😊 Transit Gateways

Transit gateways are network transit hubs that enable you to connect multiple virtual private clouds (VPCs) and on-premises networks in a centralized manner within a cloud computing environment, such as Amazon Web Services (AWS).

Instead of establishing individual connections between every VPC and on-premises network, you can use a transit gateway to simplify network architecture and management. Transit gateways act as a central point for routing traffic between VPCs and external networks, providing better scalability, easier management, and improved traffic inspection and control. They help streamline network connectivity and reduce the complexity associated with managing multiple connections.

😄 Use cases of Transit Gateways:

Transit gateways are used in various scenarios to simplify network architecture and enhance connectivity within cloud environments. Some common use cases include:

1. **Hub-and-Spoke Architecture:** Transit gateways are often deployed in a hub-and-spoke architecture where the transit gateway acts as a central hub connecting multiple VPCs (spokes) to each other and to on-premises networks. This architecture simplifies network management by reducing the number of connections required and centralizing traffic routing.
2. **Centralized Internet Access:** Transit gateways can be used to centralize internet access for multiple VPCs. Instead of each VPC having its own internet gateway, traffic from multiple VPCs can be routed through a transit gateway to a shared internet gateway or network address translation (NAT) instance, reducing the need for redundant resources and improving security posture.
3. **Cross-Region Connectivity:** Transit gateways enable connectivity between VPCs in different AWS regions, allowing organizations to build multi-region architectures for disaster recovery, data replication, and global load balancing. By using transit gateways, inter-region communication can be managed centrally and efficiently.
4. **VPN Connectivity:** Transit gateways support VPN connections, allowing on-premises networks to securely connect to multiple VPCs over encrypted tunnels. This facilitates hybrid cloud deployments, enabling organizations to extend their existing networks into the cloud while maintaining security and compliance requirements.
5. **Integration with Transit VPCs:** Transit gateways can be integrated with Transit VPC architectures, which involve using a dedicated VPC to consolidate and manage VPN connections between on-premises networks and multiple VPCs. Transit gateways simplify the implementation and management of Transit VPCs by providing a centralized routing hub.
6. **Traffic Inspection and Control:** Transit gateways allow for centralized traffic inspection and control through the use of network security features such as network access control lists (ACLs), security groups, and third-party security appliances. This enhances

visibility and security posture by providing a single point for monitoring and enforcing network policies.

Overall, transit gateways offer a scalable and efficient solution for connecting and managing network traffic within complex cloud environments, providing organizations with greater flexibility, reliability, and control over their network infrastructure.

😄 Benefit of Transit Gateways over Peering connection:

Transit gateways offer several benefits over peering connections:

1. **Centralized Management:** Transit gateways provide a centralized hub for managing connectivity between multiple VPCs and on-premises networks. This simplifies network management tasks such as routing configuration, monitoring, and troubleshooting, compared to managing individual peering connections between every pair of VPCs.
2. **Scalability:** Transit gateways are highly scalable and can handle a large number of VPC attachments, allowing organizations to easily add or remove VPCs as their infrastructure grows or changes. In contrast, managing a large number of peering connections can become complex and cumbersome, especially in environments with numerous VPCs.
3. **Transitive Routing:** Transit gateways support transitive routing, meaning traffic can be forwarded between VPCs that are attached to the same transit gateway without the need for additional peering connections. This simplifies network architectures and avoids the need for complex routing configurations to enable communication between VPCs.
4. **Global Reach:** Transit gateways can span multiple AWS regions, enabling connectivity between VPCs in different regions without the need for region-specific peering connections. This is particularly useful for organizations with multi-region deployments, as it provides a centralized solution for inter-region communication.
5. **Integration with VPN Connections:** Transit gateways seamlessly integrate with VPN connections, allowing organizations to establish secure connections between on-premises networks and multiple VPCs using standard VPN protocols. This simplifies hybrid cloud deployments and provides a unified solution for both intra-VPC and inter-VPC connectivity.
6. **Support for Network Security Features:** Transit gateways support network security features such as network ACLs and security groups, allowing organizations to enforce security policies at the transit gateway level. This enhances security posture by providing a centralized point for traffic inspection and control.

