Transit Gateway Segmentation through Route Tables Workshop

Contents

Overview

Problem: Although the Transit Gateway (TGW) has been out since 2018, it is still not a well-understood service that requires 300 level networking knowledge. In addition, those that are more focused on security instead of networking may not understand the full capabilities of TGW in a security context. While TGW is in the arena of networking, it still plays a significant role in securing the customer's network traffic and data. Many organizations are required to segment network traffic for compliance or regulatory needs. The network segmentation requirement often arises from the Data Classification area of the Directive Component within the AWS Cloud Adoption Framework (CAF) Security Perspective. Often, we have seen this result in a customer going with a multiple TGW model. While this solution may be viable, it adds additional overhead having to manage multiple TGWs. Some customers have not yet leveraged TGW due to lack of understanding, and instead are leveraging VPC peering. Again, while this solution is still viable, it is difficult to manage centrally.

Security professionals are required to understand the full capability of the TGW to not only design the network segmentation properly but also to validate that the necessary preventive controls are in place at the network layer.

What are TGW routing tables, associations, and propagations? How and when should they be used? How do they provide the preventive controls within the infrastructure while keeping-it-simple?

Solution: This workshop will allow AWS technical staff to create a testing environment that can be deployed quickly via CloudFormation inside their own Isengard accounts. Once deployed, users will follow detailed step-by-step instructions to see the results of the TGW components and configuration. Finally, users can apply the knowledge they learned to add preventive controls to protect the underlying infrastructure and ultimately the data that traverses over them.

Audience

The 1st phase of this workshop is only for Amazon employees and specifically AWS technical staff familiar with using Isengard accounts. The audience should be familiar with VPC, subnets, and VPC Route Tables, as well as basic understanding of IPv4 routing (IP address, subnet mask, default gateway).

Document Notations

Throughout the document, the following notations are used

Table 1: Document Notations

Notation	Description	Example
Step 1 > Step 2 > Step 3	Provide a series of steps in a single sentence to provide a comprehensive instruction.	Go to VPC > Transit Gateway Attachments > Create Transit Gateway Attachment
<some instruction>	Replace the text inside <> with the value that pertains to your environment. The example on the right should replace <See Table above> with the table that precedes the statement.	Attachment Name: <See Table above>

<>	Blank value that should be filled out. Example on the right is a table entry where you need to replace <> with the Account number of your Upper environment used in the workshop.	Upper Environment, <>

Prerequisites

1) We will need 3 AWS accounts (be aware of 5 VPC limit per account).
   a) It is recommended to use "clean" or new accounts.
   b) One account will be used to house "**Upper**" environments while one account will be used to house "**Lower**", and one account will be "**Shared**"
   c) The workshop will only work in "us-east-1" region within commercial in this version.
2) We will need EC2 key available in all accounts. We will not need it for access but it is required to deploy the test EC2 instance. Create a unique key in each account.
3) (Optional but recommended). We will leverage 3 AWS accounts so being able to have all 3 accounts at the same time will greatly reduce the time required to switch between accounts – you can use Firefox add on to make this possible (https://w.amazon.com/index.php/AwsConsoleFirefoxAddon)
4) 2x CloudFormation (CFN) templates – provided in the same location as this doc.
   a) Lab-single-subnet.yml
   b) Create-TGW-yml

Workshop Scenario

The workshop will be broken into multiple scenarios, and we will provide a diagram for each scenario to help the audience visualize the overall process.

- Scenario #1 - Initial set up with no Transit Gateway
- Scenario #2 - Transit Gateway with only association (no routing)
- Scenario #3 - Adding Propagations only within Routing Tables
- Scenario #4 - Adding Propagations to Shared VPCs
- Scenario #5 - Remove Propagations to other VPC's (only shared)

Architecture Diagrams

The following diagrams provide the description of each scenario.

Scenario #1 - Initial set up with no Transit Gateway

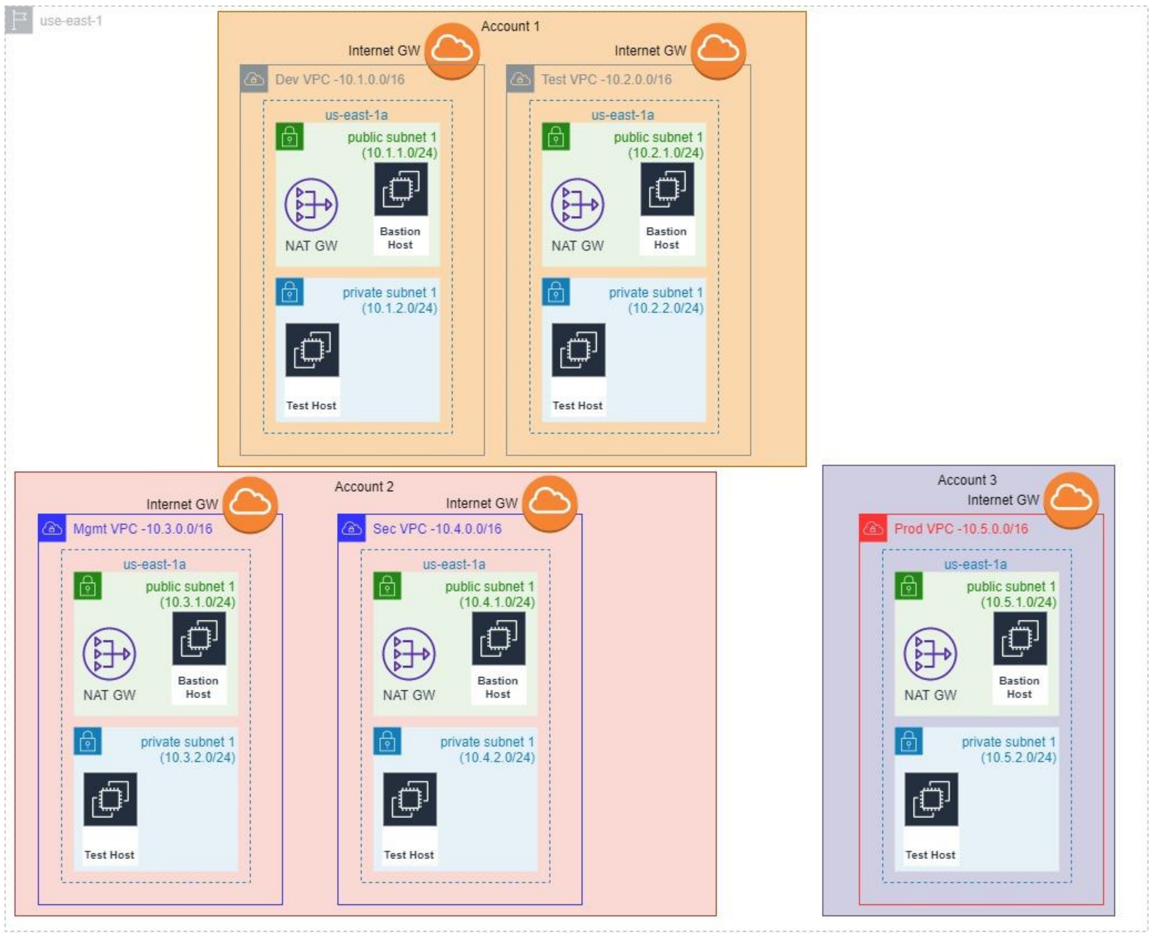


Figure 1: Initial Setup with no TGW

Once we deploy the CFN templates to create our 5x VPC stacks, the above diagram describes what we will have. Notice that each VPC is an island and no connectivity will exists between the environments.

Our connectivity testing will be done from the Private Subnet portion of the VPC (see Test Host in each VPC above). The following table provides Private Subnet CIDR for each environment.

Table 2: Environment VPC and Subnet CIDR

Environment	VPC CIDR	Private Subnet CIDR	Test EC2 Instance IP
Dev	10.1.0.0/16	10.1.2.0/24	10.1.2.11
Test	10.2.0.0/16	10.2.2.0/24	10.2.2.11
Management	10.3.0.0/16	10.3.2.0/24	10.3.2.11
Security	10.4.0.0/16	10.4.2.0/24	10.4.2.11
Prod	10.5.0.0/16	10.5.2.0/24	10.5.2.11

Scenario #2 - Transit Gateway with only association (no routing)

After we deploy the initial VPC's along with the various subnets and other components, the next CFN template will deploy the Transit Gateway (managed in Shared account) along with 3x TGW Route Tables.

We will then manually associate each VPC to the respective TGW Route Tables. There will not be any propagations or routes defined, therefore, in this scenario, all VPC's are still on their own island.

