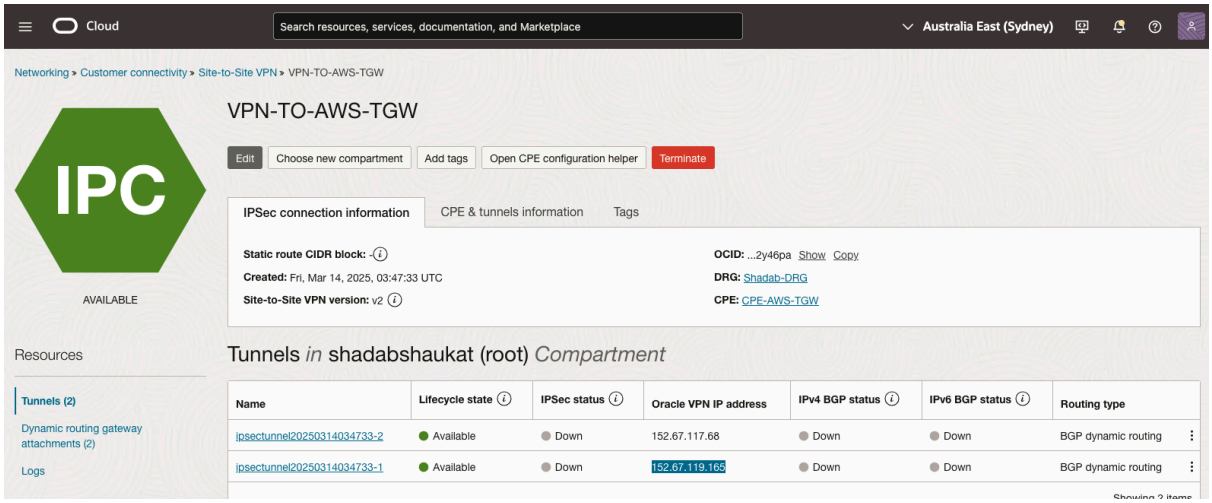


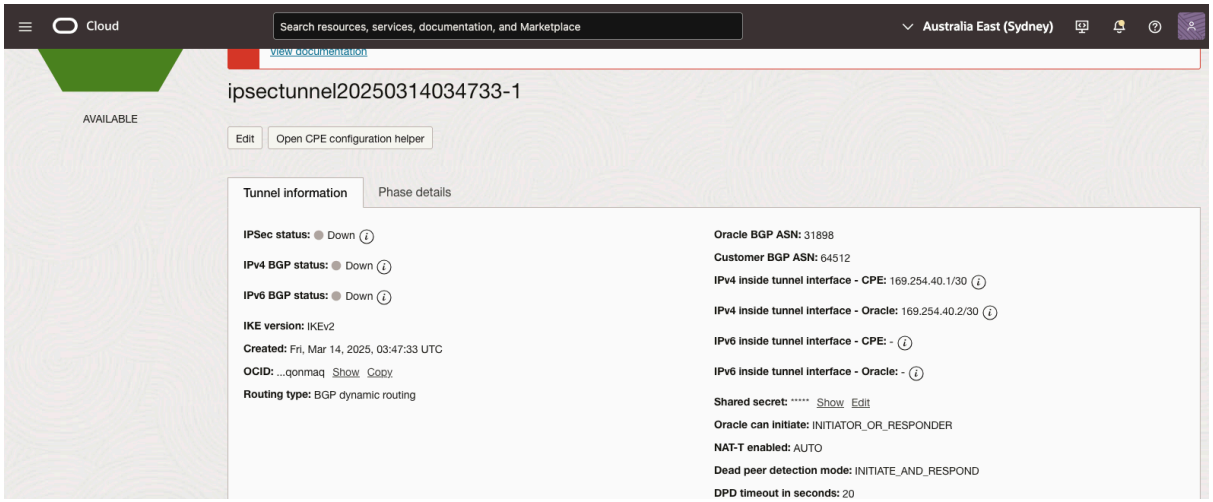


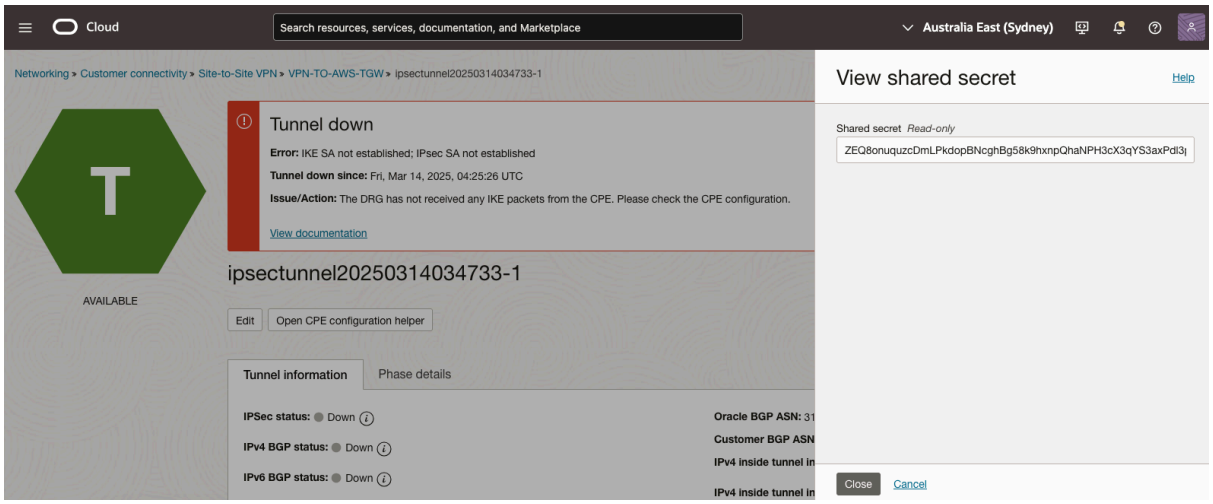




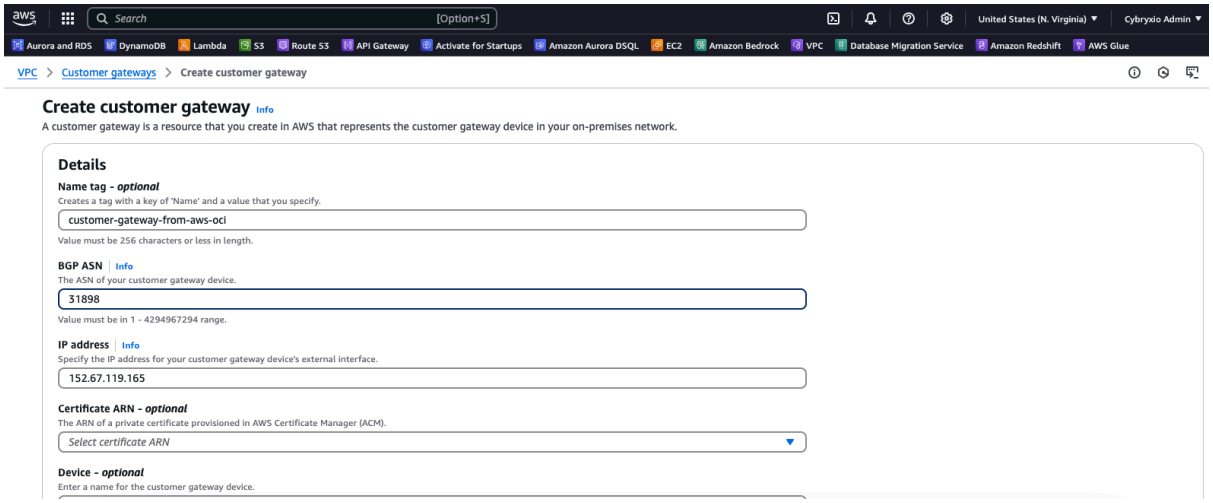
Go to Tunnel 1 and copy the Shared Secret





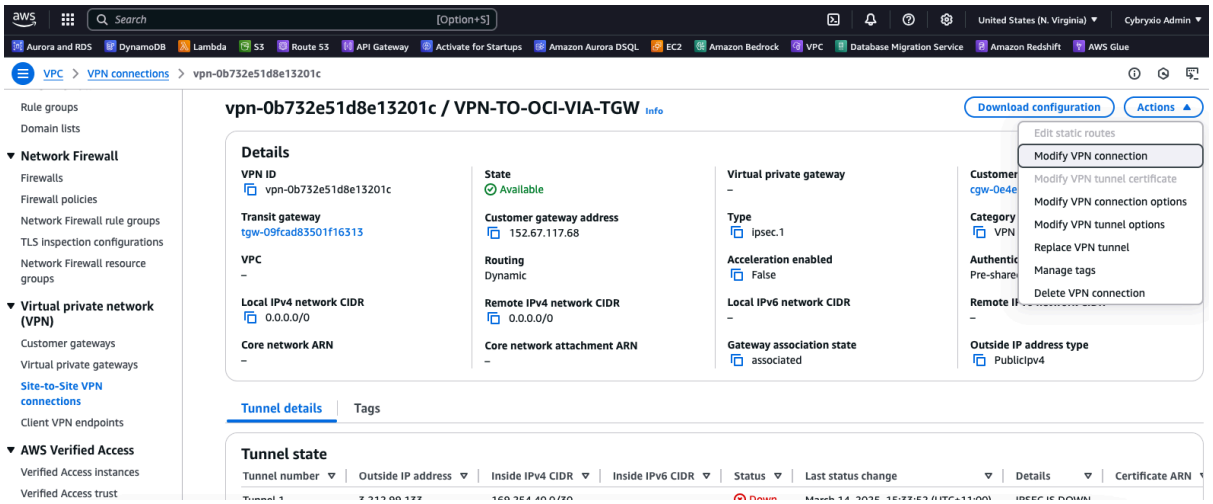


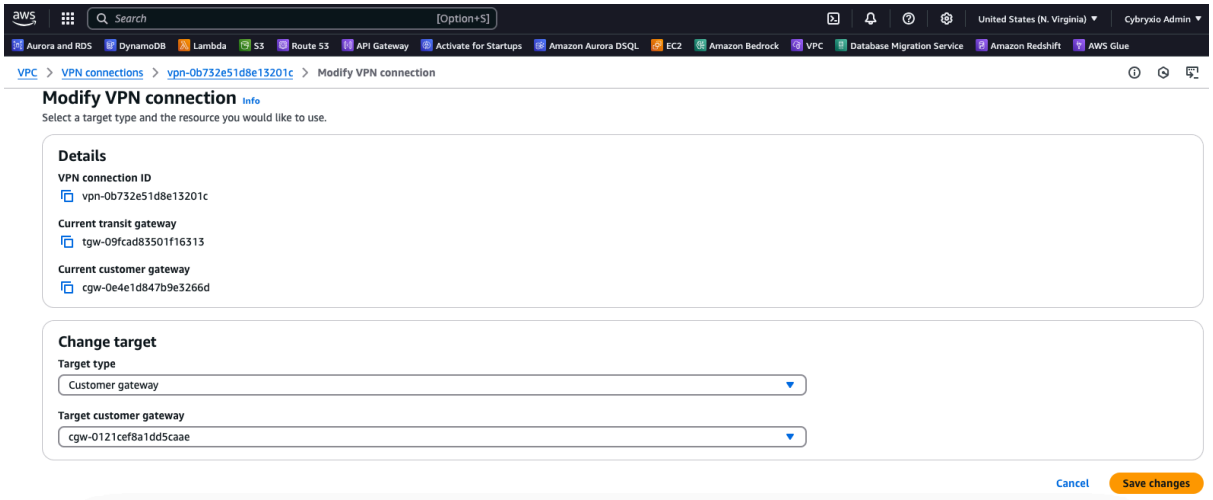
9. *Go back to AWS Console and Create new Customer Gateway with Public IP of OCI Tunnel 1 152.67.119.165 and Oracle ASN 31898*



Create Customer Gateway

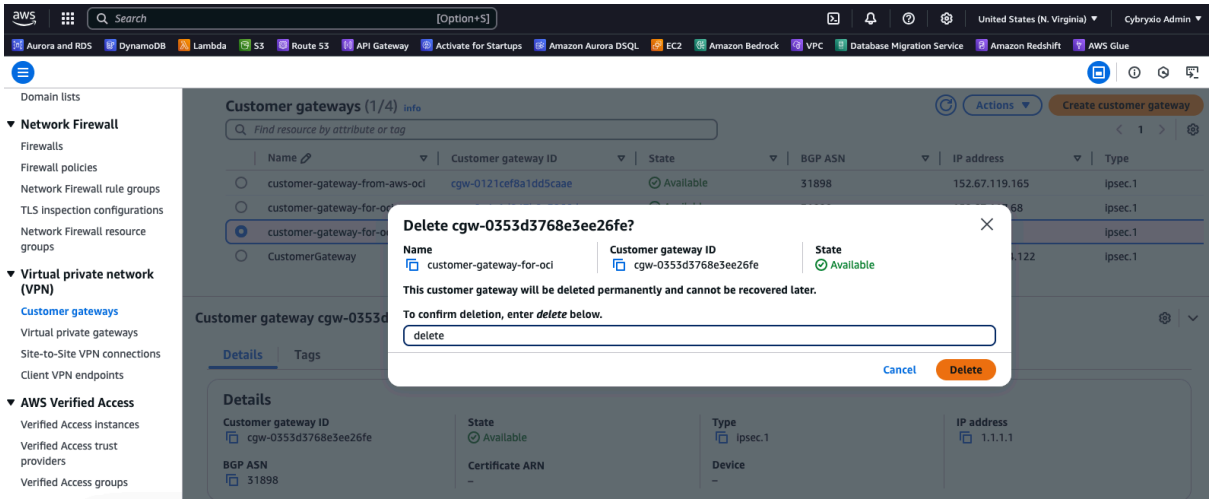
Modify the VPN connection and add the new customer gateway



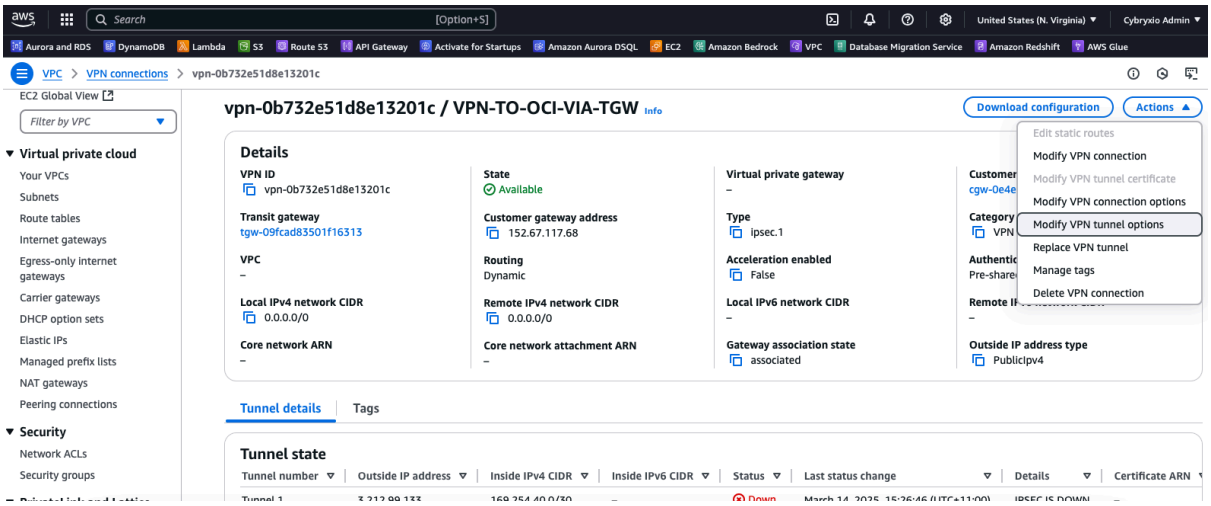


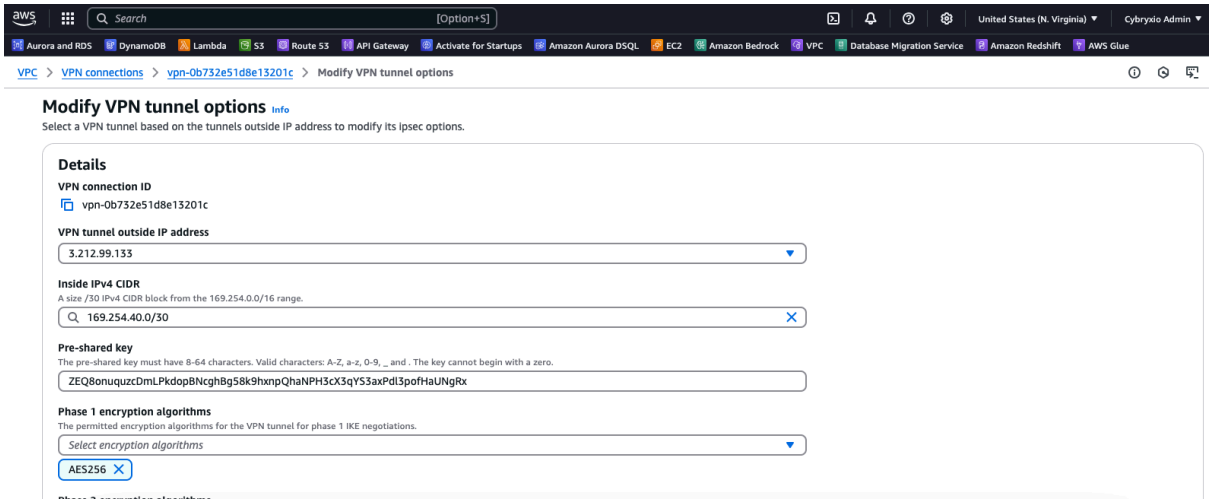
Save changes.

Make sure to delete the previous temporary customer gateway one after successfully adding new one.



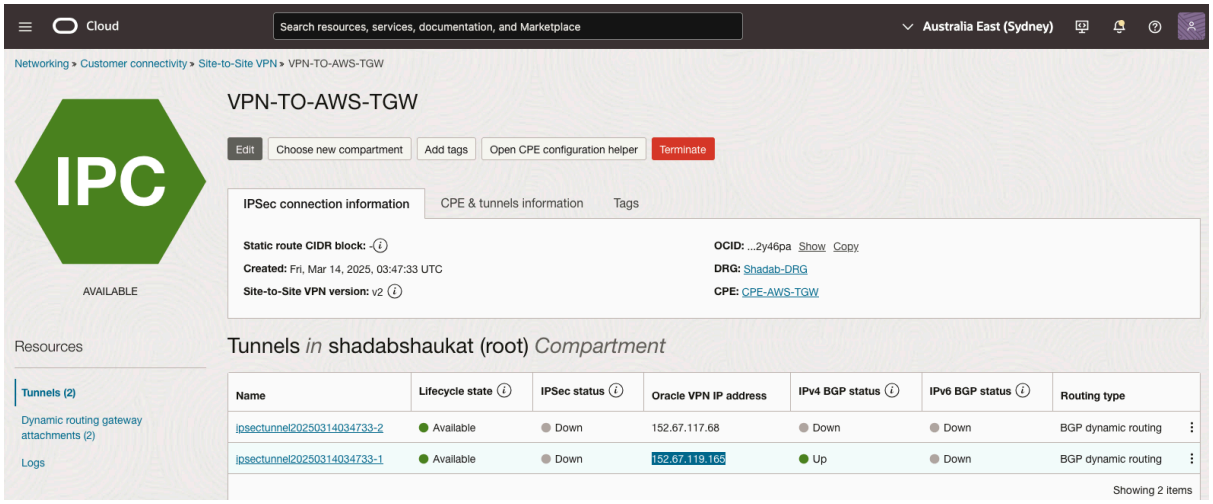
10. Modify VPN Tunnel Options and Select Tunnel 1; Add the OCI VPN Tunnel 1 secret key and save changes

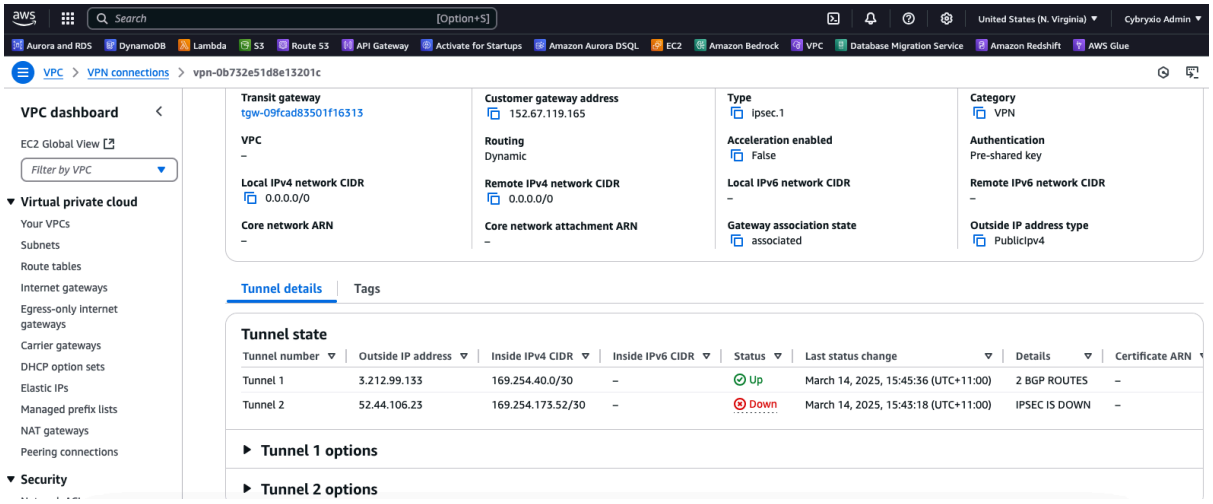




Wait for the 'modifying' to complete and VPN connection to be 'available' on AWS.

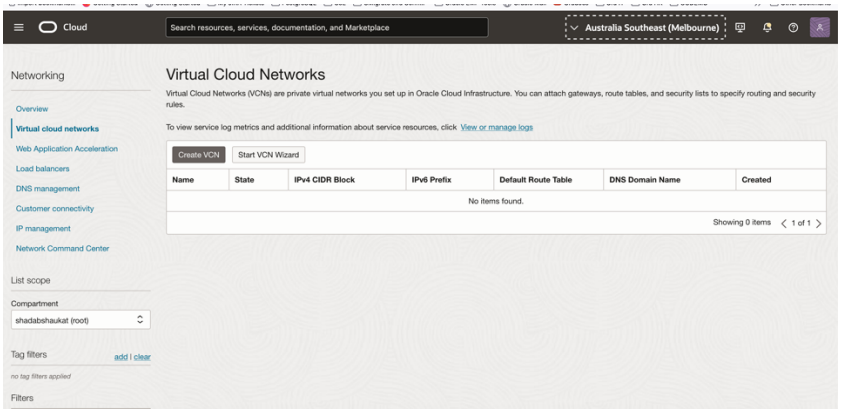
Now check the Tunnel Status on OCI and AWS. If everything was setup correctly, the Tunnel 1 on AWS will be connected to IPSEC VPN connection Tunnel on OCI.

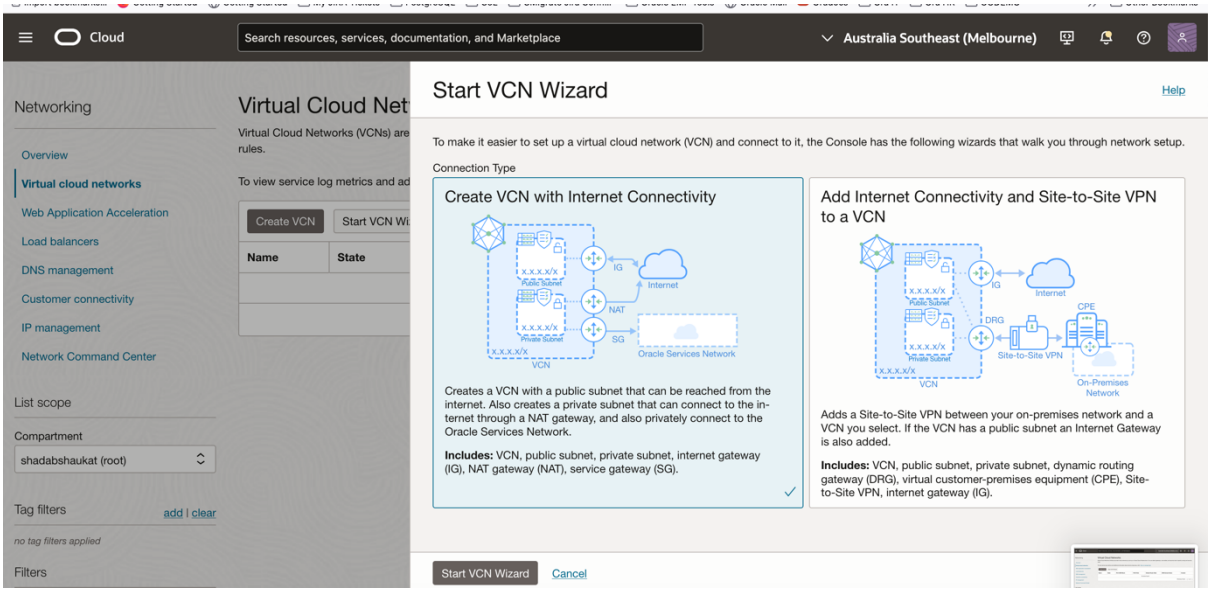


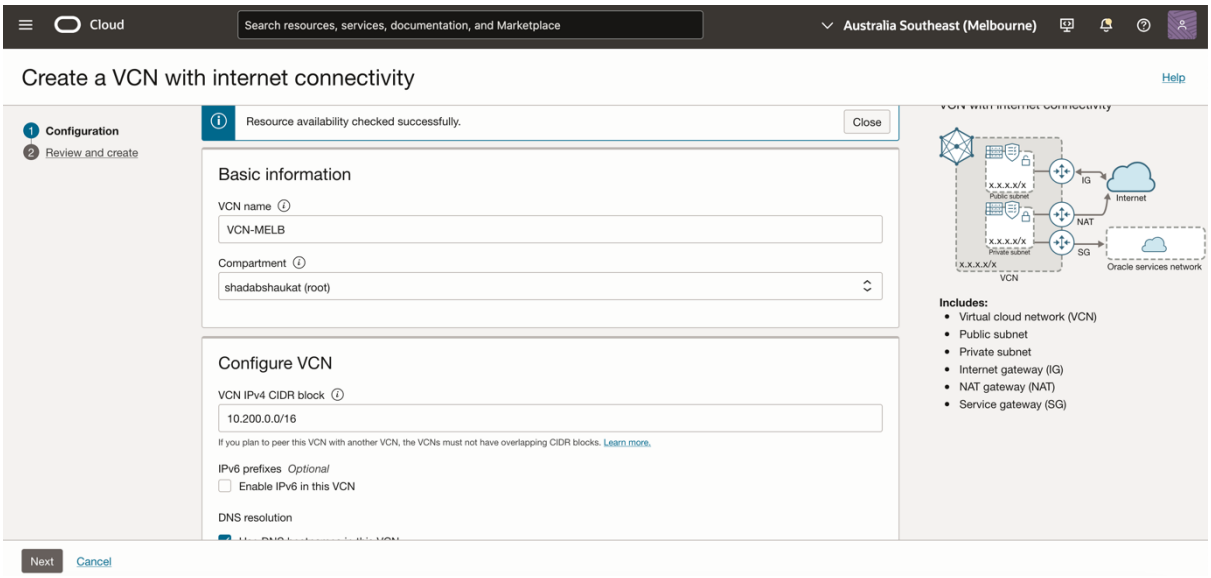


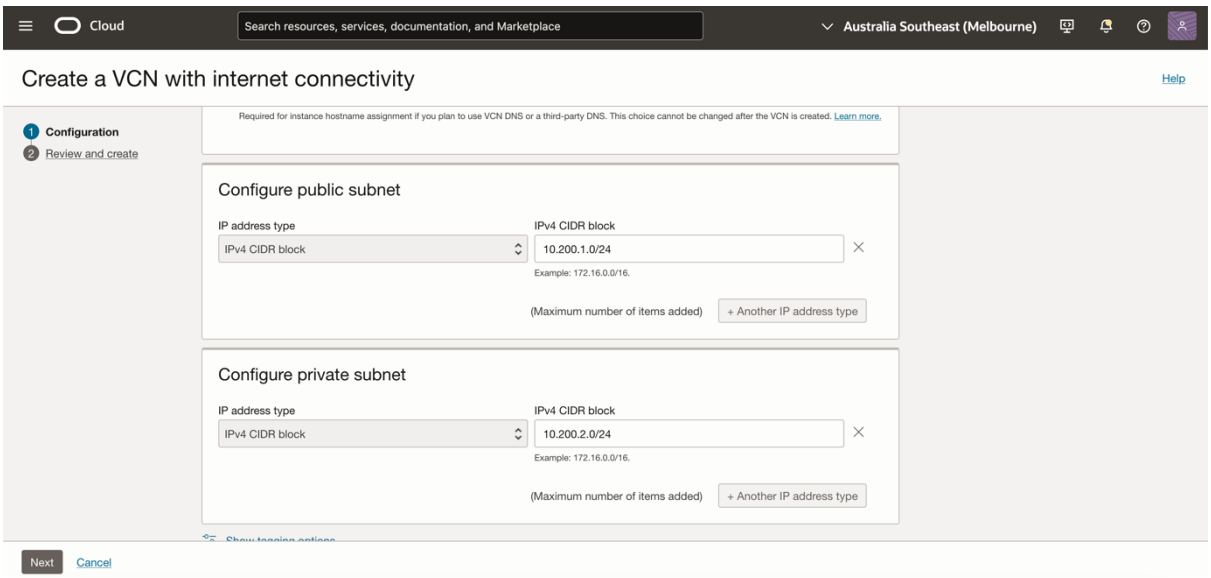
11. Now we will Create the Cross-Region RPC from OCI Region Sydney to Melbourne