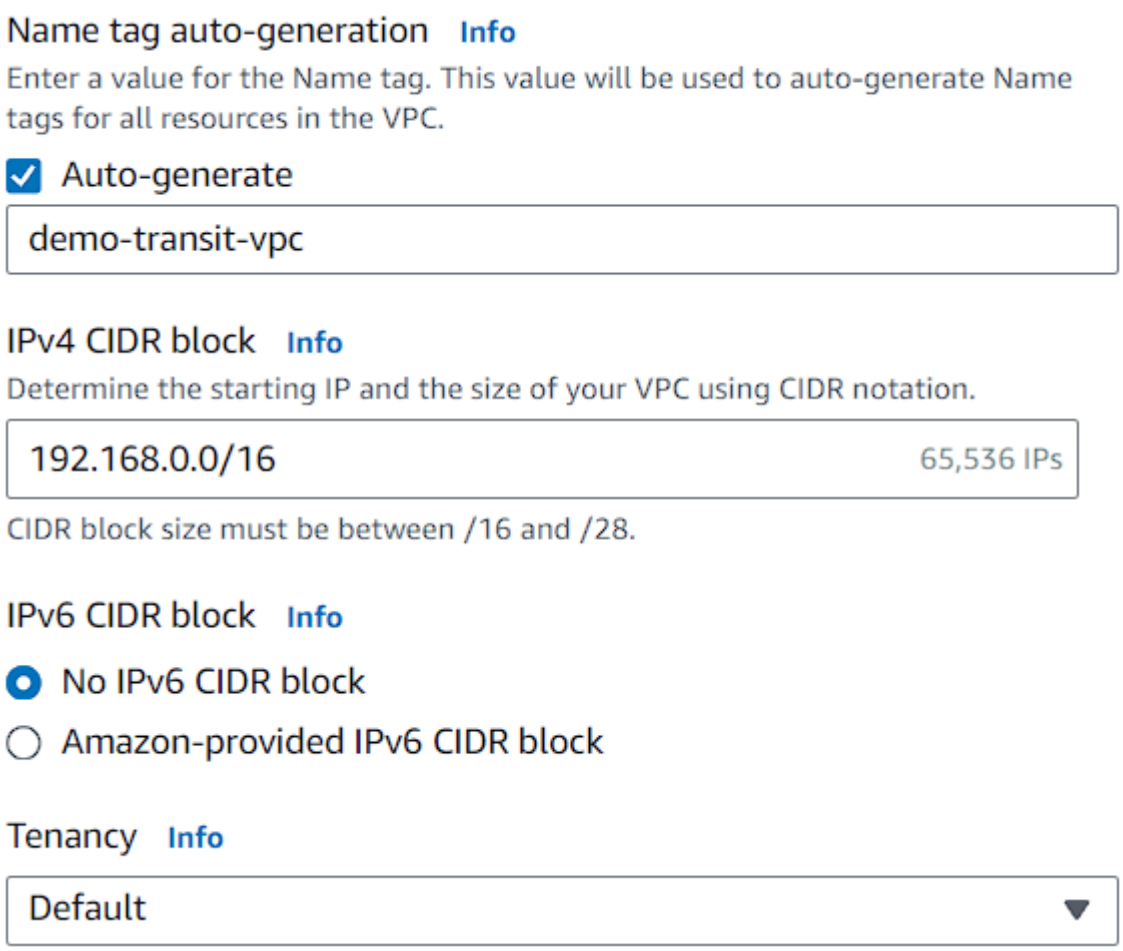
Overall, transit gateways offer a more scalable, centralized, and feature-rich solution for managing network connectivity within cloud environments compared to peering connections. They streamline network management tasks, improve scalability, and enhance security and flexibility, making them a preferred choice for many organizations deploying complex cloud architectures.

In this setup, you're configuring networking components in AWS to establish connectivity between multiple Virtual Private Clouds (VPCs) using a Transit Gateway. The end goal is to enable communication between instances in different VPCs while ensuring network isolation and security. This setup facilitates efficient and scalable network architecture in AWS, allowing for centralized management and streamlined connectivity between VPCs.

😄 To begin with the Lab:

😁 Step 1: Create VPC

1. Login to AWS Console. Then navigate to VPC.
2. There you have to create a VPC with CIDR block 192.168.0.0/16, there is no specific need to create with same CIDR block you can choose one your choice. Keep it to one availability zone only.
3. Then we really don't need a public subnet just keep one private subnet. There is no requirement of NAT gateways and VPC endpoints. Just create your VPC.



Number of Availability Zones (AZs) Info

Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.