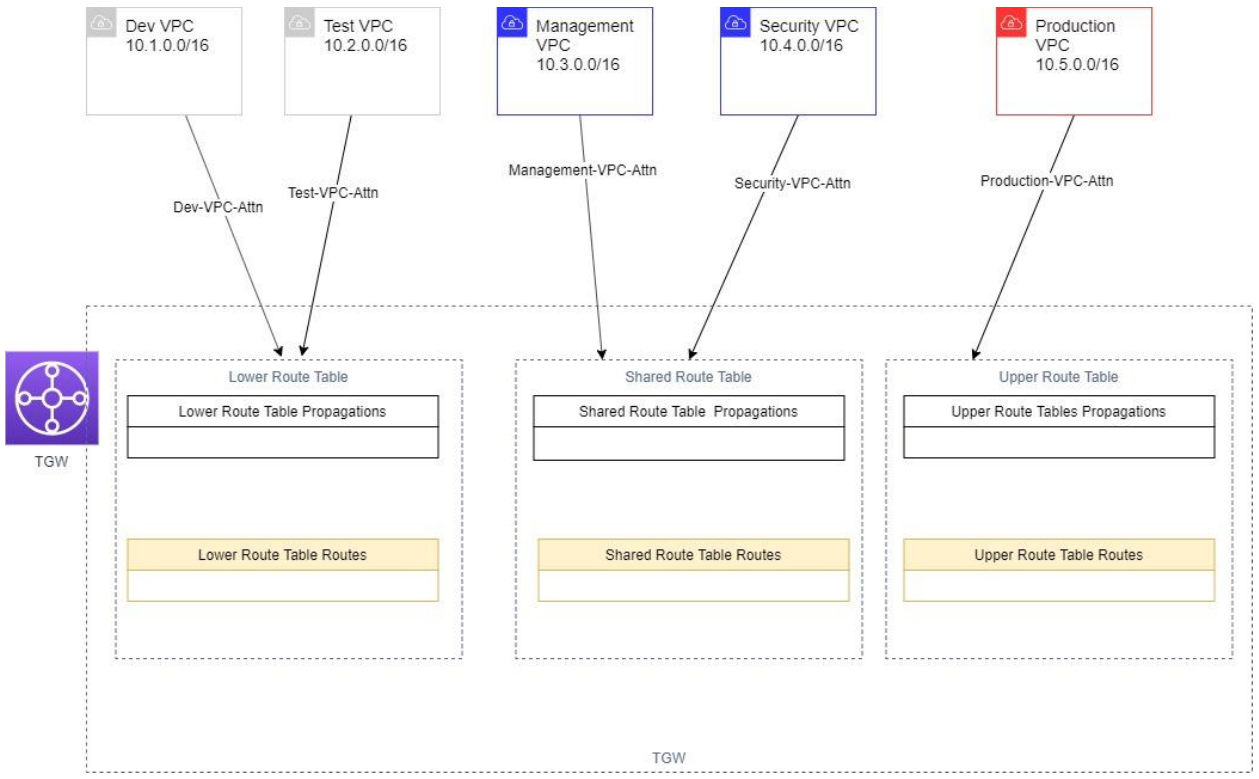


Figure 2: Add TGW and Association but No Routing

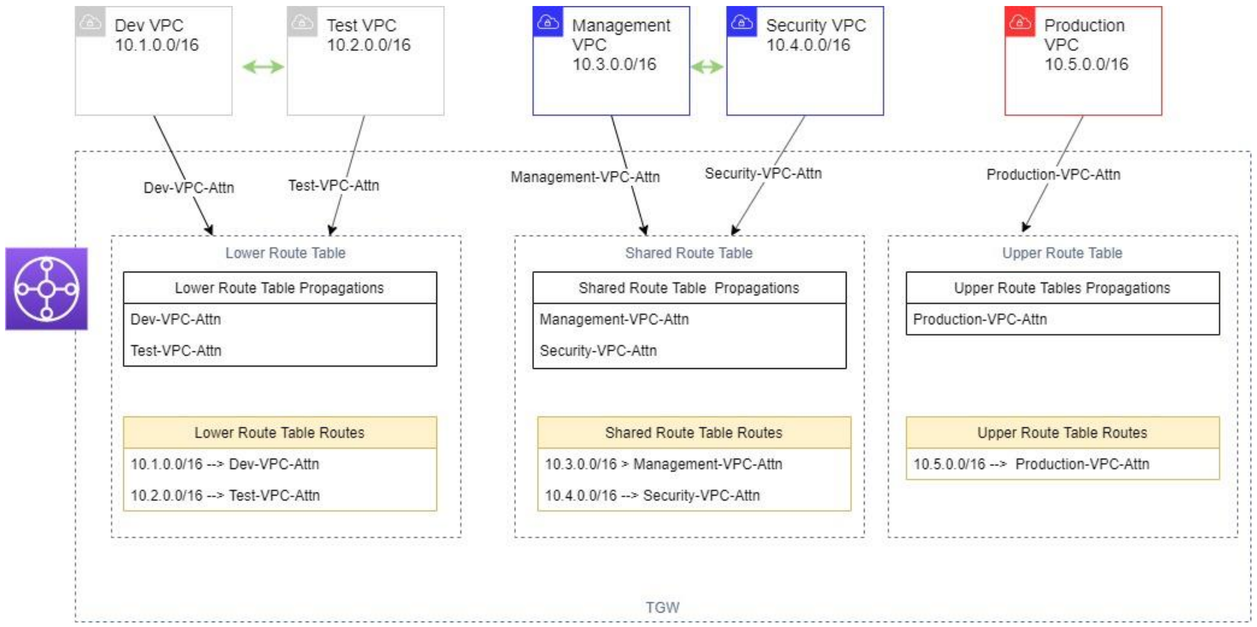
We will be using the following Transit Gateway terms in this workshop. They will be explained more during the detailed steps but here are the official definitions from the AWS website.

https://docs.aws.amazon.com/vpc/latest/tgw/what-is-transit-gateway.html

- **Attachment** — You can attach a VPC, an AWS Direct Connect gateway, or a VPN connection to a transit gateway.
- **Transit gateway route table** — A transit gateway has a default route table and can optionally have additional route tables. A route table includes dynamic and static routes that decide the next hop based on the destination IP address of the packet. The target of these routes could be a VPC or a VPN connection. By default, the VPCs and VPN connections that you attach to a transit gateway are associated with the default transit gateway route table.
- **Associations** — Each attachment is associated with *exactly one route table*. Each route table can be associated with zero to many attachments.
- **Route propagation** — A VPC or VPN connection can dynamically propagate routes to a transit gateway route table. With a VPC, you must create static routes to send traffic to the transit gateway. With a VPN connection, routes are propagated from the transit gateway to your on-premises router using Border Gateway Protocol (BGP).

Scenario #3 - Adding Propagations only within Routing Tables

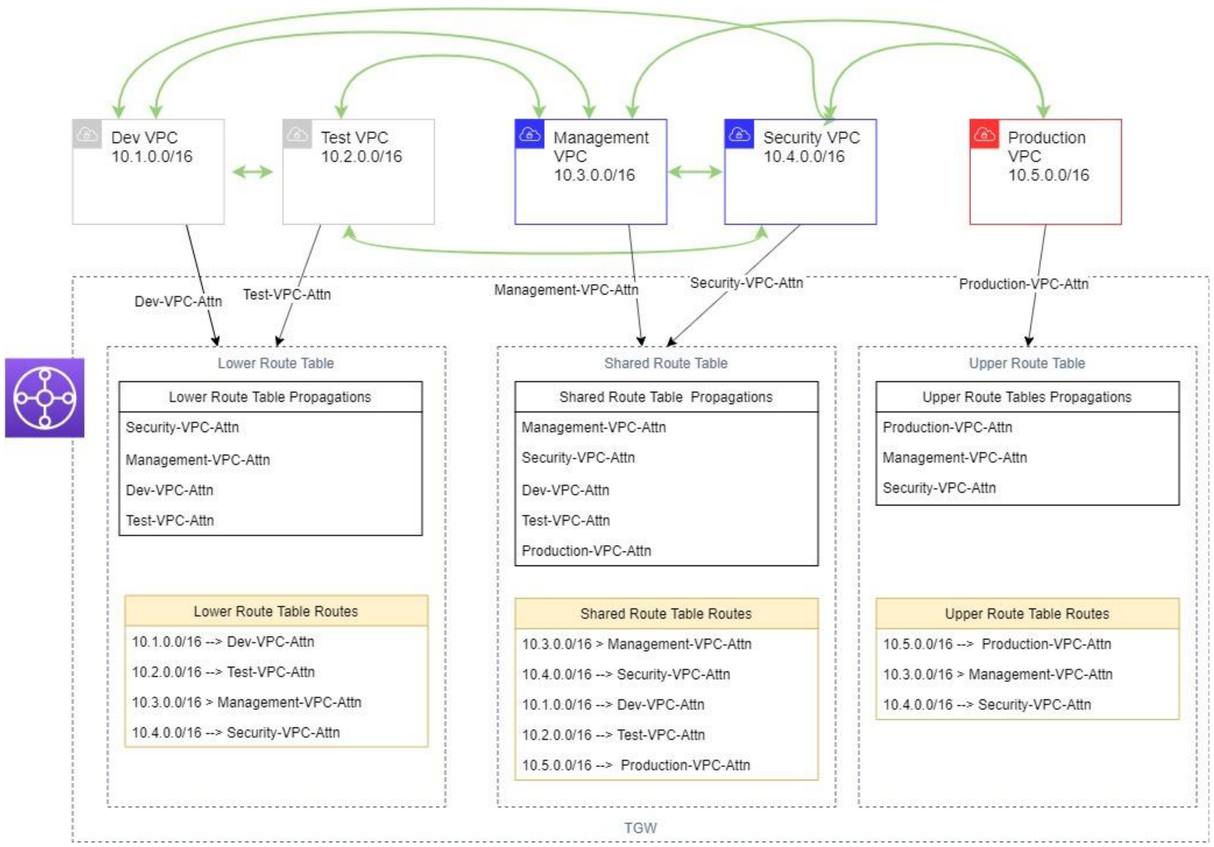
Once we have the TGW associations complete, we will add Propagations to the respective TGW Route Tables. Once we add the Propagations, the respective VPC routes (VPC CIDR) will be added the TGW Route Table. See below for visual representation.


Dev VPC 10.1.0.0/16
Test VPC 10.2.0.0/16
Management VPC 10.3.0.0/16
Security VPC 10.4.0.0/16
Production VPC 10.5.0.0/16
Dev-VPC-Attn
Test-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Production-VPC-Attn
Lower Route Table
Lower Route Table Propagations
Dev-VPC-Attn
Test-VPC-Attn
Lower Route Table Routes
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
Shared Route Table
Shared Route Table Propagations
Management-VPC-Attn
Security-VPC-Attn
Shared Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
Upper Route Table
Upper Route Tables Propagations
Production-VPC-Attn
Upper Route Table Routes
10.5.0.0/16 --> Production-VPC-Attn
TGW
Figure 3: Add Propagations in the Same Routing Tales

The above diagrams shows (green) that communication is possible within the same TGW Route Table. However, the communication across the TGW Route tables is not allowed.

Scenario #4 - Adding Propagations to Shared VPCs

We add additional Propagations so that each VPC's (dev, test and production) are allowed to communicate to the shared VPC's (Management and Security). By adding the Propagations, the TGW route tables dynamically updated to provide routes to those VPC's.



Dev VPC 10.1.0.0/16
Test VPC 10.2.0.0/16
Management VPC 10.3.0.0/16
Security VPC 10.4.0.0/16
Production VPC 10.5.0.0/16
Dev-VPC-Attn
Test-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Production-VPC-Attn
Lower Route Table
Lower Route Table Propagations
Security-VPC-Attn
Management-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Lower Route Table Routes
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
Shared Route Table
Shared Route Table Propagations
Management-VPC-Attn
Security-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Production-VPC-Attn
Shared Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.5.0.0/16 --> Production-VPC-Attn
Upper Route Table
Upper Route Tables Propagations
Production-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Upper Route Table Routes
10.5.0.0/16 --> Production-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
TGW

Figure 4: Add Propagations to Shared VPC's

Scenario #5 - Remove Propagations to other VPC's (only shared)

This scenario, we will isolate the VPC's even within the same TGW Route Table so that while they are allowed to communicate to the Shared VPC's, they are NOT able to communicate with even the VPC's associated with the same TGW Route table (e.g. Dev <- -> Test).

