Inter-Region VPC Connectivity Using AWS Transit Gateway

Technical Documentation

Index

1. Introduction

Organizations running workloads in multiple AWS regions often require **secure and scalable communication between VPCs across regions**.

Using **Amazon Virtual Private Cloud (VPC)** combined with **AWS Transit Gateway** and **Transit Gateway Inter-Region Peering**, organizations can create a **centralized hub-and-spoke network architecture** that enables efficient connectivity between VPCs across regions.

This document provides a guide to implementing **inter-region VPC connectivity using Transit Gateway peering**.

2. Business Case Scenario

Scenario

A multinational company operates applications across multiple AWS regions:

- **US-EAST-1 Region**: Customer-facing web applications
- **US-WEST-2 Region**: Backend analytics platform
- **EU-WEST-1 Region**: Disaster recovery environment

The company must allow:

- Application servers in US-EAST-1 to access analytics services in the US-WEST-2.
- Disaster recovery replication between EU-WEST-1, US-EAST-1 and US-WEST-2.
- Secure communication without exposing services to the public internet.

Communication Scenarios

Customer Access (Public Traffic)

- Customers access the web app via **public internet**
- Traffic enters through **Internet Gateway**
- Web servers have public IPs for inbound access
- This is **NOT** using Transit Gateway

Web App to Analytics (Private Cross-Region)

- Web app sends user behavior data to analytics
- Uses private IPs (10.1.x.x → 10.2.x.x)
- Traffic through Transit Gateway peering
- Completely private, never touches internet
- Low latency, high throughput

Database Replication (Production to Disaster Recovery)

- Continuous replication from production to DR in EU-WEST-1
- Uses **private connectivity** through TGW
- Secure, encrypted over AWS backbone
- No exposure to public internet

Analytics to DR (Cross-Region Backup)

- Analytics results backed up to DR region
- Direct private connectivity via TGW mesh
- No need to route through us-east-1

Business Requirements

- Private connectivity between VPCs in different regions
- High scalability for future VPC expansion
- Centralized network management
- Reduced operational complexity

Proposed Solution

Deploy **regional Transit Gateways** and connect them using **inter-region Transit Gateway peering**.

Benefits:

- Simplifies VPC connectivity
- Reduces the need for multiple VPC peering connections
- Provides scalable hub-and-spoke architecture
- Enables encrypted traffic between regions

3. Scope

This document covers:

- Inter-region connectivity using Transit Gateway
- VPC attachment configuration
- Routing setup
- Security and monitoring considerations

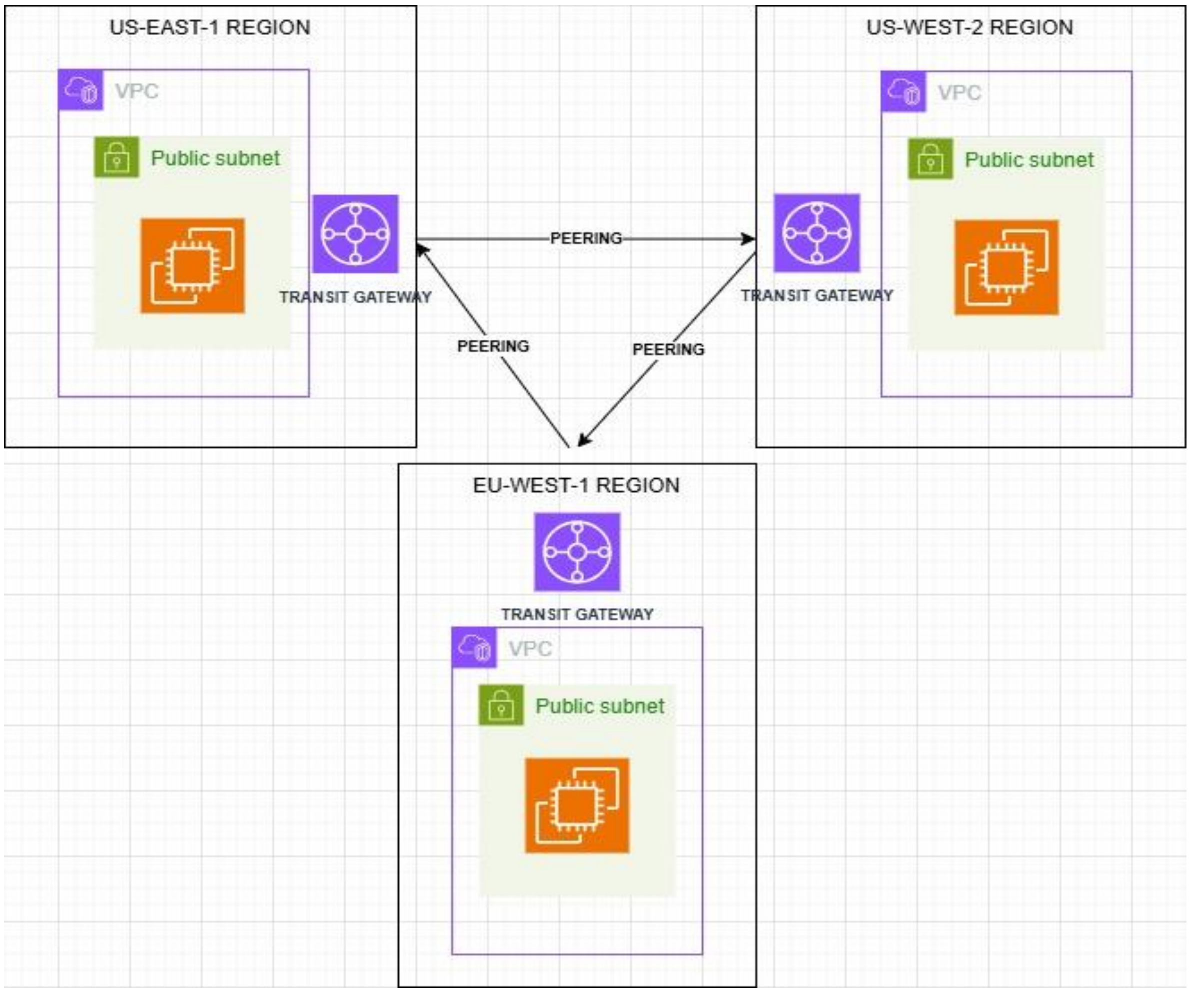
This document does **not** cover:

- Hybrid connectivity (VPN / Direct Connect)
- Multi-account AWS networking using AWS Organizations

4. Definitions and Terminology

Term	**Description**
VPC	Virtual Private Cloud that hosts AWS resources
Transit Gateway	Central hub for connecting multiple VPCs and networks
Attachment	Logical connection between TGW and VPC
TGW Peering	Connection between Transit Gateways in different regions
Route Table	Controls traffic flow between attachments

5. Architecture Overview



The architecture consists of:

- One **Transit Gateway per region**
- VPCs attached to the local Transit Gateway
- Transit Gateways connected via **inter-region peering**

6. Prerequisites

Before implementation ensure:

- AWS account access
- Appropriate IAM permissions

- AWS CLI configured with appropriate credentials
- Terraform installed

Services involved:

- AWS Identity and Access Management
- Amazon Virtual Private Cloud
- AWS Transit Gateway
- VPCs, Subnets and EC2 instances

7. Solution Architecture

Components:

1. Regional Transit Gateway (EU)
2. Regional Transit Gateway (US)
3. VPC Attachments
4. Inter-Region Transit Gateway Peering
5. Route Table Configuration

Traffic Flow:

EC2 US → VPC Route Table → TGW US
TGW US → TGW Peering → TGW EU
TGW EU → VPC Route Table → EC2 EU

8. Implementation Steps (Terraform Modules-Based)

This implementation leverages **Terraform modules** for VPCs, Transit Gateways, and EC2 instances to automate inter-region connectivity.