1 | 2 | 3

▶ Customize AZs

Number of public subnets Info

The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

0 | 1

Number of private subnets Info

The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

0 | 1 | 2

▶ Customize subnets CIDR blocks

4. In less than a minute your VPC will be ready for use.

Create VPC workflow

Success

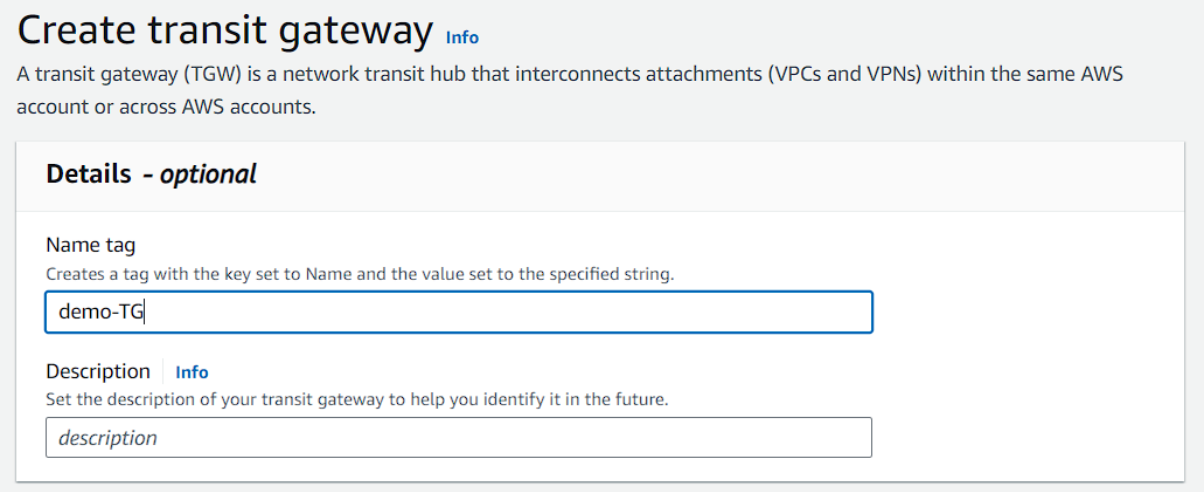
Details

- Create VPC: vpc-05205fb085af28c81
- Enable DNS hostnames
- Enable DNS resolution
- Verifying VPC creation: vpc-05205fb085af28c81
- Create subnet: subnet-0d563ed2e125addc6
- Create route table: rtb-075ab1472791bd837
- Associate route table
- Verifying route table creation