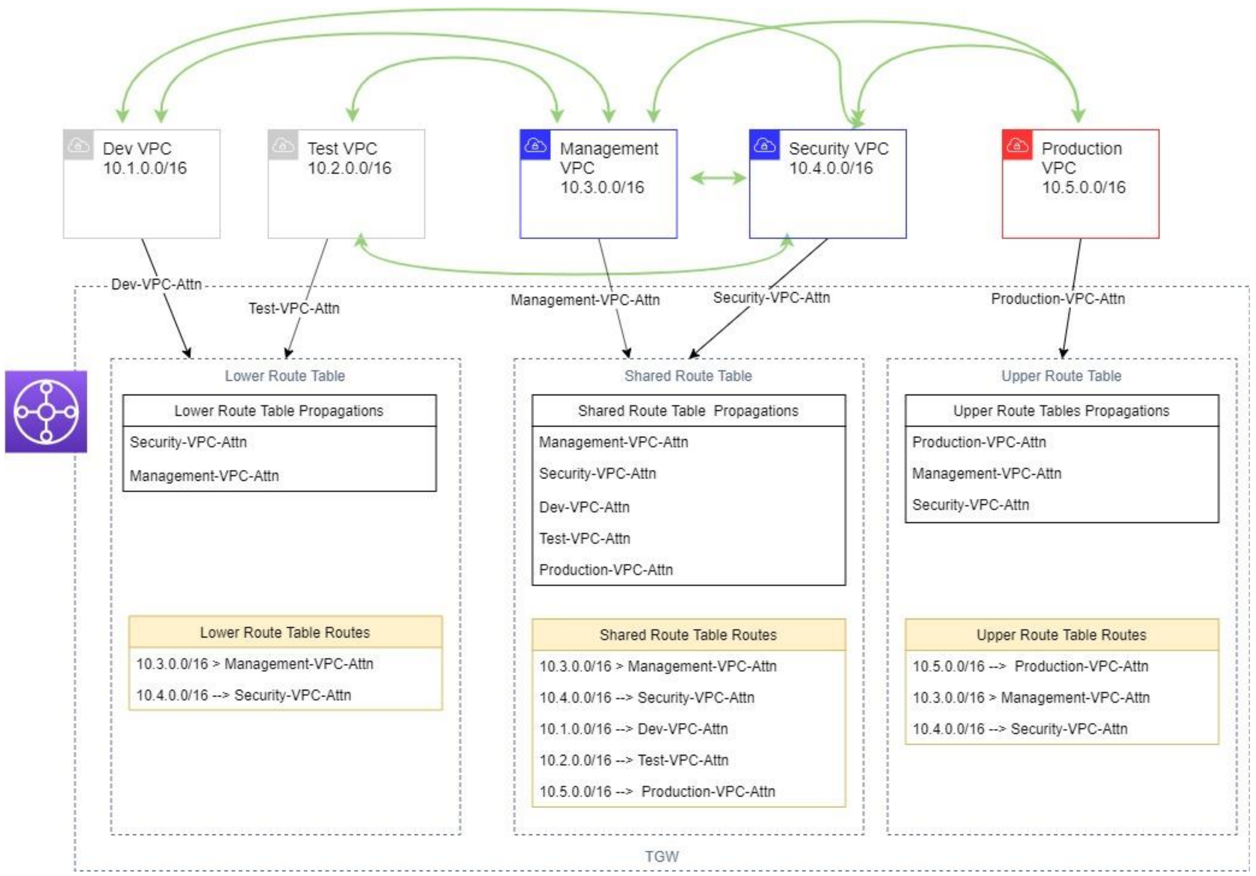


Figure 5: Remove Propagations to Other VPCs

Detailed Steps

Make sure to complete all prerequisites as outlined above before proceeding. The rest of the sections will detail the exact steps to deploy the environment.

Before continuing, gather the information that you will need for the rest of the setup. Fill out the table below in your own “scratch” area. We will need this info later when deploying the CFN templates. as input parameters.

Table 3: Workshop Prerequisites

Workshop Prerequisites	**Value**
“Lower” AWS Account Number (for Dev and Test VPC)	<>
“Shared” AWS Account Number (for Management and Security VPC)	<>
“Upper” Account Number (for Production VPC)	<>

Scenario #1 - Initial set up with no Transit Gateway

1) Login to the AWS console of **Lower** environment account that you have designated.
2) Change your region to **us-east-1** if not already there.
3) Use "**Lab-single-subnet.yml**" to deploy your stacks using CloudFormation (CFN). We will deploy Dev-Lab stack using 10.1.0.0/16 as VPC CIDR
   a) CloudFormation > Create Stack (select "with new resources")
   b) Template is Ready > Upload a template file > select the **lab-single-subnet.yml** from your hard-drive
   c) Click **Next** and enter/select the following parameters. **NOTE**: that the 2nd Octet of all IP related information should be "1" (e.g. 10.1.x.x). These are predefined, so you just need to select the correct values.

Parameters	**Value**
Stack Name	Dev-Lab
VPC CIDR	10.1.0.0/16
Public Subnet 1 CIDR	10.1.1.0/24
Private Subnet 1 CIDR	10.1.2.0/24
Availability Zone 1	us-east-1a
Private IP address of the test EC2 instance to assign	10.1.2.11
Bastion host key	<key pair for which you have your private key>
Instance type for EC2 host	t2.micro
Prefix ID for Amazon Network	pl-60b85b09

4) Use the wizard (accept all default) to finish deploying the stack. **NOTE**: You need to accept the prompt for "**I acknowledge that AWS CloudFormation might create IAM resources with custom names.**" The CFN template deploys a new role for **ec2SSMRole-<stackname>** with managed policy **AmazonSSMManagedInstanceCore** which allows the Session Manager to work.
5) Once complete, deploy the 2nd stack in the same "**Lower**" account using the value below with the same CFN template "**lab-single-subnet.yml**". We will deploy Test-Lab stack using 10.2.0.0/16 as VPC CIDR.
6) **NOTE**: that the 2nd Octet of all IP related information should be "2" (e.g. 10.2.x.x).

Parameters	**Value**
Stack Name	Test-Lab
VPC CIDR	10.2.0.0/16
Public Subnet 1 CIDR	10.2.1.0/24
Private Subnet 1 CIDR	10.2.2.0/24
Availability Zone 1	us-east-1a
Private IP address of the test EC2 instance to assign	10.2.2.11
Bastion host key	<key pair for which you have your private key>
Instance type for EC2 host	t2.micro
Prefix ID for Amazon Network	pl-60b85b09

7) **Login to your AWS account** that is going to serve as "**Shared**" AWS account. We will be deploying the <u>management</u> and <u>security</u> VPC.
8) Change your region to **us-east-1** if not already there.

9) Deploy the 1st stack in the "**Shared**" account using the value below with the same CloudFormation template "**lab-single-subnet.yml**". We will deploy Management-Lab stack using 10.3.0.0/16 as VPC CIDR.
10) NOTE: that the 2nd Octet of all IP related information should be "3" (e.g. 10.3.x.x).

Parameters	**Value**
Stack Name	Management-Lab
VPC CIDR	10.3.0.0/16
Public Subnet 1 CIDR	10.3.1.0/24
Private Subnet 1 CIDR	10.3.2.0/24
Availability Zone 1	us-east-1a
Private IP address of the test EC2 instance to assign	10.3.2.11
Bastion host key	<key pair for which you have your private key>
Instance type for EC2 host	t2.micro
Prefix ID for Amazon Network	pl-60b85b09

11) Once complete, deploy the 2nd stack in the same "**Shared**" account using the value below with the same CloudFormation template "**lab-single-subnet.yml**". We will deploy Security-Lab stack using 10.4.0.0/16 as VPC CIDR.
12) NOTE: that the 2nd Octet of all IP related information should be "4" (e.g. 10.4.x.x).

Parameters	**Value**
Stack Name	Security-Lab
VPC CIDR	10.4.0.0/16
Public Subnet 1 CIDR	10.4.1.0/24
Private Subnet 1 CIDR	10.4.2.0/24
Availability Zone 1	us-east-1a
Private IP address of the test EC2 instance to assign	10.4.2.11
Bastion host key	<key pair for which you have your private key>
Instance type for EC2 host	t2.micro
Prefix ID for Amazon Network	pl-60b85b09

13) **Login to your AWS account** that is going to serve as "**Upper**" AWS account. We will be deploying the Production VPC.
14) Change your region to **us-east-1** if not already there.
15) Deploy the 1st stack in the "**Upper**" account using the value below with the same CloudFormation template "**lab-single-subnet.yml**". We will deploy Production-Lab stack using 10.5.0.0/16 as VPC CIDR.
16) NOTE: that the 2nd Octet of all IP related information should be "5" (e.g. 10.5.x.x).

Parameters	**Value**
Stack Name	Production-Lab
VPC CIDR	10.5.0.0/16
Public Subnet 1 CIDR	10.5.1.0/24
Private Subnet 1 CIDR	10.5.2.0/24
Availability Zone 1	us-east-1a

Private IP address of the test EC2 instance to assign	10.5.2.11
Bastion host key	<key pair for which you have your private key>
Instance type for EC2 host	t2.micro
Prefix ID for Amazon Network	pl-60b85b09

17) Once all VPC's are deployed, the following diagram provides what is deployed.