Workflow Overview:

1. **Deploy VPCs using vpc-module:**
    a. Instantiate the VPC module for each region.
    b. Provide inputs such as CIDR block, subnets, and tags.

2. **Create Transit Gateways using tgw-module:**
    a. Instantiate one TGW per region using the module.
    b. Provide inputs including region, and tags.
3. **Attach VPCs to Transit Gateways using tgw-module:**
    a. Use module outputs from the VPC module (VPC IDs, TGW IDs and subnets) as inputs.
4. **Deploy EC2 Instances using ec2-module:**
    a. Provision EC2 instances in each VPC for workloads.
    b. Use module inputs for instance type, AMI, key pairs, subnet, and security groups.
5. **Establish Inter-Region TGW Peering using tgw-module:**
    a. Configure peering between regional TGWs using module inputs.
    b. Handles both requester and accepter roles across regions.
6. **Configure Routing via Modules:**
    a. Use module outputs to configure TGW route tables and update VPC route tables.
    b. Routes direct traffic between VPCs through the local TGW and peering attachment.

Key Benefits:

- Full automation and repeatability for multi-region deployments
- Consistency in VPC and EC2 configurations across regions
- Simplified hub-and-spoke network management
- Easily extendable for new regions or additional workloads

For detailed module usage and variables, refer to the **README.md** accompanying the Terraform code in this GitHub Repo.

9. Security Considerations

Important controls:

- Use **Security Groups** to restrict traffic
- Implement **Network ACLs where necessary**

Inter-region Transit Gateway traffic is **encrypted by AWS automatically**.

10. Monitoring and Logging

Recommended monitoring tools:

- Amazon CloudWatch (metrics and alarms)
- AWS VPC Flow Logs (traffic inspection)
- AWS CloudTrail (audit logs)

Key metrics:

- Packet drop count
- Attachment status
- Throughput

11. Cost Considerations

Costs involved:

1. Transit Gateway hourly charge
2. Data processing charges
3. Inter-region data transfer

Cost optimization tips:

- Consolidate VPC connections

- Use centralized TGW design
- Monitor cross-region traffic volume

12. Limitations

- No support for **transitive routing through TGW peering**
- CIDR ranges must not overlap
- Additional latency compared to same-region traffic

13. Best Practices

Recommended practices:

- Use **hub-and-spoke network design**
- Separate route tables for environments (Prod / Dev)
- Implement tagging strategy

14. Troubleshooting

Common issues:

Connectivity Failure

Check:

- VPC route tables
- TGW route tables
- Security group rules

Peering Not Active

Verify:

- Peering request accepted
- Correct region configuration

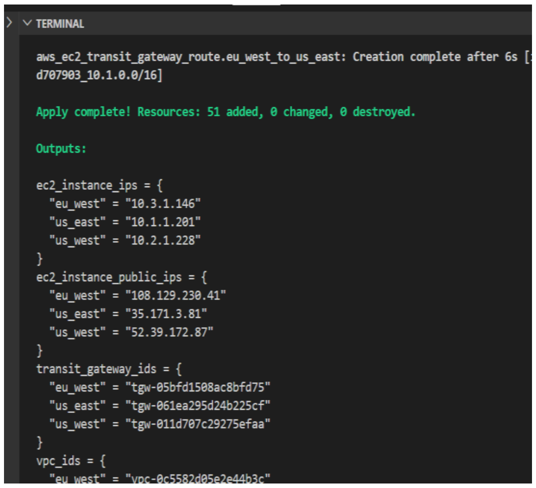
Packet Loss

Use:

- VPC Flow Logs
- CloudWatch metrics

15. Proof Of Concept

These are various outputs after Terraform apply



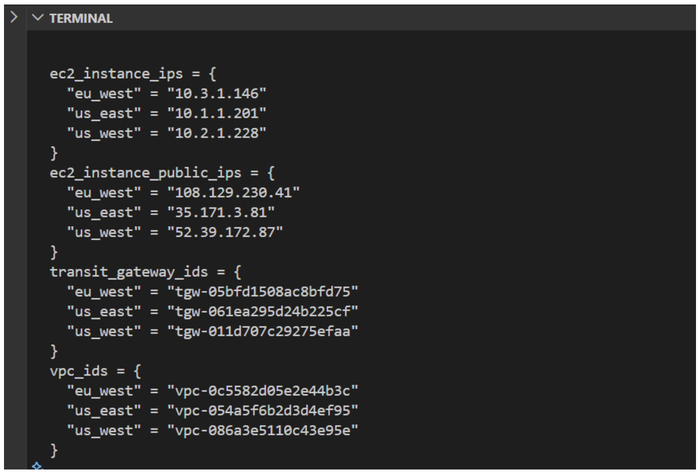
```
TERMINAL

aws_ec2_transit_gateway_route.eu_west_to_us_east: Creation complete after 6s [
d707903_10.1.0.0/16]

Apply complete! Resources: 51 added, 0 changed, 0 destroyed.

Outputs:

ec2_instance_ips = {
  "eu_west" = "10.3.1.146"
  "us_east" = "10.1.1.201"
  "us_west" = "10.2.1.228"
}
ec2_instance_public_ips = {
  "eu_west" = "108.129.230.41"
  "us_east" = "35.171.3.81"
  "us_west" = "52.39.172.87"
}
transit_gateway_ids = {
  "eu_west" = "tgw-05bfd1508ac8bfd75"
  "us_east" = "tgw-061ea295d24b225cf"
  "us_west" = "tgw-011d707c29275efaa"
}
vpc_ids = {
  "eu_west" = "vpc-0c5582d05e2e44b3c"
```