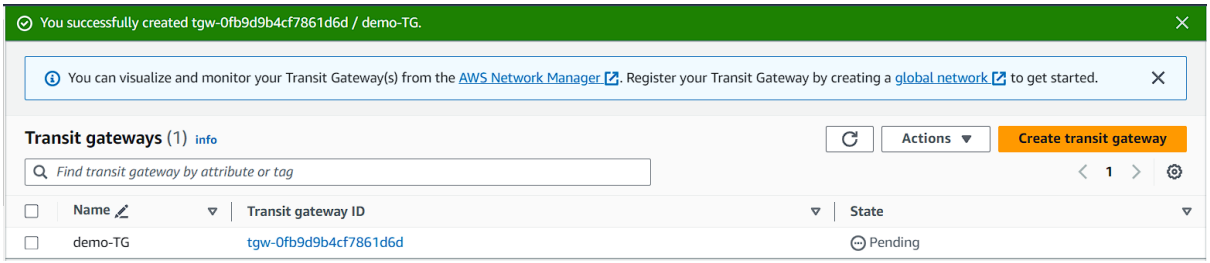
View VPC

😁 Step 2: Create Transit Gateway and Lunch EC2 instances

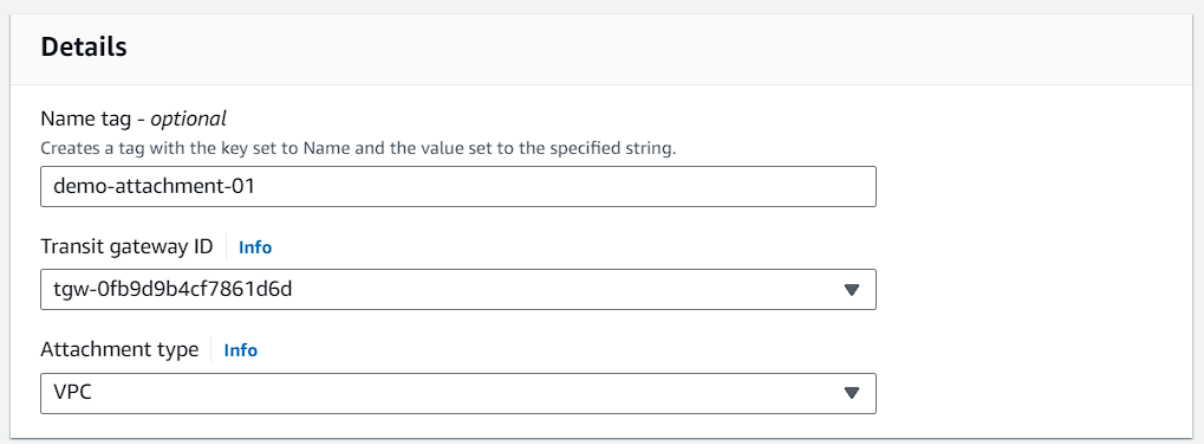
1. Now from the left pane scroll it down and navigate to Transit gateways. There you need to click on create transit gateway.
2. Then you just need to give it a name and directly create a Transit gateway.