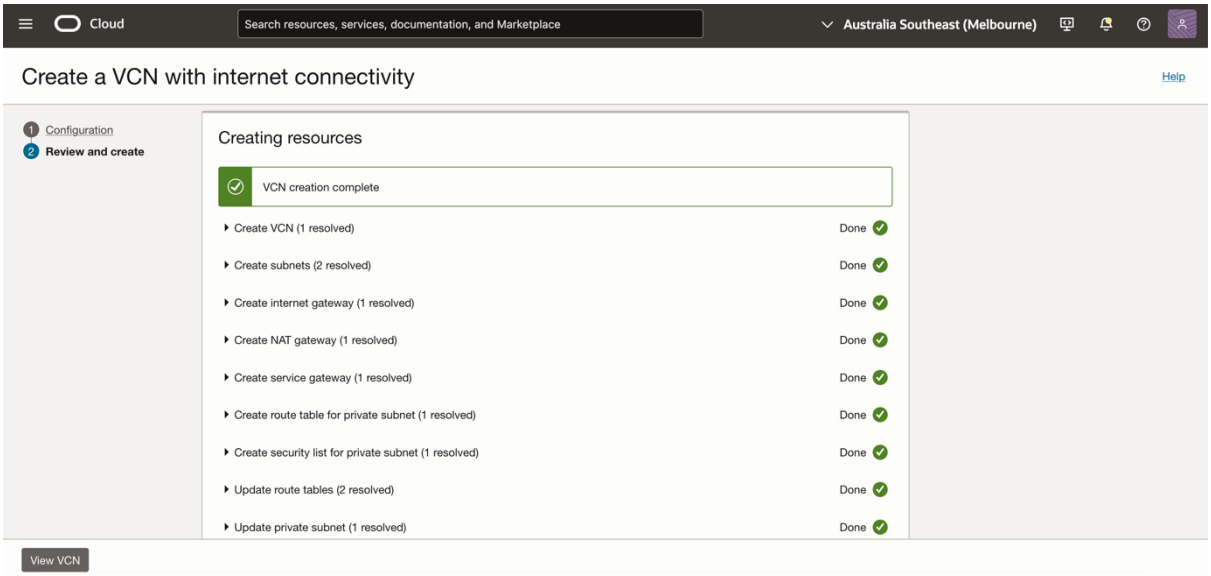
Create OCI VCN in Melbourne with CIDR 10.200.0.0/16



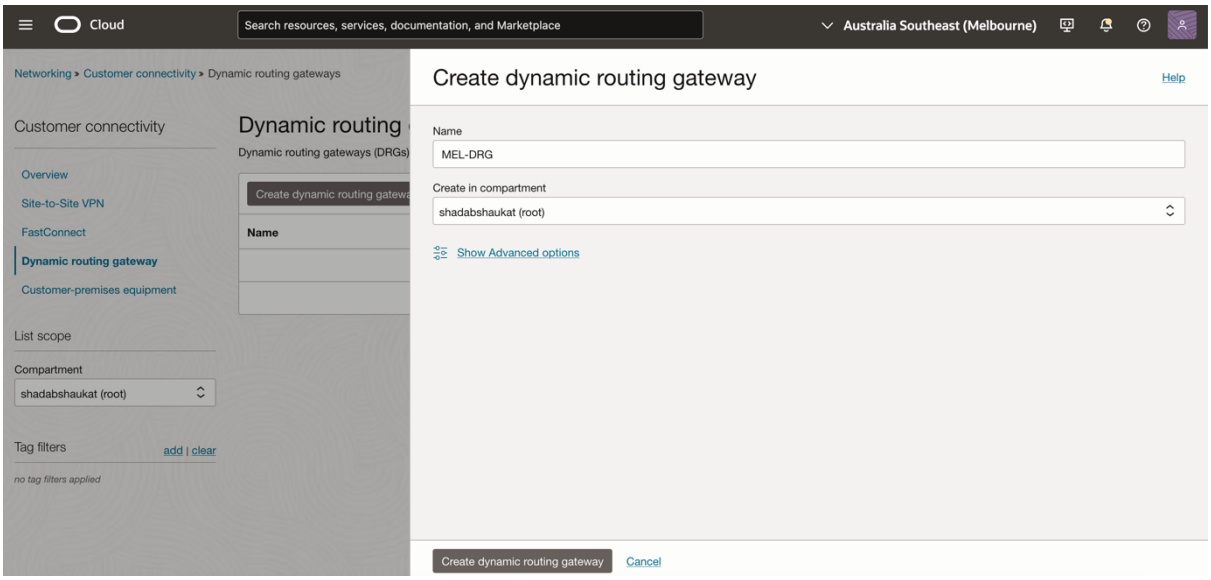

Cloud
Search resources, services, documentation, and Marketplace
Australia Southeast (Melbourne)
Networking
Overview
Virtual cloud networks
Web Application Acceleration
Load balancers
DNS management
Customer connectivity
IP management
Network Command Center
List scope
Compartment
shadabshaukat (root)
Tag filters
add | clear
no tag filters applied
Filters
Virtual Cloud Net
Virtual Cloud Networks (VCNs) are
rules.
To view service log metrics and ad
Create VCN
Start VCN Wi
Name
State
Start VCN Wizard
Help
To make it easier to set up a virtual cloud network (VCN) and connect to it, the Console has the following wizards that walk you through network setup.
Connection Type
Create VCN with Internet Connectivity
Creates a VCN with a public subnet that can be reached from the internet. Also creates a private subnet that can connect to the internet through a NAT gateway, and also privately connect to the Oracle Services Network.
Includes: VCN, public subnet, private subnet, internet gateway (IG), NAT gateway (NAT), service gateway (SG).
Add Internet Connectivity and Site-to-Site VPN to a VCN
Adds a Site-to-Site VPN between your on-premises network and a VCN you select. If the VCN has a public subnet an Internet Gateway is also added.
Includes: VCN, public subnet, private subnet, dynamic routing gateway (DRG), virtual customer-premises equipment (CPE), Site-to-Site VPN, internet gateway (IG).
Start VCN Wizard
Cancel

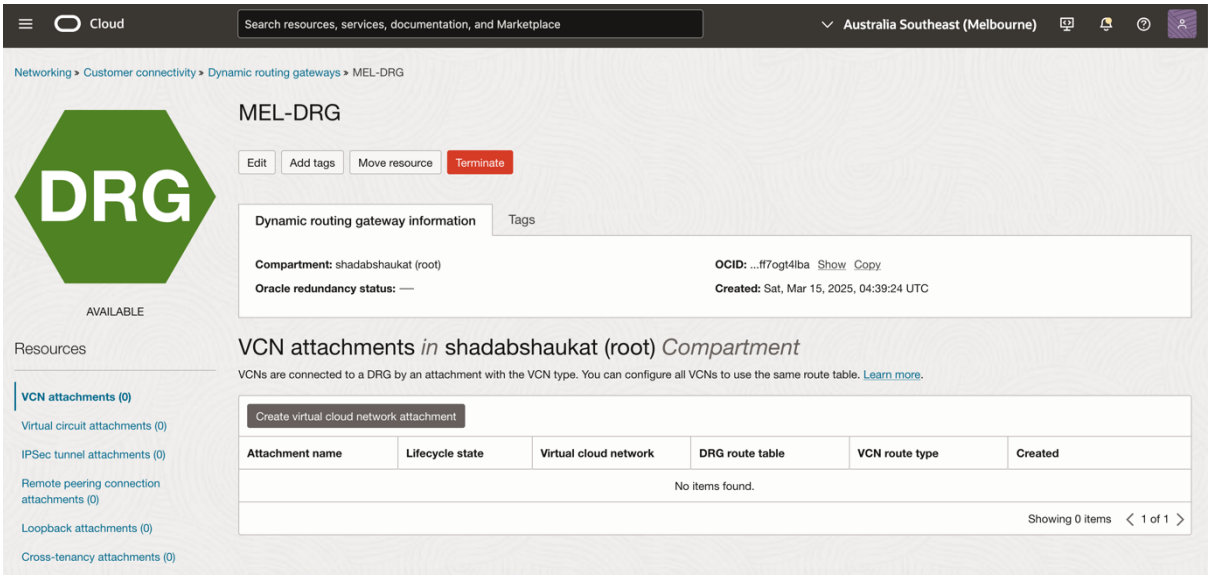

Cloud
Search resources, services, documentation, and Marketplace
Australia Southeast (Melbourne)
Create a VCN with internet connectivity
Help
1 Configuration
2 Review and create
Resource availability checked successfully.
Close
Basic information
VCN name
VCN-MELB
Compartment
shadabshaukat (root)
Configure VCN
VCN IPv4 CIDR block
10.200.0.0/16
If you plan to peer this VCN with another VCN, the VCNs must not have overlapping CIDR blocks. Learn more.
IPv6 prefixes Optional
Enable IPv6 in this VCN
DNS resolution
Includes:
Virtual cloud network (VCN)
Public subnet
Private subnet
Internet gateway (IG)
NAT gateway (NAT)
Service gateway (SG)
Oracle services network
Next
Cancel

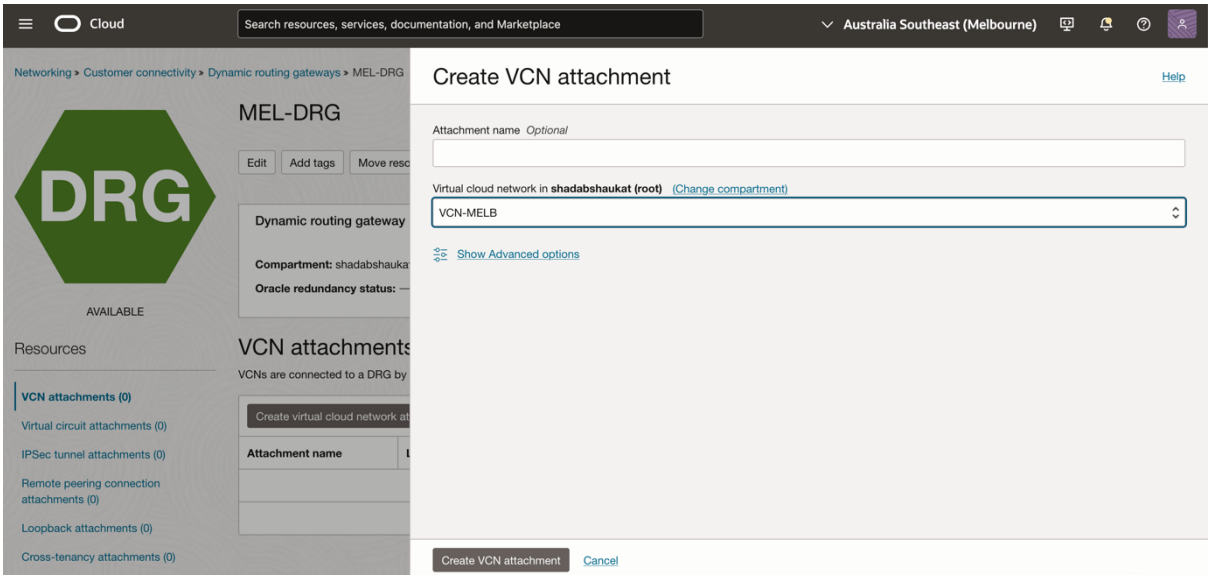

Cloud
Search resources, services, documentation, and Marketplace
Australia Southeast (Melbourne)
Create a VCN with internet connectivity
Help
1 Configuration
2 Review and create
Required for instance hostname assignment if you plan to use VCN DNS or a third-party DNS. This choice cannot be changed after the VCN is created. Learn more.
Configure public subnet
IP address type
IPv4 CIDR block
IPv4 CIDR block
10.200.1.0/24
Example: 172.16.0.0/16.
(Maximum number of items added)
+ Another IP address type
Configure private subnet
IP address type
IPv4 CIDR block
IPv4 CIDR block
10.200.2.0/24
Example: 172.16.0.0/16.
(Maximum number of items added)
+ Another IP address type
Next
Cancel

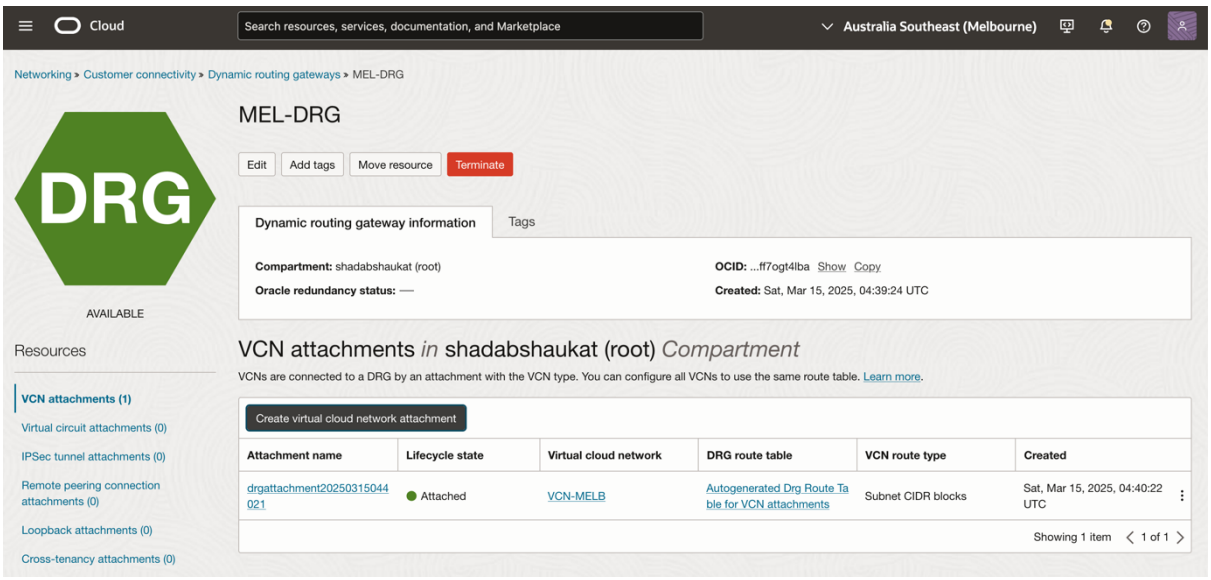


Add DRG in Melbourne region and Attach VCN

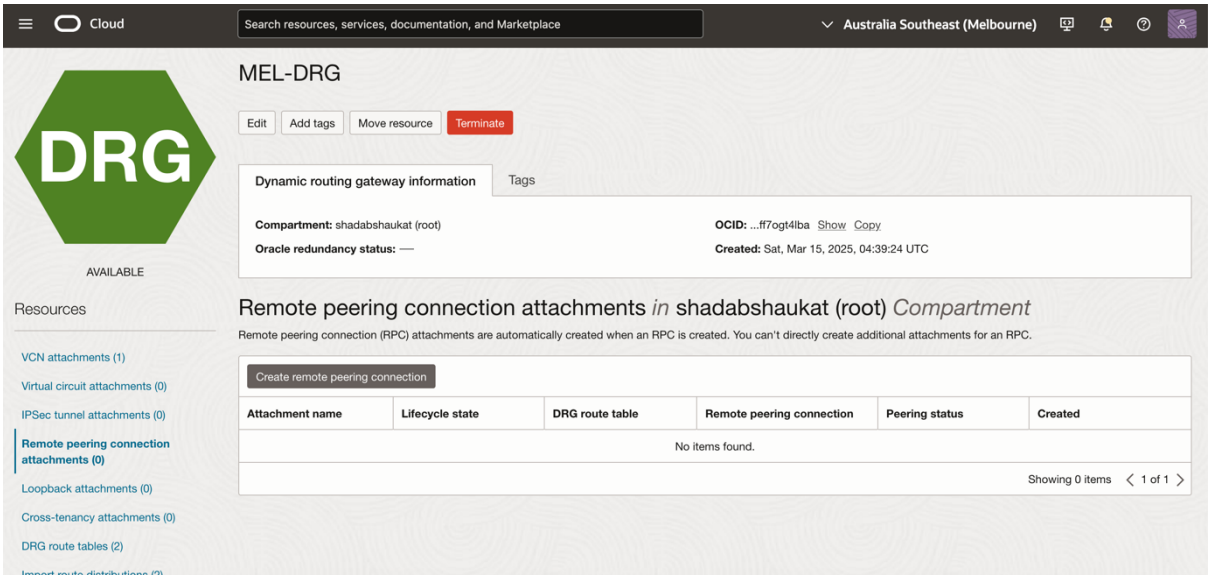


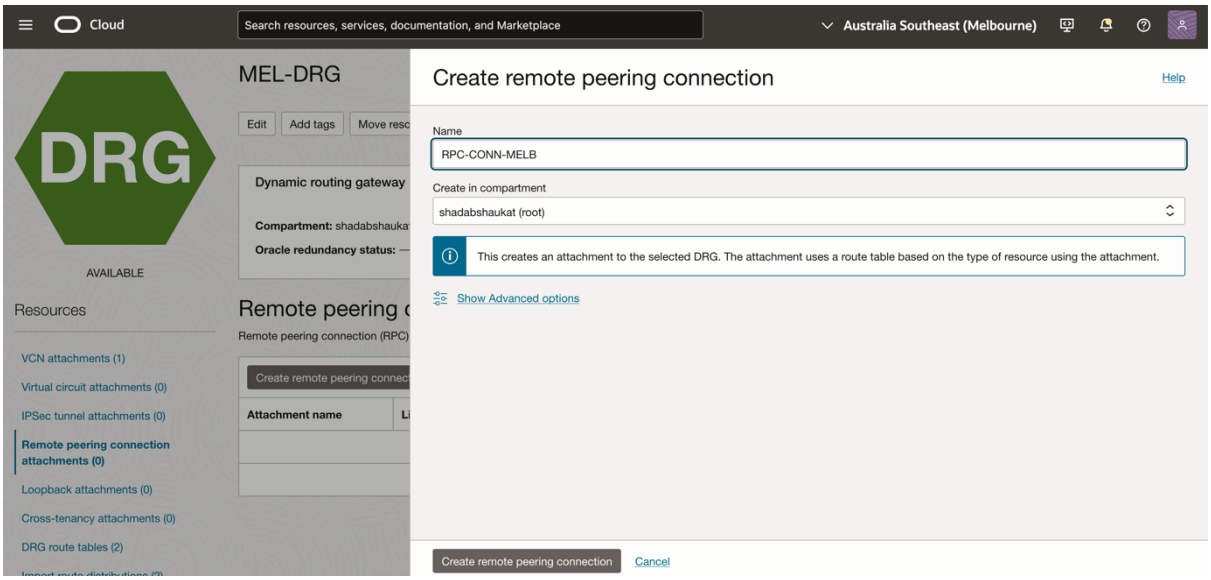


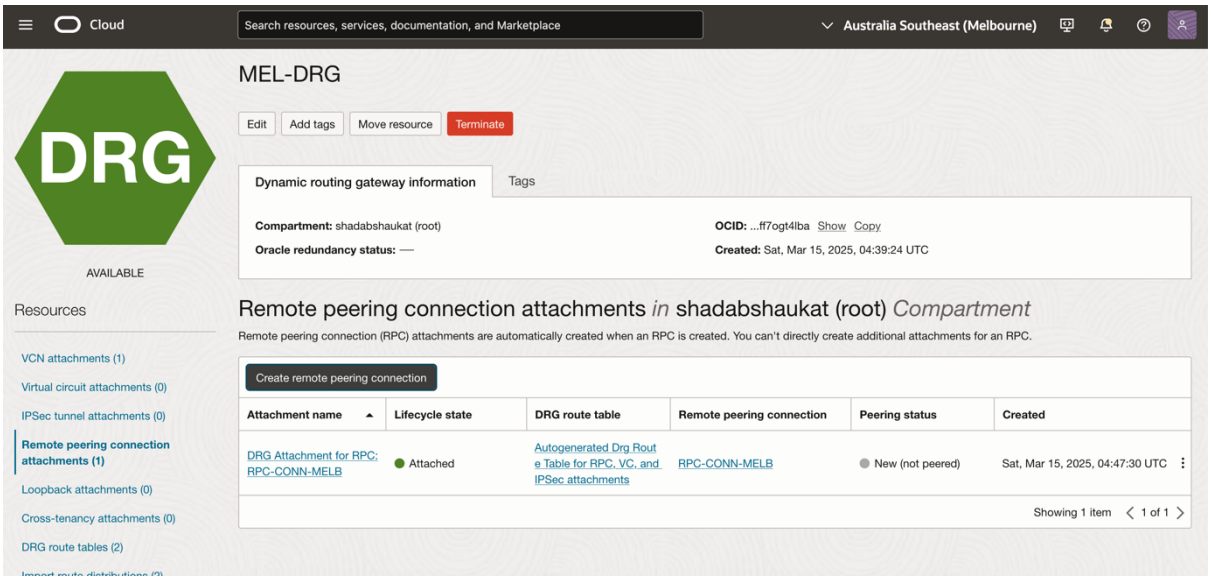




Create Remote Peering Connection (RPC) in DRG in Melbourne

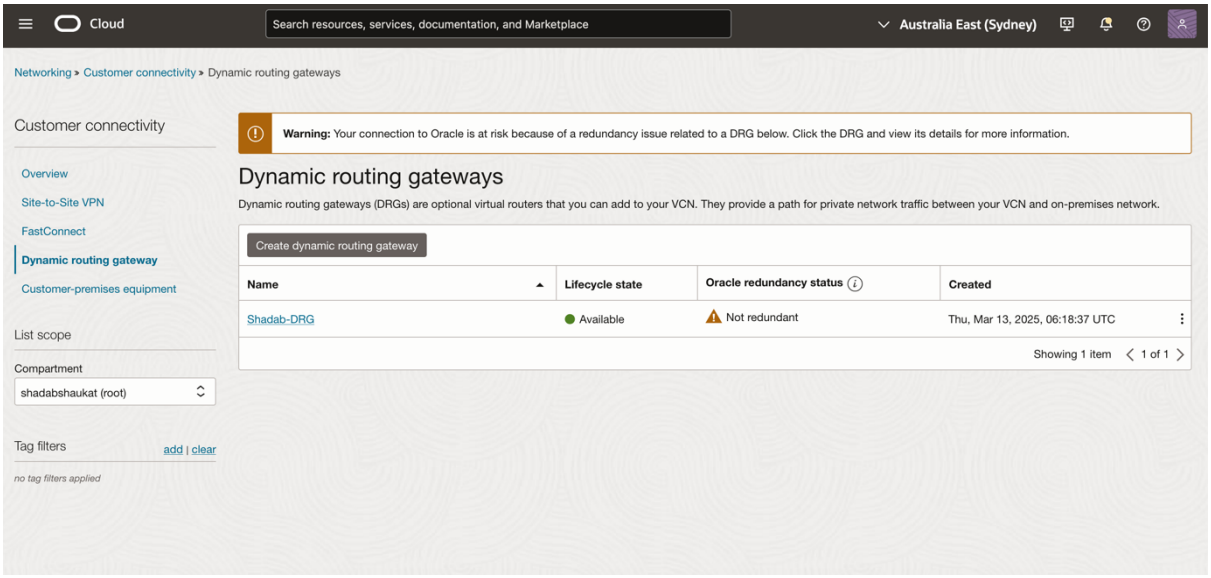

Cloud
Search resources, services, documentation, and Marketplace
Australia Southeast (Melbourne)
MEL-DRG
Edit
Add tags
Move resource
Terminate
Dynamic routing gateway information
Tags
Compartment: shadabshaukat (root)
OCID: ...ff7ogt4lba Show Copy
Oracle redundancy status: —
Created: Sat, Mar 15, 2025, 04:39:24 UTC
AVAILABLE
Resources
VCN attachments (1)
Virtual circuit attachments (0)
IPSec tunnel attachments (0)
Remote peering connection attachments (0)
Loopback attachments (0)
Cross-tenancy attachments (0)
DRG route tables (2)
Remote peering connection attachments in shadabshaukat (root) Compartment
Remote peering connection (RPC) attachments are automatically created when an RPC is created. You can't directly create additional attachments for an RPC.
Create remote peering connection
Attachment name
Lifecycle state
DRG route table
Remote peering connection
Peering status
Created
No items found.
Showing 0 items
1 of 1

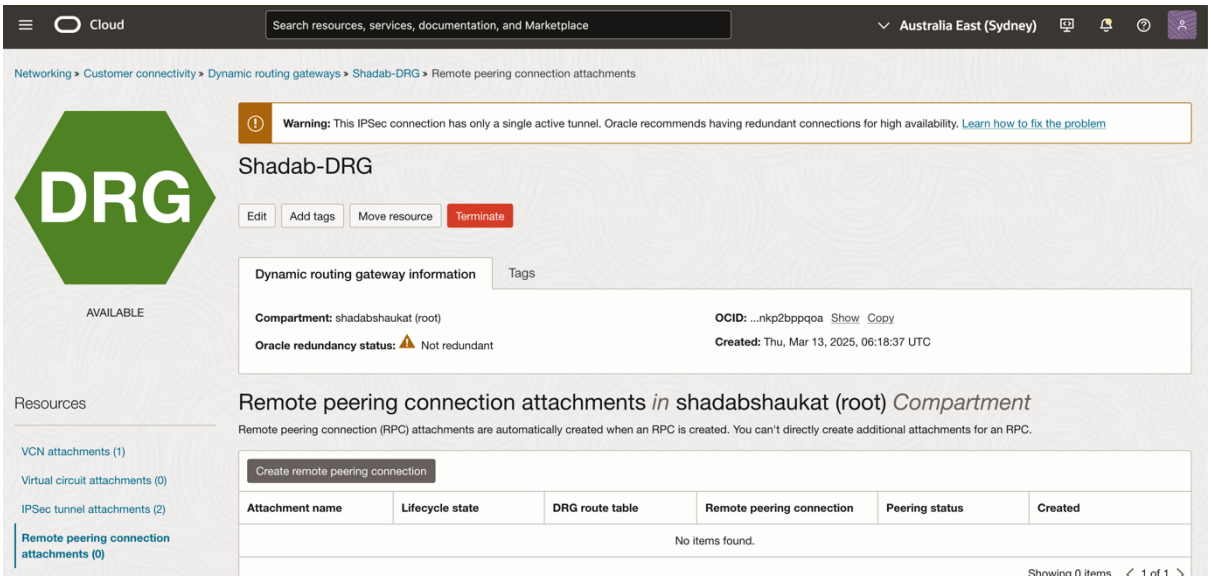

Cloud
Search resources, services, documentation, and Marketplace
Australia Southeast (Melbourne)
MEL-DRG
Create remote peering connection
Help
Name
RPC-CONN-MELB
Create in compartment
shadabshaukat (root)
This creates an attachment to the selected DRG. The attachment uses a route table based on the type of resource using the attachment.
Show Advanced options
Create remote peering connection
Cancel
AVAILABLE
Resources