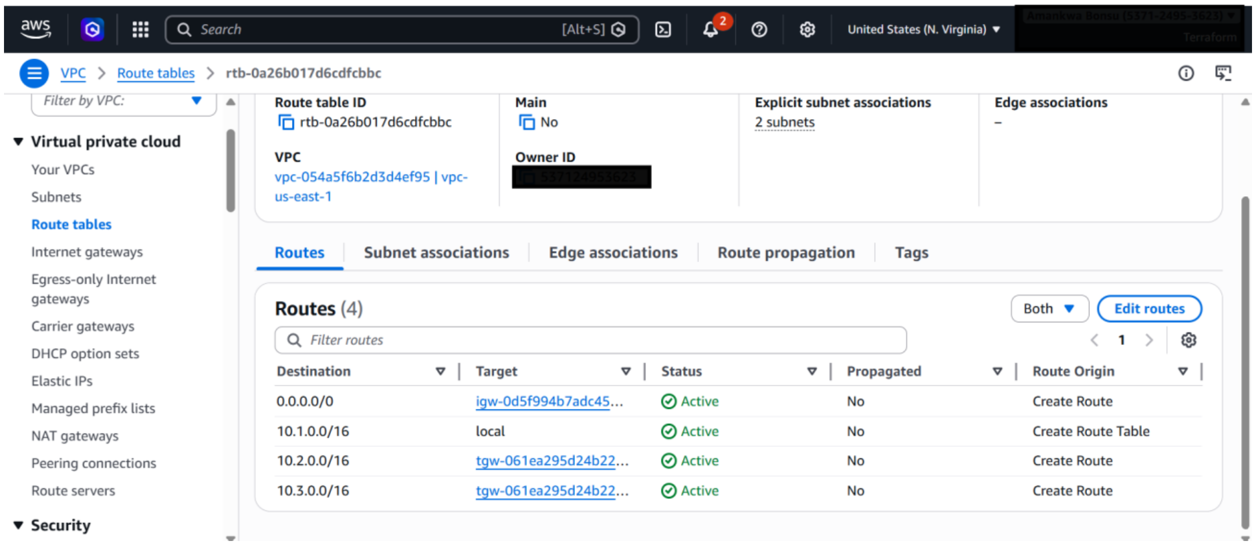


```
TERMINAL

ec2_instance_ips = {
  "eu_west" = "10.3.1.146"
  "us_east" = "10.1.1.201"
  "us_west" = "10.2.1.228"
}
ec2_instance_public_ips = {
  "eu_west" = "108.129.230.41"
  "us_east" = "35.171.3.81"
  "us_west" = "52.39.172.87"
}
transit_gateway_ids = {
  "eu_west" = "tgw-05bfd1508ac8bfd75"
  "us_east" = "tgw-061ea295d24b225cf"
  "us_west" = "tgw-011d707c29275efaa"
}
vpc_ids = {
  "eu_west" = "vpc-0c5582d05e2e44b3c"
  "us_east" = "vpc-054a5f6b2d3d4ef95"
  "us_west" = "vpc-086a3e5110c43e95e"
}
```

VPC Route Table verification:

US-EAST-1

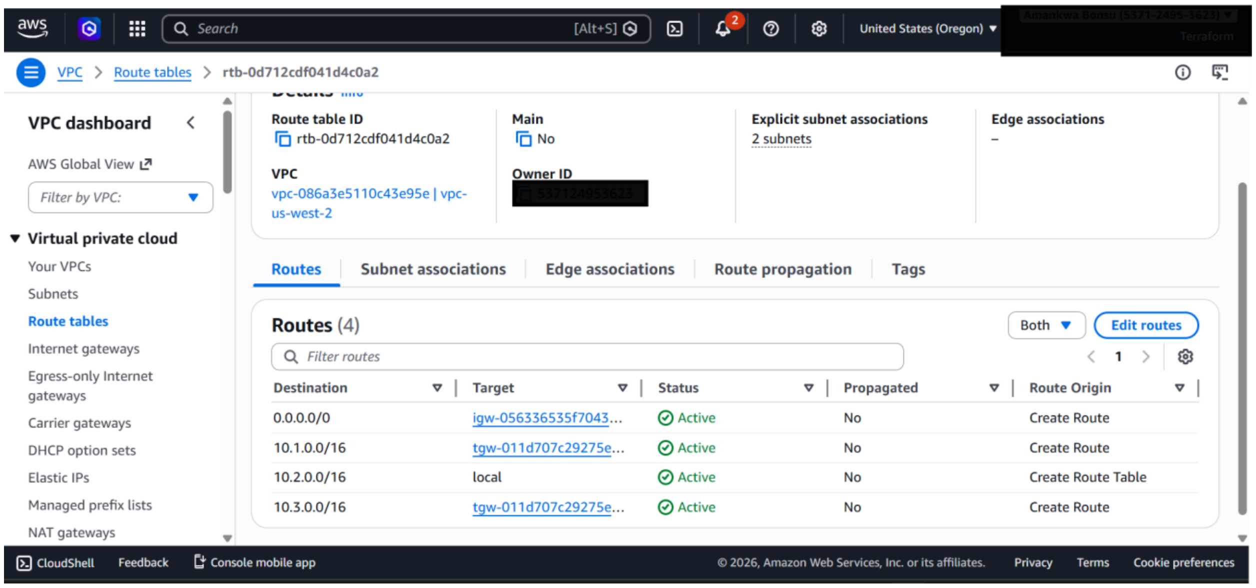


Route table ID: rtb-0a26b017d6cdfcbbc
Main: No
Explicit subnet associations: 2 subnets
Edge associations: –
VPC: vpc-054a5f6b2d3d4ef95 | vpc-us-east-1
Owner ID: [redacted]

Routes (4)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0d5f994b7adc45...	Active	No	Create Route
10.1.0.0/16	local	Active	No	Create Route Table
10.2.0.0/16	tgw-061ea295d24b22...	Active	No	Create Route
10.3.0.0/16	tgw-061ea295d24b22...	Active	No	Create Route

US-WEST-2



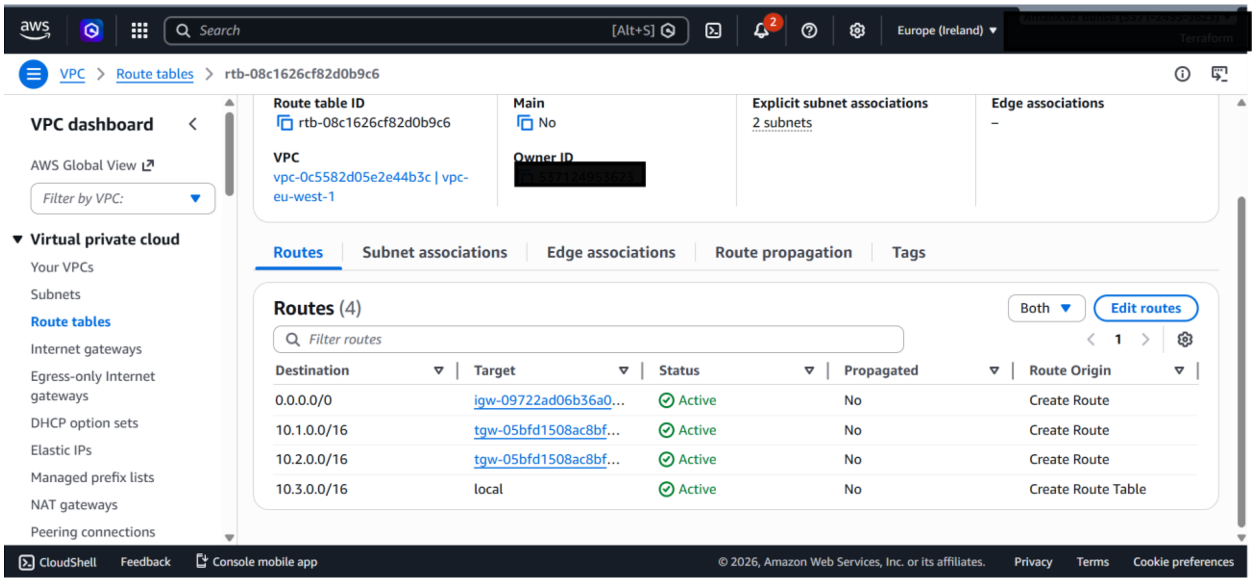
VPC > Route tables > rtb-0d712cdf041d4c0a2