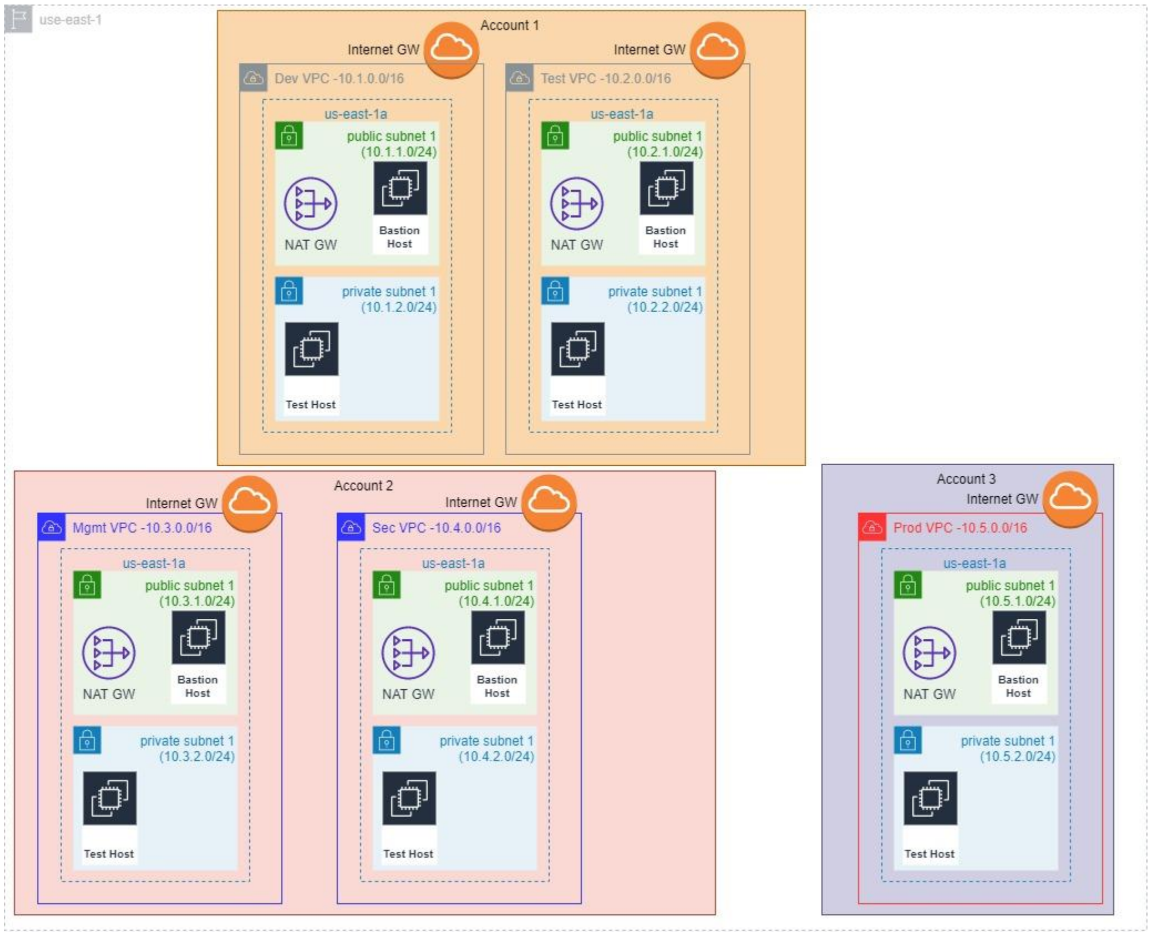


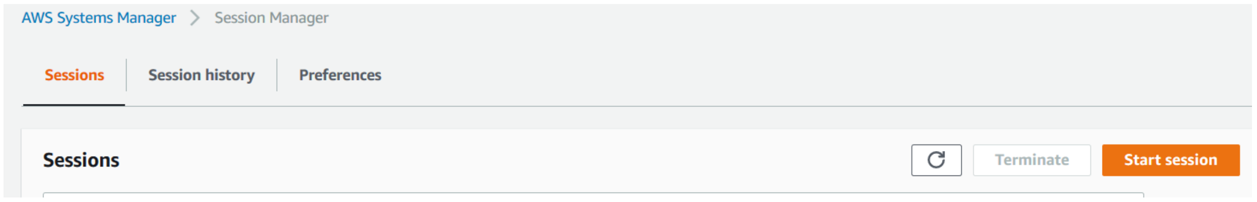
Figure 6: Scenario 1 Diagram

18) We will validate the initial deployment.
19) For each Environment, you will need the Test EC2 Instance names. The name tag should be with "TestEC2 - <stack name>".
20) We will be using System Manager Session Manager (https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager.html) to access the Test instances. They are pre-configured to be used.
21) The following table provides the info you will need. As long as you have used the correct CFN stack names and the IP CIDR as outlined earlier, the following should map to your environment.

Table 4: Environment VPC and Test EC2 Instance IP and Tag Name

Stack/Environment	**VPC CIDR**	**Test EC2 Instance IP**	**Test EC2 Instance Name**
Dev-Lab	10.1.0.0/16	10.1.2.11/24	TestEC2 - Dev-Lab

Test-Lab	10.2.0.0/16	10.2.2.11/24	TestEC2 - Test-Lab
Management-Lab	10.3.0.0/16	10.3.2.11/24	TestEC2 - Management-Lab
Security-Lab	10.4.0.0/16	10.4.2.11/24	TestEC2 - Security-Lab
Production-Lab	10.5.0.0/16	10.5.2.11/24	TestEC2 - Production-Lab

22) Test your deployment via PING test. We will use the same method for all subsequent PING test.
23) From the **TestEC2** instance attempt to ping the other Private IP address of the other EC2 instance. These should all fail since all VPC's are still isolated from each other. Repeat the following steps to confirm the ping test.
    a) Login to the AWS account where the instance resides (e.g. **Lower** for both Dev and Test)
    b) Go to Services > Systems Manager
    c) Select Session Manager from the left pane. Start Session
    d) 

    e) Select the particular instance (use the instance name to identify correct host) from the list and Start Session.
    f) **NOTE**: Ignore the Bastion hosts. They are left-over from previous version of this lab.
    g) You should now be in a console. Type "ifconfig" and validate IP address matches what's on the table (inet addr under "eth0").
    h) Perform "ping <target IP>". For example, to test ping to Security VPC, type "ping 10.4.2.11"
24) Record the result here. Use Yes/No or "success/fail" to denote the results using the System Manager Session Manager.

Table 5: Ping Test #1

	To TestEC2 Dev 10.1.2.11	To TestEC2 Test 10.2.2.11	To TestEC2 Management 10.3.2.11	To TestEC2 Security 10.4.2.11	To TestEC2 Production 10.5.2.11
From TestEC2 Dev 10.1.2.11	Same host – skip				
From TestEC2 Test 10.2.2.11		Same host – skip			
From TestEC2 Management 10.3.2.11			Same host – skip		
From TestEC2 Security 10.4.2.11				Same host – skip	
From TestEC2 Production 10.5.2.11					Same host – skip

25) Once the confirmation is done, proceed to the next Section.

Scenario #2 - Transit Gateway with only association (no routing)

This Scenario will deploy the Transit Gateway and attach each VPC to the respective Routing Tables. You will need your AWS account numbers for your "**Upper**" and "**Lower**" environments. Use Table 3: Workshop Prerequisites above.

1) **Login** to the **Shared** AWS account.
2) Change your region to **us-east-1** if not already there.
3) Use "**create-TGW.yml**" (see attached) to deploy your stacks using CFN.
   a) CloudFormation > Create Stack
   b) Template is Ready > Upload a template file > select the **create-TGW.yml** from your hard-drive
   c) Next

Parameters	**Value**
Stack Name	TGW-Lab
Lower AWS Account	<Enter your Lower AWS Account from the table>
Upper AWS Account	<Enter your Upper AWS Account from the table>
TGW ASN Number to Assign	64512

4) Follow the CFN wizard to deploy.
5) The CFN will deploy a Transit Gateway along with 3 Routing Tables and share the TGW via Resource Access Manager (RAM) to the 2 other AWS accounts you specified in the CFN stack.
6) Once the template is deployed, check your deployment:
   a) From **Shared** account, go to VPC > Transit Gateways. You should see the TGW named "**TGW TGW-Workshop**"
   b) VPC > Transit Gateway Route Tables. You should see 3 Route Tables.
   c) Login into the AWS **Lower** account, go to Resource Access Manager > Shared With me > ResourceShareTGW > accept the resource share.
   d) Login into the AWS **Upper** account, go to Resource Access Manager > Shared With me > ResourceShareTGW > accept the resource share.
   e) **NOTE**: If you accidently rejected the resource share, then manually share the Transit Gateway resource from the Shared account again to the other accounts.
7) Use the following table to attach each VPC to the Transit Gateway.

AWS Account	**VPC**	**TGW Attachment Name**
"Lower"	VPC Dev-Lab	Dev-VPC-Attn
"Lower"	VPC Test-Lab	Test-VPC-Attn
"Shared"	VPC Management-Lab	Management-VPC-Attn
"Shared"	VPC Security-Lab	Security-VPC-Attn
"Upper"	VPC Production-Lab	Production-VPC-Attn

   a) Login to the proper AWS account as outlined in the table above.
   b) Go to VPC > Transit Gateway Attachments > Create Transit Gateway Attachment
   c) Transit Gateway ID: <Select the TGW created earlier> with description "TGW Workshop"
   d) Attachment Type: VPC
   e) Attachment Name: <See Table above for the environment>
   f) DNS support: **Enable (Default)**
   g) IPv6 support: **Disable (Default)**
   h) VPC ID: <select from the drop down and select the environment>

i) Availability Zone: <select the only subnet available>

j) Select the **Private** Subnet. **NOTE**: It is recommended to have a dedicated Subnet for the Transit Gateway per AZ. However for the simplicity of the lab, we are using the existing subnet.

k) Repeat above for each VPC.

8) Go back to **Shared** account > VPC > Transit Gateway Attachments. You will see **3 pending acceptance** for the 3 external VPC (Dev, Test, Production). Select each and Action > accept.

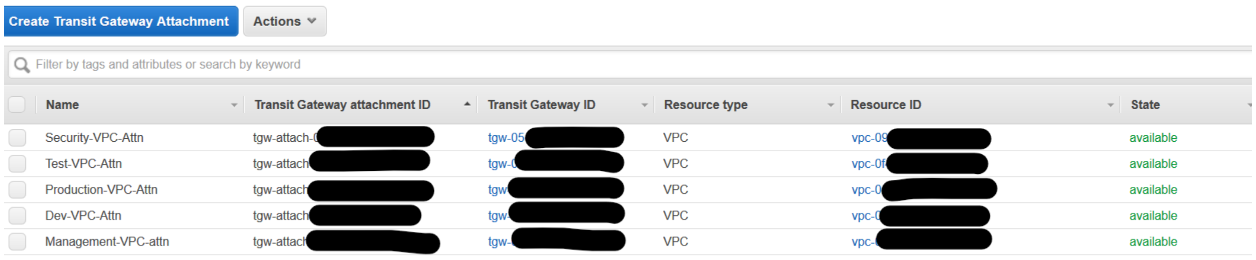
a) Wait several minutes until the state shows "**available**".