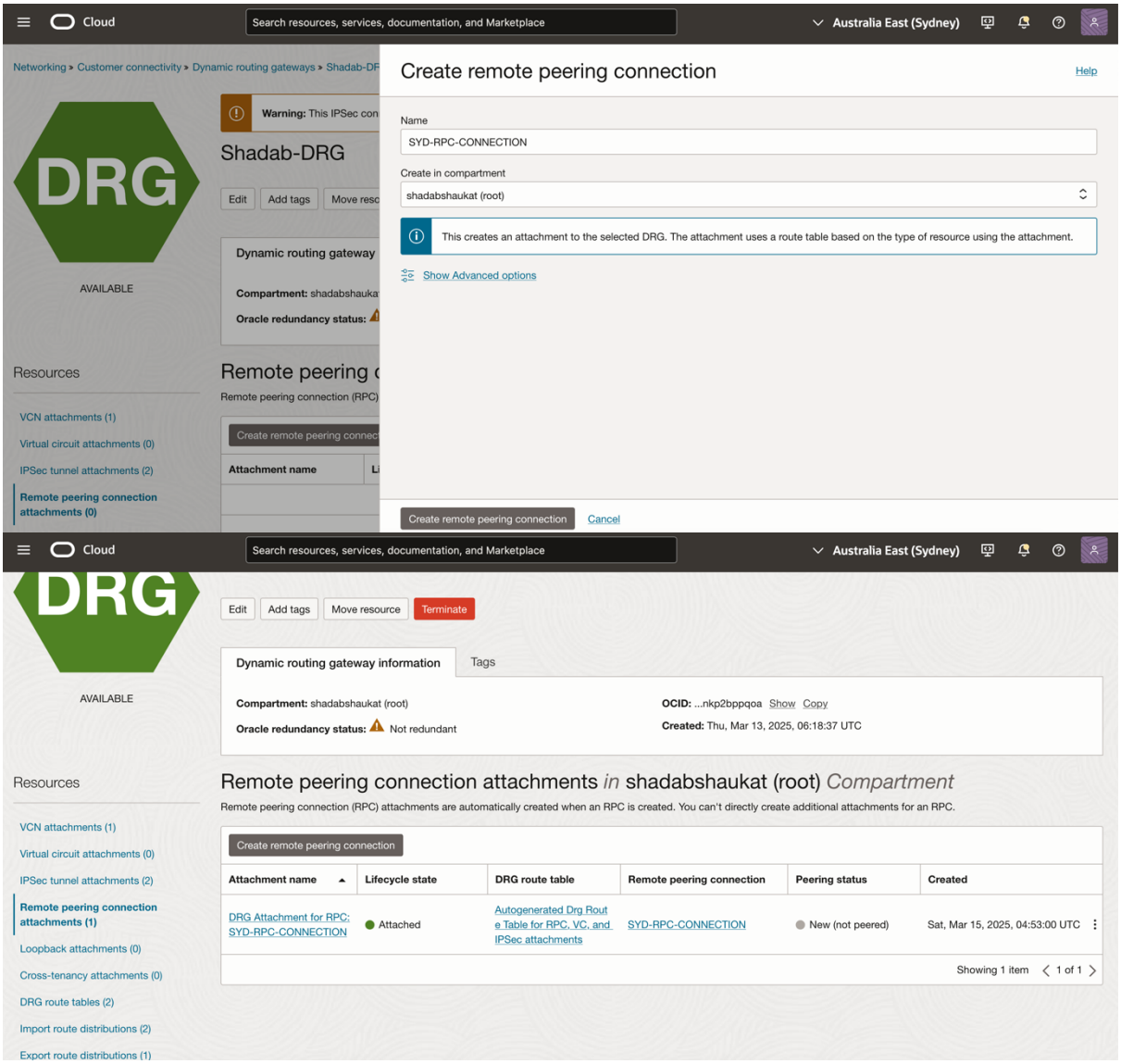

Cloud
Search resources, services, documentation, and Marketplace
Australia Southeast (Melbourne)
MEL-DRG
Edit
Add tags
Move resource
Terminate
Dynamic routing gateway information
Tags
Compartment: shadabshaukat (root)
OCID: ...ff7ogt4lba Show Copy
Oracle redundancy status: —
Created: Sat, Mar 15, 2025, 04:39:24 UTC
AVAILABLE
Resources
VCN attachments (1)
Virtual circuit attachments (0)
IPSec tunnel attachments (0)
Remote peering connection attachments (1)
Loopback attachments (0)
Cross-tenancy attachments (0)
DRG route tables (2)
Remote peering connection attachments in shadabshaukat (root) Compartment
Remote peering connection (RPC) attachments are automatically created when an RPC is created. You can't directly create additional attachments for an RPC.
Create remote peering connection
Attachment name
Lifecycle state
DRG route table
Remote peering connection
Peering status
Created
DRG Attachment for RPC: RPC-CONN-MELB
Attached
Autogenerated Drg Route Table for RPC, VC, and IPSec attachments
RPC-CONN-MELB
New (not peered)
Sat, Mar 15, 2025, 04:47:30 UTC
Showing 1 item
1 of 1

Create Remote Peering Connection (RPC) in DRG in Sydney

Go to Sydney DRG

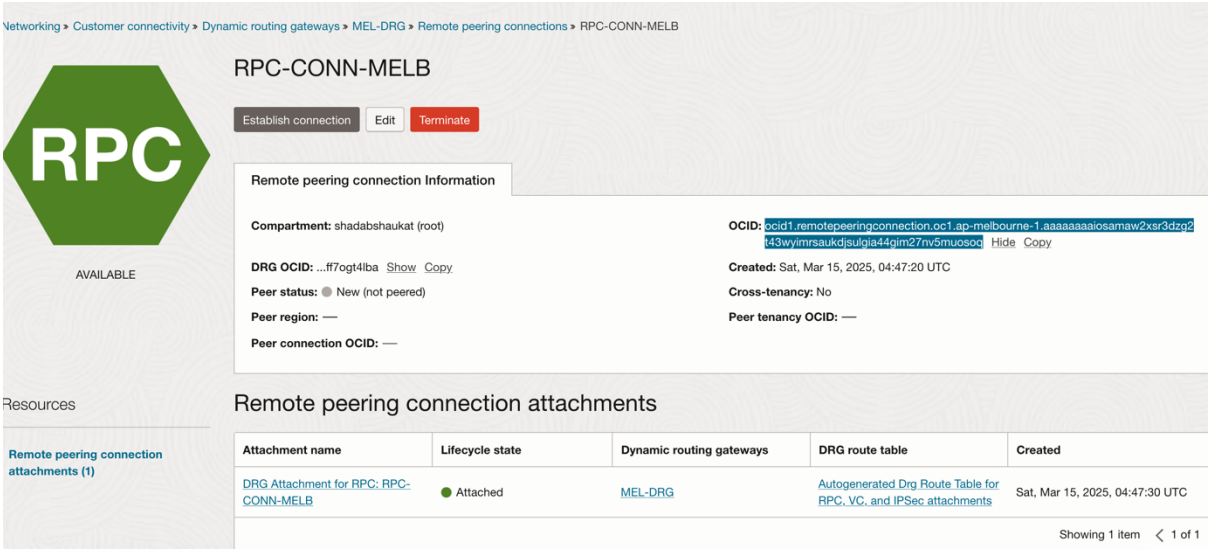






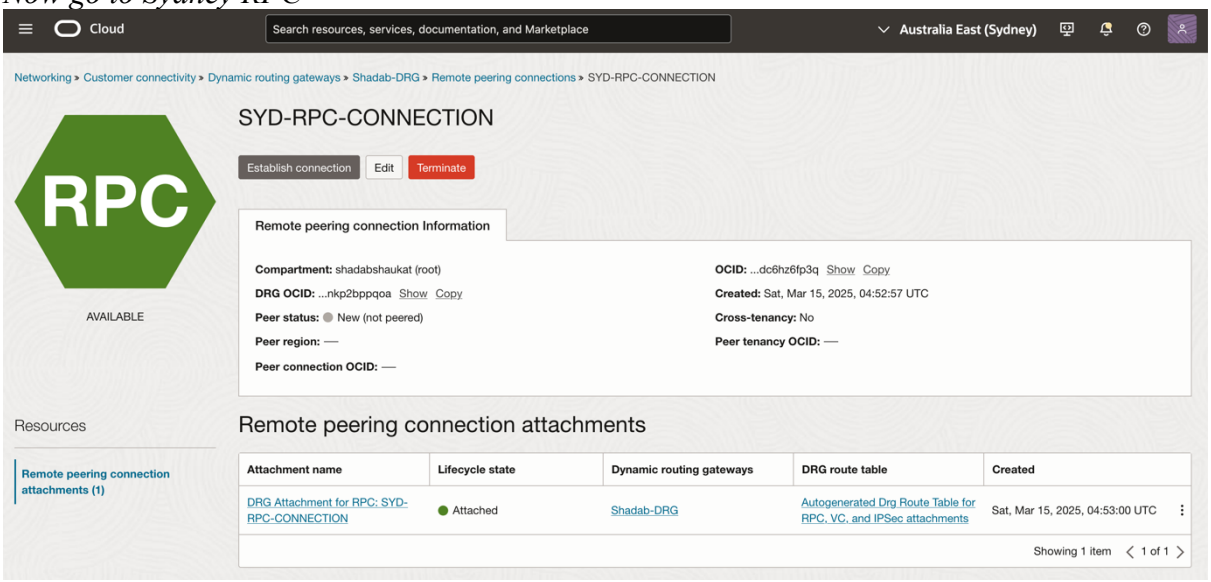
Now click on the Sydney remote peering connection and establish connection with Melbourne RPC

First get OCID of the Melbourne RPC connection peer

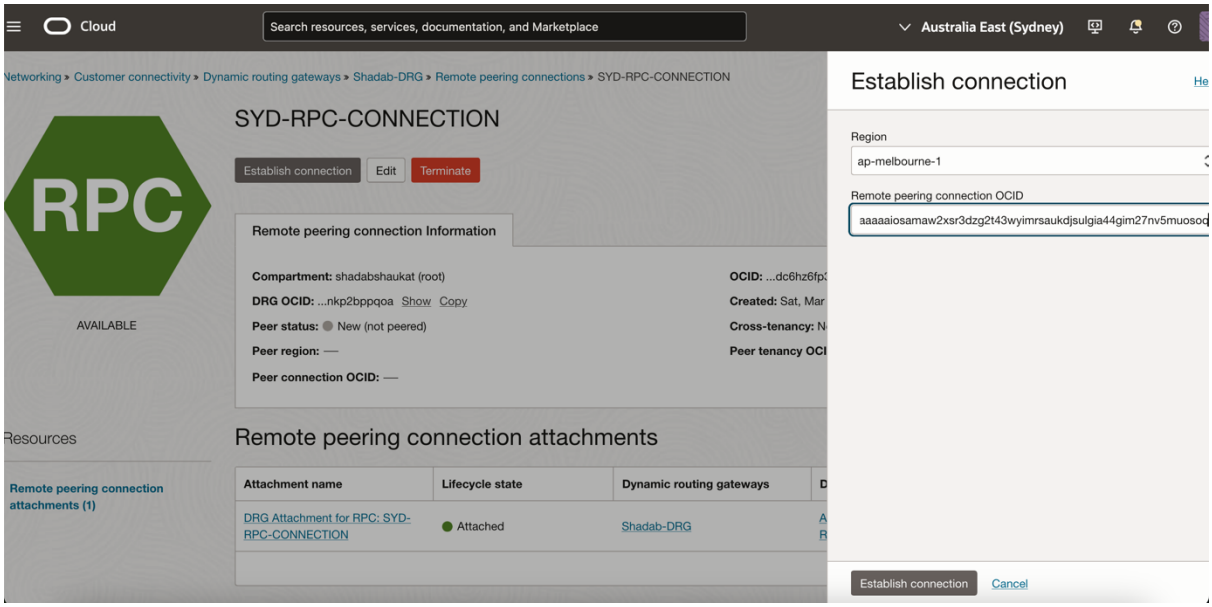


ocid1.remotepeeringconnection.oc1.ap-melbourne-1.aaaaaaaaiosamaw2xsr3dzg2t43wyimrsaukdjsulgia44gim27nv5muosoq

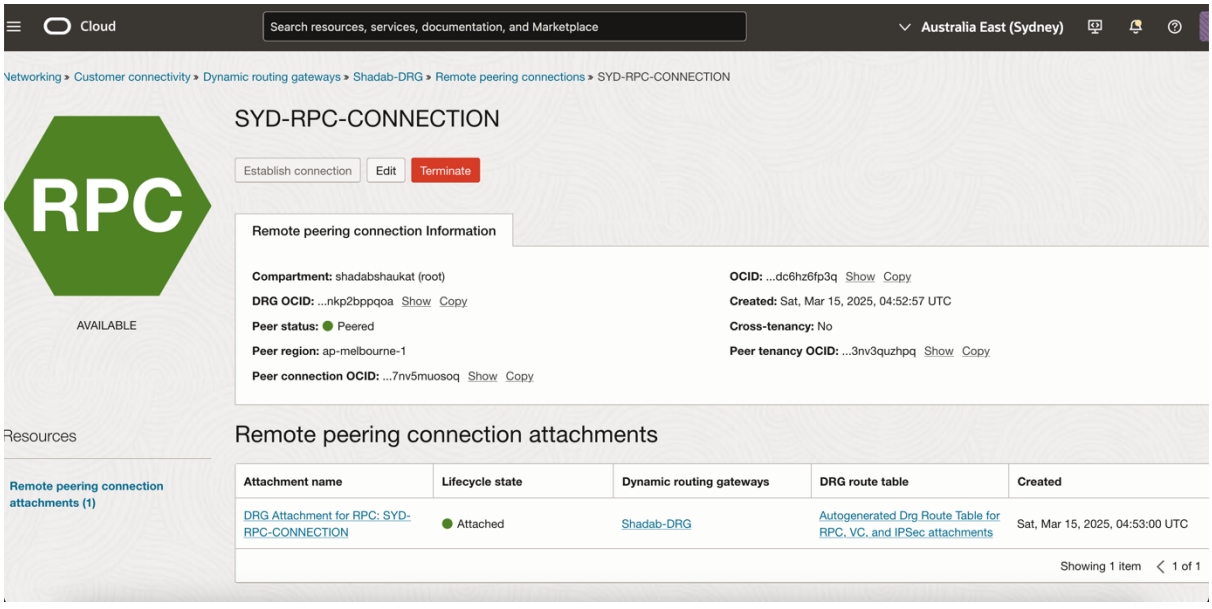
Now go to Sydney RPC



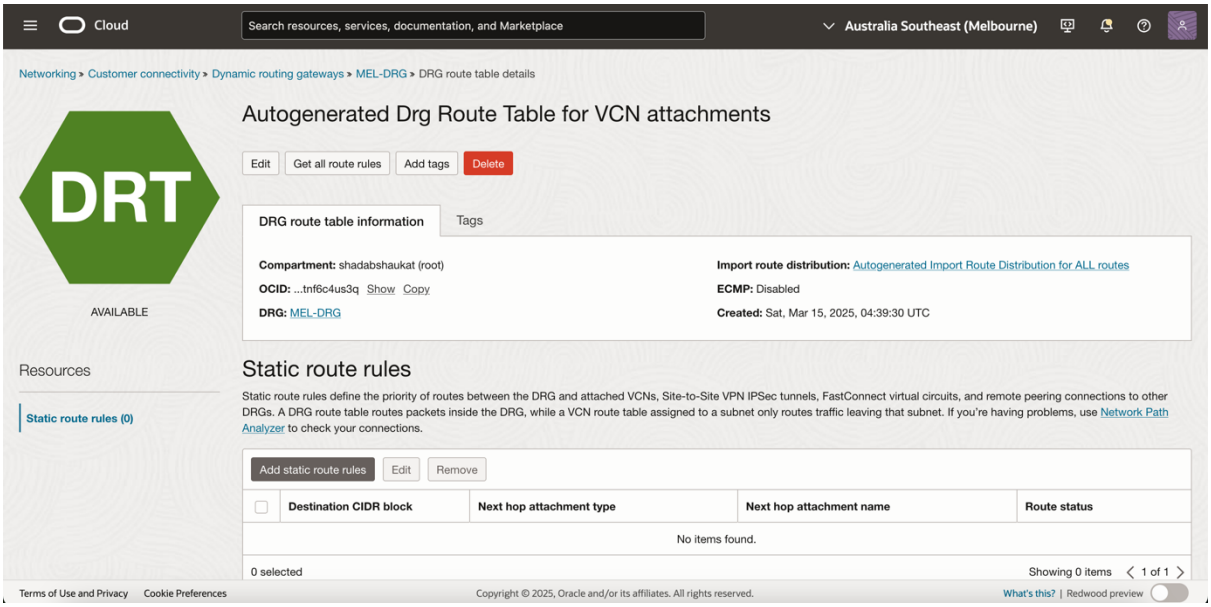
Click on Establish Connection and Add the Peer OCID :

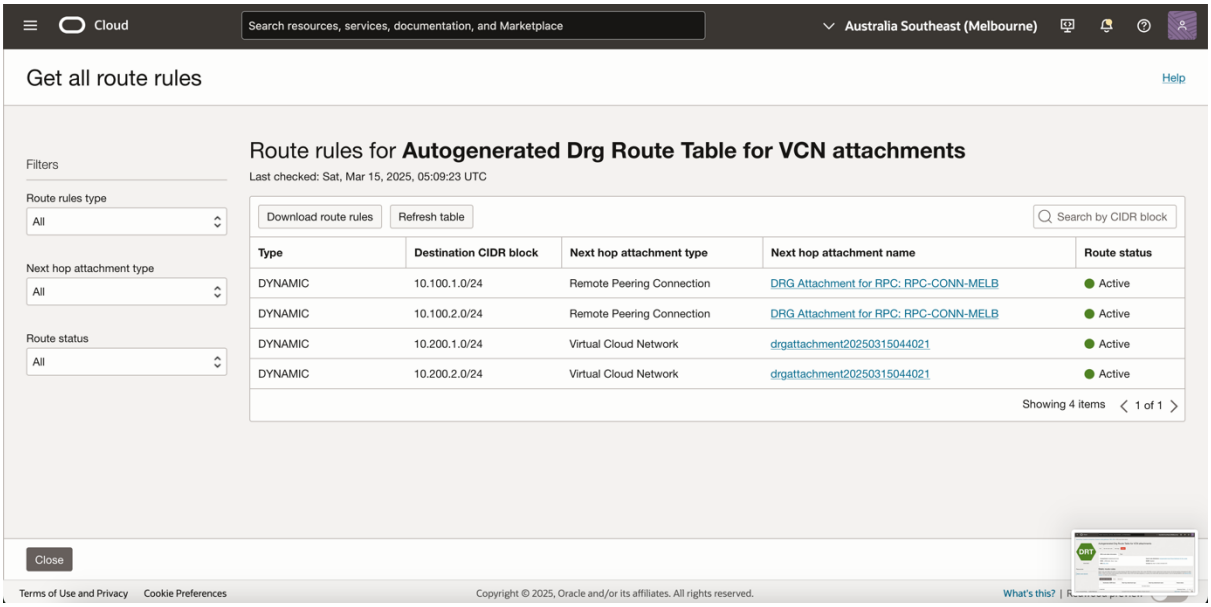


If everything was setup properly the peer status will be green and 'Peered'



Your Routes for both Sydney and Melbourne VCN should be automatically published. You can go to VCN attachment in both regions DRG and check VCN Attachment's DRG Route Table

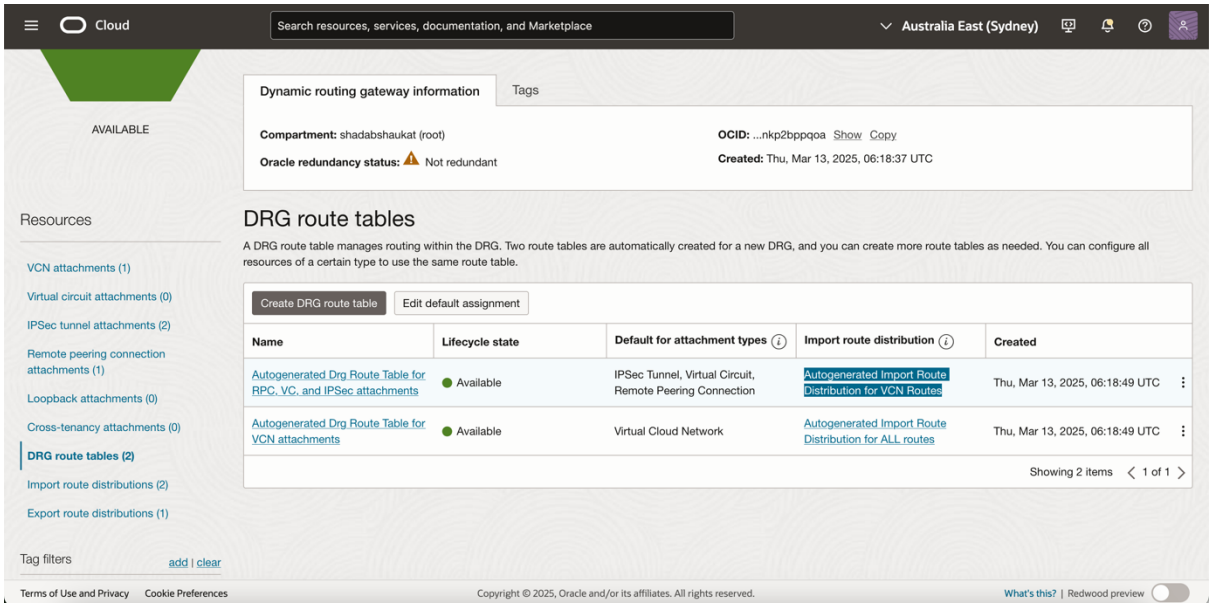




Configure DRG in OCI Sydney for Transit connectivity

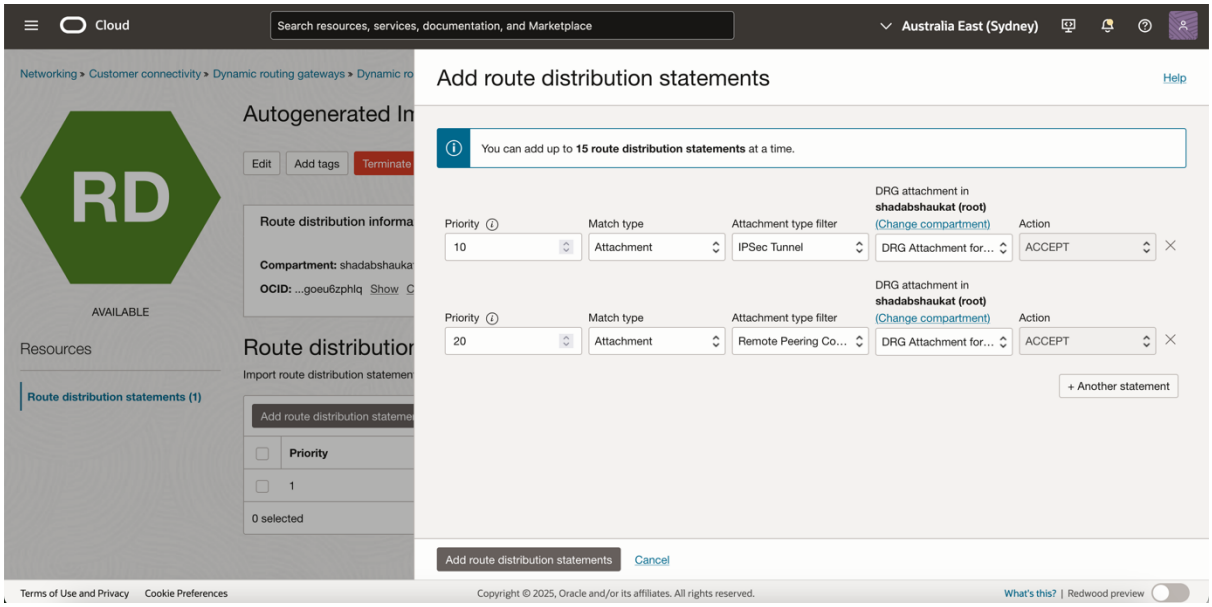
Setup the DRG in Sydney for transit routing.

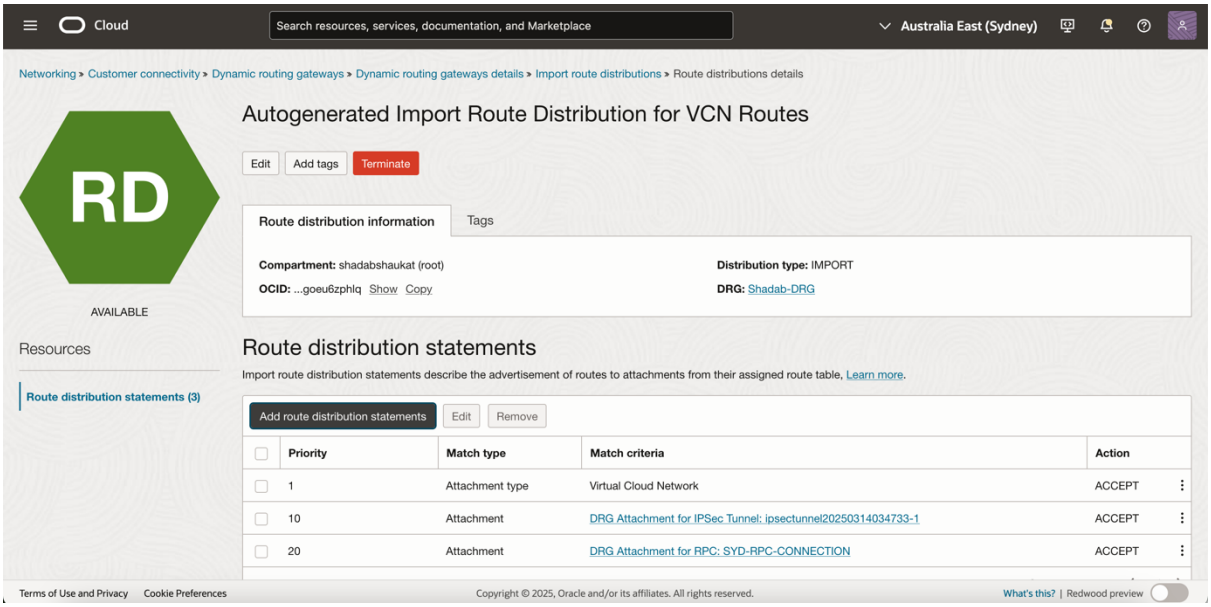
Since by default IPSec VPN and RPC are using the same DRG route, we will will just modify the route. In real world production setup, you should have a separate route.



Go to DRG in Sydney > DRG Route Table > 'Import Route Distribution' for 'RPC, VC and IPSec attachments'

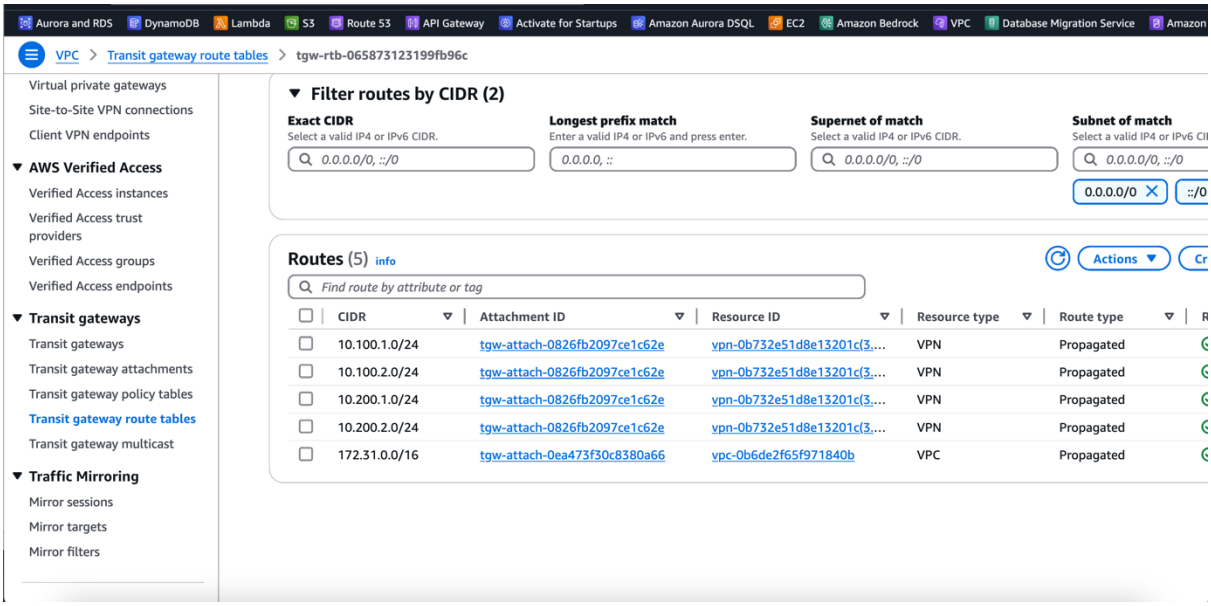
Add 2 New Route Distribution Statements



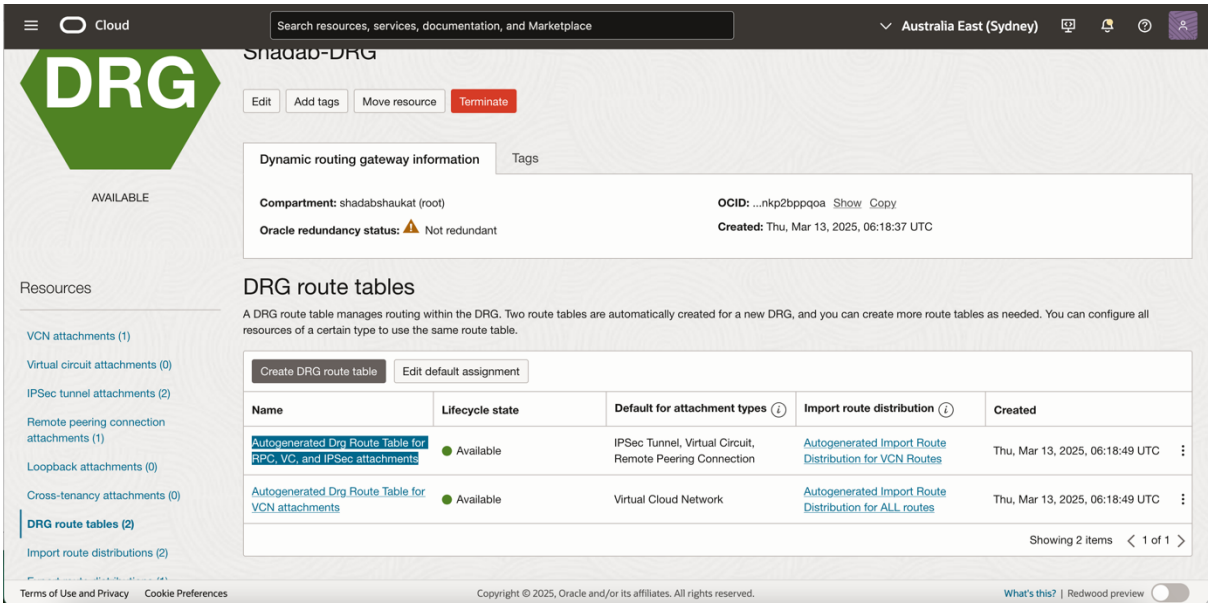


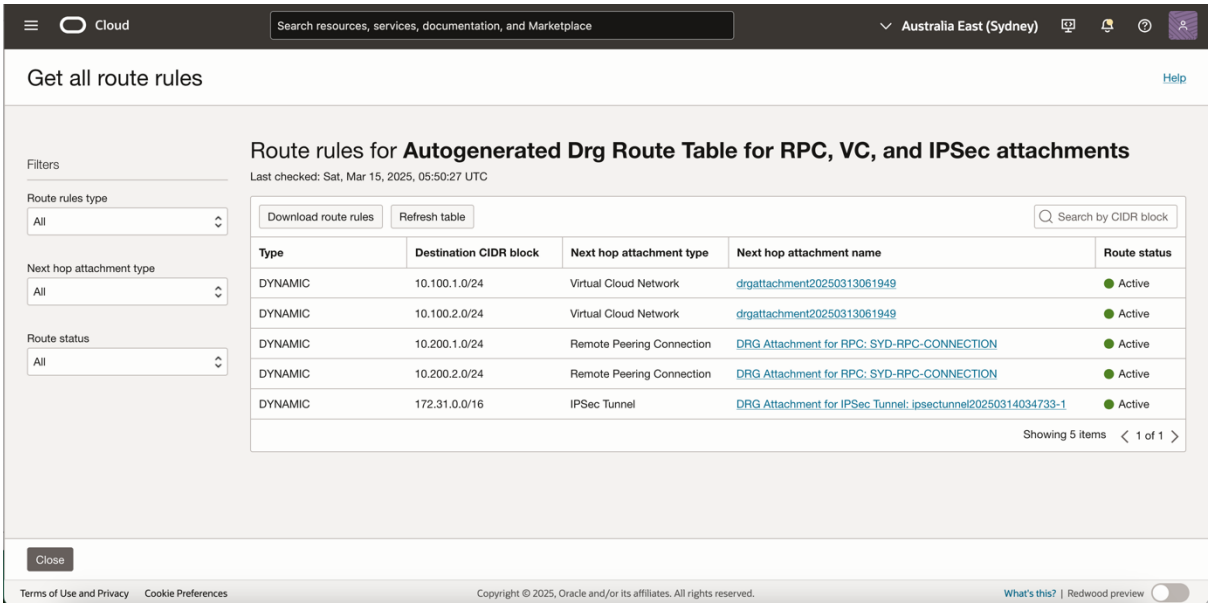
If you setup everything correctly in your Transit Gateway Route Tables you will see