Route table ID: rtb-0d712cdf041d4c0a2	Main: No	Explicit subnet associations: 2 subnets	Edge associations: –
VPC: vpc-086a3e5110c43e95e | vpc-us-west-2	Owner ID: [redacted]		

Routes (4)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-056336535f7043…	Active	No	Create Route
10.1.0.0/16	tgw-011d707c29275e…	Active	No	Create Route
10.2.0.0/16	local	Active	No	Create Route Table
10.3.0.0/16	tgw-011d707c29275e…	Active	No	Create Route

EU-WEST-1



VPC > Route tables > rtb-08c1626cf82d0b9c6

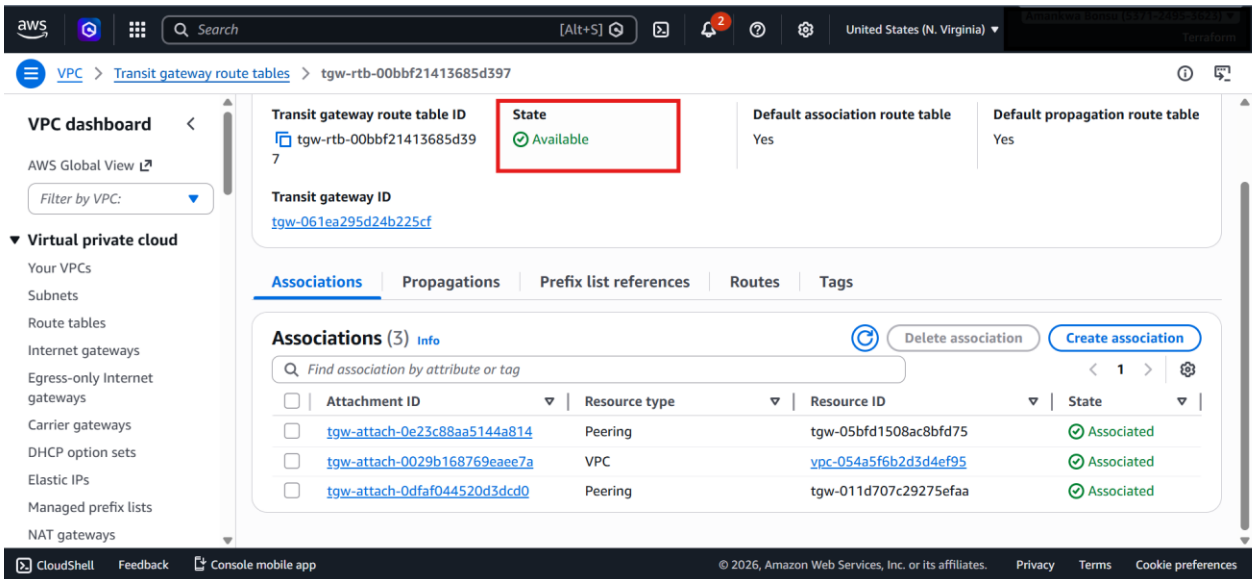
Route table ID: rtb-08c1626cf82d0b9c6	Main: No	Explicit subnet associations: 2 subnets	Edge associations: –
VPC: vpc-0c5582d05e2e44b3c | vpc-eu-west-1	Owner ID: [redacted]		

Routes (4)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-09722ad06b36a0…	Active	No	Create Route
10.1.0.0/16	tgw-05bfd1508ac8bf…	Active	No	Create Route
10.2.0.0/16	tgw-05bfd1508ac8bf…	Active	No	Create Route
10.3.0.0/16	local	Active	No	Create Route Table

Transit Gateway Peering Status and Route tables:

US-EAST-1



Search [Alt+S] | United States (N. Virginia) | [redacted]

VPC > Transit gateway route tables > tgw-rtb-00bbf21413685d397

VPC dashboard
AWS Global View
Filter by VPC:
Virtual private cloud
- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only Internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- NAT gateways

Transit gateway route table ID	State	Default association route table	Default propagation route table
tgw-rtb-00bbf21413685d397	Available	Yes	Yes

Transit gateway ID
tgw-061ea295d24b225cf

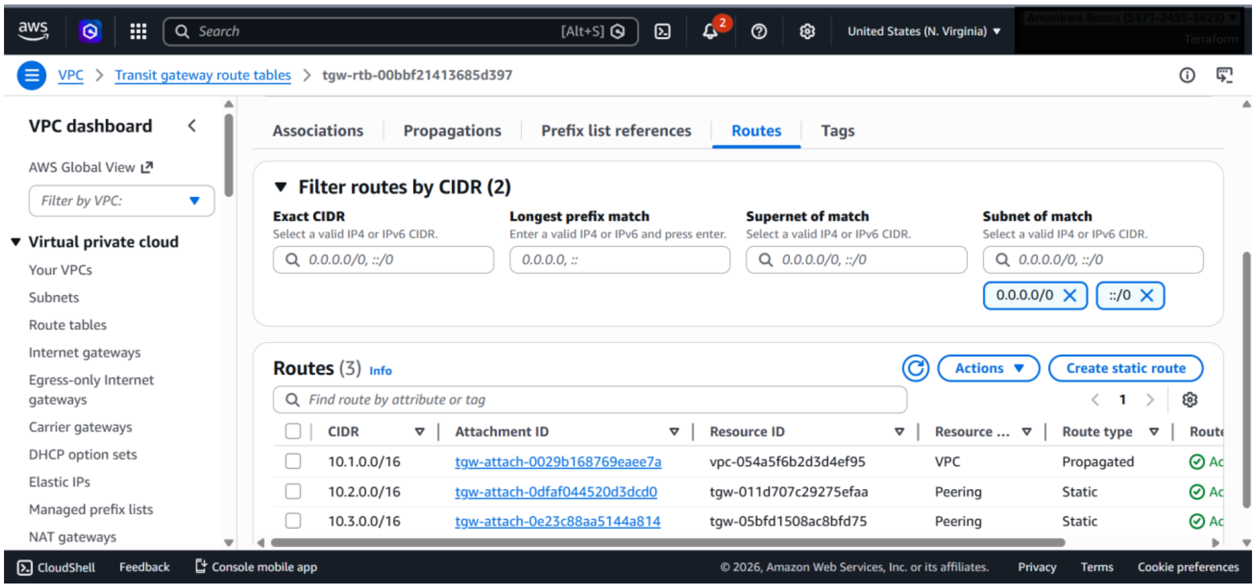
Associations | Propagations | Prefix list references | Routes | Tags