9) **NOTE**: Even though you have named your VPC attachment inside the other accounts, the name tags do not show up within the **Shared** account. We will go ahead and add the names manually again with the steps outlined below.

a) Login to the **Lower** account. Under VPC > Transit Gateway Attachments, populate the below table. Also login to the **Upper** account. Under VPC > Transit Gateway Attachments, populate the below table. Copy and paste the attachment ID for each respective attachment into the table. You will need these information to properly tag the attachments.

TGW Attachment Name	**TGW Attachment ID**
Dev-VPC-Attn	<>
Test-VPC-Attn	<>
Production-VPC-Attn	<>

b) Go back to **Shared** account > VPC > Transit Gateway Attachments. And based upon the attachment ID from the table above, update the Attachment Name to match what is in the table above.


Create Transit Gateway Attachment
Actions
Filter by tags and attributes or search by keyword

Name	Transit Gateway attachment ID	Transit Gateway ID	Resource type	Resource ID	State
Security-VPC-Attn	tgw-attach-0[redacted]	tgw-05[redacted]	VPC	vpc-09[redacted]	available
Test-VPC-Attn	tgw-attach[redacted]	tgw-0[redacted]	VPC	vpc-0f[redacted]	available
Production-VPC-Attn	tgw-attach[redacted]	tgw[redacted]	VPC	vpc-0[redacted]	available
Dev-VPC-Attn	tgw-attach[redacted]	tgw[redacted]	VPC	vpc-0[redacted]	available
Management-VPC-attn	tgw-attac[redacted]	tgw-[redacted]	VPC	vpc-[redacted]	available

c) You should have 5 attachments when complete with proper names (should look similar to above).

10) Now, we will **associate** each VPC attachment to a single TGW Route Table (aka Routing Domain).

a) **NOTE**: an attachment within a TGW can only be associated with a single Route Table.

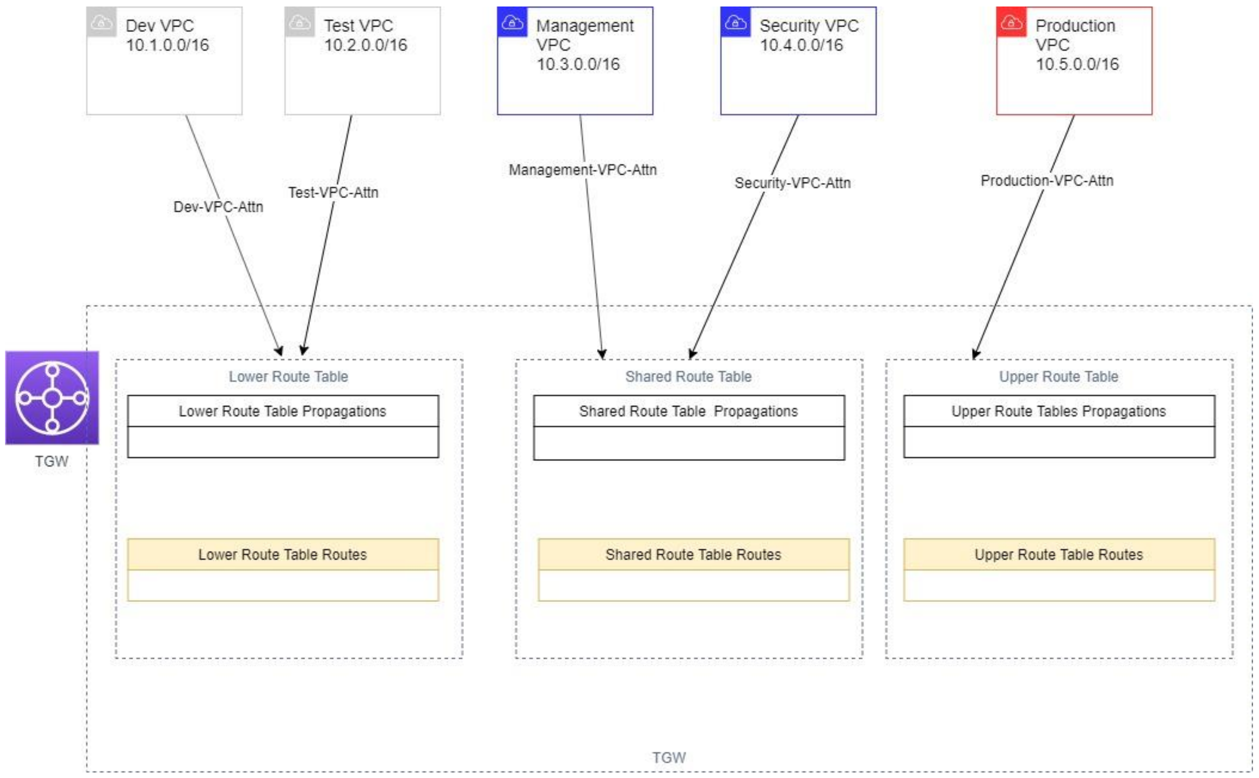
b) Use the table below for the associations.

TGW Route Tables	**VPC Attachment to TGW Association**
Lower Route Table <CFN stack name>	Dev-VPC-Attn Test-VPC-Attn
Shared Route Table <CFN stack name>	Management-VPC-Attn Security-VPC-Attn
Upper Route Table <CFN stack name>	Production-VPC-Attn

c) Login to the **Shared** AWS account (where TGW was created).

d) Go to VPC > Transit Gateway Route Tables > Select the respective Route Table > Change to Association tab > Create Association > Select the proper VPC attachment from the table above > Create Association
e) Repeat above for all 5 attachments and associate with the correct TGW Route Table.
f) Wait until the state changes to **associated**.

11) When you have finished creating the Associations, your environment will look like the following diagram.



12) Since each VPC is still isolated, there is no need to check/test the PING from the Test EC2 hosts.
13) Review the Knowledge Check below and proceed to the next section.

Knowledge Check – Scenario #2

We have introduced 3 terms within Transit Gateway. You should be familiar with these terms as well as how to create them.

- **Transit Gateway Route Table**: TGW Route tables provide a flexible segmentation that isolates traffic within the TGW. For this lab, we are using "**Upper**", "**Lower**", and "**Shared**" Route Tables to provide a model of how you can segment your traffic leveraging TGW Route Tables. We will cover more about the TGW Route Tables in next sections. In this lab, the TGW Route Tables were provisioned for you through the CFN Template.
- **Transit Gateway attachment** (VPC). This is simply attaching a VPC to the Transit Gateway so that the respective VPC can interact with the Transit Gateway. With only the attachment, the VPC still has no connectivity via Transit Gateway. Other constructs are still needed (mainly the Association, and TGW Route tables). A VPC can only be attached to a maximum of 5 distinct Transit Gateways within the region.

- **Transit Gateway Associations**. Within a Transit Gateway, an attachment (whether DXGW, VPC or VPN) can only be associated with one TGW Route Table. We will provide more explanation on the impact of having Transit Gateway Associations in the subsequent sections. The main take-away in this section is that you will need to associate the attachment to a TGW Route Table.

Scenario #3 - Adding Propagations only within TGW Routing Tables

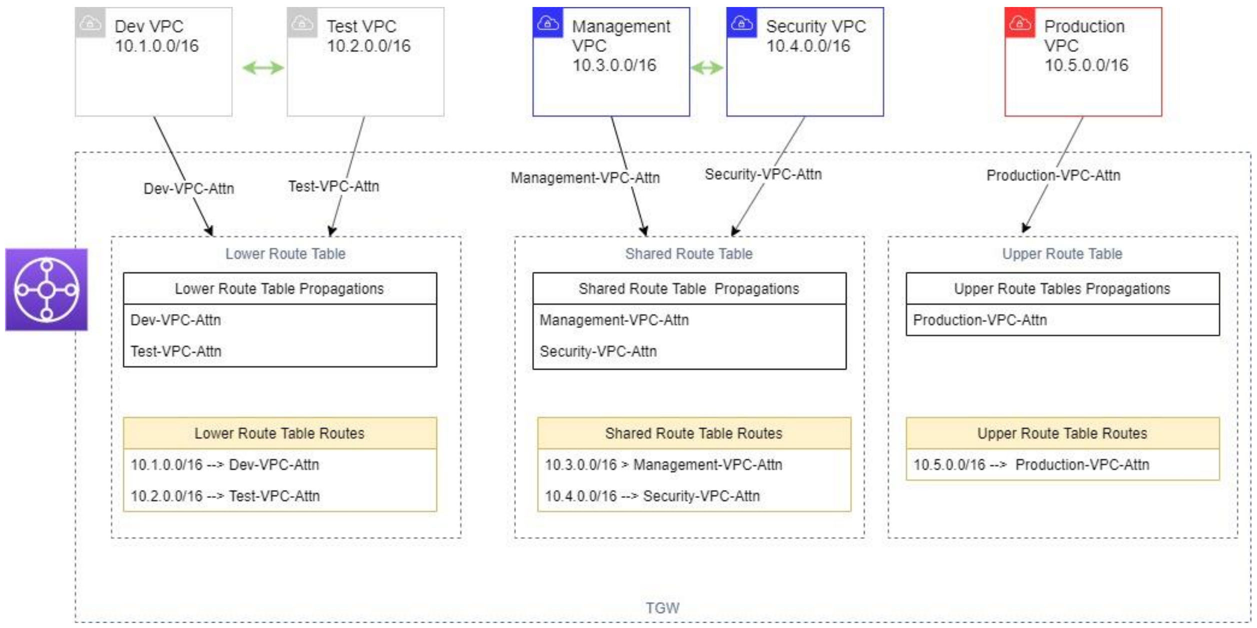
Once we have attached the VPC's to the TGW as well as associating each VPC attachment to the respective TGW Route Table, we will be adding the TGW Propagations to the respective TGW Route Tables. Route Propagation allows the CIDR blocks of the respective attached VPC to be automatically added to the TGW Route Table (see https://docs.aws.amazon.com/vpc/latest/tgw/how-transit-gateways-work.html for more info). By adding the VPC attachment to a TGW Route Table, we allow traffic ***TO*** the VPC from the TGW Route table to be possible.

1) Login to the **Shared** account.
2) VPC > Transit Gateway Route Tables > Use **Propagation** tab.
3) Use the following table to add the TGW Propagations.