All routes of your 2 OCI Regions plus the AWS VPC route

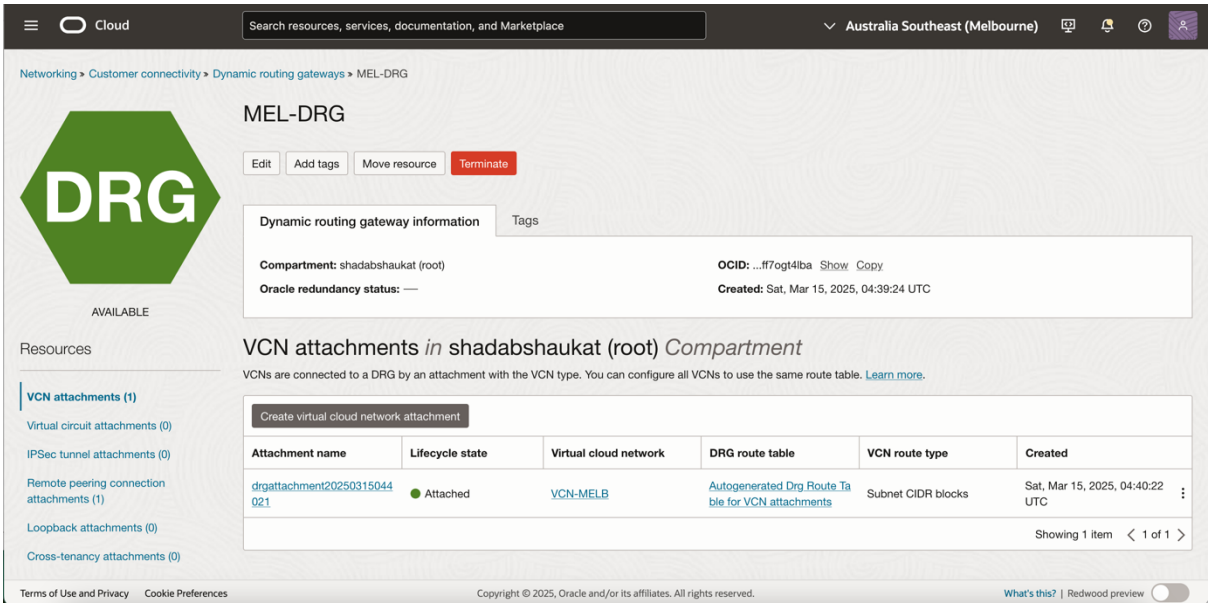


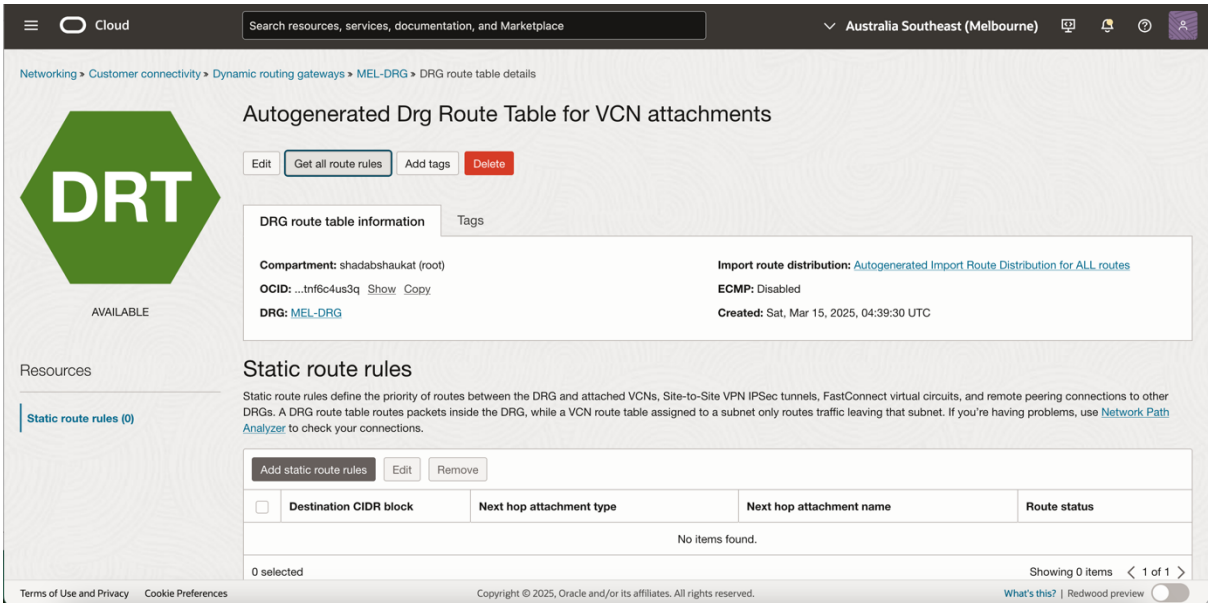
You can verify the same from Sydney DRG

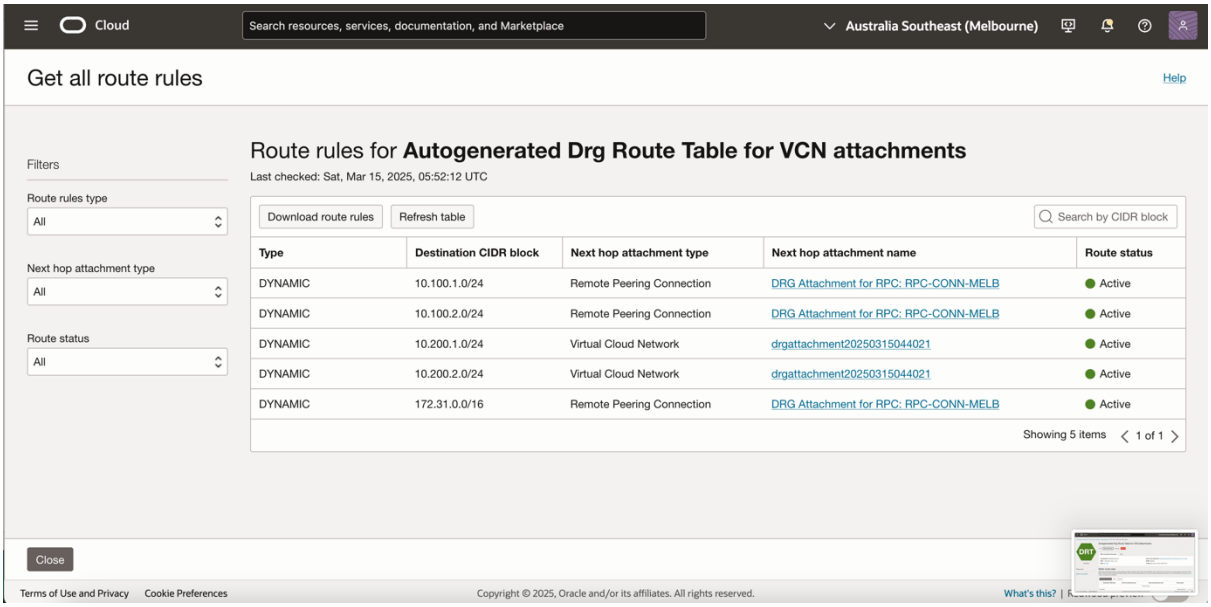




Do the same from Melbourne DRG

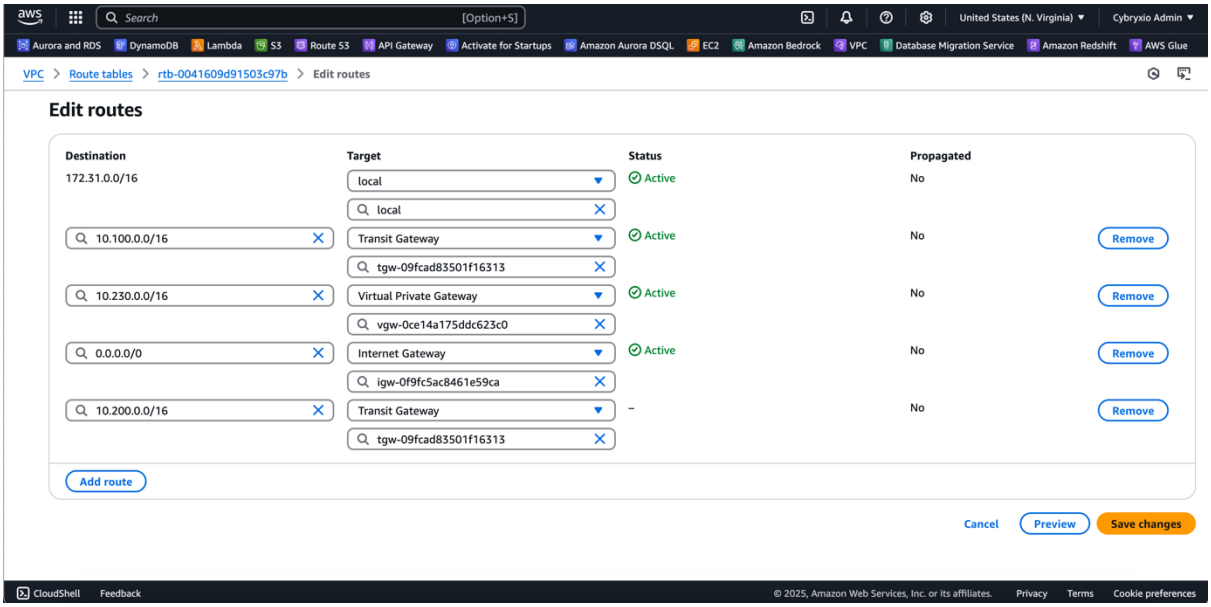


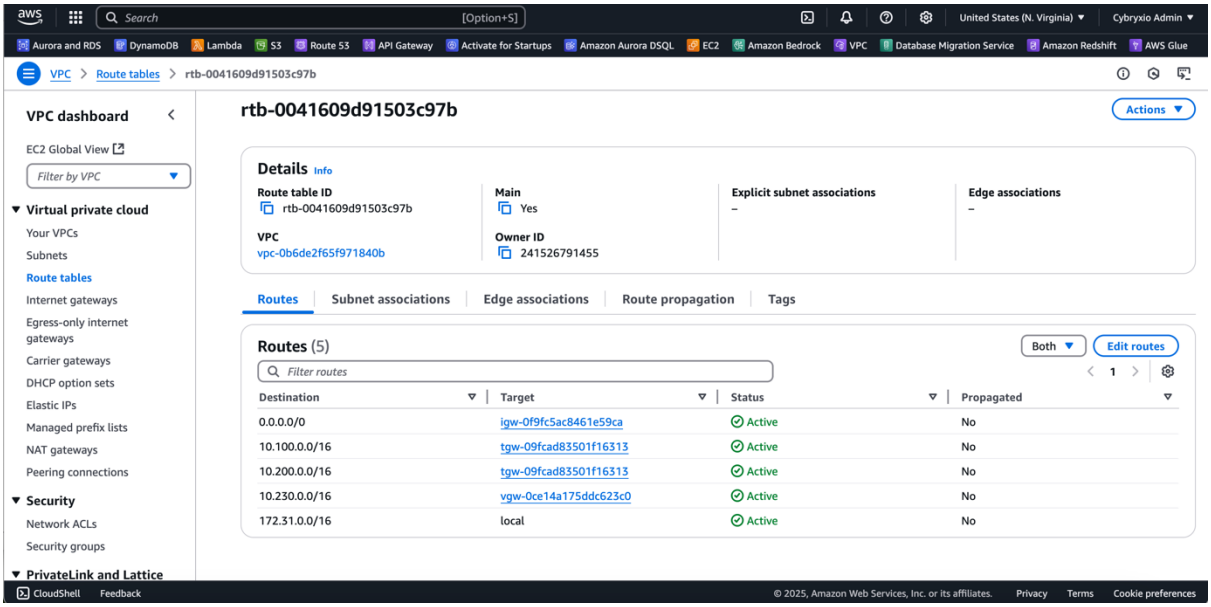




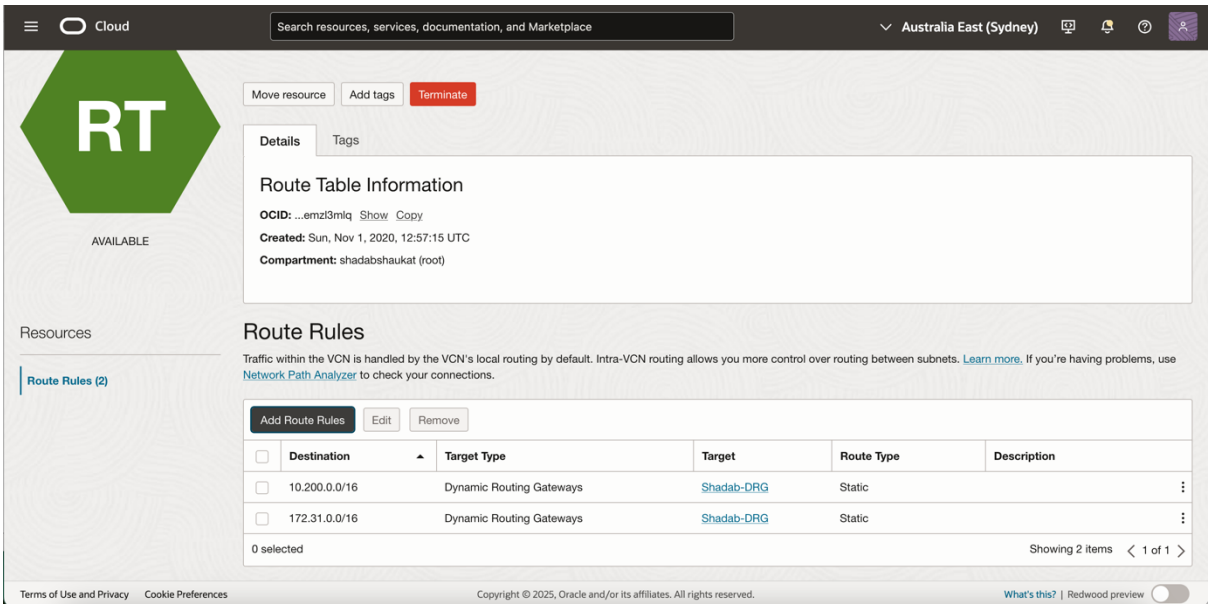
Ref : https://ystatit.medium.com/azure-ipsec-vpn-to-oci-plus-oci-remote-peering-0b09e62e50a2

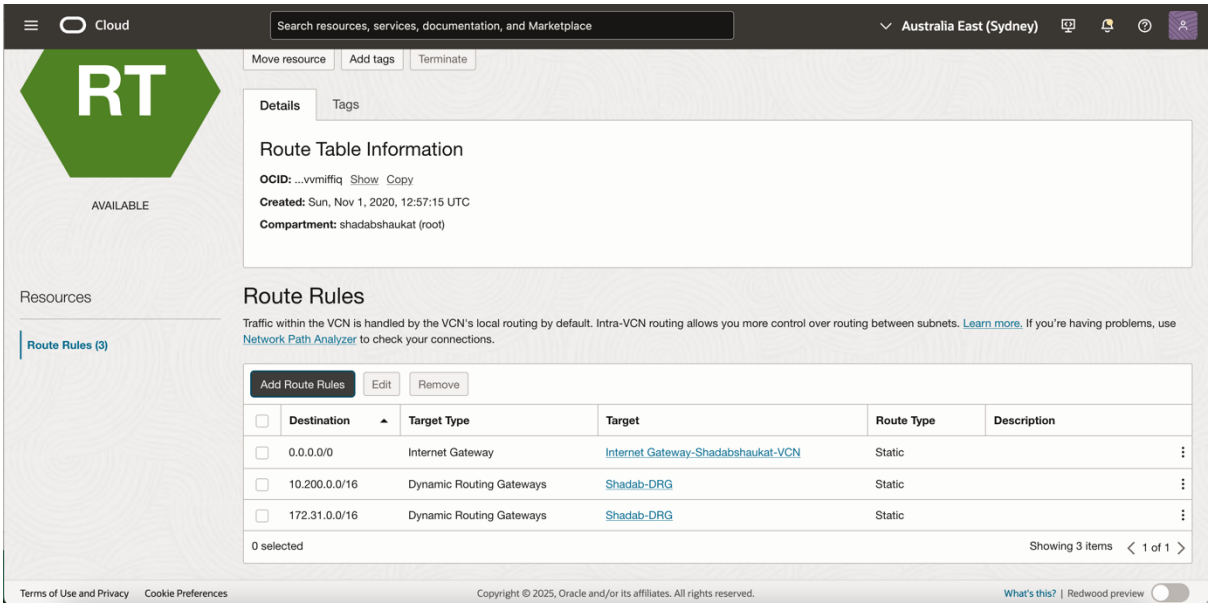
Ensure AWS VPC Route Table has your routes for Sydney and Melbourne VPC added



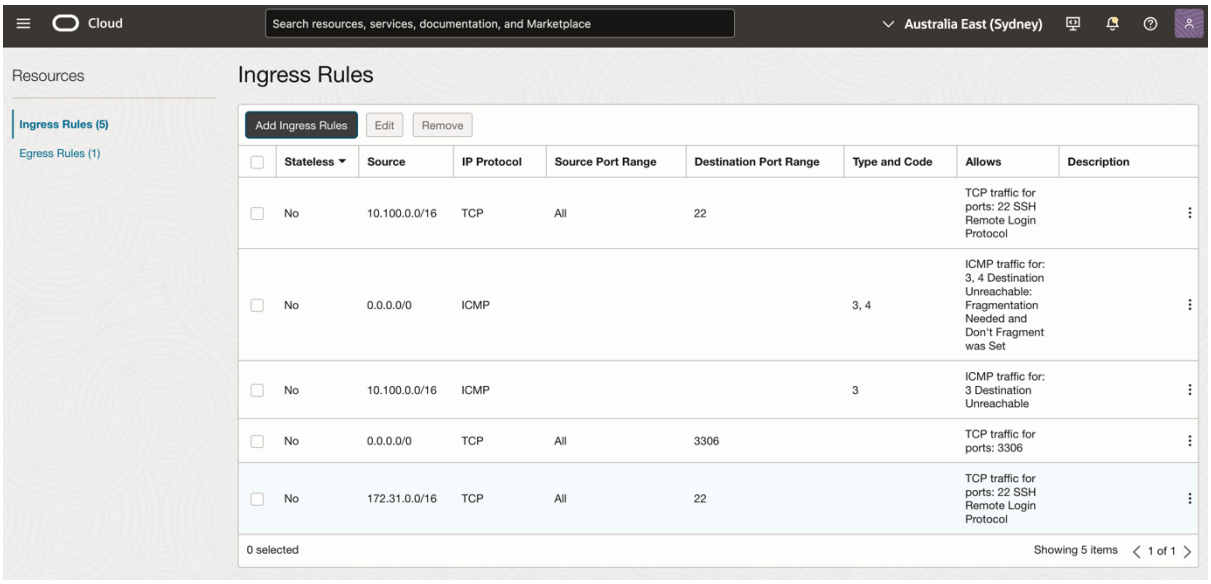


Ensure your Sydney VCN Route tables have routes for Melbourne VCN and AWS VPC via DRG

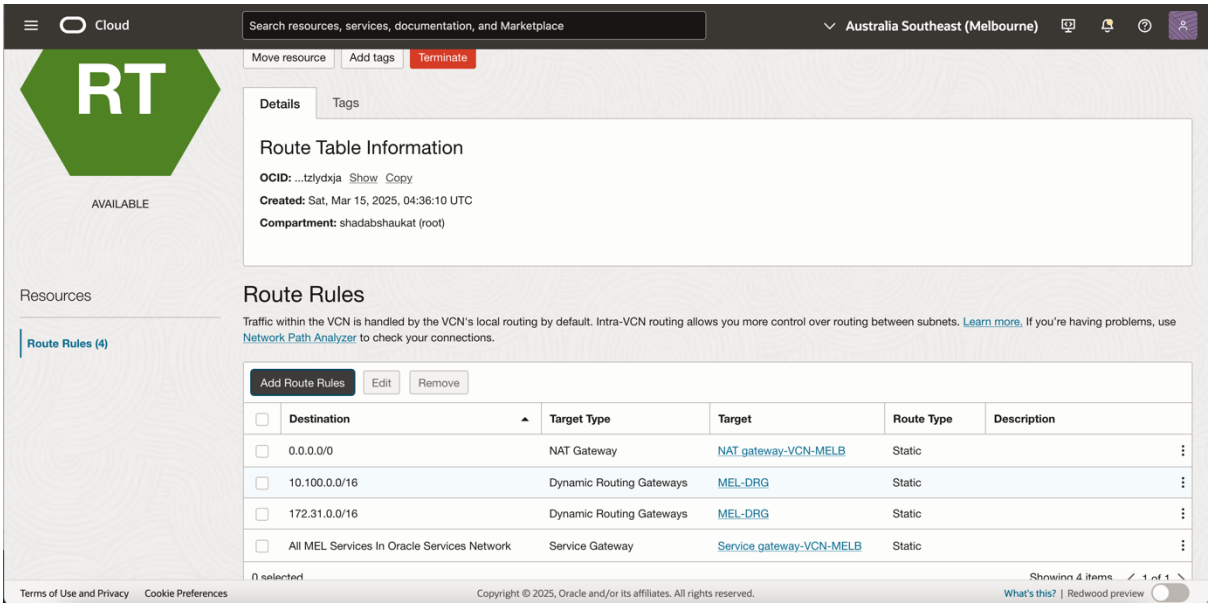


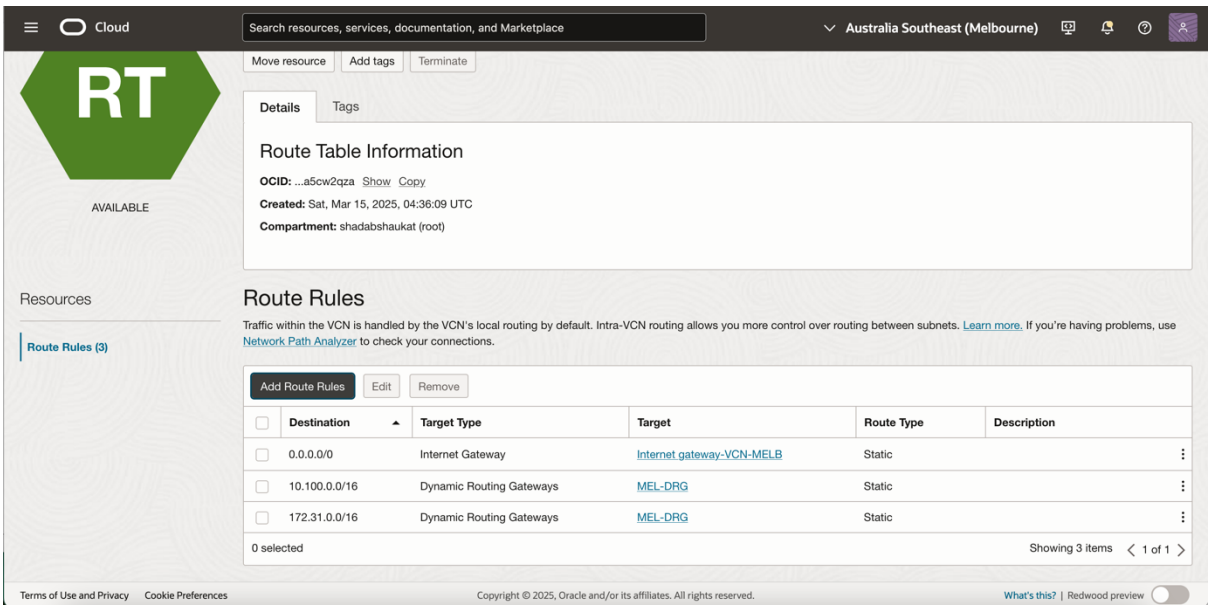


Add ingress rule for ports in Route Table security list in SYDNEY for AWS VPC CIDR 172.31.0.0/16

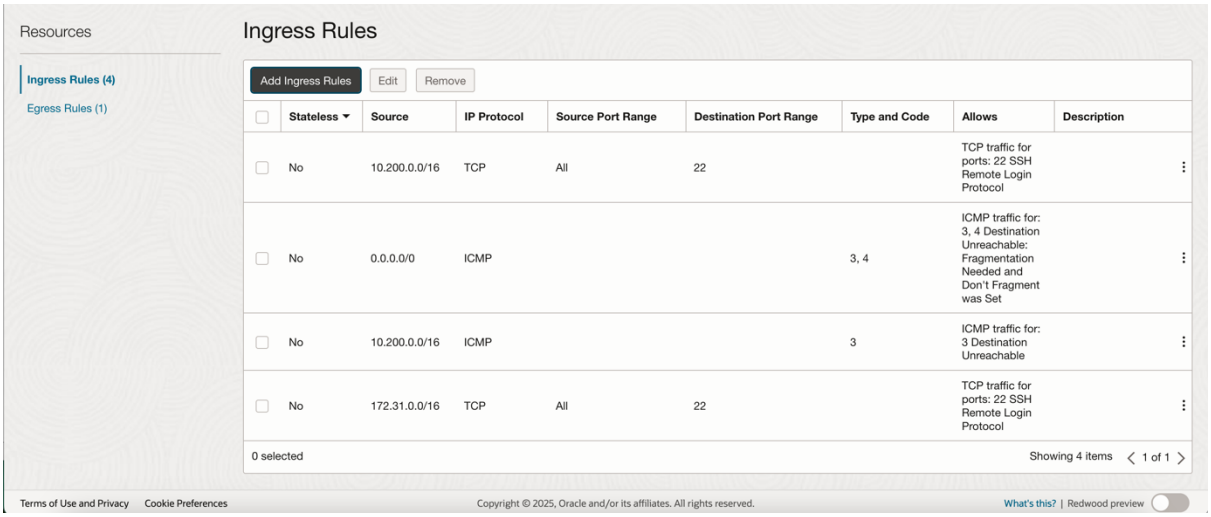


And do the same in Melbourne VCN Route Tables





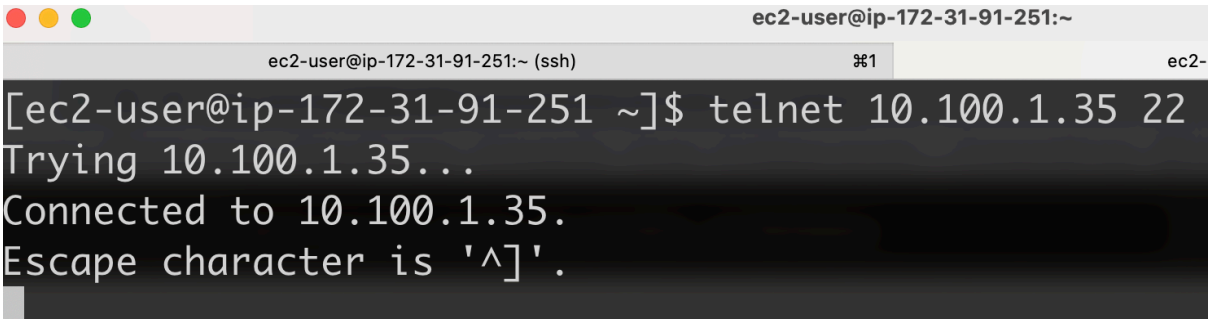
Add ingress rule for ports in Route Table security list in Melbourne for AWS VPC CIDR 172.31.0.0/16