Associations (3) Info — Delete association | Create association

Find association by attribute or tag

Attachment ID	Resource type	Resource ID	State
tgw-attach-0e23c88aa5144a814	Peering	tgw-05bfd1508ac8bfd75	Associated
tgw-attach-0029b168769eaee7a	VPC	vpc-054a5f6b2d3d4ef95	Associated
tgw-attach-0dfaf044520d3dcd0	Peering	tgw-011d707c29275efaa	Associated

CloudShell Feedback Console mobile app © 2026, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Search [Alt+S] | United States (N. Virginia) | [redacted]

VPC > Transit gateway route tables > tgw-rtb-00bbf21413685d397

VPC dashboard
AWS Global View
Filter by VPC:
Virtual private cloud
- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only Internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- NAT gateways

Associations | Propagations | Prefix list references | Routes | Tags

Filter routes by CIDR (2)

Exact CIDR	Longest prefix match	Supernet of match	Subnet of match
Select a valid IP4 or IPv6 CIDR.	Enter a valid IP4 or IPv6 and press enter.	Select a valid IP4 or IPv6 CIDR.	Select a valid IP4 or IPv6 CIDR.
0.0.0.0/0, ::/0	0.0.0.0, ::	0.0.0.0/0, ::/0	0.0.0.0/0, ::/0
			0.0.0.0/0 ::/0

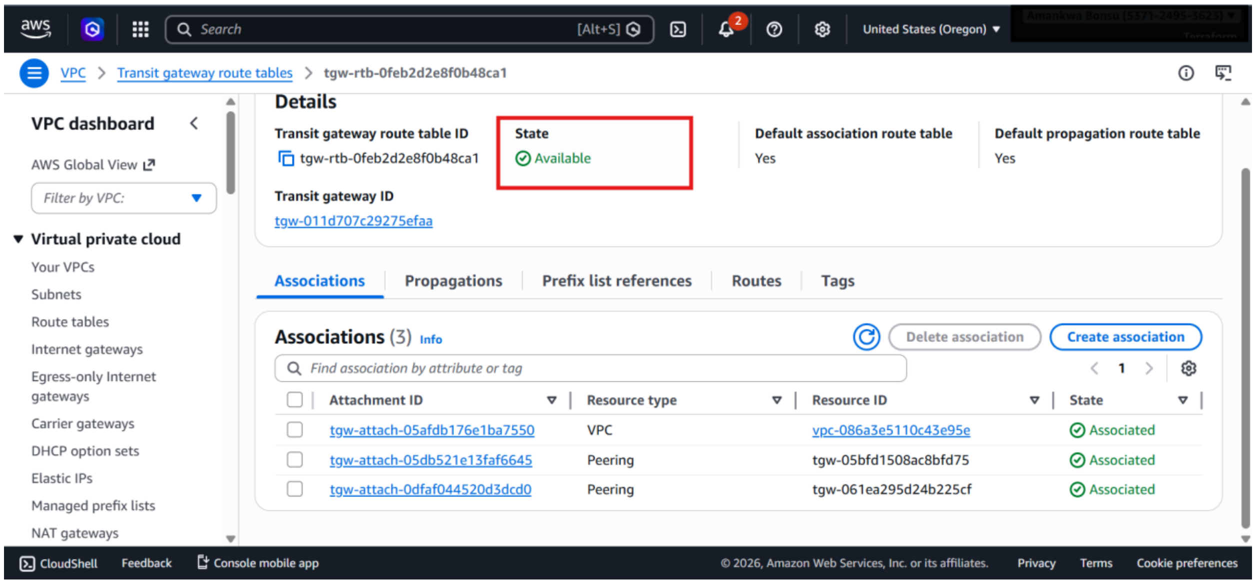
Routes (3) Info — Actions | Create static route

Find route by attribute or tag

CIDR	Attachment ID	Resource ID	Resource ...	Route type	Route
10.1.0.0/16	tgw-attach-0029b168769eaee7a	vpc-054a5f6b2d3d4ef95	VPC	Propagated	Ac
10.2.0.0/16	tgw-attach-0dfaf044520d3dcd0	tgw-011d707c29275efaa	Peering	Static	Ac
10.3.0.0/16	tgw-attach-0e23c88aa5144a814	tgw-05bfd1508ac8bfd75	Peering	Static	Ac

CloudShell Feedback Console mobile app © 2026, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

US-WEST-2



United States (Oregon) [redacted]

VPC > Transit gateway route tables > tgw-rtb-0feb2d2e8f0b48ca1

Details

Transit gateway route table ID	State	Default association route table	Default propagation route table
tgw-rtb-0feb2d2e8f0b48ca1	Available	Yes	Yes

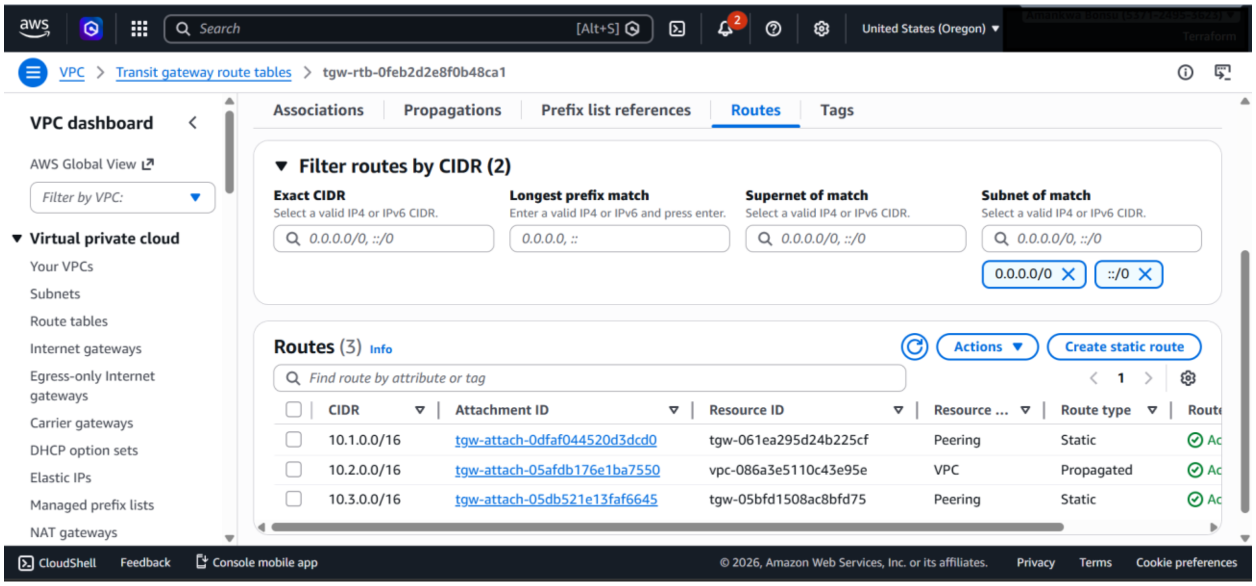
Transit gateway ID: tgw-011d707c29275efaa

Associations | Propagations | Prefix list references | Routes | Tags

Associations (3) Info

Attachment ID	Resource type	Resource ID	State
tgw-attach-05afdb176e1ba7550	VPC	vpc-086a3e5110c43e95e	Associated
tgw-attach-05db521e13faf6645	Peering	tgw-05bfd1508ac8bfd75	Associated
tgw-attach-0dfaf044520d3dcd0	Peering	tgw-061ea295d24b225cf	Associated

© 2026, Amazon Web Services, Inc. or its affiliates.