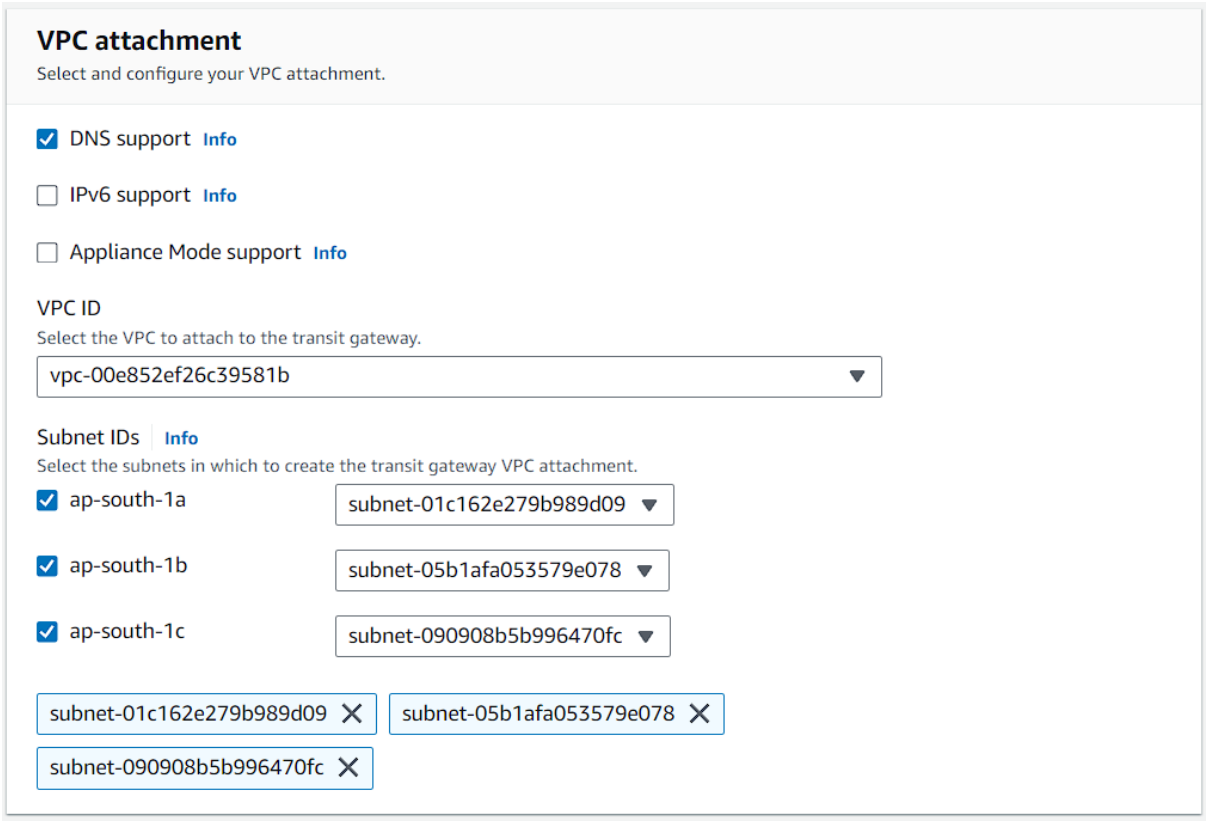


3. It generally takes some minutes for the transit gateway to be available.

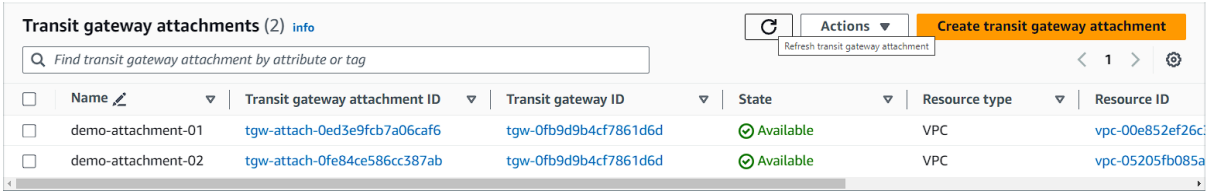


4. Once it is available then you are going to attach VPCs to your transit gateway. Currently your will be having two VPCs, first one is your default VPC and other is the one that you just created.
5. Now you have to navigate to Transit gateway attachment. There click on create attachment.
6. There you will be creating two attachments for both of the VPCs.
7. First you will be creating attachment for default VPC.
8. Similarly, you have to create attachment for other VPC.

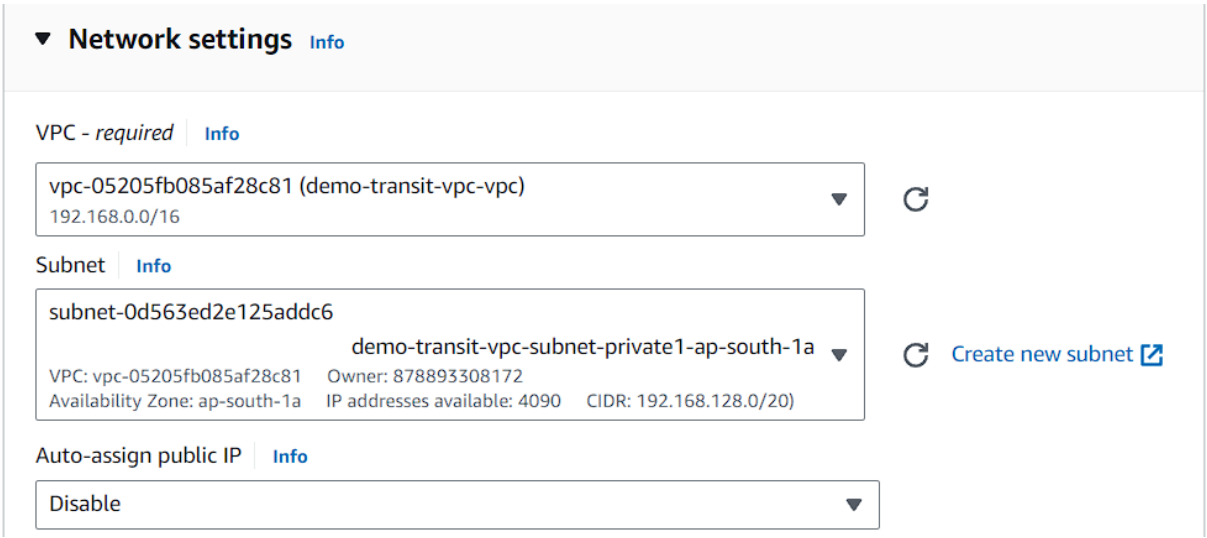




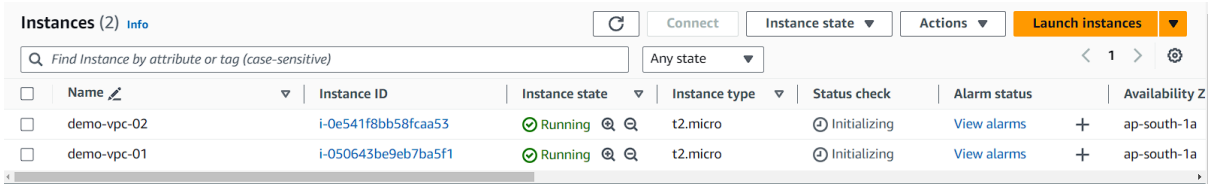
9. Once they are created, now you have to navigate to EC2 and create 2 instances. One should be in default VPC and other should be in the new VPC.