TGW Route Tables	**VPC Attachment to be added in TGW Propagation**
Lower Route Table <CFN stack name>	• Dev-VPC-Attn • Test-VPC-Attn
Shared Route Table <CFN stack name>	• Management-VPC-Attn • Security-VPC-Attn
Upper Route Table <CFN stack name>	• Production-VPC-Attn

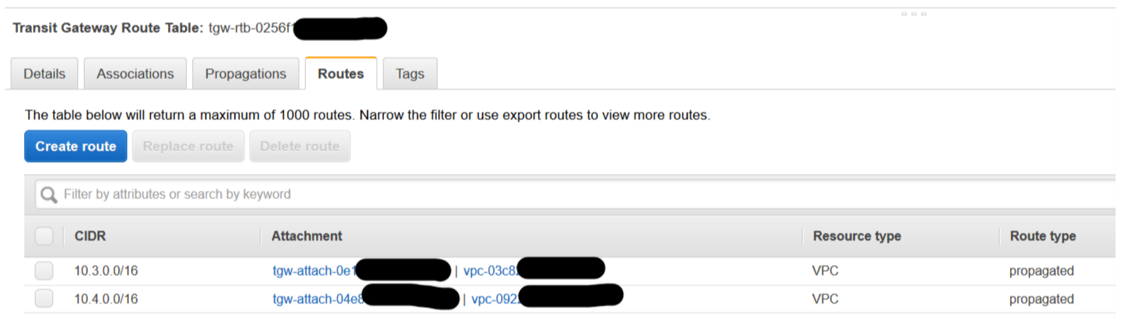
   a) Go to the **Propagation** Tab under each TGW Route Table.
   b) Add the attachments as outlined in the table above to the proper TGW Route Tables.
   c) **NOTE**: In this scenario, each VPC associated in respective TGW Route Table propagation also has a Propagation in the same TGW Route Table.

4) The following diagram provides a visual of what your environment will look like.



Dev VPC 10.1.0.0/16
Test VPC 10.2.0.0/16
Management VPC 10.3.0.0/16
Security VPC 10.4.0.0/16
Production VPC 10.5.0.0/16
Dev-VPC-Attn
Test-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Production-VPC-Attn
Lower Route Table
Lower Route Table Propagations
Dev-VPC-Attn
Test-VPC-Attn
Lower Route Table Routes
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
Shared Route Table
Shared Route Table Propagations
Management-VPC-Attn
Security-VPC-Attn
Shared Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
Upper Route Table
Upper Route Tables Propagations
Production-VPC-Attn
Upper Route Table Routes
10.5.0.0/16 --> Production-VPC-Attn
TGW

5) For each TGW Route Table, click on the **Routes** tab. Notice that the VPC CIDR (e.g. 10.1.0.0/16 for Dev VPC) is automatically added to the TGW Route Table routes. This is the result of adding the VPC attachment to the TGW Route Table Propagation; the VPC CIDR's are dynamically added. Thus, if you were to add a 2nd CIDR to the VPC, that 2nd CIDR will automatically appear in the *Routes* under the TGW Route Table.



Transit Gateway Route Table: tgw-rtb-0256f[redacted]

Details | Associations | Propagations | Routes | Tags

The table below will return a maximum of 1000 routes. Narrow the filter or use export routes to view more routes.

Create route | Replace route | Delete route

Filter by attributes or search by keyword

CIDR	Attachment	Resource type	Route type
10.3.0.0/16	tgw-attach-0e1[redacted] | vpc-03c8[redacted]	VPC	propagated
10.4.0.0/16	tgw-attach-04e8[redacted] | vpc-092[redacted]	VPC	propagated

6) Once you have added each Propagation as outlined in the table above, review the Routes on each TGW Route Table and confirm that it matches what is shown in the diagram above.
7) From the Test EC2 instance, attempt to ping the other Private IP address of the other EC2 instance.
8) Record the result here. Use Yes/No or "success/fail" to denote the results using the System Manager Session Manager.

Table 6: Ping Test #2

	To TestEC2 Dev 10.1.2.11	To TestEC2 Test 10.2.2.11	To TestEC2 Management 10.3.2.11	To TestEC2 Security 10.4.2.11	To TestEC2 Production 10.5.2.11
From TestEC2 Dev 10.1.2.11	Same host – skip				
From TestEC2 Test 10.2.2.11		Same host – skip			
From TestEC2 Management 10.3.2.11			Same host – skip		
From TestEC2 Security 10.4.2.11				Same host – skip	
From TestEC2 Production 10.5.2.11					Same host – skip

9) But wait…. All PING still fails, what happened? Did you think that the PING should work between the VPC's associated with the same Routing Table?
10) We have missed one step. Even though the TGW Route Tables have the correct Routes to the various VPC's, there is **no** route defined to go to the Transit GW from the VPC. This is NOT added automatically, and you must go ahead and configure this. The VPC Route tables must be updated to allow this. Follow steps below:
    a) Login to the each AWS account (where the respective VPC resides).
    b) Go to the VPC > Route Tables
    c) Select "Private subnet route table - <Environment>" > Select Routes.
    d) There should be 2 existing Routes already there.
        i) 10.x.0.0/16 with local (VPC)
        ii) 0.0.0.0/0 to NAT gateway.
    e) Add a new route entry
        i) **Edit** Route > **Add** Route
        ii) Destination: **10.0.0.0/8**
        iii) Target: **Transit Gateway**, select the only TGW listed.
        iv) **Save** Route
    f) Repeat above for each VPC, so each private subnet route table will have 10.0.0.0/8 route to the TGW. See the table below for summary

VPC Route Table	**Route Add**	**Destination**
Private subnet route table - Dev-Lab	10.0.0.0/8	Transit Gateway
Private subnet route table - Test-Lab	10.0.0.0/8	Transit Gateway
Private subnet route table - Management-Lab	10.0.0.0/8	Transit Gateway
Private subnet route table - Security-Lab	10.0.0.0/8	Transit Gateway
Private subnet route table - Production-Lab	10.0.0.0/8	Transit Gateway

11) Note that the route that we are adding to the VPC route table is a static route. You can specify a different CIDR (e.g. add TGW as the default route 0.0.0.0/0 or something that is less specific than the VPC itself such as 10.0.0.0/8).
12) Once the VPC Route tables is updated as outlined above, we should retest our PING test.
13) From the Test EC2 instance, attempt to ping the other Private IP address of the other EC2 instance.
14) Record the result here. Use Yes/No or "success/fail" to denote the results using the System Manager Session Manager.

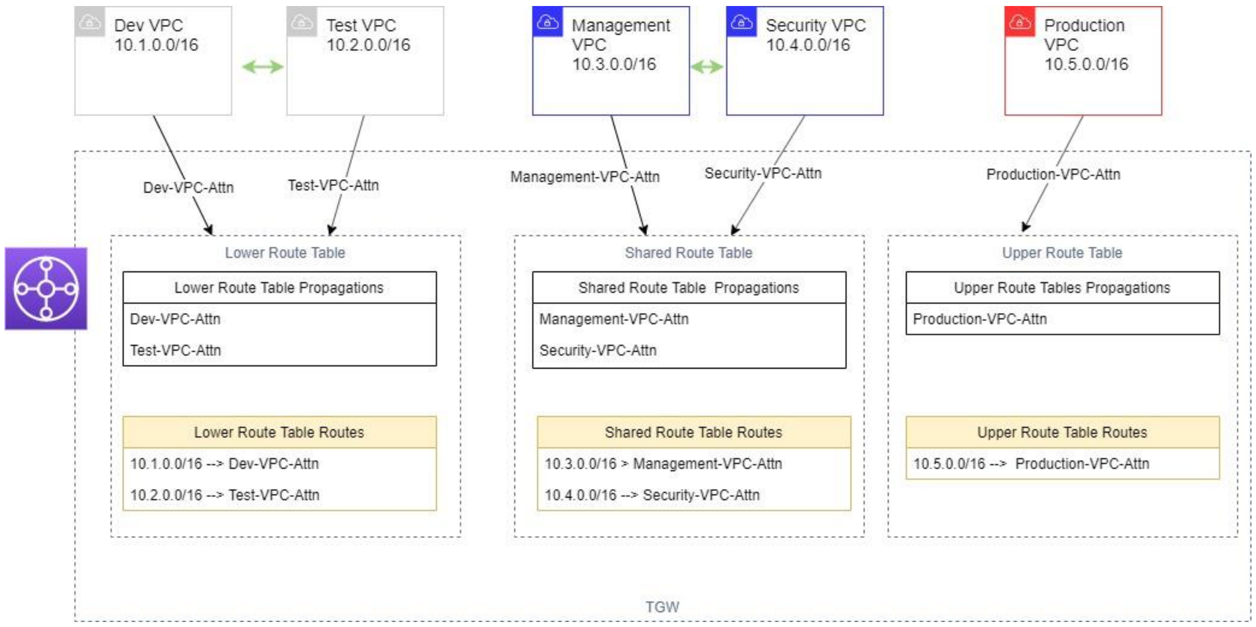
Table 7: Ping Test #3 – after subnet route table change

	To TestEC2 Dev 10.1.2.11	To TestEC2 Test 10.2.2.11	To TestEC2 Management 10.3.2.11	To TestEC2 Security 10.4.2.11	To TestEC2 Production 10.5.2.11
From TestEC2 Dev 10.1.2.11	Same host – skip				
From TestEC2 Test 10.2.2.11		Same host – skip			
From TestEC2 Management 10.3.2.11			Same host – skip		
From TestEC2 Security 10.4.2.11				Same host – skip	
From TestEC2 Production 10.5.2.11					Same host – skip

15) The test should confirm that we have now allowed EC2 instances from these environments to communicate with each other but nothing else.
    a) Dev <- -> Test
    b) Management <- -> Security
16) Since Production VPC is still isolated, it is NOT able to communicate with any other VPC's.
17) Review the Knowledge Check below and proceed to the next section.

Knowledge Check – Scenario #3

- You can add Transit Gateway Propagations to dynamically add the routes to the VPC CIDR.
- You must add static route to the TGW within the VPC Route Table(s). This is not automatic.
- TGW Association determines which TGW Route Tables is used to evaluate for the source VPC traffic. Let's use the diagram below to demonstrate.