United States (Oregon) [redacted]

VPC > Transit gateway route tables > tgw-rtb-0feb2d2e8f0b48ca1

Associations | Propagations | Prefix list references | Routes | Tags

Filter routes by CIDR (2)

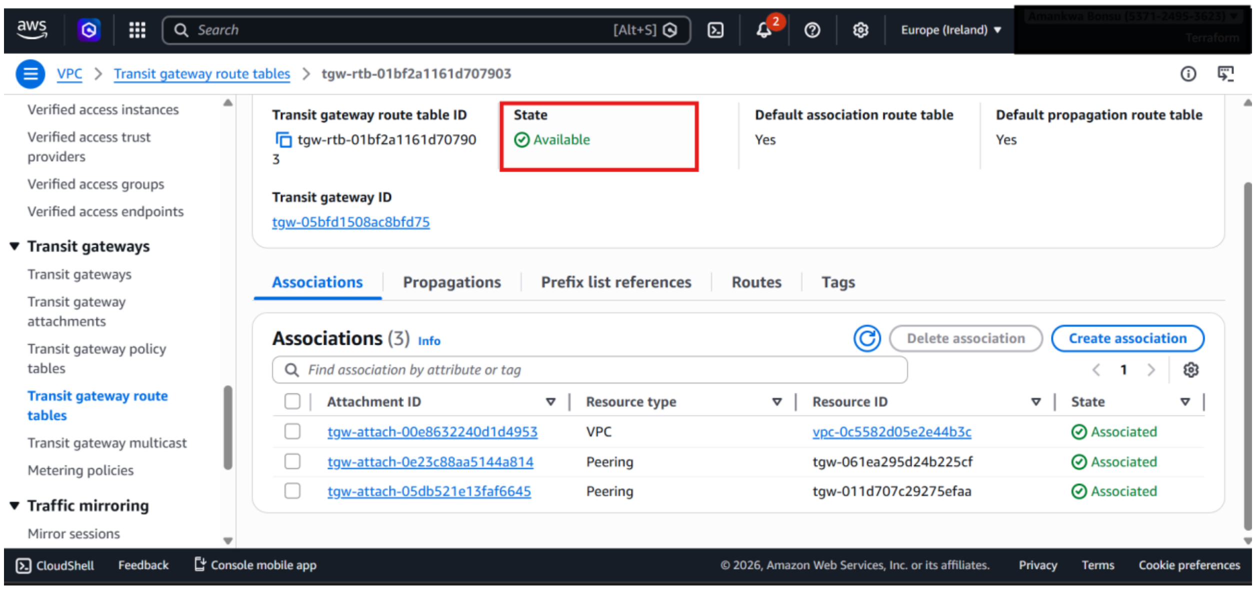
Subnet of match: 0.0.0.0/0, ::/0

Routes (3) Info

CIDR	Attachment ID	Resource ID	Resource ...	Route type	Route
10.1.0.0/16	tgw-attach-0dfaf044520d3dcd0	tgw-061ea295d24b225cf	Peering	Static	Ac
10.2.0.0/16	tgw-attach-05afdb176e1ba7550	vpc-086a3e5110c43e95e	VPC	Propagated	Ac
10.3.0.0/16	tgw-attach-05db521e13faf6645	tgw-05bfd1508ac8bfd75	Peering	Static	Ac

© 2026, Amazon Web Services, Inc. or its affiliates.

EU-WEST-1


aws
Search [Alt+S]
Europe (Ireland)
[redacted]
VPC > Transit gateway route tables > tgw-rtb-01bf2a1161d707903
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
Metering policies
Traffic mirroring
Mirror sessions
CloudShell
Feedback
Console mobile app
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Transit gateway route table ID: tgw-rtb-01bf2a1161d707903

State: Available

Default association route table: Yes

Default propagation route table: Yes

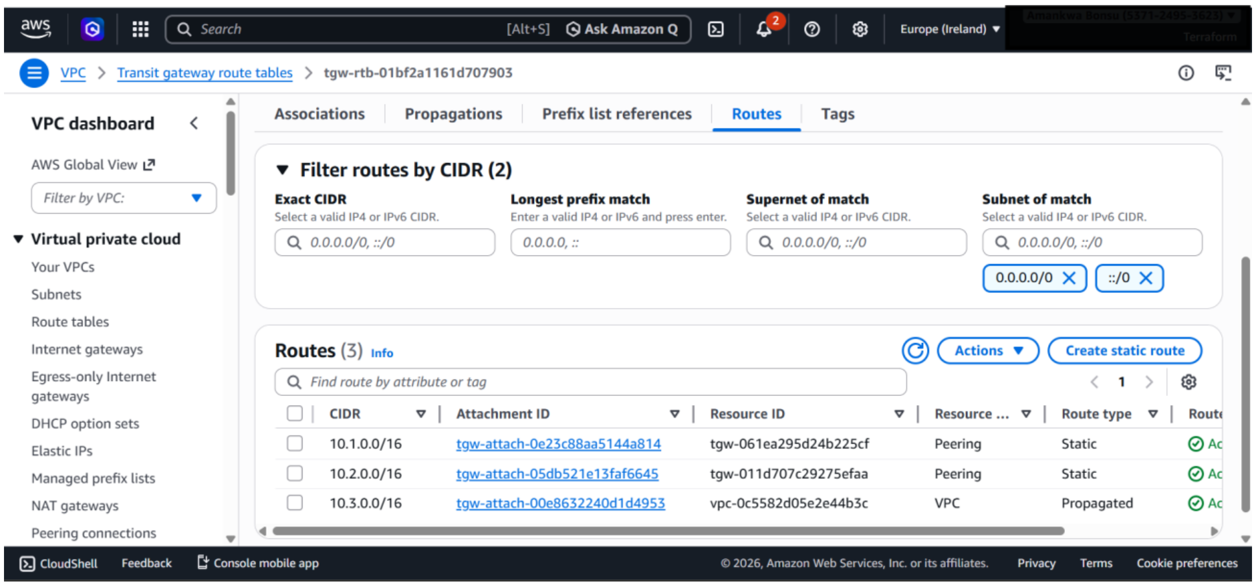
Transit gateway ID: tgw-05bfd1508ac8bfd75

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-00e8632240d1d4953	VPC	vpc-0c5582d05e2e44b3c	Associated
tgw-attach-0e23c88aa5144a814	Peering	tgw-061ea295d24b225cf	Associated
tgw-attach-05db521e13faf6645	Peering	tgw-011d707c29275efaa	Associated