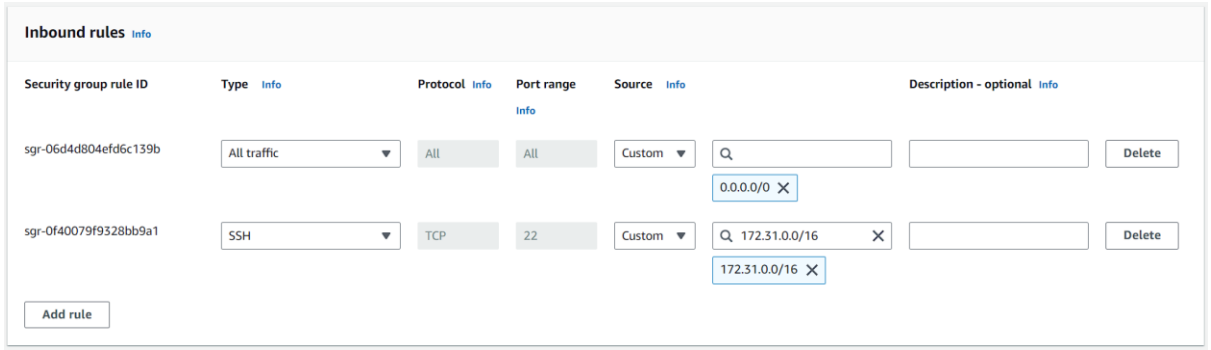
10. While launching your instance in your new VPC disable auto assign public IP.



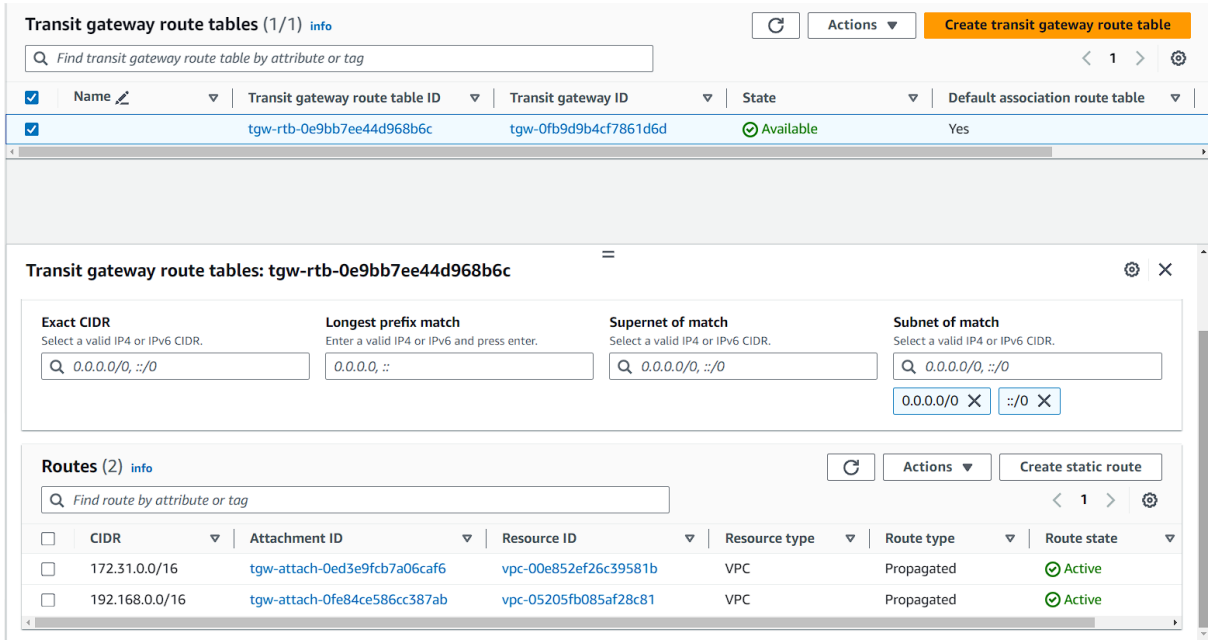
11. Once both of the instances are up and running.