- Example #1 - The traffic from Dev VPC (10.1/16) going to Test VPC (10.2/16)
  - The local VPC route table states, for 10.2.0.0/24 destined traffic, the route is to the TGW.
  - The packet arrives at the "**Lower Route Table**" since the Dev VPC is **associated** with that TGW Route Table.
  - There is a route defined for 10.2.0.0/24 within the "Lower Route Table" which is destined for Test-VPC-Attn
  - The packet is then routed to the Test VPC.
- Example #2 – The traffic from Dev VPC (10.1/16) going to Production VPC (10.5/16)
  - The local VPC route table states, for 10.5.0.0/24 destined traffic, the route is to the TGW.
  - The traffic arrives at the "**Lower Route Table**" since the Dev VPC is **associated** with that TGW Route table.
  - There is NO route defined for 10.5.0.0/24 within the Lower Route Table
  - The packet dies since there is no way to get to 10.5/16 address.

Scenario #4 - Adding Propagations to Shared VPCs

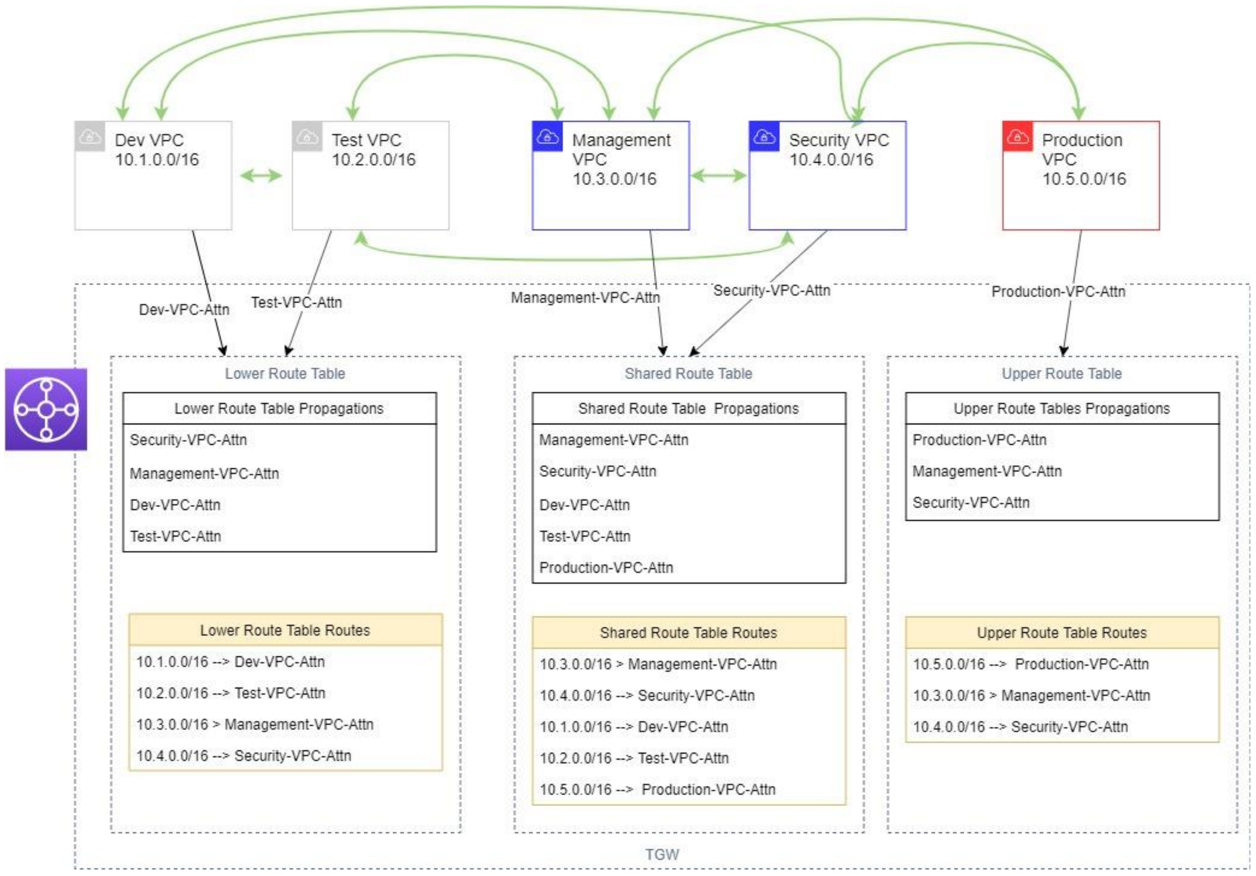
This scenario, we add additional TGW propagation to show how you can allow Shared VPC's to be reachable by other VPC's even though they are associated to a different TGW Route Table.

From the previous Scenario, we demonstrated that both the management and the security VPC are only reachable by each other and not by any other VPC's. Now, we want to make both management and security VPC reachable by 3 other VPC's in our workshop.

1) Login to the **Shared** AWS account.
2) Change your region to **us-east-1** if not already there.
3) VPC > Transit Gateway Route Tables > Use Propagation tab.
4) Use the following table to add the *additional* TGW Propagations. These are additional Propagations. Leave the existing Propagations alone.

TGW Route Tables	**VPC Attachment to be added in TGW Propagation**
Lower Route Table <CFN stack name>	Management-VPC-Attn Security-VPC-Attn
Shared Route Table <CFN stack name>	Dev-VPC-Attn Test-VPC-Attn Production-VPC-Attn
Upper Route Table <CFN stack name>	Management-VPC-Attn Security-VPC-Attn

5) Once complete, your environment will look like the following diagram.


Dev VPC 10.1.0.0/16
Test VPC 10.2.0.0/16
Management VPC 10.3.0.0/16
Security VPC 10.4.0.0/16
Production VPC 10.5.0.0/16
Dev-VPC-Attn
Test-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Production-VPC-Attn
Lower Route Table
Lower Route Table Propagations
Security-VPC-Attn
Management-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Lower Route Table Routes
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
Shared Route Table
Shared Route Table Propagations
Management-VPC-Attn
Security-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Production-VPC-Attn
Shared Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.5.0.0/16 --> Production-VPC-Attn
Upper Route Table
Upper Route Tables Propagations
Production-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Upper Route Table Routes
10.5.0.0/16 --> Production-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
TGW

6) Check the Routes tab of each TGW Route Table to confirm that the VPC CIDRs have been dynamically added. Again, this is the result of the adding of the Propagation.
7) Do you think that now, the Dev VPC should be able to reach Management VPC? What about to the Production VPC? Let's find out.
8) From the Test EC2 instance, attempt to ping the other Private IP address of the other EC2 instance.
9) Record the result here. Use Yes/No or "success/fail" to denote the results using the System Manager Session Manager.

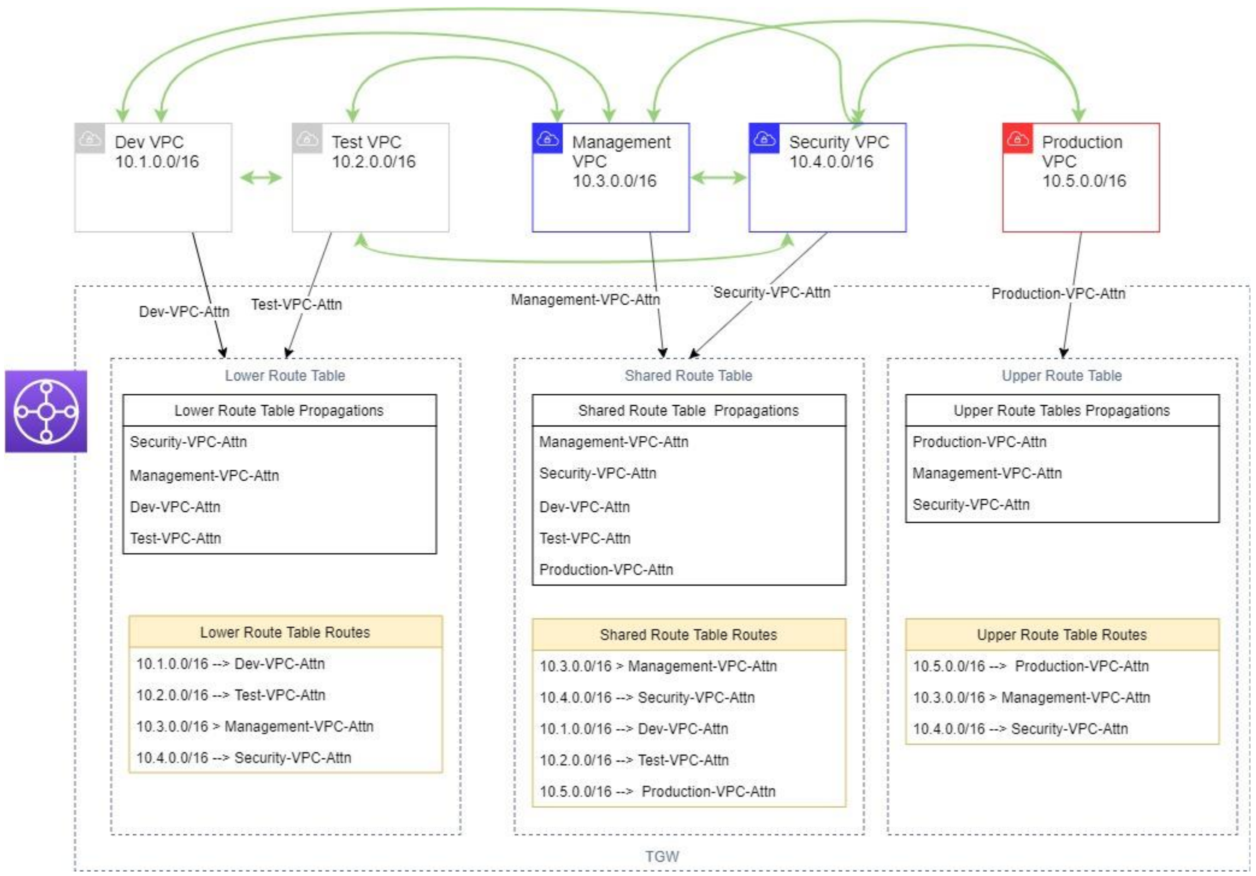
Table 8: Ping Test #4 – After Adding Propagation

	To TestEC2 Dev 10.1.2.11	To TestEC2 Test 10.2.2.11	To TestEC2 Management 10.3.2.11	To TestEC2 Security 10.4.2.11	To TestEC2 Production 10.5.2.11
From TestEC2 Dev 10.1.2.11	Same host – skip				
From TestEC2 Test 10.2.2.11		Same host – skip			
From TestEC2 Management 10.3.2.11			Same host – skip		
From TestEC2 Security 10.4.2.11				Same host – skip	
From TestEC2 Production 10.5.2.11					Same host – skip

10) Were you able to validate your guess? Here's the summary of what is allowed after we have added the additional Propagations.
    a) From Dev -> Test, Management, and Security
    b) From Test -> Dev, Management, and Security
    c) From Management -> all
    d) From Security -> all
    e) From Production -> Management, and Security
11) Review the Knowledge Check below and proceed to the next section.