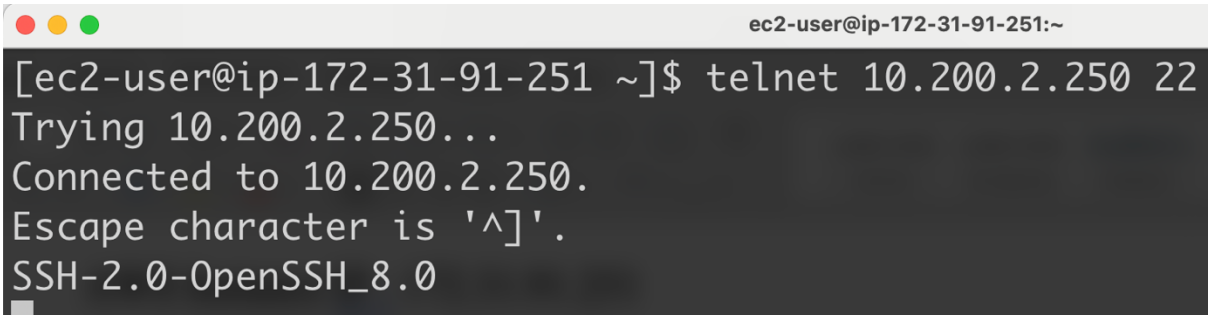
Create an EC2 instance in the AWS VPC and connect to 2 VMs in OCI one in Sydney and Melbourne respectively

AWS Instance IP : 172.31.91.251
OCI Sydney VM : 10.100.1.35
OCI Melbourne VM : 10.200.2.250

AWS → OCI Sydney SSH Connection

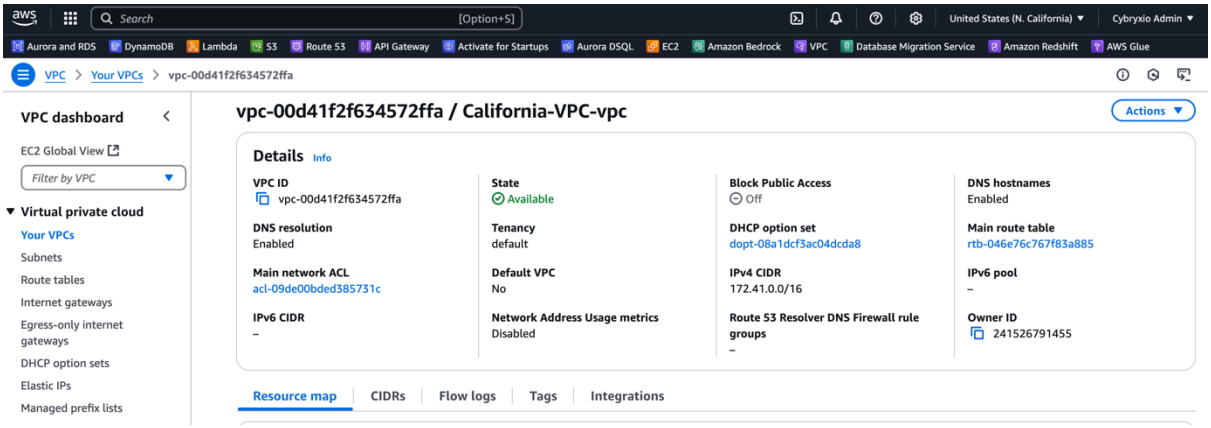


AWS → OCI Melbourne SSH Connection

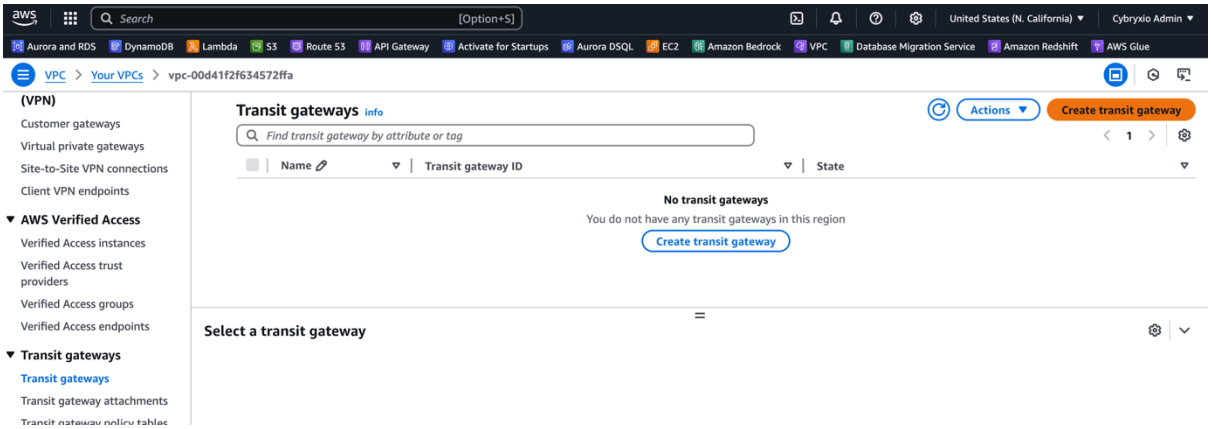


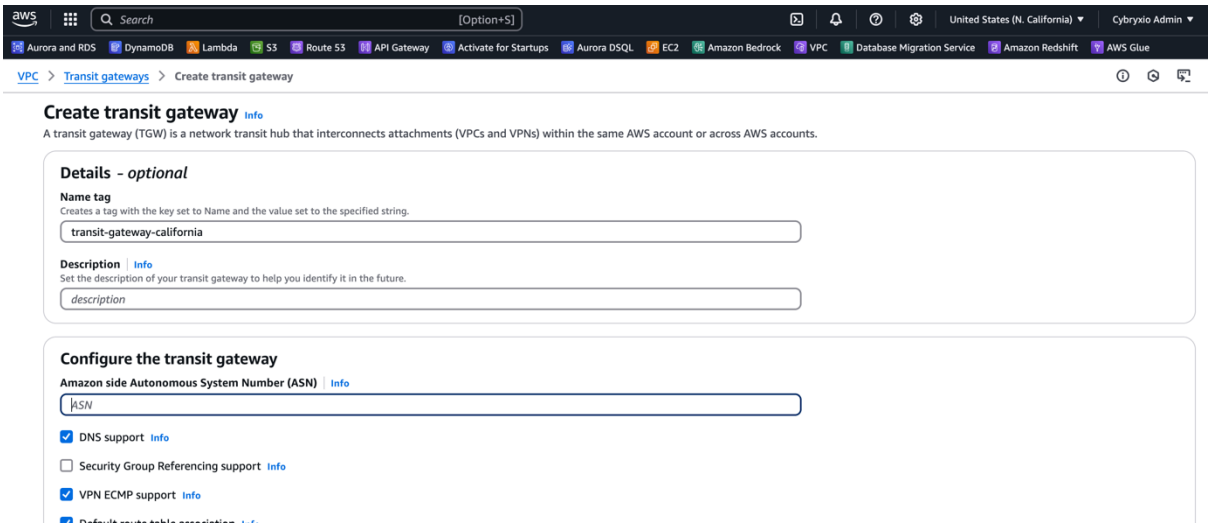
We can further extend this solution by Creating another Transit Gateway in another AWS Region, attach your VPCs to them, and then create a Transit Gateway peering attachment between the two Transit Gateways, ensuring proper route table updates.

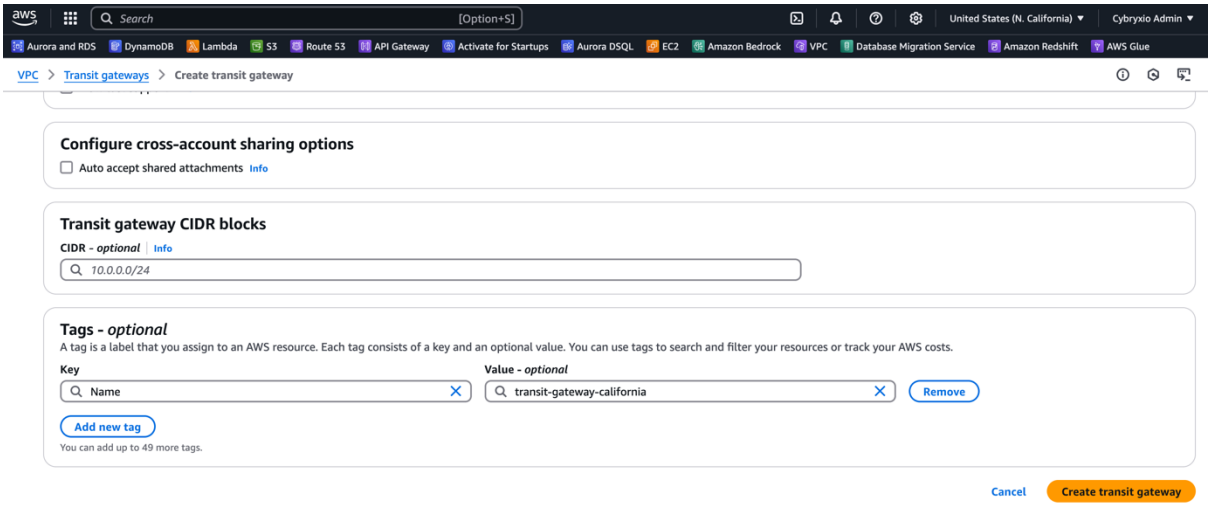
We now have another VPC in us-west-1 (N. California) region.

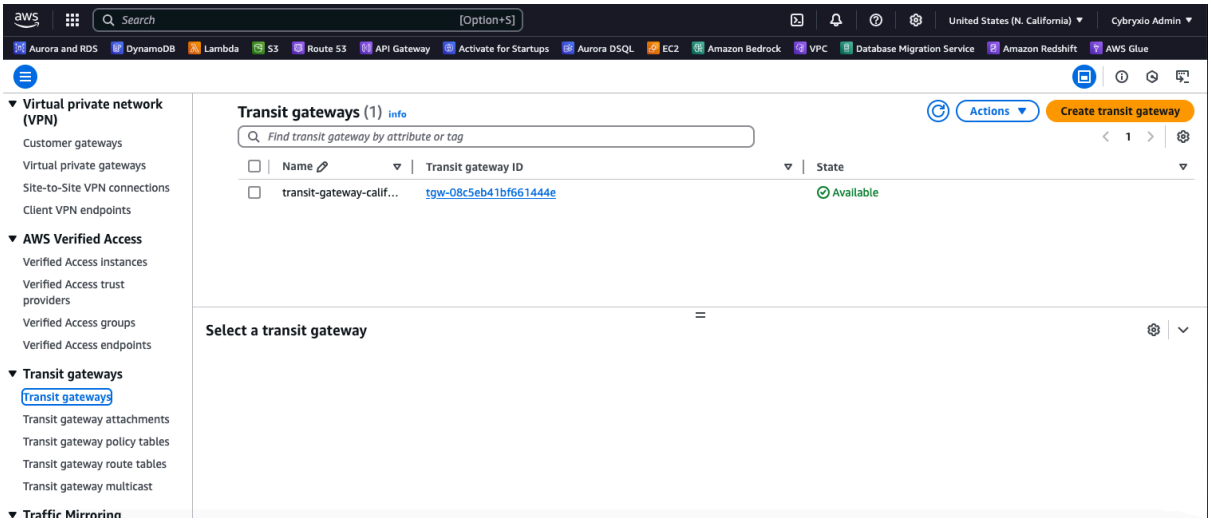


First let us create another Transit Routing Gateway in us-west-1 and attach the VPC to it.

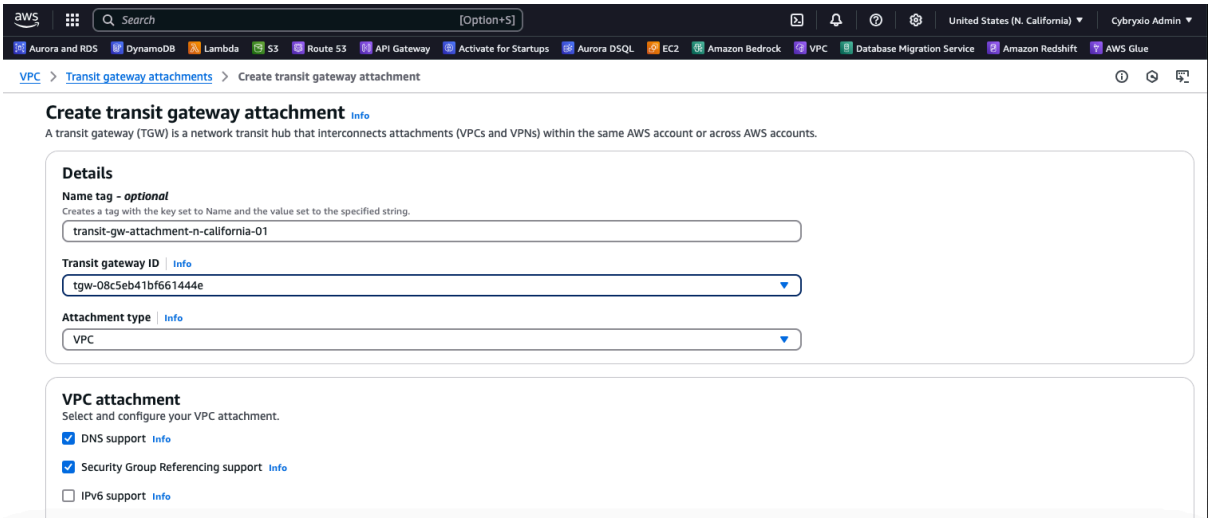


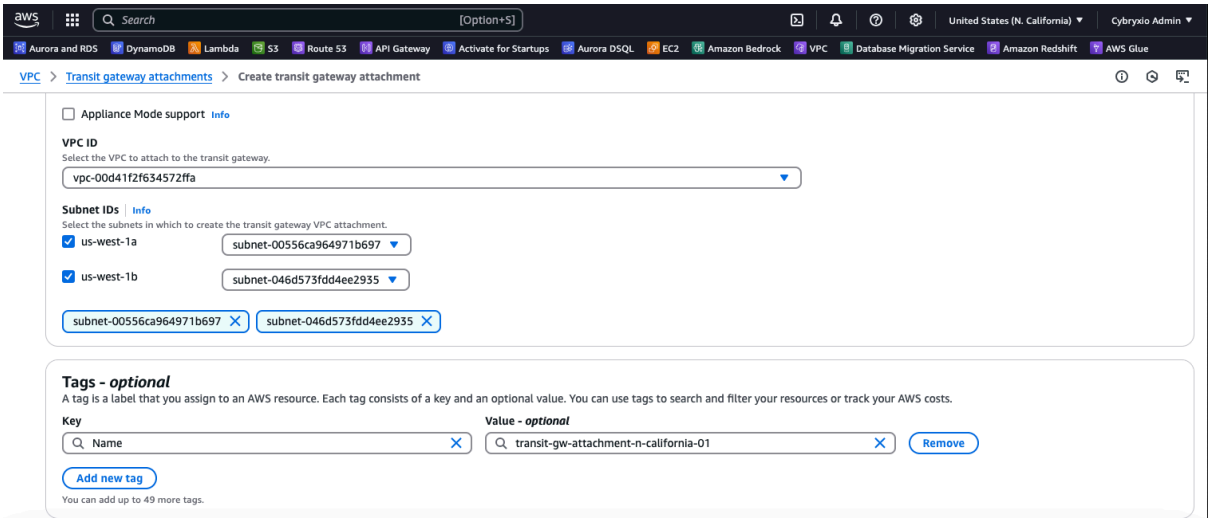




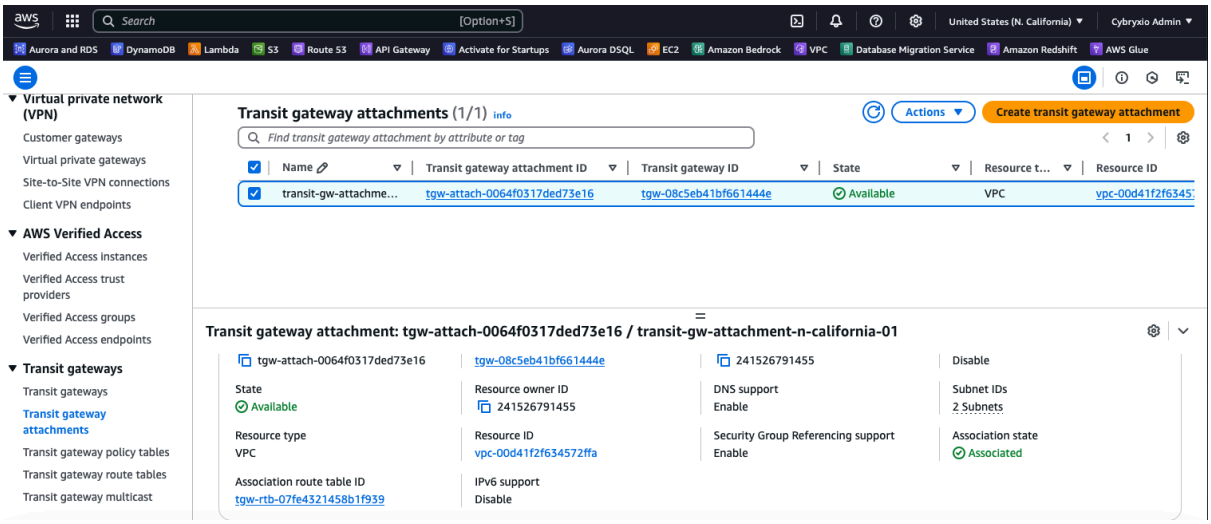


Create Transit Gateway attachment





Create Transit Gateway Attachment

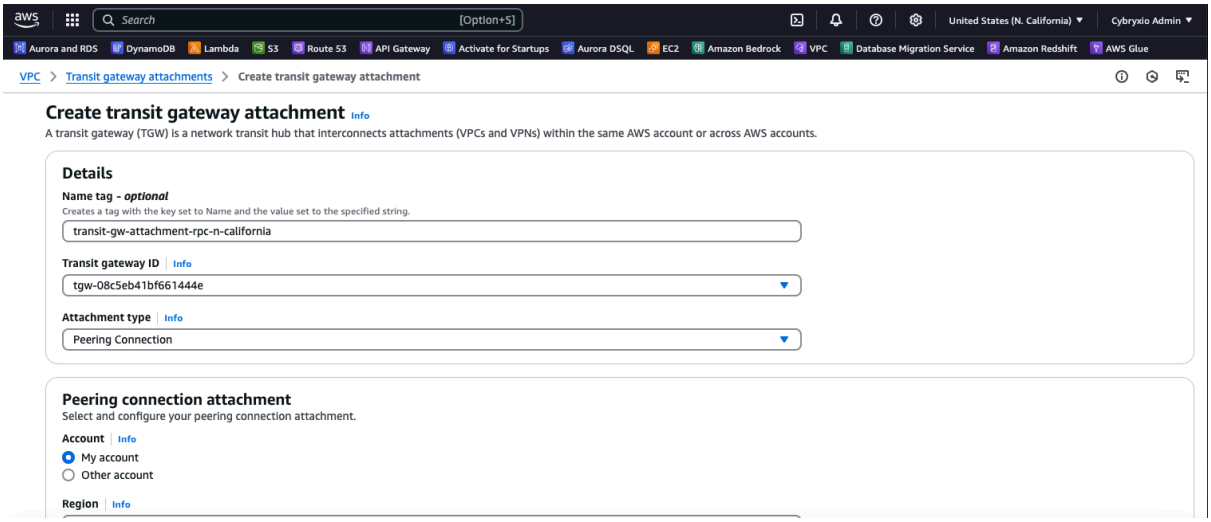


We now need to connect the Transit Routing Gateways in us-east-1 and us-west-1 via Peering connection.
Ref : https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering.html

Before you begin, ensure that you have the ID of the transit gateway that you want to attach. If the transit gateway is in another AWS account, ensure that you have the AWS account ID of the owner of the transit gateway.

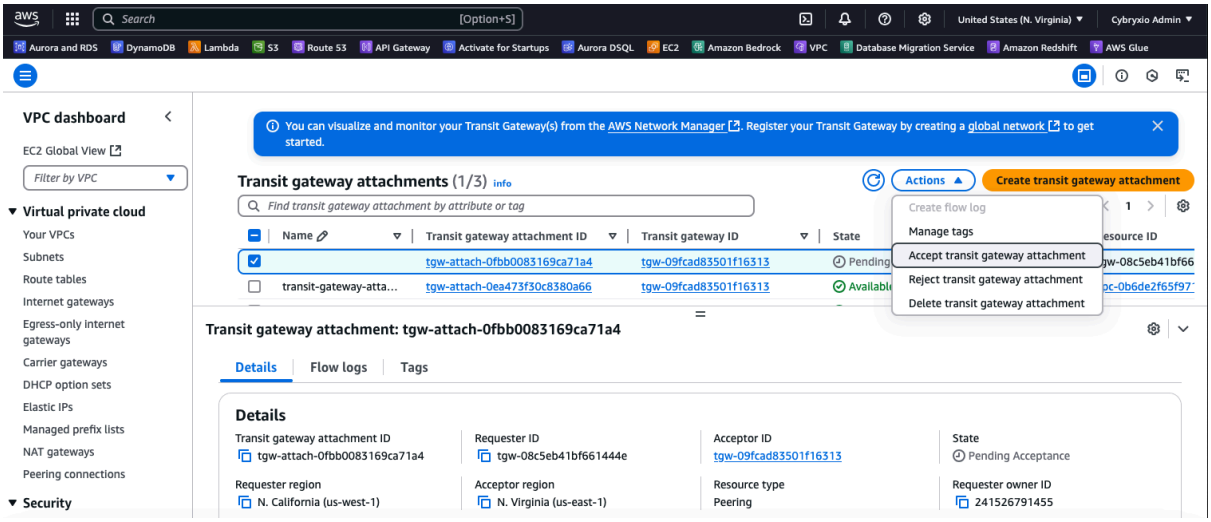
My Transit Gateway ID in us-east-1 is tgw-09fcad83501f16313

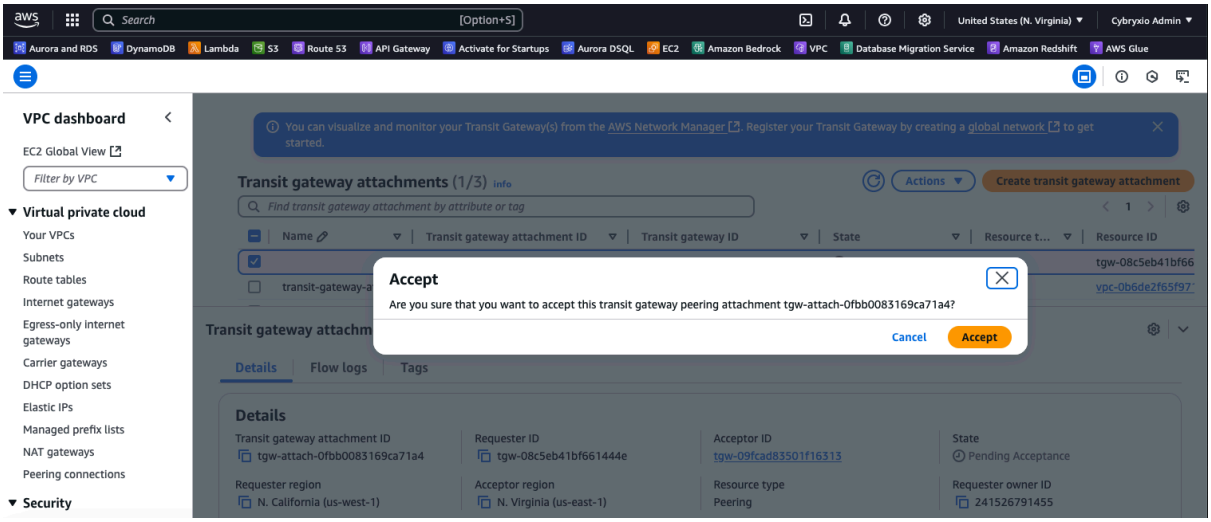
Go back to Transit Gateway Attachments and Create Transit Gateway Attachment in us-west-1
Ref : https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering-create.html

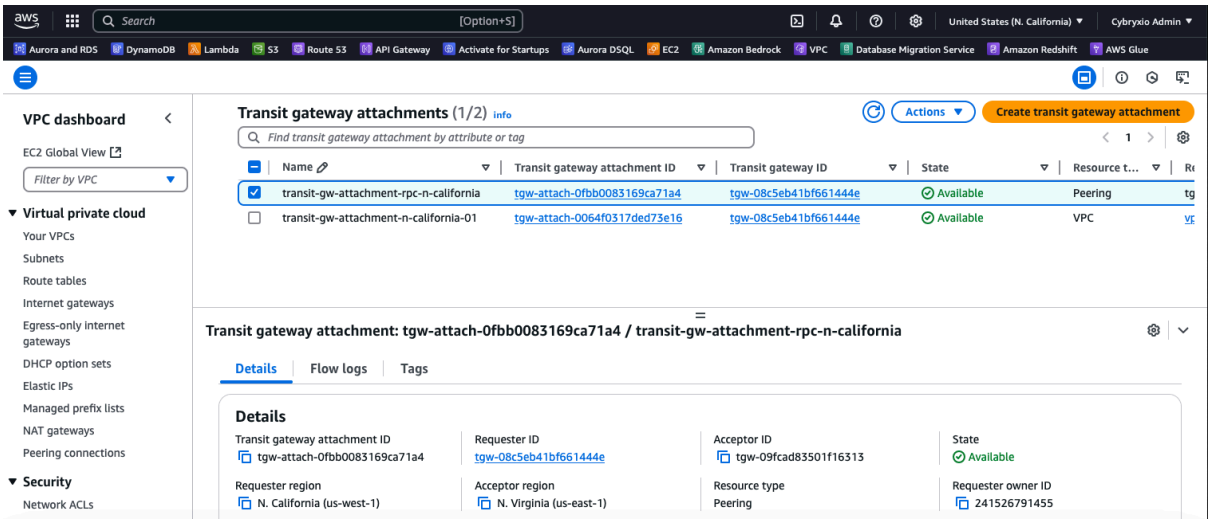


Make sure there are no empty spaces in the string when you paste the Transit Gateway ID

Create the peering connection attachment and wait for it to become ‘Pending Acceptance’. Go to us-east-1 TRG Attachments and accept the request