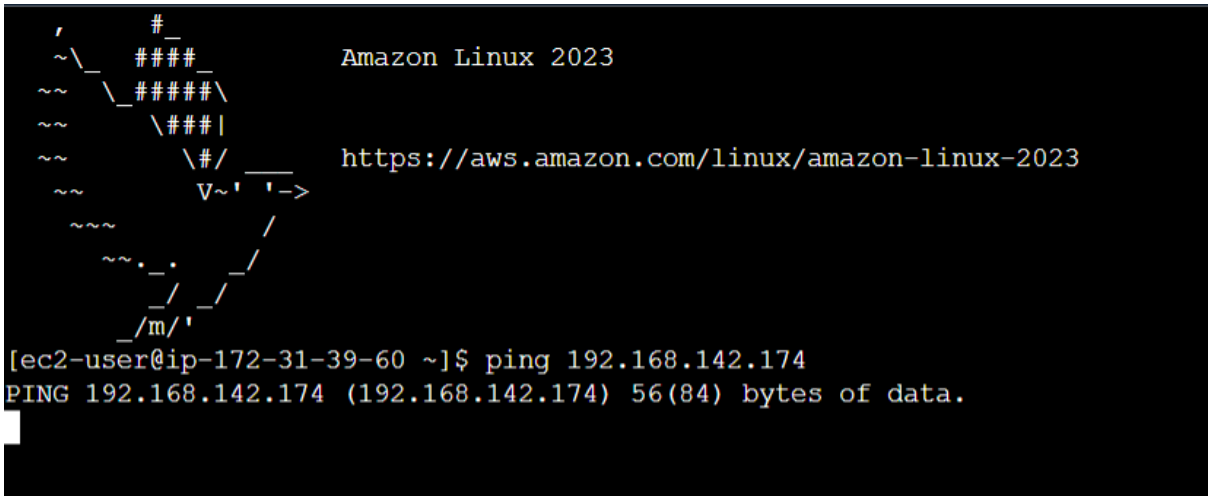
12. So, what we'll be doing from the demo-vpc-01 we will test the connectivity towards the demo-vpc-02 through ping.
13. So, if you want to test the ping connectivity, the security group of the demo-vpc-02 should allow the ICMP traffic as well. But for this demo to make things simple just allow all traffic from your default VPC. So, add the IP of your default VPC in source.



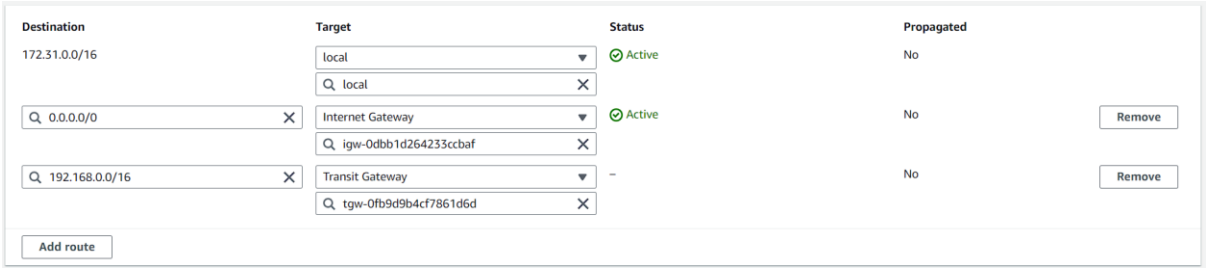
14. Once this is all done now you have to go back to VPC and navigate to Transit gateway route table. There you have to select your route table and then go to routes.
15. There you have to ensure that two route entries should be present.



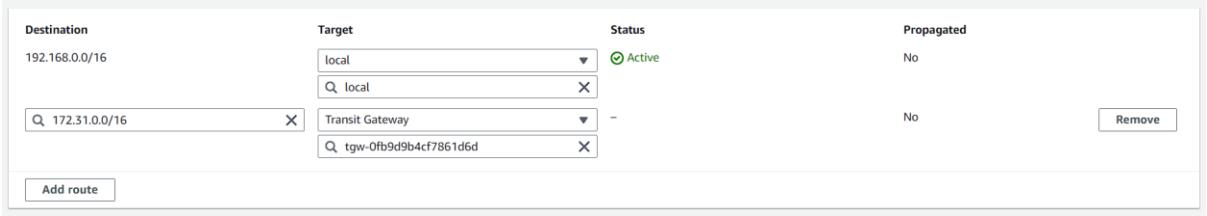
16. Now you have to connect with your demo-vpc-01 instance and once it is connected try to ping the other instance.
17. There you will see that things are not working at this stage not because of the transit gateways but because of the VPC route table.