Knowledge Check – Scenario #4

- A TGW Attachment (VPC attachment as an example) can be added to multiple TGW Route Tables Propagations as Route Destination.
- Adding a VPC attachment (destination) to a TGW Route Table **Propagation** provides the ability for the VPC's **associated** (source) with another TGW Route Table to route traffic to the destination VPC attachment
    - As an example, we add the Management VPC attachment to the "**Lower Route Table**" that allows both the Test VPC and the Dev VPC to be able to route the traffic to the Management VPC.
- Summary Take-Away:
    - TGW **Associations** controls traffic originating from that VPC by ***assigning*** that TGW Route Table as the "next" hop for the VPC.
    - TGW **Propagation** (and thus Routes) controls which destination attachments are routable from the associated VPC's in the TGW Route Tables.


Dev VPC 10.1.0.0/16
Test VPC 10.2.0.0/16
Management VPC 10.3.0.0/16
Security VPC 10.4.0.0/16
Production VPC 10.5.0.0/16
Dev-VPC-Attn
Test-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Production-VPC-Attn
Lower Route Table
Shared Route Table
Upper Route Table
Lower Route Table Propagations
Security-VPC-Attn
Management-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Shared Route Table Propagations
Management-VPC-Attn
Security-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Production-VPC-Attn
Upper Route Tables Propagations
Production-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Lower Route Table Routes
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
Shared Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.5.0.0/16 --> Production-VPC-Attn
Upper Route Table Routes
10.5.0.0/16 --> Production-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
TGW

Scenario #5 - Remove Propagations to other VPC's (Allow only To Shared)

Finally, what if we do not want the VPC's associated with the same TGW routing table to be able to communicate? For example, what if we want to prevent traffic between the Dev VPC and the Test VPC?

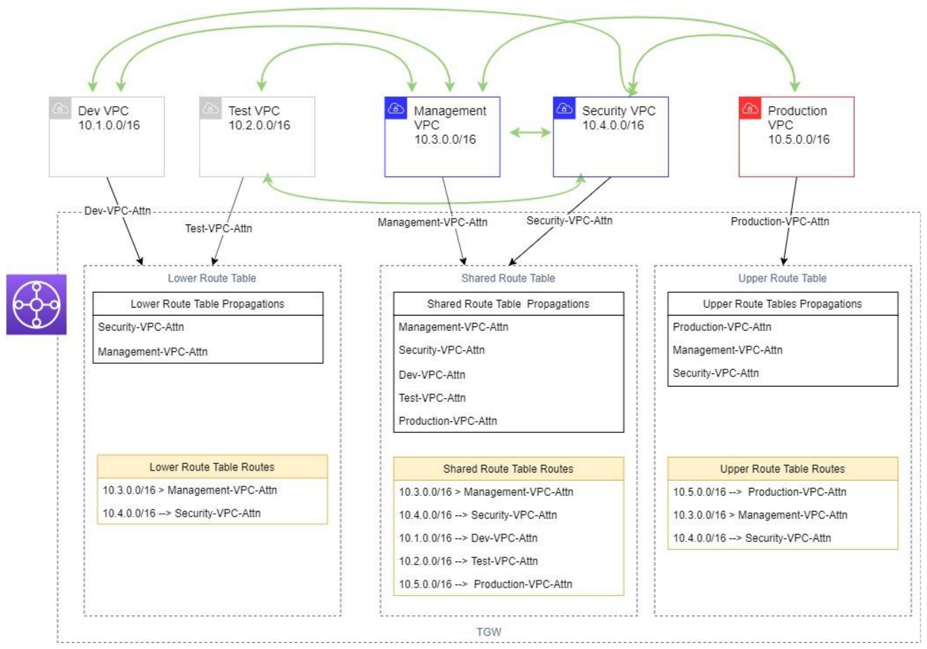
Based upon previous sections Knowledge Check, this is controlled by managing the **Propagations** under the TGW Route Table where the VPC's are associated. In our case, we need to update the **Lower Route Table** Propagations.

1) Login to the **Shared** AWS account.
2) Change your region to **us-east-1** if not already there.
3) You will need to find the Transit Gateway Attachment ID of the respective VPC attachments,
   a) VPC > Transit Gateway Attachments
   b) Map out Transit Gateway Names to the Attachment ID and update the table with ID of your attachments.

Table 9: TGW Attachment ID to Remove

TGW Route Tables	Remove VPC Attachment to TGW Propagation	TGW Attachment ID to Remove
Lower Route Table <CFN stack name>	Dev-VPC-Attn	<>
Lower Route Table <CFN stack name>	Test-VPC-Attn	<>

4) Once you have the ID's to be removed, go to VPC > Transit Gateway Route Tables > Use Propagation tab.
5) Go to the **Lower** Route Table <CNF stack name> and **remove** the respective TGW Attachments from the **Propagations**. Use the table above for the specific TGW Attachment ID's to remove. You should only remove 2 attachments from the **Propagations** in the **Lower** Route Table.
6) Go the Routes tab and verify that both the Dev and the Test VPC CIDR are no longer present. Again, this is done dynamically since the attachment was removed from the Propagations.
7) Once you remove the VPC attachments from the Propagation, the following diagram provides the visual representation of the environment. Notice that the Dev and the Test VPC are no longer able to route to each other through TGW.


Dev VPC 10.1.0.0/16
Test VPC 10.2.0.0/16
Management VPC 10.3.0.0/16
Security VPC 10.4.0.0/16
Production VPC 10.5.0.0/16
Dev-VPC-Attn
Test-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Production-VPC-Attn
Lower Route Table
Lower Route Table Propagations
Security-VPC-Attn
Management-VPC-Attn
Lower Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
Shared Route Table
Shared Route Table Propagations
Management-VPC-Attn
Security-VPC-Attn
Dev-VPC-Attn
Test-VPC-Attn
Production-VPC-Attn
Shared Route Table Routes
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
10.1.0.0/16 --> Dev-VPC-Attn
10.2.0.0/16 --> Test-VPC-Attn
10.5.0.0/16 --> Production-VPC-Attn
Upper Route Table
Upper Route Tables Propagations
Production-VPC-Attn
Management-VPC-Attn
Security-VPC-Attn
Upper Route Table Routes
10.5.0.0/16 --> Production-VPC-Attn
10.3.0.0/16 > Management-VPC-Attn
10.4.0.0/16 --> Security-VPC-Attn
TGW

12) From the Test EC2 instance, attempt to ping the other Private IP address of the other EC2 instance.
13) Record the result here. Use Yes/No or “success/fail” to denote the results using the System Manager Session Manager.

Table 10: Ping Test #5 – After Removing Propagations

	To TestEC2 Dev 10.1.2.11	To TestEC2 Test 10.2.2.11	To TestEC2 Management 10.3.2.11	To TestEC2 Security 10.4.2.11	To TestEC2 Production 10.5.2.11
From TestEC2 Dev 10.1.2.11	Same host – skip				
From TestEC2 Test 10.2.2.11		Same host – skip			
From TestEC2 Management 10.3.2.11			Same host – skip		
From TestEC2 Security 10.4.2.11				Same host – skip	
From TestEC2 Production 10.5.2.11					Same host – skip

8) So what happened? The result is the same as the previous scenario except that now the Dev VPC and Test VPC are no longer able to communicate. We accomplished it by managing the **Propagation** of the TGW Route table where the source VPC is associated with.
9) Review the Knowledge Check below and proceed to the next section.

Knowledge Check – Scenario #5

- It is NOT necessary that the attachment associated with the particular TGW Route table to also be added to the Propagation. In our example, we have both the Test and the Dev VPC attachments associated to the Lower Route Table but neither VPC attachments are added to the Propagation of the TGW Route Table.
- Again, we reinforce what we have learned before. TGW **Propagation** (and thus Routes) controls which destination attachments are routable from the associated VPC’s in the TGW Route Tables.

Clean up!

The workshop is complete, and the remaining step is to go ahead and destroy the resources deployed in your environment.

1) Login to the **Shared** AWS account.
2) Change your region to **us-east-1** if not already there.
3) VPC > Transit Gateway Attachments
4) Delete 5 attachments that were created during the workshop. Deletion can take several minutes. Wait until all attachments are deleted before proceeding to the next steps..
5) Once deleted, we will delete the TGW CFN stack.
    a) Login to the **Shared** AWS account.
    b) Change your region to **us-east-1** if not already there.
    c) Go to CloudFormation Service console.
    d) Delete **TGW-Lab** stack.
6) Delete the respective VPC's and the associated resources.
    a) Login to the **Upper** AWS account.
    b) Change your region to **us-east-1** if not already there.
    c) Go to CloudFormation Service console.
    d) Delete **Production-Lab** stack.
    e) Login to the **Shared** AWS account.
    f) Change your region to **us-east-1** if not already there.
    g) Go to CloudFormation Service console.
    h) Delete **Management-Lab** stack.
    i) Delete **Security-Lab** stack.
    j) Login to the **Lower** AWS account.
    k) Change your region to **us-east-1** if not already there.
    l) Go to CloudFormation Service console.
    m) Delete **Test-Lab** stack.
    n) Delete **Dev-Lab** stack.
7) (Optional) If the key pair is no longer needed or used, delete it from your AWA accounts and on your machine.

Future Workshop Enhancements

As time allows, we will enhance the overall workshop with the following features.

1) Support additional regions
2) Use SSM managed parameter store to query the latest AMI instead of hardcoding the AMI ID
    a. https://aws.amazon.com/blogs/compute/query-for-the-latest-amazon-linux-ami-ids-using-aws-systems-manager-parameter-store/
3) Add VPN Attachments via EC2 VPC Appliance to simulate on-premises connection via VPN connection.
4) Provide CLI commands to perform the test as an option instead of using console.