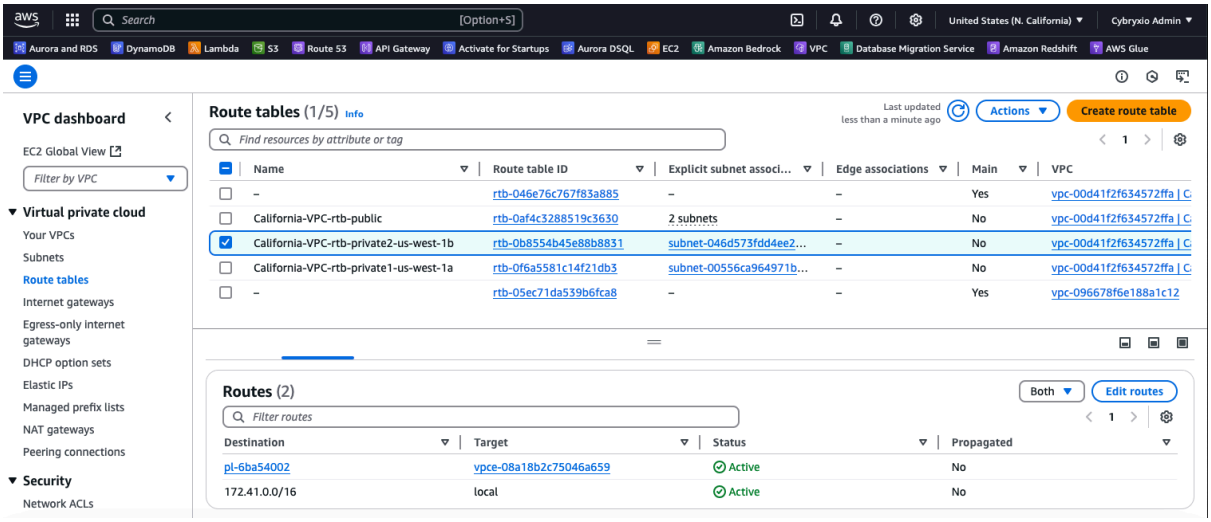


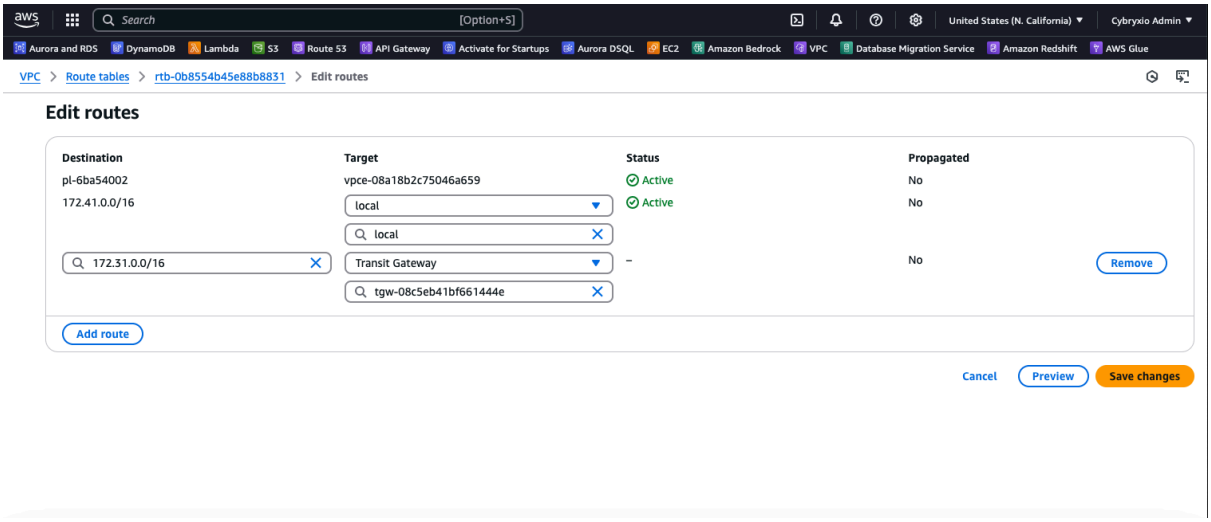


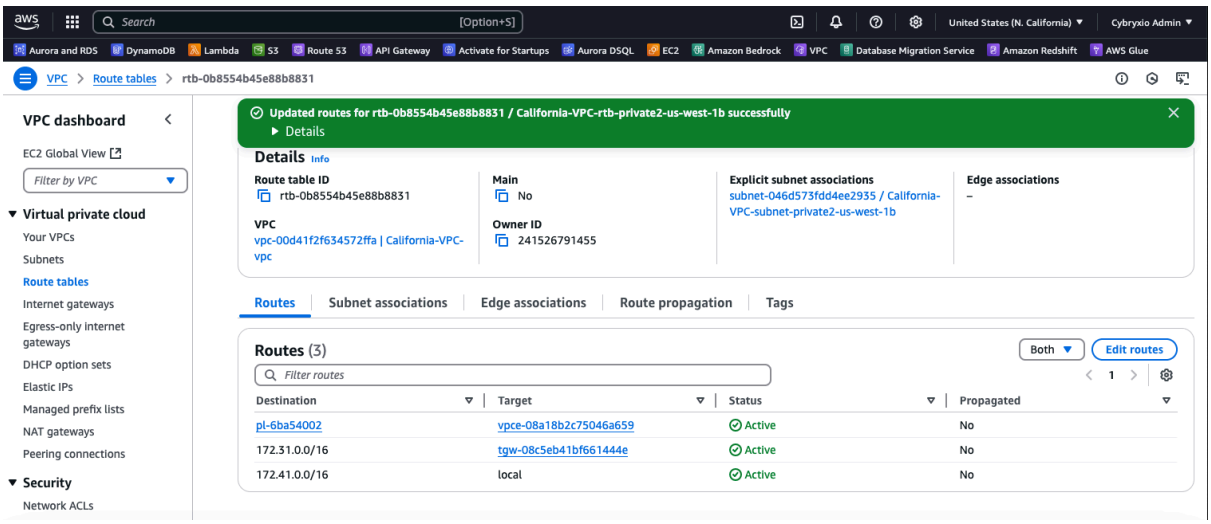


Once it is completed, add the routes of both VPC's to the respective Route tables in each regions

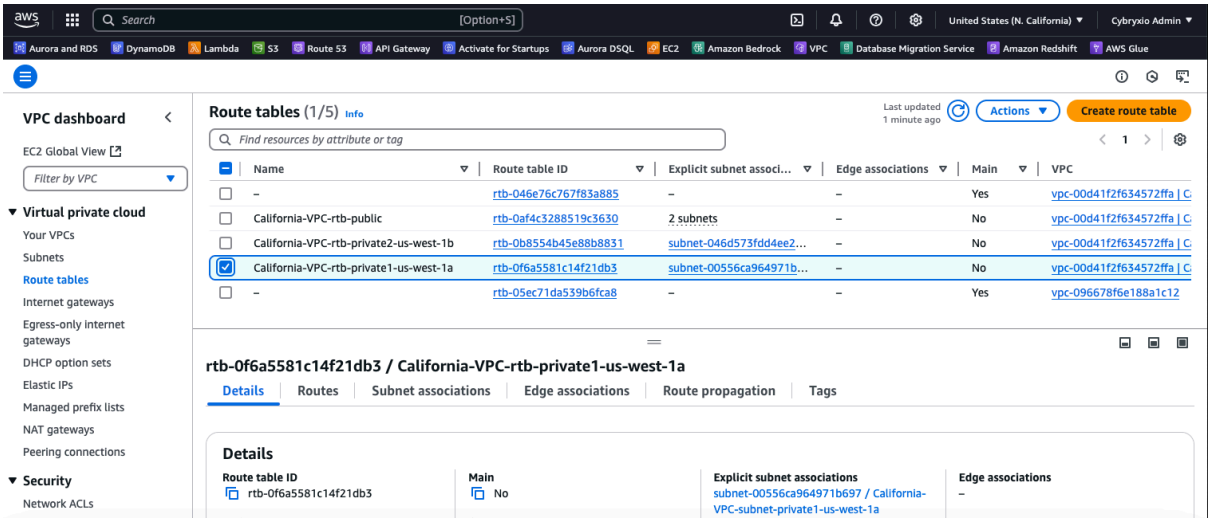
In California VPC Route tables add 172.31.0.0/16 (CIDR for us-east-1) to the private route tables for which you created the TRG attachments

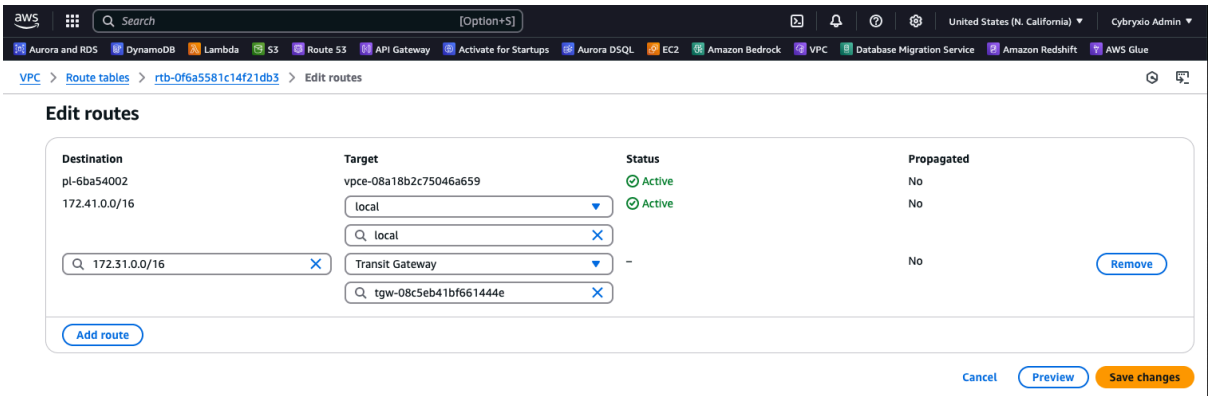


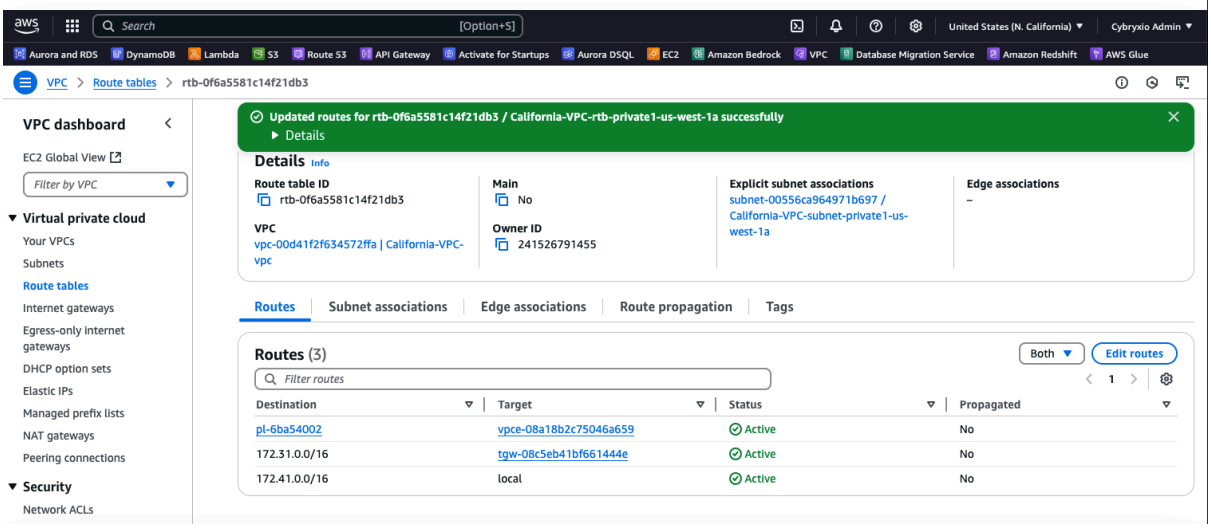




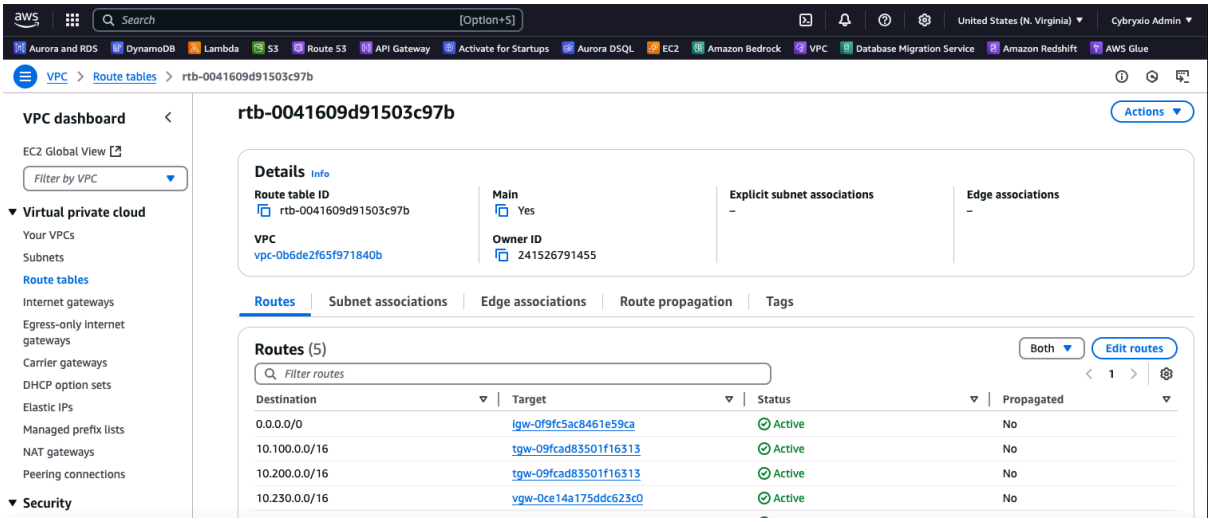
Add it for the other private subnet as well

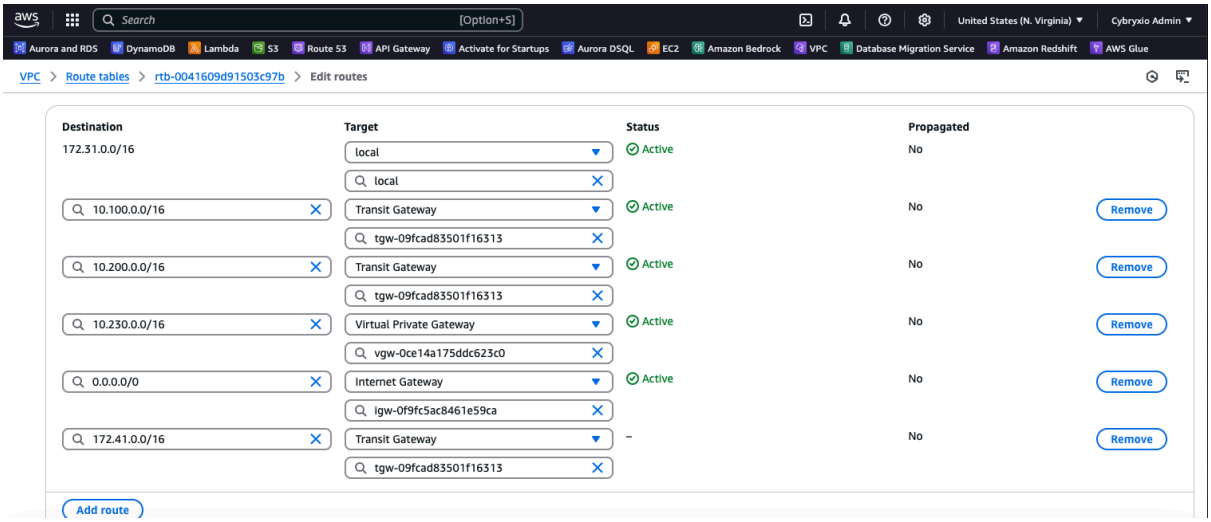


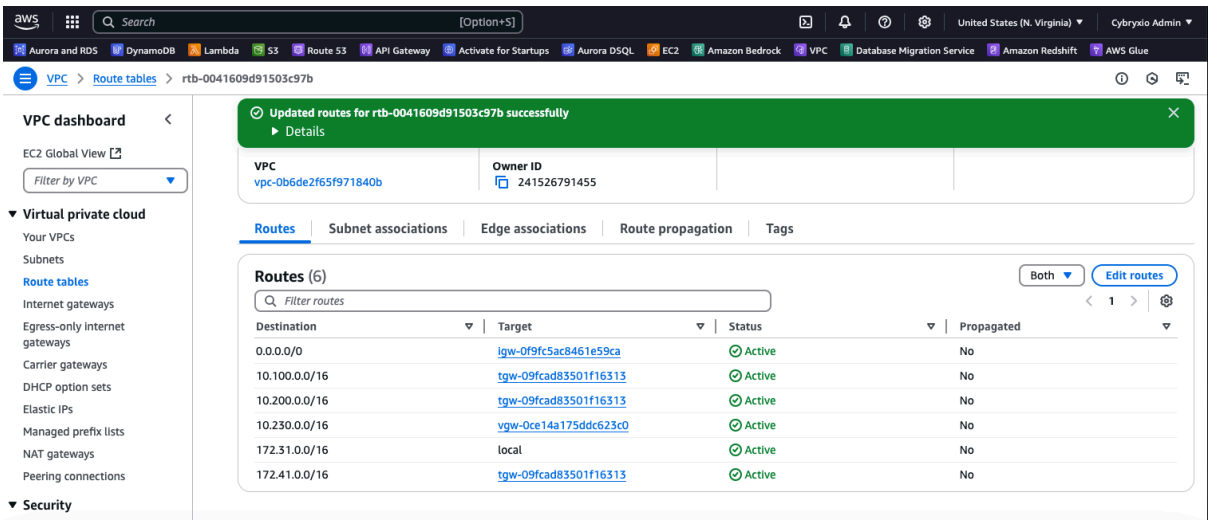




Now In N Virginia VPC Route tables add 172.41.0.0/16 (CIDR for us-west-1) to the private route tables for which you created the TRG attachments

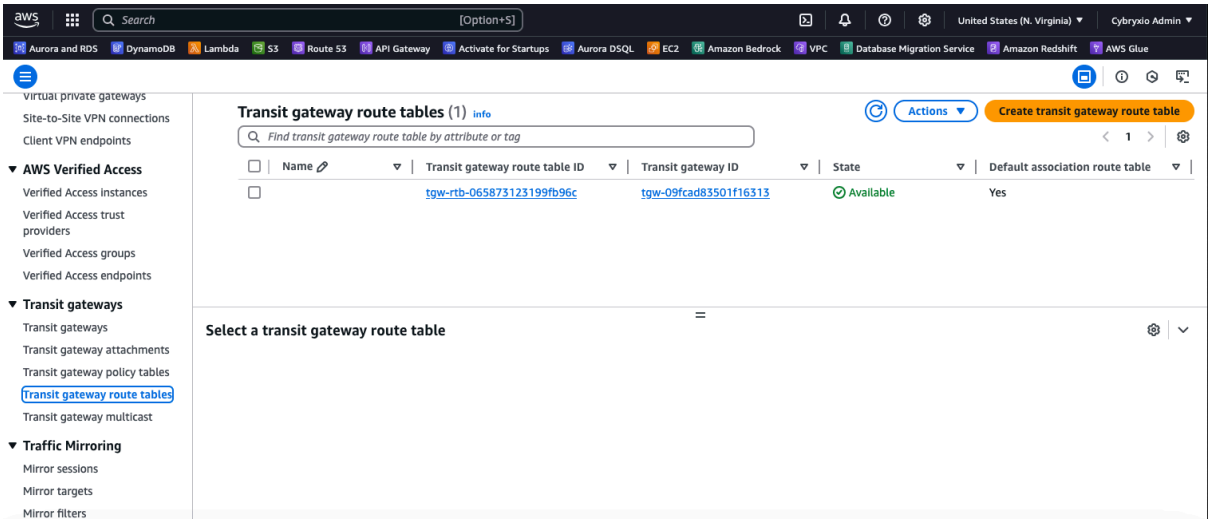


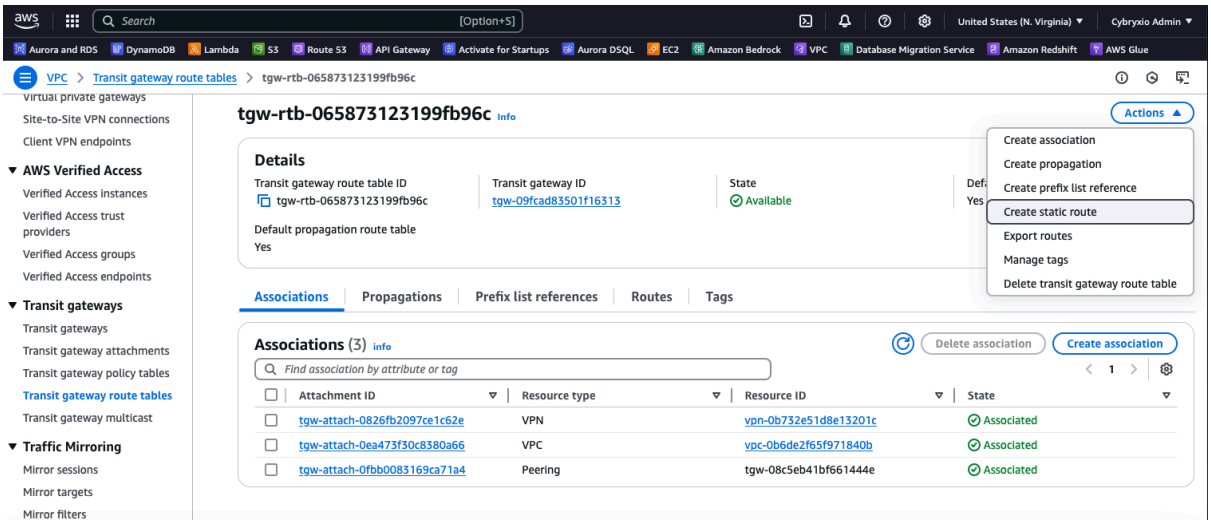


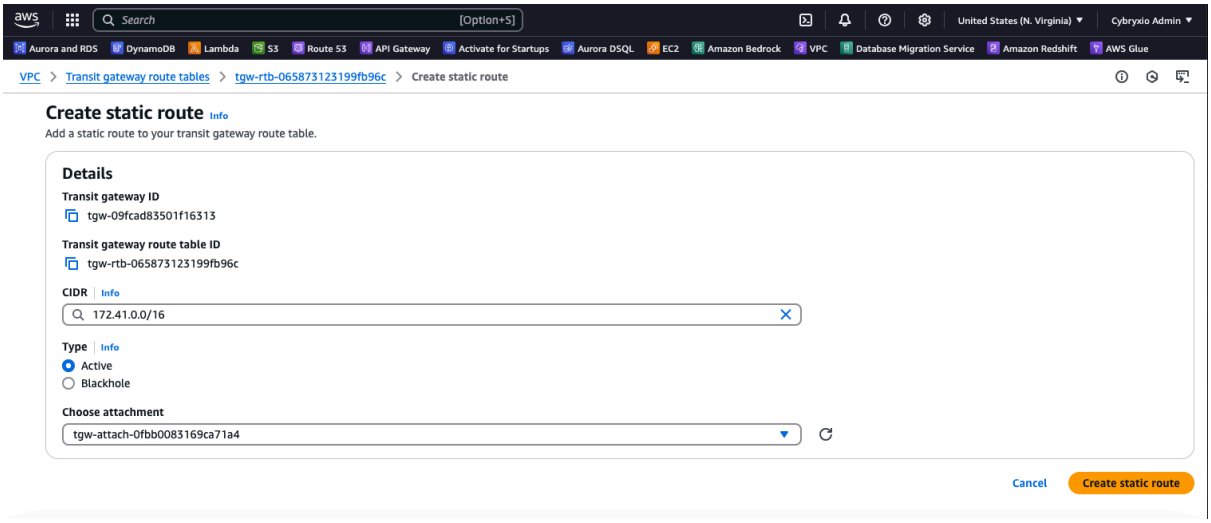


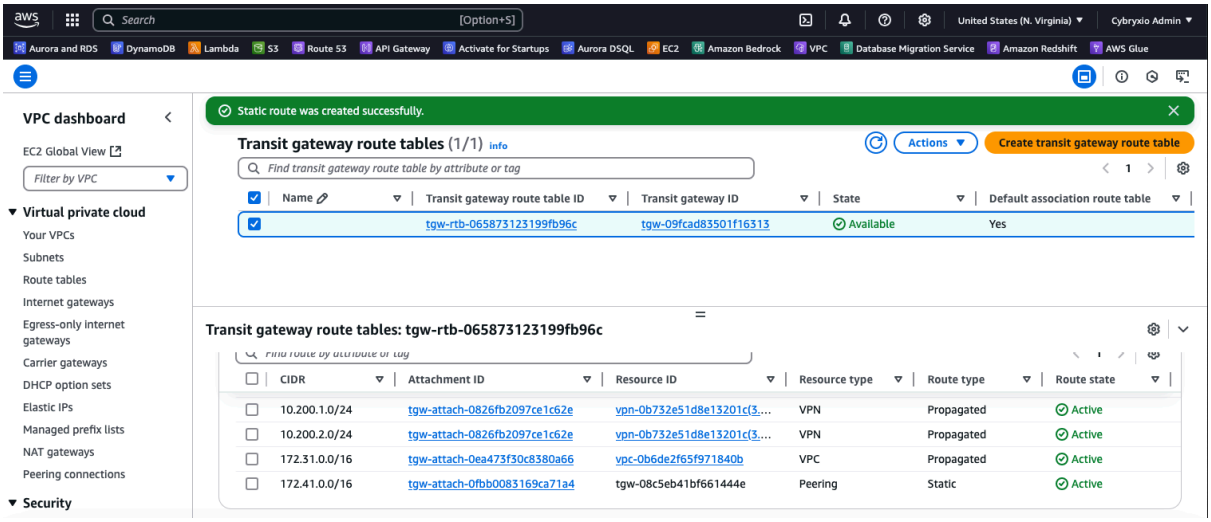
Add a route to a transit gateway route table using Amazon VPC Transit Gateways
Ref : https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering-add-route.html

In us-east-1


Search
[Option+S]
United States (N. Virginia)
Cybryxio Admin
Aurora and RDS
DynamoDB
Lambda
S3
Route 53
API Gateway
Activate for Startups
Aurora DSQL
EC2
Amazon Bedrock
VPC
Database Migration Service
Amazon Redshift
AWS Glue
Virtual private gateways
Site-to-Site VPN connections
Client VPN endpoints
AWS Verified Access
Verified Access instances
Verified Access trust providers
Verified Access groups
Verified Access endpoints
Transit gateways
Transit gateways
Transit gateway attachments
Transit gateway policy tables
Transit gateway route tables
Transit gateway multicast
Traffic Mirroring
Mirror sessions
Mirror targets
Mirror filters
Transit gateway route tables (1) info
Find transit gateway route table by attribute or tag
Actions
Create transit gateway route table
Name
Transit gateway route table ID
Transit gateway ID
State
Default association route table
tgw-rtb-065873123199fb96c
tgw-09fcad83501f16313
Available
Yes
Select a transit gateway route table

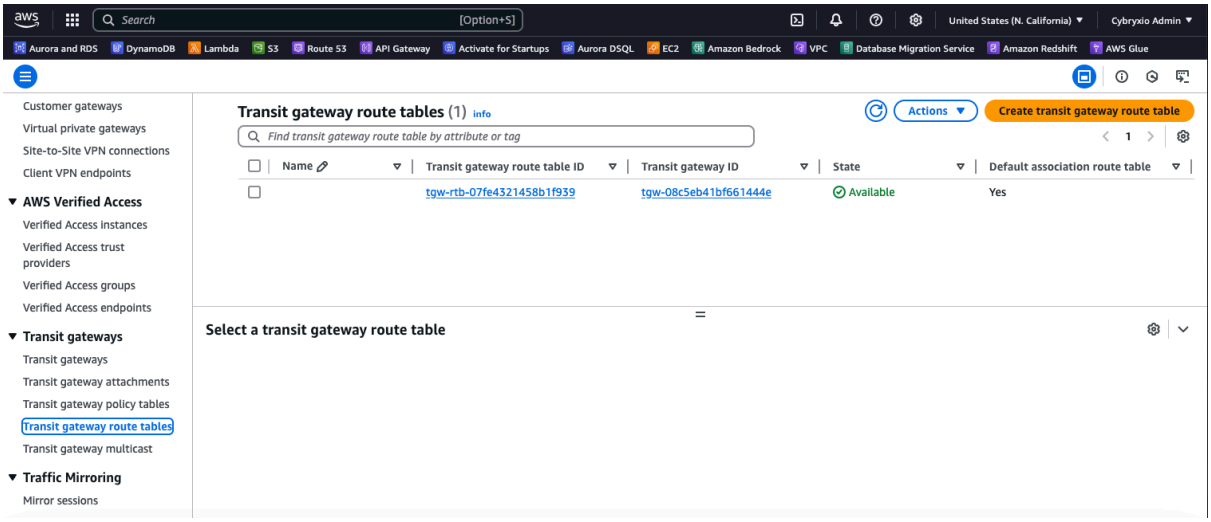

Search
[Option+S]
United States (N. Virginia)
Cybryxio Admin
VPC
Transit gateway route tables
tgw-rtb-065873123199fb96c
tgw-rtb-065873123199fb96c info
Actions
Create association
Create propagation
Create prefix list reference
Create static route
Export routes
Manage tags
Delete transit gateway route table
Details
Transit gateway route table ID
tgw-rtb-065873123199fb96c
Transit gateway ID
tgw-09fcad83501f16313
State
Available
Default propagation route table
Yes
Associations
Propagations
Prefix list references
Routes
Tags
Associations (3) info
Delete association
Create association
Find association by attribute or tag
Attachment ID
Resource type
Resource ID
State
tgw-attach-0826fb2097ce1c62e
VPN
vpn-0b732e51d8e13201c
Associated
tgw-attach-0ea473f30c8380a66
VPC
vpc-0b6de2f65f971840b
Associated
tgw-attach-0fbb0083169ca71a4
Peering
tgw-08c5eb41bf661444e
Associated

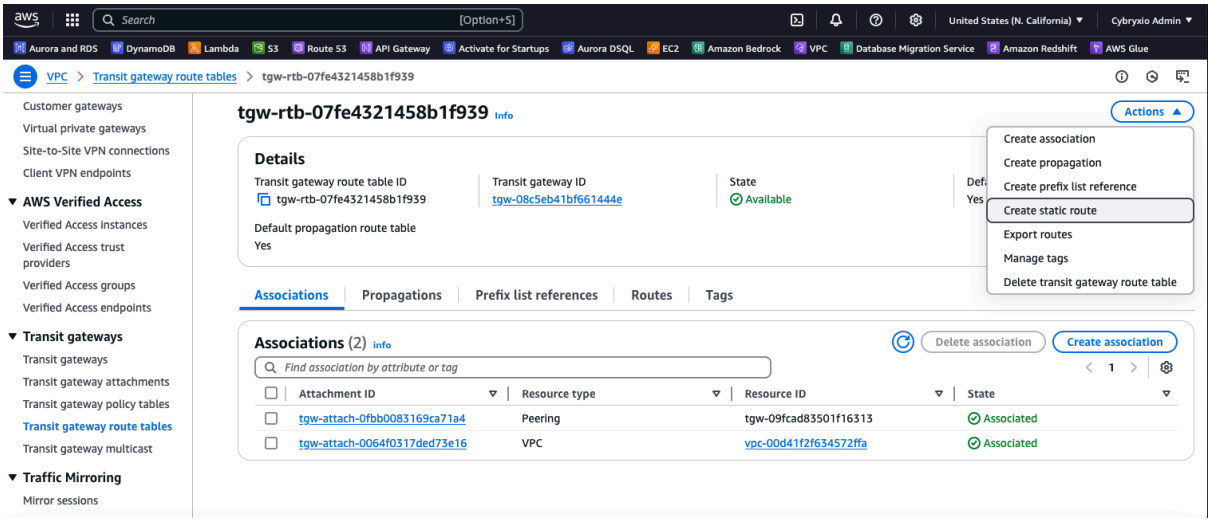

Search
[Option+S]
United States (N. Virginia)
Cybryxio Admin
VPC
Transit gateway route tables
tgw-rtb-065873123199fb96c
Create static route
Create static route info
Add a static route to your transit gateway route table.
Details
Transit gateway ID
tgw-09fcad83501f16313
Transit gateway route table ID
tgw-rtb-065873123199fb96c
CIDR info
172.41.0.0/16
Type info
Active
Blackhole
Choose attachment
tgw-attach-0fbb0083169ca71a4
Cancel
Create static route