Associations | Propagations | Prefix list references | Routes | Tags

Filter routes by CIDR (2)

Exact CIDR — Select a valid IP4 or IPv6 CIDR. — 0.0.0.0/0, ::/0

Longest prefix match — Enter a valid IP4 or IPv6 and press enter. — 0.0.0.0, ::

Supernet of match — Select a valid IP4 or IPv6 CIDR. — 0.0.0.0/0, ::/0

Subnet of match — Select a valid IP4 or IPv6 CIDR. — 0.0.0.0/0, ::/0 — 0.0.0.0/0 ✕ — ::/0 ✕

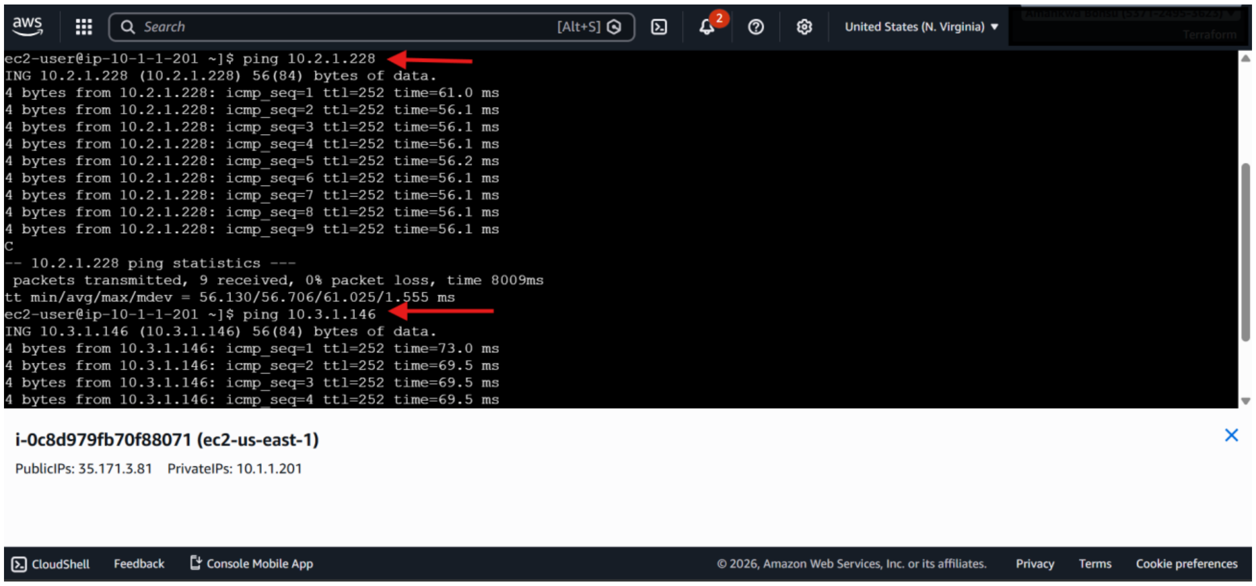
Routes (3) Info — Actions — Create static route

Find route by attribute or tag

CIDR	Attachment ID	Resource ID	Resource ...	Route type	Route
10.1.0.0/16	tgw-attach-0e23c88aa5144a814	tgw-061ea295d24b225cf	Peering	Static	Ac
10.2.0.0/16	tgw-attach-05db521e13faf6645	tgw-011d707c29275efaa	Peering	Static	Ac
10.3.0.0/16	tgw-attach-00e8632240d1d4953	vpc-0c5582d05e2e44b3c	VPC	Propagated	Ac

Connectivity Testing

Test from US-EAST-1 to US-WEST-2 and EU-WEST-1

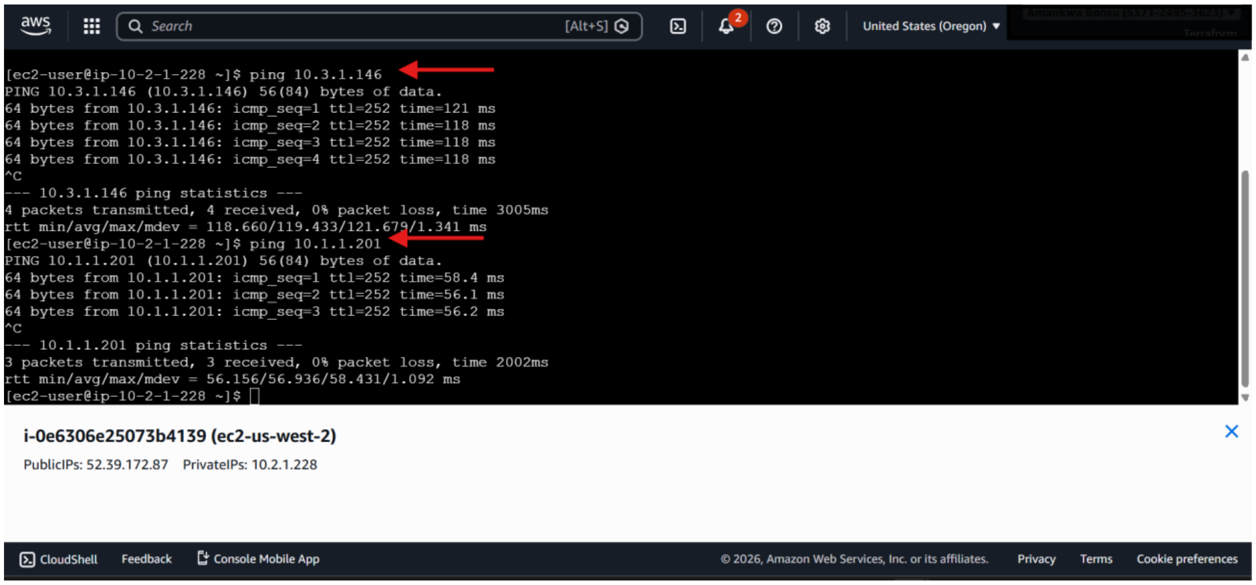


```
ec2-user@ip-10-1-1-201 ~]$ ping 10.2.1.228
ING 10.2.1.228 (10.2.1.228) 56(84) bytes of data.
4 bytes from 10.2.1.228: icmp_seq=1 ttl=252 time=61.0 ms
4 bytes from 10.2.1.228: icmp_seq=2 ttl=252 time=56.1 ms
4 bytes from 10.2.1.228: icmp_seq=3 ttl=252 time=56.1 ms
4 bytes from 10.2.1.228: icmp_seq=4 ttl=252 time=56.1 ms
4 bytes from 10.2.1.228: icmp_seq=5 ttl=252 time=56.2 ms
4 bytes from 10.2.1.228: icmp_seq=6 ttl=252 time=56.1 ms
4 bytes from 10.2.1.228: icmp_seq=7 ttl=252 time=56.1 ms
4 bytes from 10.2.1.228: icmp_seq=8 ttl=252 time=56.1 ms
4 bytes from 10.2.1.228: icmp_seq=9 ttl=252 time=56.1 ms
C
-- 10.2.1.228 ping statistics ---
 packets transmitted, 9 received, 0% packet loss, time 8009ms
tt min/avg/max/mdev = 56.130/56.706/61.025/1.555 ms
ec2-user@ip-10-1-1-201 ~]$ ping 10.3.1.146
ING 10.3.1.146 (10.3.1.146) 56(84) bytes of data.
4 bytes from 10.3.1.146: icmp_seq=1 ttl=252 time=73.0 ms
4 bytes from 10.3.1.146: icmp_seq=2 ttl=252 time=69.5 ms
4 bytes from 10.3.1.146: icmp_seq=3 ttl=252 time=69.5 ms
4 bytes from 10.3.1.146: icmp_seq=4 ttl=252 time=69.5 ms
```

i-0c8d979fb70f88071 (ec2-us-east-1)