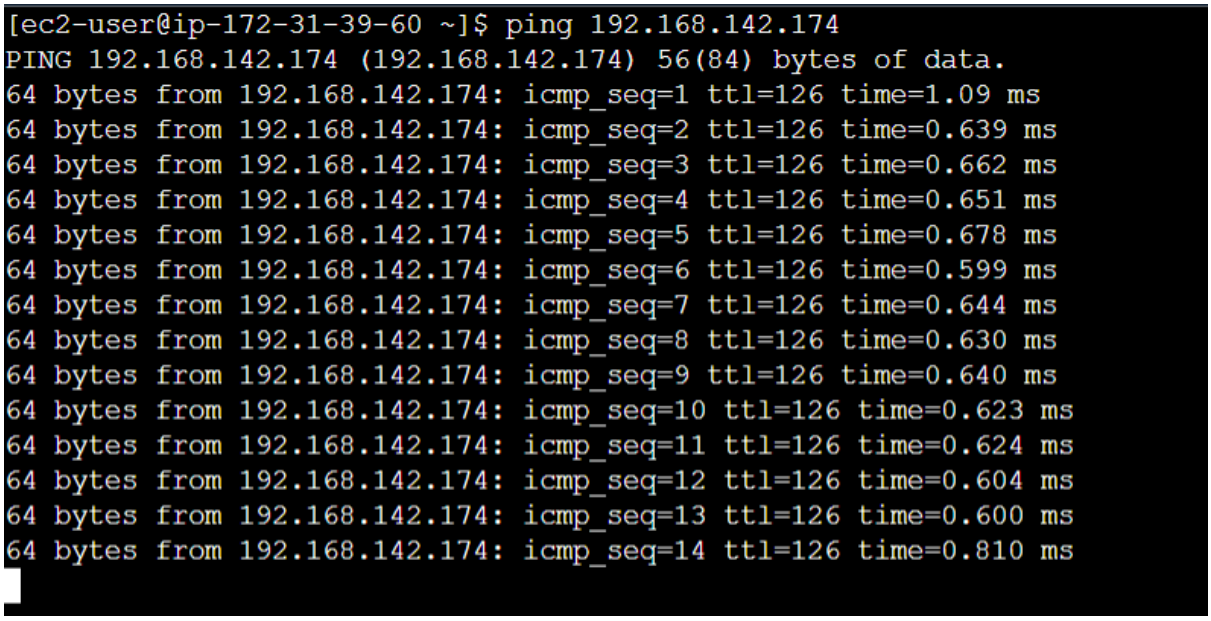
18. Now you have to navigate to VPC and open route table of default VPC and in the routes you have to add the route for other VPC and the target is your transit gateway.



19. Then you have to go to the route table your new VPC and add route for default VPC and the target is transit gateway.



20. Once the setup is done you will see that things are working fine.


[ec2-user@ip-172-31-39-60 ~]$ ping 192.168.142.174
PING 192.168.142.174 (192.168.142.174) 56(84) bytes of data.
64 bytes from 192.168.142.174: icmp_seq=1 ttl=126 time=1.09 ms
64 bytes from 192.168.142.174: icmp_seq=2 ttl=126 time=0.639 ms
64 bytes from 192.168.142.174: icmp_seq=3 ttl=126 time=0.662 ms
64 bytes from 192.168.142.174: icmp_seq=4 ttl=126 time=0.651 ms
64 bytes from 192.168.142.174: icmp_seq=5 ttl=126 time=0.678 ms
64 bytes from 192.168.142.174: icmp_seq=6 ttl=126 time=0.599 ms
64 bytes from 192.168.142.174: icmp_seq=7 ttl=126 time=0.644 ms
64 bytes from 192.168.142.174: icmp_seq=8 ttl=126 time=0.630 ms
64 bytes from 192.168.142.174: icmp_seq=9 ttl=126 time=0.640 ms
64 bytes from 192.168.142.174: icmp_seq=10 ttl=126 time=0.623 ms
64 bytes from 192.168.142.174: icmp_seq=11 ttl=126 time=0.624 ms
64 bytes from 192.168.142.174: icmp_seq=12 ttl=126 time=0.604 ms
64 bytes from 192.168.142.174: icmp_seq=13 ttl=126 time=0.600 ms
64 bytes from 192.168.142.174: icmp_seq=14 ttl=126 time=0.810 ms