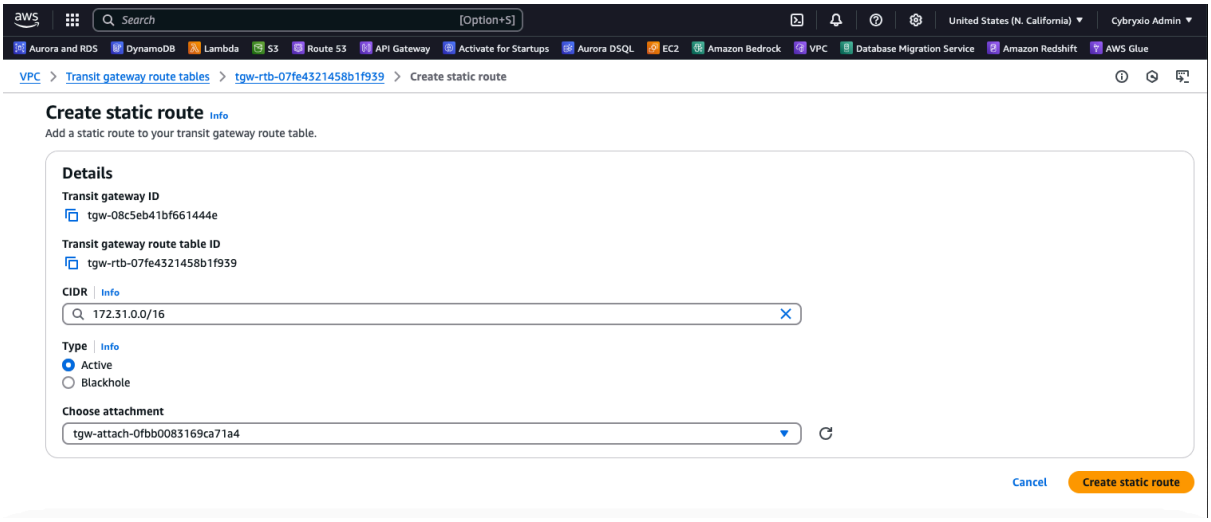


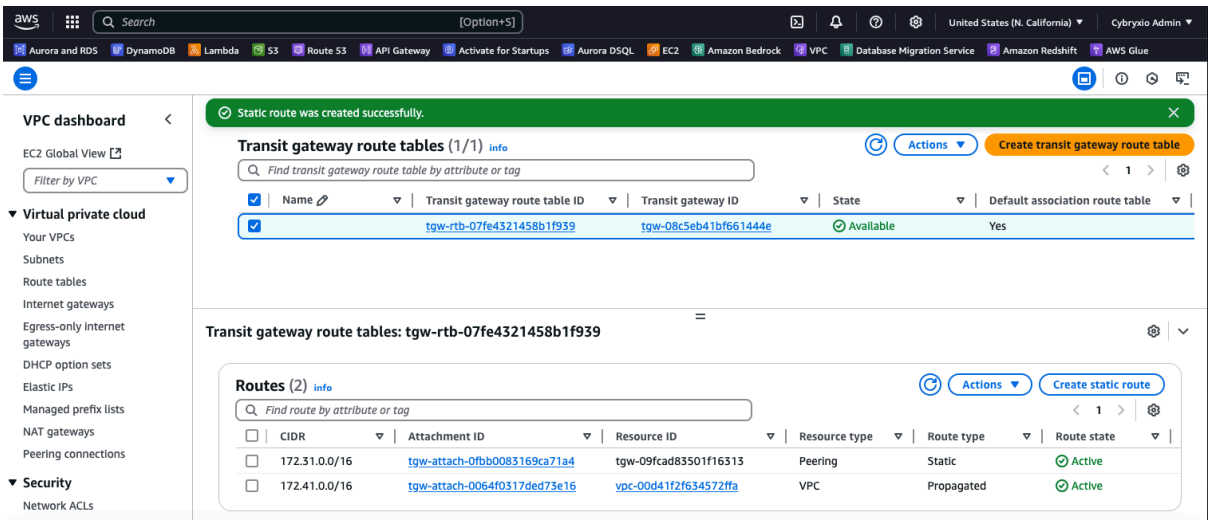
Add a route to a transit gateway route table using Amazon VPC Transit Gateways
Ref : https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering-add-route.html

In us-west-1







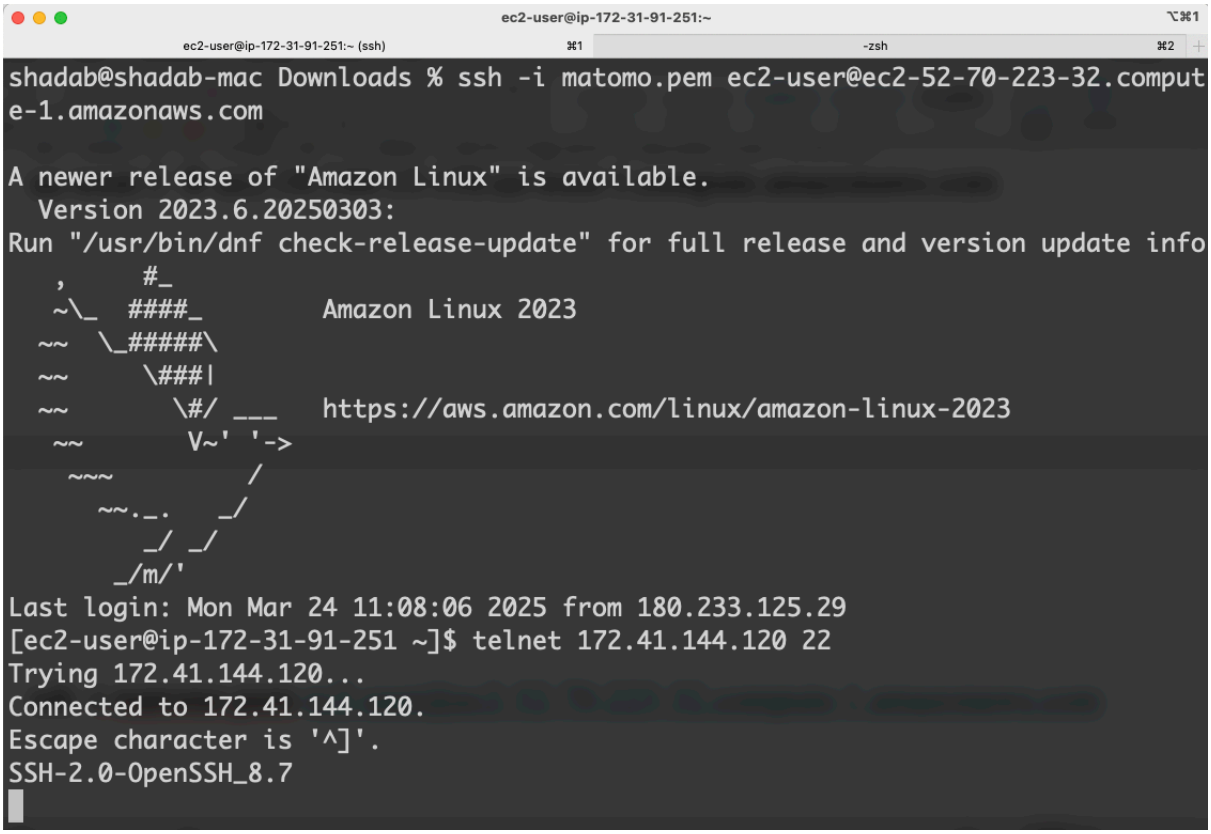


Create an Instance in us-west-2 and test connectivity to Instance in us-east-1

us -east-1 → ec2-52-70-223-32.compute-1.amazonaws.com
us-west-1 → ec2-18-144-13-158.us-west-1.compute.amazonaws.com

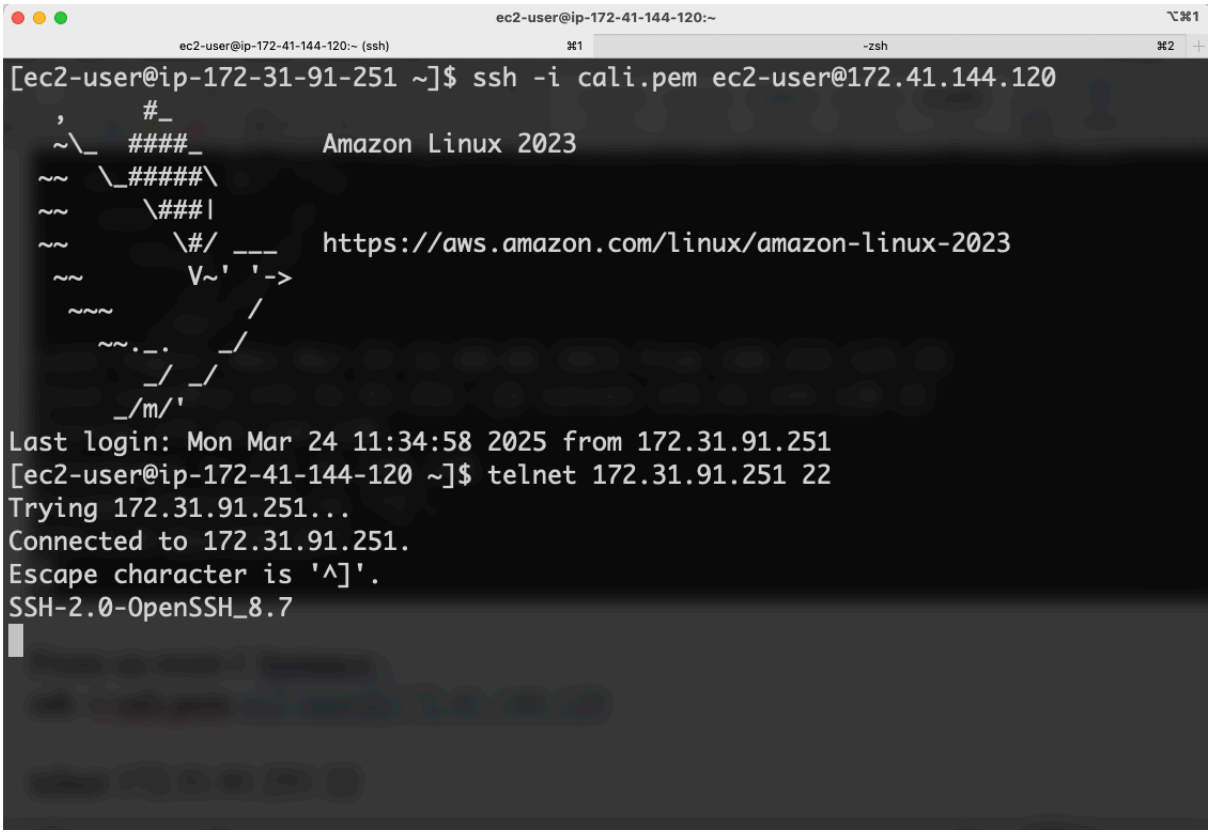
From us-east-1 Instance :
ssh -i matomo.pem ec2-user@ec2-52-70-223-32.compute-1.amazonaws.com

telnet 172.41.144.120 22



From us-west-1 Instance :
ssh -i cali.pem ec2-user@172.41.144.120

telnet 172.31.91.251 22

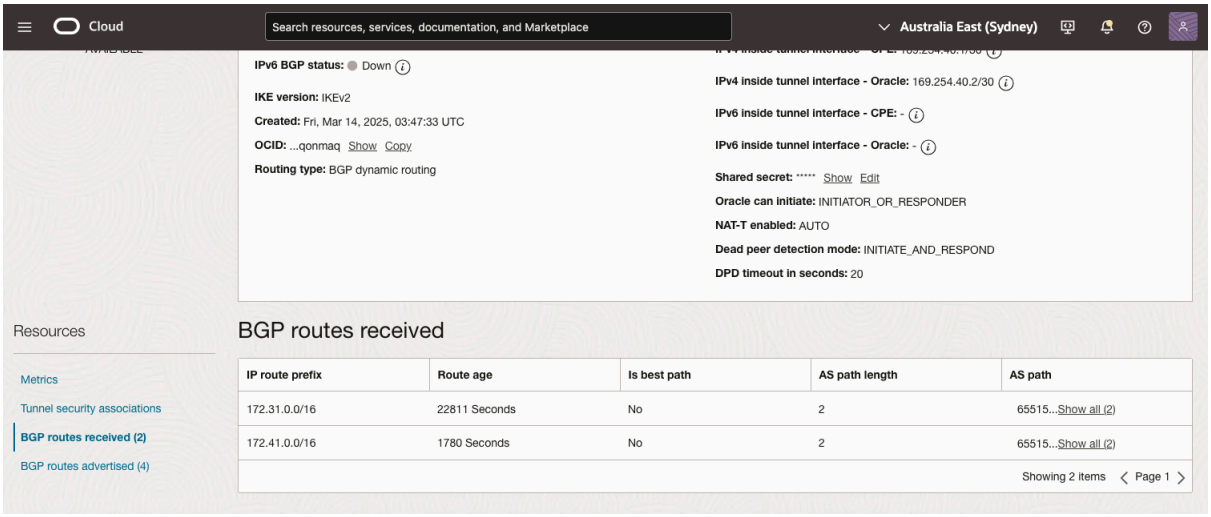


As we can see both us-east-1 and us-west-1 VPC's are able to communicate with each other.

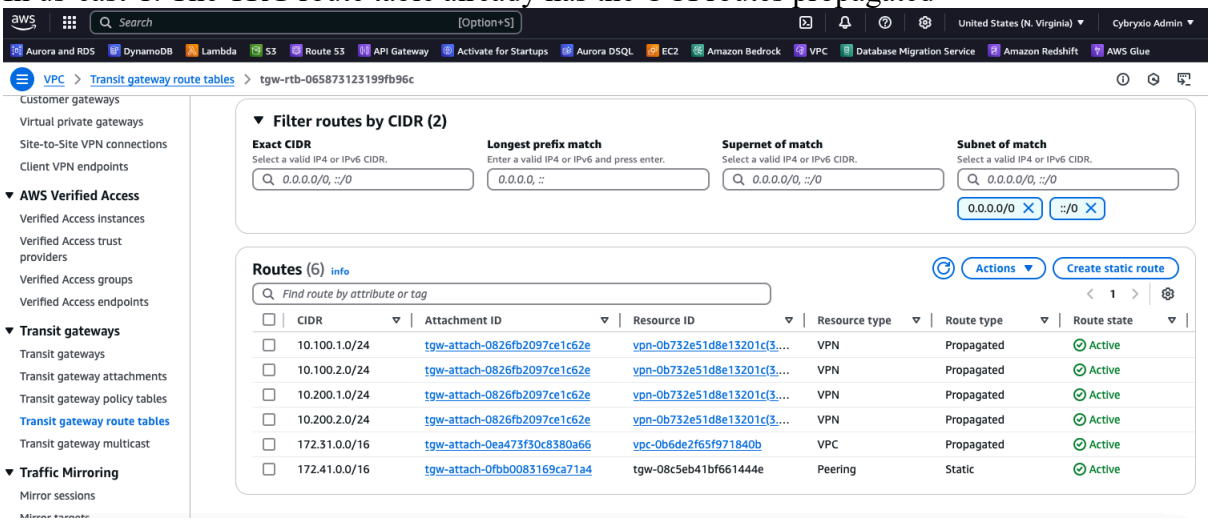
Our next step is now to enable us-west-1 VPC to talk to OCI Regions Sydney and Melbourne VCN's

AWS us-east-1 Instance IP : 172.31.91.251
AWS us-west-1 Instance IP : 172.41.144.120
OCI Sydney VM : 10.100.1.35
OCI Melbourne VM : 10.200.2.250

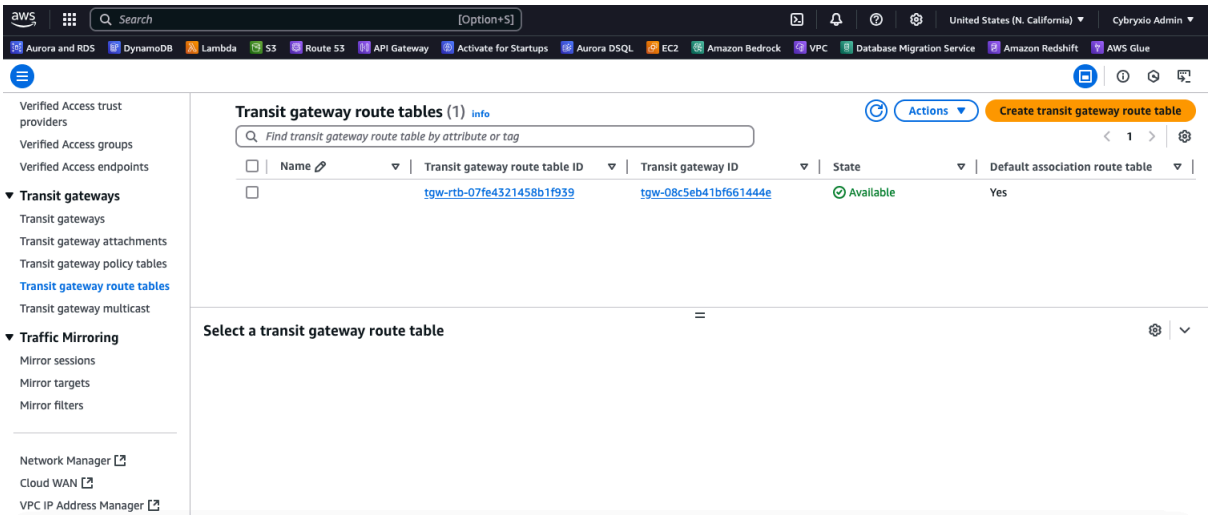
If you check the IPSec VPN tunnel Route propagation in OCI us-ashburn-1 which is the point where the IPSec Tunnel from AWS us-east-1 is connected, you can see both the VPC CIDR's published due to BGP and the Static Route added in AWS TRG Route Table

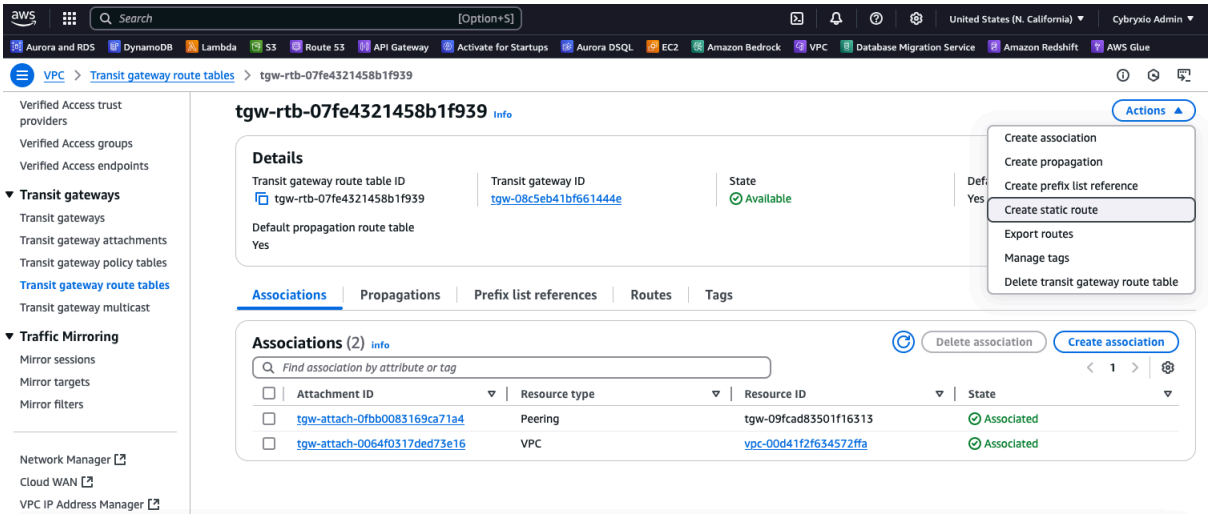


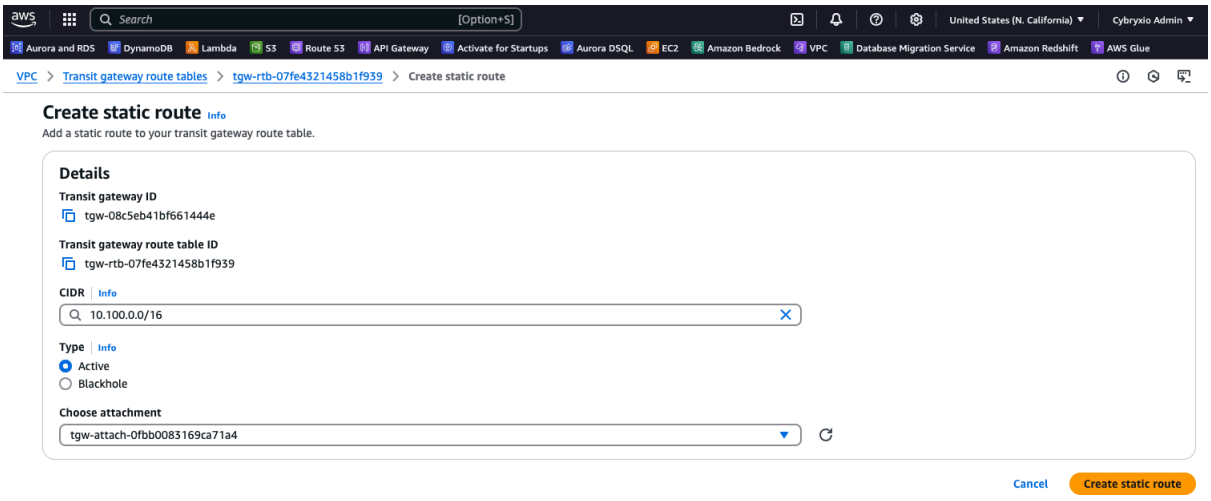
In us-east-1. The TRG route table already has the OCI routes propagated

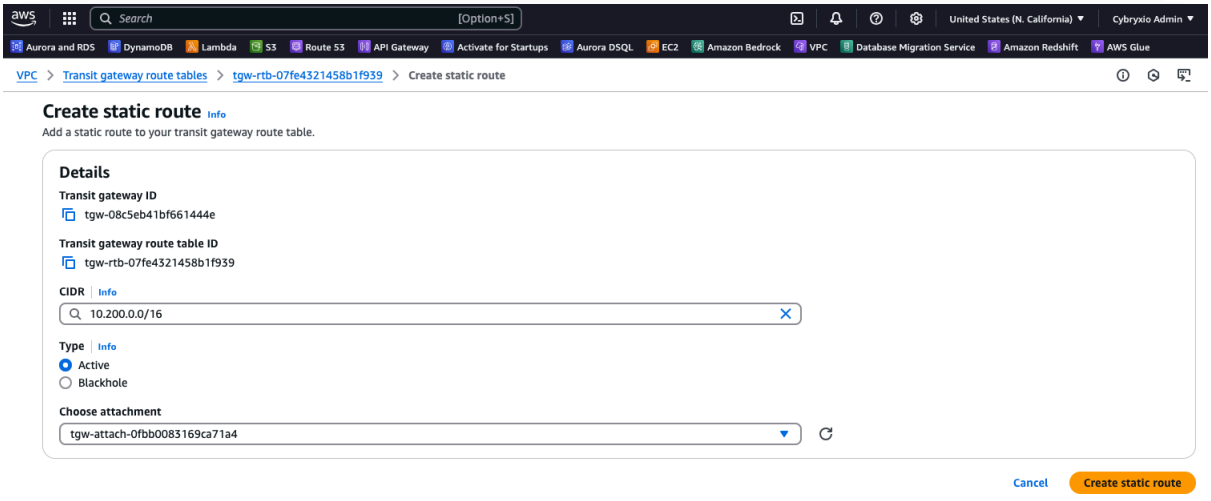


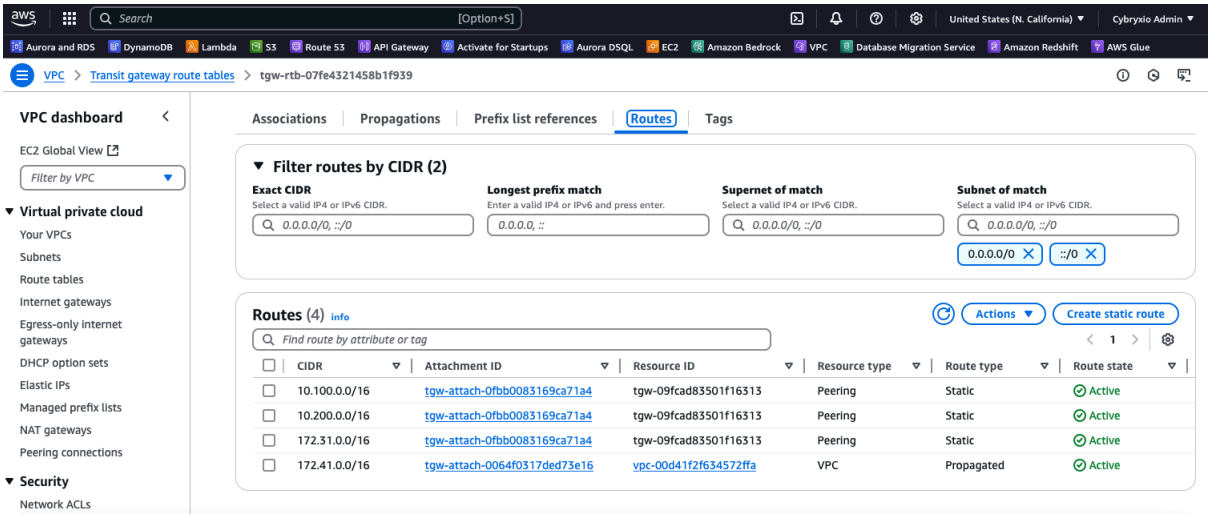
But we need to add the OCI routes in us-west-1 TRG Route table as static routes