PublicIPs: 35.171.3.81 PrivateIPs: 10.1.1.201

Test from US-WEST-2 to US-EAST-1 and EU-WEST-1

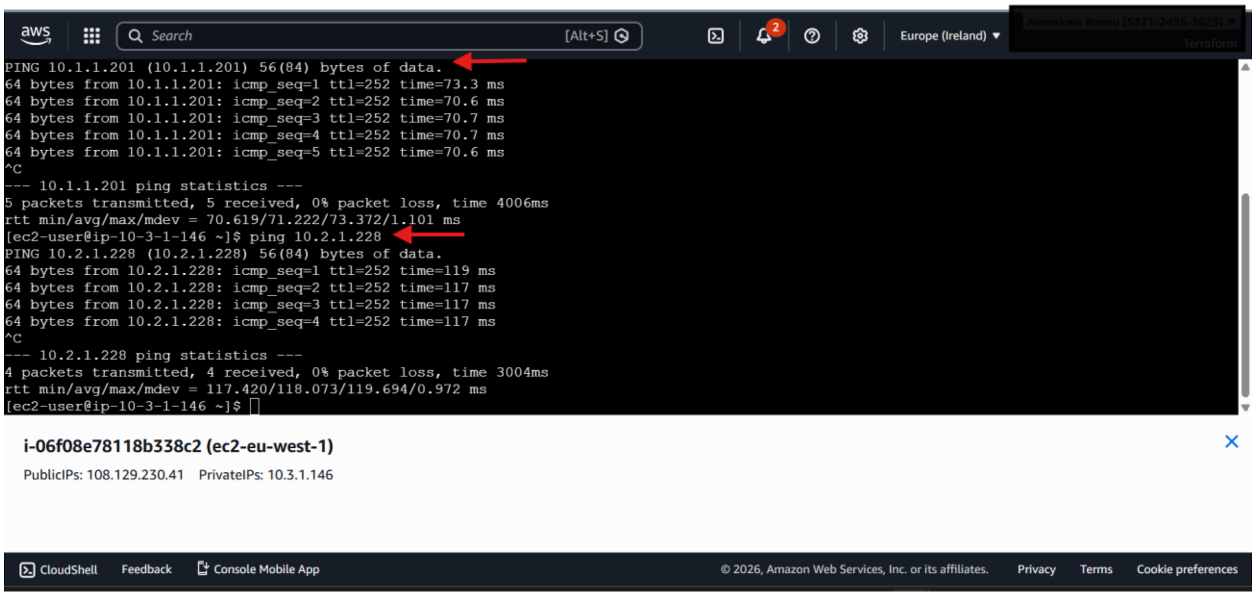


```
[ec2-user@ip-10-2-1-228 ~]$ ping 10.3.1.146
PING 10.3.1.146 (10.3.1.146) 56(84) bytes of data.
64 bytes from 10.3.1.146: icmp_seq=1 ttl=252 time=121 ms
64 bytes from 10.3.1.146: icmp_seq=2 ttl=252 time=118 ms
64 bytes from 10.3.1.146: icmp_seq=3 ttl=252 time=118 ms
64 bytes from 10.3.1.146: icmp_seq=4 ttl=252 time=118 ms
^C
--- 10.3.1.146 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 118.660/119.433/121.679/1.341 ms
[ec2-user@ip-10-2-1-228 ~]$ ping 10.1.1.201
PING 10.1.1.201 (10.1.1.201) 56(84) bytes of data.
64 bytes from 10.1.1.201: icmp_seq=1 ttl=252 time=58.4 ms
64 bytes from 10.1.1.201: icmp_seq=2 ttl=252 time=56.1 ms
64 bytes from 10.1.1.201: icmp_seq=3 ttl=252 time=56.2 ms
^C
--- 10.1.1.201 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 56.156/56.936/58.431/1.092 ms
[ec2-user@ip-10-2-1-228 ~]$
```

i-0e6306e25073b4139 (ec2-us-west-2)

PublicIPs: 52.39.172.87 PrivateIPs: 10.2.1.228

Test from EU-WEST-1 to US-EAST-1 and US-WEST-2



```
PING 10.1.1.201 (10.1.1.201) 56(84) bytes of data.
64 bytes from 10.1.1.201: icmp_seq=1 ttl=252 time=73.3 ms
64 bytes from 10.1.1.201: icmp_seq=2 ttl=252 time=70.6 ms
64 bytes from 10.1.1.201: icmp_seq=3 ttl=252 time=70.7 ms
64 bytes from 10.1.1.201: icmp_seq=4 ttl=252 time=70.7 ms
64 bytes from 10.1.1.201: icmp_seq=5 ttl=252 time=70.6 ms
^C
--- 10.1.1.201 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 70.619/71.222/73.372/1.101 ms
[ec2-user@ip-10-3-1-146 ~]$ ping 10.2.1.228
PING 10.2.1.228 (10.2.1.228) 56(84) bytes of data.
64 bytes from 10.2.1.228: icmp_seq=1 ttl=252 time=119 ms
64 bytes from 10.2.1.228: icmp_seq=2 ttl=252 time=117 ms
64 bytes from 10.2.1.228: icmp_seq=3 ttl=252 time=117 ms
64 bytes from 10.2.1.228: icmp_seq=4 ttl=252 time=117 ms
^C
--- 10.2.1.228 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 117.420/118.073/119.694/0.972 ms
[ec2-user@ip-10-3-1-146 ~]$
```

i-06f08e78118b338c2 (ec2-eu-west-1)

PublicIPs: 108.129.230.41 PrivateIPs: 10.3.1.146

Key Findings

- All Transit Gateway peering connections are in "Available" state
- VPC route tables correctly route remote CIDR blocks to Transit Gateway
- TGW route tables have static routes to peering attachments
- Full mesh connectivity achieved, all regions can communicate
- Traffic uses private IPs and AWS backbone network (not internet)

Conclusion

The multi-region Transit Gateway setup has been successfully deployed and verified. All three regions (us-east-1, us-west-2, and eu-west-1) can communicate privately through TGW peering connections. The architecture provides:

- **Private connectivity** between all regions
- **Low latency** via AWS backbone network
- **Full mesh topology** for redundancy
- **Scalable design** for future expansion