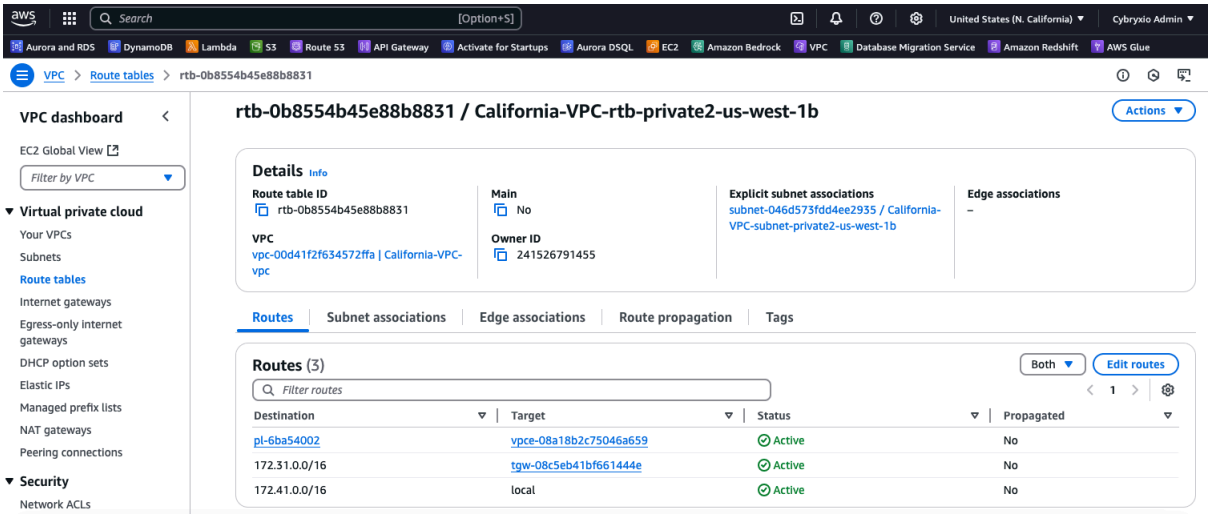


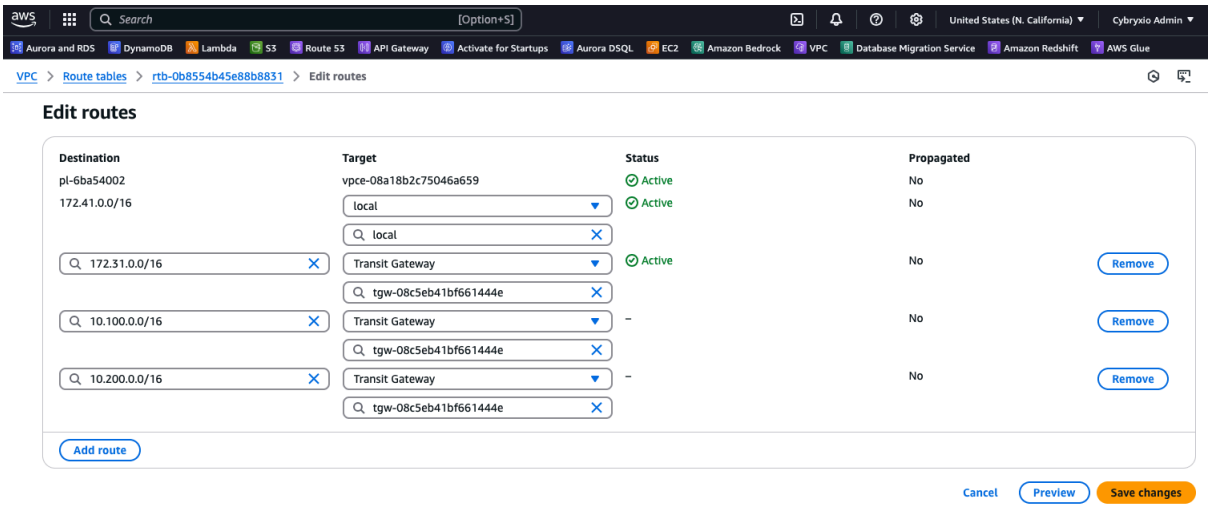


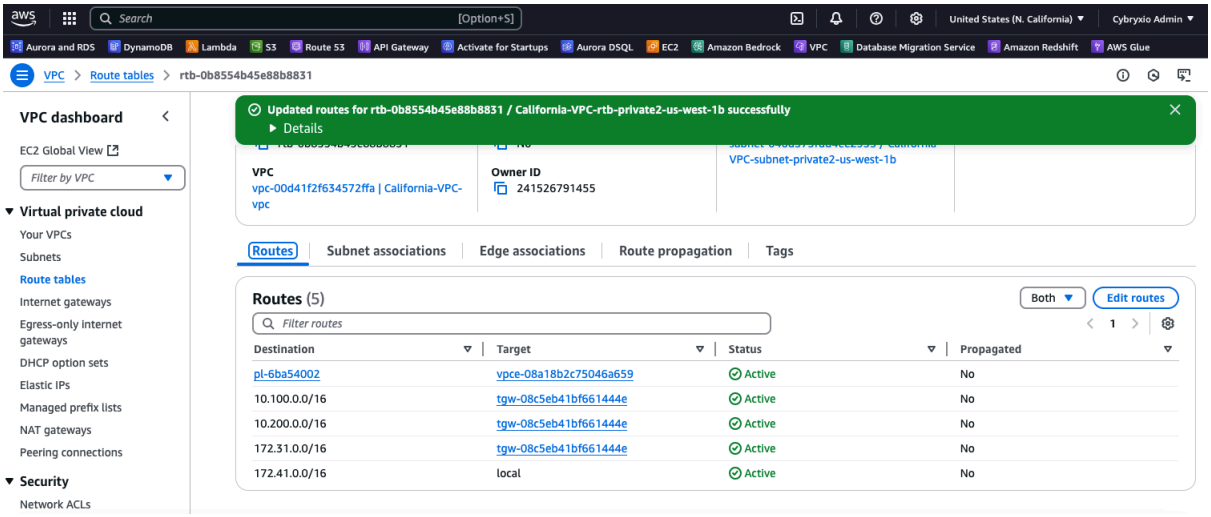


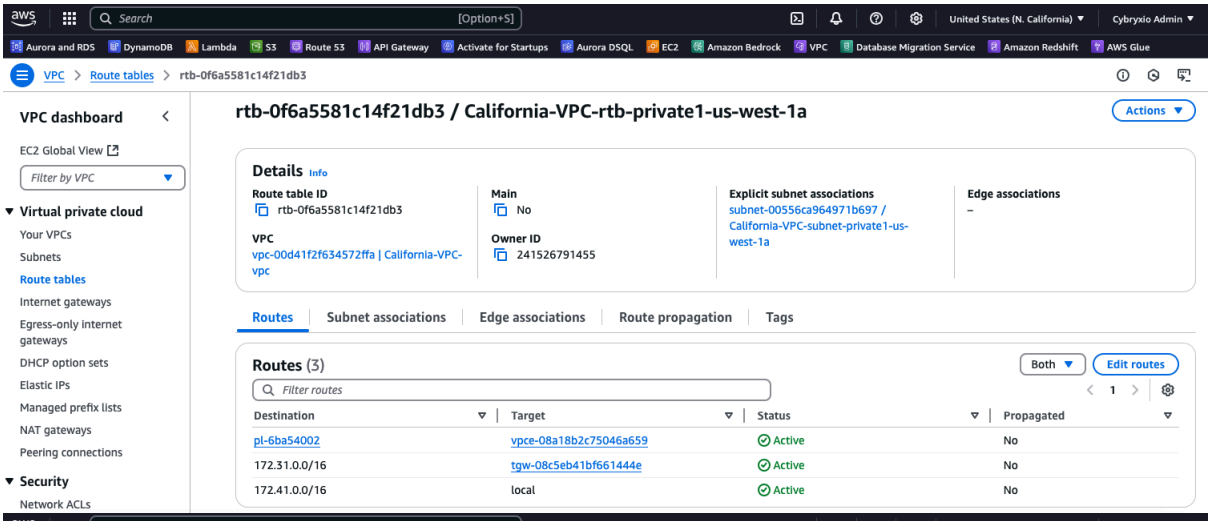


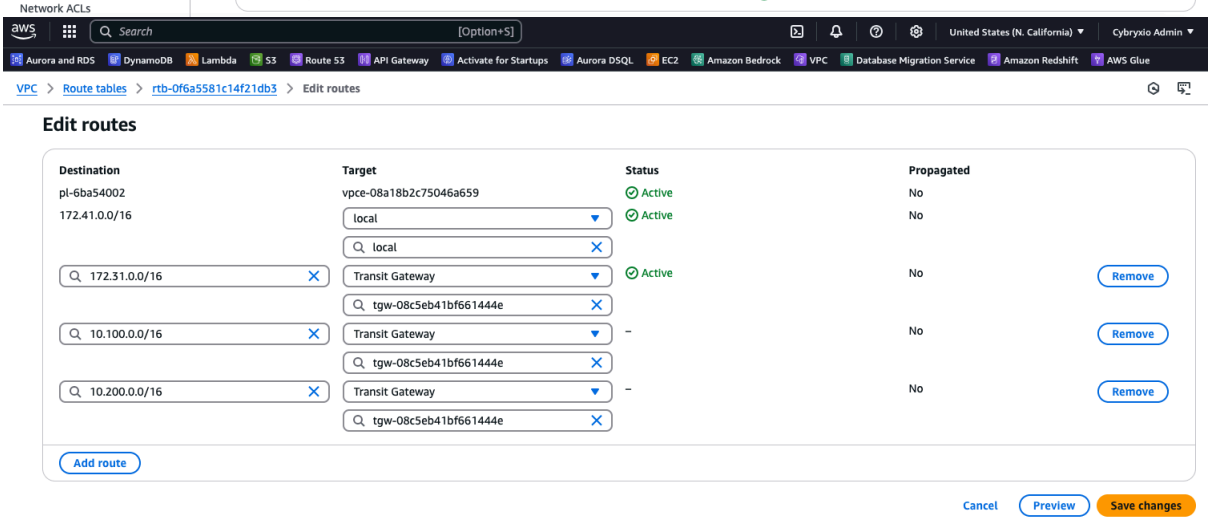
Add 10.100.0.0/16 and 10.200.0.0/16 OCI VCN CIDR to us-west-1 VPC (172.41.0.0/16) Route Table

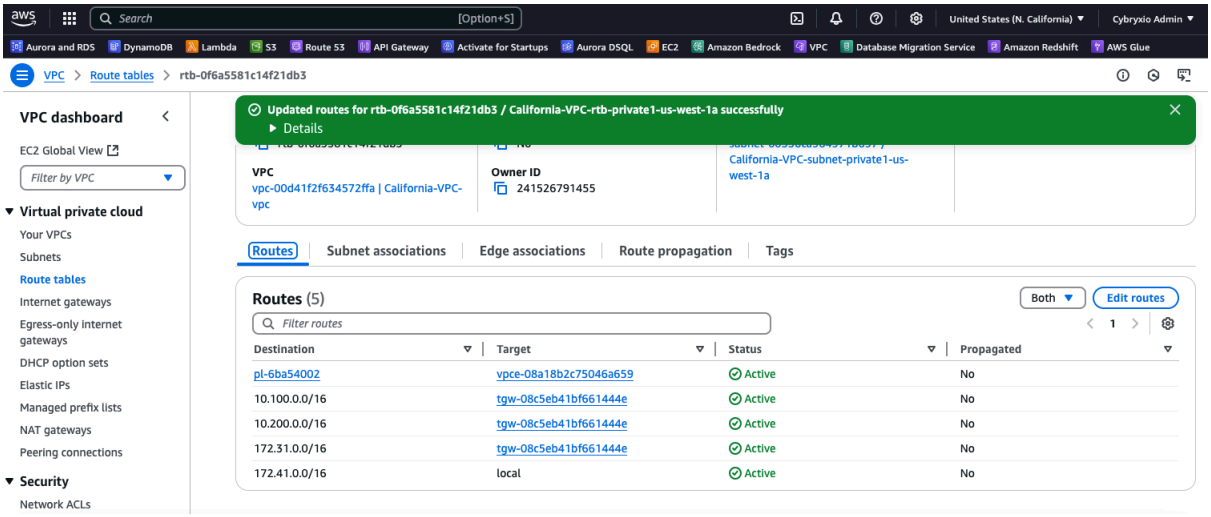






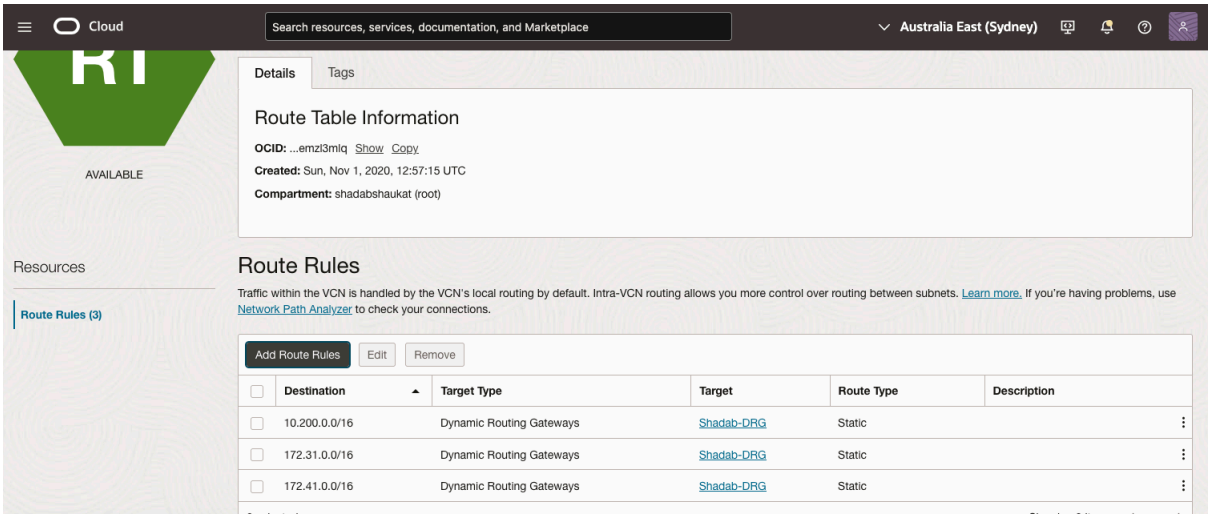


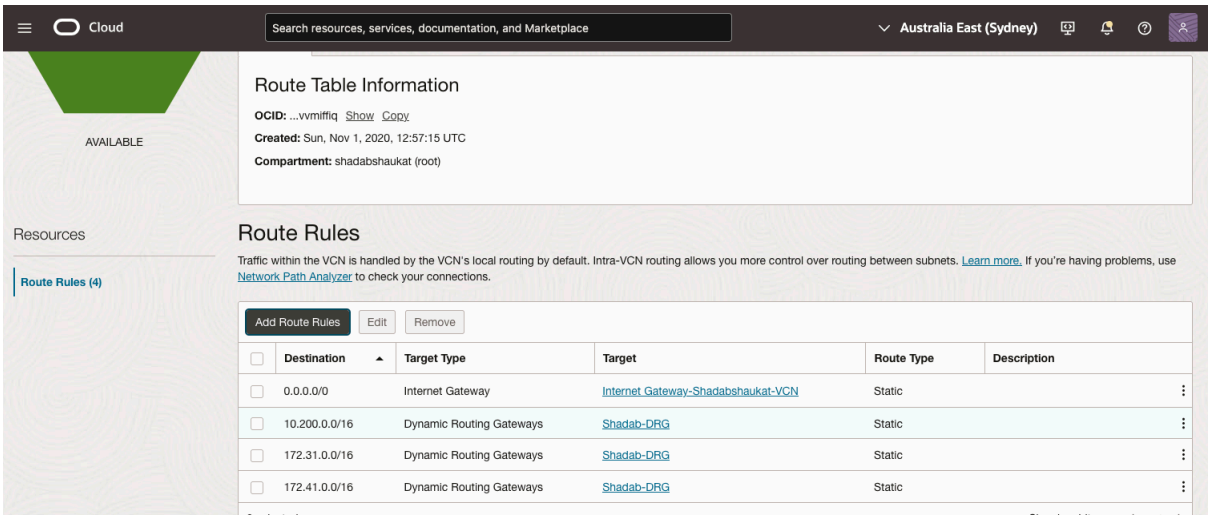




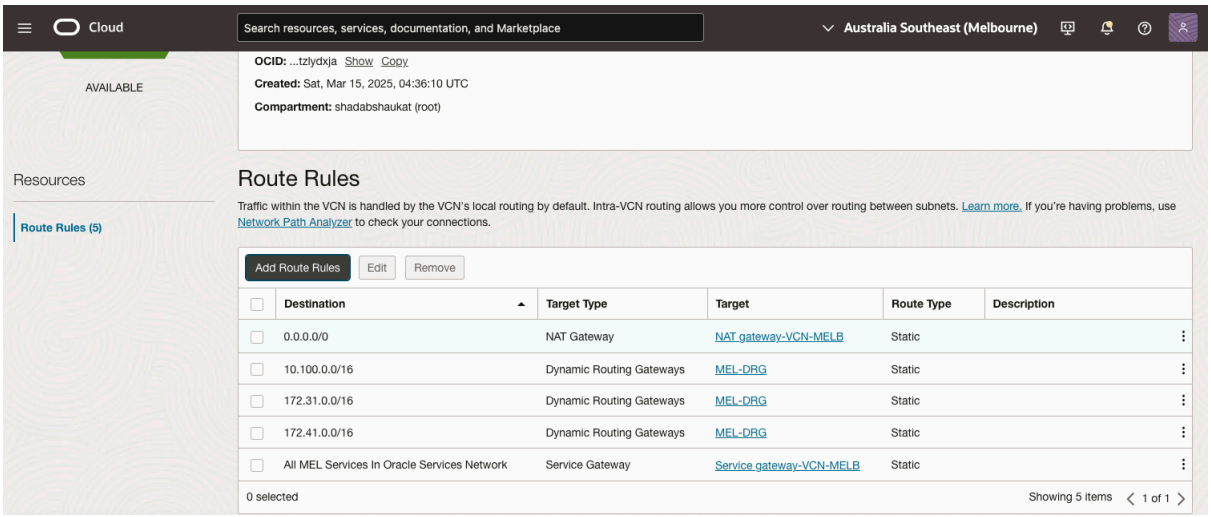
Now add the AWS VPC new CIDR 172.41.0.0/16 to both OCI regions VCN's Route tables

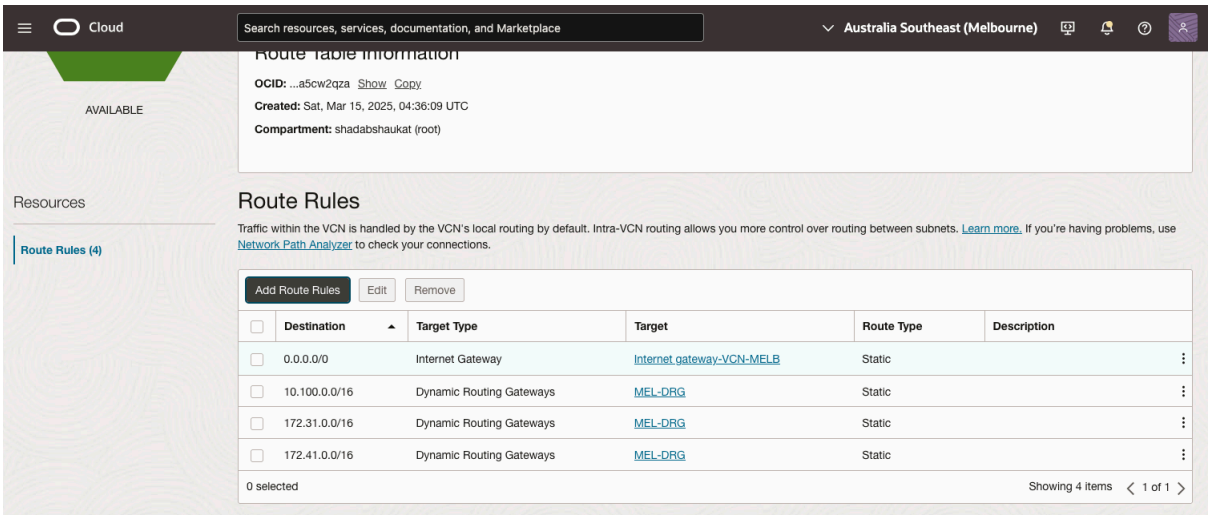
In OCI Sydney VCN in both Route tables



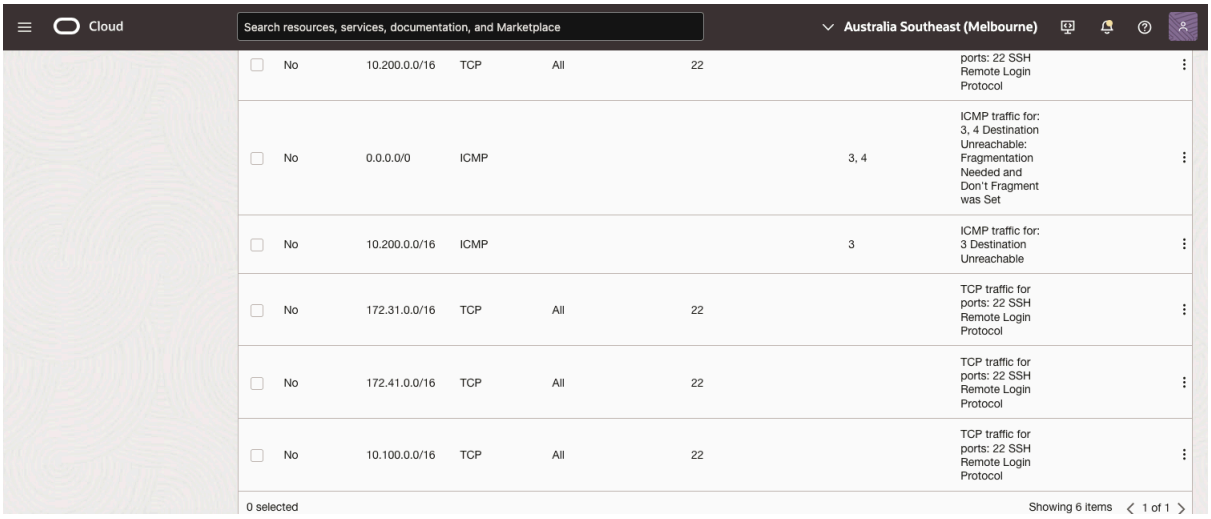


In OCI Melbourne VCN in both Route tables





Make sure your Security list on OCI VCNs in Sydney and Melbourne allows port 22 connection from both AWS VPC CIDRs



Now Lets do a connectivity test from each instance in 1 AWS region to other regions in OCI and repeat for all regions in both AWS and OCI.

AWS us-east-1 Instance IP : 172.31.91.251
AWS us-west-1 Instance IP : 172.41.144.120
OCI Sydney VM : 10.100.1.35
OCI Melbourne VM : 10.200.2.250

From *AWS us-east-1 Instance IP : 172.31.91.251 to :*
AWS us-west-1 Instance IP : 172.41.144.120
[ec2-user@ip-172-31-91-251 ~]$ telnet 172.41.144.120 22
Trying 172.41.144.120...
Connected to 172.41.144.120.
Escape character is '^]'.
SSH-2.0-OpenSSH_8.7

OCI Sydney VM : 10.100.1.35
[ec2-user@ip-172-31-91-251 ~]$ telnet 10.100.1.35 22
Trying 10.100.1.35...
Connected to 10.100.1.35.
Escape character is '^]'.
SSH-2.0-OpenSSH_8.0

OCI Melbourne VM : 10.200.2.250
[ec2-user@ip-172-31-91-251 ~]$ telnet 10.200.2.250 22
Trying 10.200.2.250...
Connected to 10.200.2.250.
Escape character is '^]'.
SSH-2.0-OpenSSH_8.0

From AWS us-west-1 Instance IP : 172.41.144.120 to :
AWS us-east-1 Instance IP : 172.31.91.251
[ec2-user@ip-172-41-144-120 ~]$ telnet 172.31.91.251 22
Trying 172.31.91.251...
Connected to 172.31.91.251.
Escape character is '^]'.
SSH-2.0-OpenSSH_8.7

OCI Sydney VM : 10.100.1.35
[ec2-user@ip-172-41-144-120 ~]$ telnet 10.100.1.35 22
Trying 10.100.1.35...
Connected to 10.100.1.35.
Escape character is '^]'.
SSH-2.0-OpenSSH_8.0

OCI Melbourne VM : 10.200.2.250
[ec2-user@ip-172-41-144-120 ~]$ telnet 10.200.2.250 22
Trying 10.200.2.250...

Connected to 10.200.2.250.
Escape character is '^]'.
SSH-2.0-OpenSSH_8.0

From OCI Sydney VM : 10.100.1.35 to :
AWS us-east-1 Instance IP : 172.31.91.251
[opc@ords-secondary ~]$ curl -v telnet://172.31.91.251:22
** Rebuilt URL to: telnet://172.31.91.251:22/*
** Trying 172.31.91.251...*
** TCP_NODELAY set*
** Connected to 172.31.91.251 (172.31.91.251) port 22 (#0)*

AWS us-west-1 Instance IP : 172.41.144.120
[opc@ords-secondary ~]$ curl -v telnet://172.41.144.120:22
** Rebuilt URL to: telnet://172.41.144.120:22/*
** Trying 172.41.144.120...*
** TCP_NODELAY set*
** Connected to 172.41.144.120 (172.41.144.120) port 22 (#0)*

OCI Melbourne VM : 10.200.2.250
[opc@ords-secondary ~]$ curl -v telnet://10.200.2.250:22
* Rebuilt URL to: telnet://10.200.2.250:22/
* Trying 10.200.2.250...
* TCP_NODELAY set
* Connected to 10.200.2.250 (10.200.2.250) port 22 (#0)
SSH-2.0-OpenSSH_8.0

From OCI Melbourne VM : 10.200.2.250 to :
AWS us-east-1 Instance IP : 172.31.91.251
[opc@instance-20250324-2246 ~]$ curl -v telnet://172.31.91.251:22
** Rebuilt URL to: telnet://172.31.91.251:22/*
** Trying 172.31.91.251...*
** TCP_NODELAY set*
** Connected to 172.31.91.251 (172.31.91.251) port 22 (#0)*

AWS us-west-1 Instance IP : 172.41.144.120
[opc@instance-20250324-2246 ~]$ curl -v telnet://172.41.144.120:22
** Rebuilt URL to: telnet://172.41.144.120:22/*
** Trying 172.41.144.120...*
** TCP_NODELAY set*
** Connected to 172.41.144.120 (172.41.144.120) port 22 (#0)*

OCI Sydney VM : 10.100.1.35
[opc@instance-20250324-2246 ~]$ curl -v telnet://10.100.1.35:22
** Rebuilt URL to: telnet://10.100.1.35:22/*
** Trying 10.100.1.35...*
** TCP_NODELAY set*
** Connected to 10.100.1.35 (10.100.1.35) port 22 (#0)*

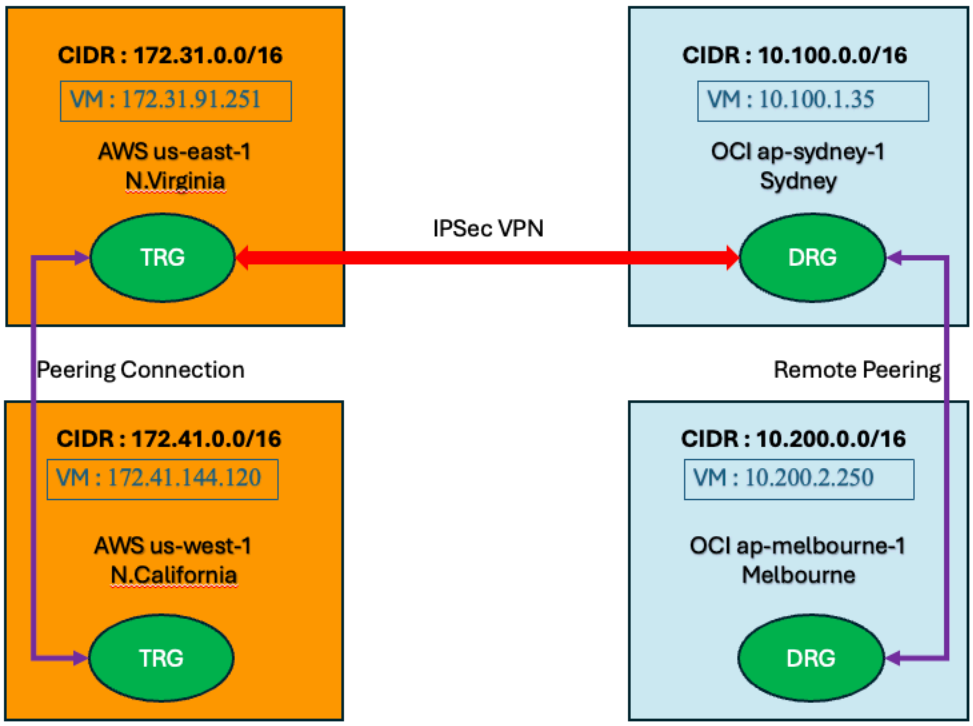
Connectivity Tests:

Source Instance	Destination Instance	Test Method	Result
AWS us-east-1 (172.31.91.251)	OCI Sydney (10.100.1.35)	SSH/Telnet	✅ Pass
AWS us-east-1 (172.31.91.251)	OCI Melbourne (10.200.2.250)	SSH/Telnet	✅ Pass
AWS us-west-1 (172.41.144.120)	OCI Sydney (10.100.1.35)	SSH/Telnet	✅ Pass
AWS us-west-1 (172.41.144.120)	OCI Melbourne (10.200.2.250)	SSH/Telnet	✅ Pass
OCI Sydney (10.100.1.35)	AWS us-east-1 (172.31.91.251)	SSH/Telnet	✅ Pass
OCI Sydney (10.100.1.35)	AWS us-west-1 (172.41.144.120)	SSH/Telnet	✅ Pass
OCI Melbourne (10.200.2.250)	AWS us-east-1 (172.31.91.251)	SSH/Telnet	✅ Pass
OCI Melbourne (10.200.2.250)	AWS us-west-1 (172.41.144.120)	SSH/Telnet	✅ Pass

SUMMARY

We've now created routing and connection over port 22 from all regions in AWS(x2) to all regions in OCI (x2) using Transit Routing Gateway in AWS and Dynamic Routing Gateway in OCI.

TRG are paired with Peering connections; DRG are paired with Remote Peering Connection. IPSec Tunnel in OCI has Import Route Distribution Added for the IPSec Tunnel connection. AWS TRG has Route Table Static routes added for OCI and other AWS region.



All the route rules, security group, security list in this Whitepaper for both AWS and OCI is only tested for port 22. For any other ports ensure you add the required tcp or udp ports in each of the security groups in AWS and security list in OCI.

Conclusion

This whitepaper outlines an extendable multi-cloud routing architecture integrating AWS Transit Gateway and OCI DRG. By leveraging site-to-site VPNs, dynamic BGP routing, and remote peering connections, enterprises can achieve scalable and secure connectivity across multiple AWS and OCI regions.

This solution provides the following benefits:

- **Scalability:** Extendable to additional regions and VPCs.
- **Security:** Encrypted connectivity using IPSec VPN.
- **Resilience:** High availability with redundant tunnels and peering.
- **Operational Efficiency:** Centralized routing management across multiple cloud regions.

REFERENCES

https://docs.oracle.com/en-us/iaas/Content/Network/Tasks/vpn_to_aws.htm
https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering-create.html
https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering.html
https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering-add-route.html
https://www.ateam-oracle.com/post/setting-up-secure-sitetosite-vpn-connectivity-from-oci-to-aws-transit-gateway
https://medium.com/@ystatit/azure-ipsec-vpn-to-oci-plus-oci-remote-peering-0b09e62e50a2