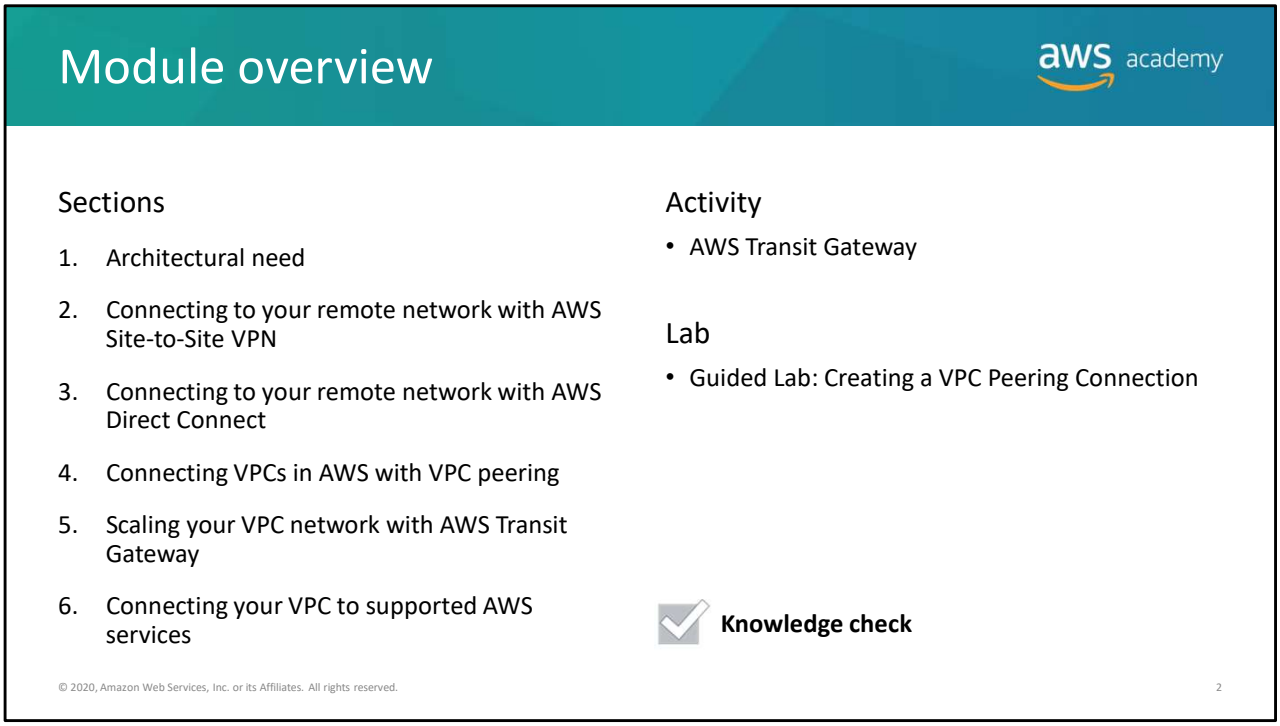
Welcome to Module 7: Connecting Networks.



This module includes the following sections:

1. Architectural need
2. Connecting to your remote network with AWS Site-to-Site VPN
3. Connecting to your remote network with AWS Direct Connect
4. Connecting VPCs in AWS with VPC peering
5. Scaling your VPC network with AWS Transit Gateway
6. Connecting your VPC to supported AWS services

This module also includes:

- An activity in which you discuss how you would use AWS Transit Gateway to connect three virtual private clouds (VPCs)
- A guided lab where you will create a VPC peering connection

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

aws academy

At the end of this module, you should be able to:

- Describe how to connect an on-premises network to the Amazon Web Services (AWS) Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud by using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services

At the end of this module, you should be able to:

- Describe how to connect an on-premises network to the Amazon Web Services (AWS) Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud by using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services

Module 7: Connecting Networks

Section 1: Architectural need

Introducing Section 1: Architectural need.

Café business requirement

The workloads for the café are increasing in complexity. The architecture must support connectivity between multiple VPCs, and be highly available and fault tolerant.

The café has started a loyalty rewards program, where a customer gets a free beverage or dessert after they purchase 10 or more similar items. When customers order online, they must provide some personally identifiable information (PII), such as an email address and credit card number. The café can't store this information in the cloud because of compliance reasons. Thus, Sofía and Nikhil need a way to connect their on-premises database (which stores sensitive customer information) to their cloud system (which stores transactions data). They must then map the data between the two systems to offer the rewards that their customers have earned.

Additionally, for security purposes, Sofía tells Olivia that she wants to isolate the development environment in one VPC and the production environment in another VPC, but still have connectivity between the two. Olivia agrees that this is a good idea, and advises Sofía to design the networking environment so that it's highly available and fault-tolerant.

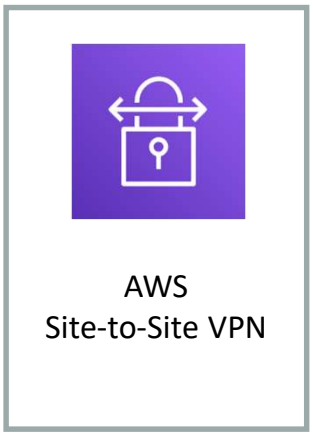
Module 7: Connecting Networks

Section 2: Connecting to your remote network with AWS Site-to-Site VPN

aws academy

Introducing Section 2: Connecting to your remote network with AWS Site-to-Site VPN.

AWS Site-to-Site VPN



AWS Site-to-Site is a highly available solution that enables you to securely connect your on-premises network or branch office site to your VPC.

- Uses internet protocol security (IPSec) communications to create encrypted virtual private network (VPN) tunnels
- Provides two encrypted tunnels per VPN connection
- Charged per VPN connection-hour

By default, instances that you launch into a virtual private cloud (VPC) on AWS cannot communicate with your on-premises network.

You can use AWS Site-to-Site Virtual Private Network (AWS Site-to-Site VPN) to securely connect your on-premises network or branch office site to your VPC. Each AWS Site-to-Site VPN connection uses internet protocol security (IPSec) communications to create encrypted VPN tunnels between two locations. A *VPN tunnel* is an encrypted link where data can pass from the customer network to or from AWS. The AWS side of the connection is the *virtual private gateway*. (Note that instead of a virtual private gateway, you can also create a Site-to-Site VPN connection as an attachment on a transit gateway. You will learn more about AWS Transit Gateway later in this module.) The on-premises side of the connection is the *customer gateway*.

AWS Site-to-Site VPN provides two VPN tunnels across multiple Availability Zones that you can use simultaneously for high availability. You can stream primary traffic through the first tunnel and use the second tunnel for redundancy. If one tunnel goes down, traffic will still get delivered to your VPC.

If you create a Site-to-Site VPN connection to your VPC, you are charged for each VPN connection-hour that your VPN connection is provisioned and available. For more information about pricing, see AWS Site-to-Site VPN and Accelerated Site-to-Site VPN Connection Pricing.

Static and dynamic routing

Static routing

- Requires you to specify all routes (IP prefixes)
- Specify *static routing* if your customer gateway device does not support BGP

Dynamic routing

- Uses the Border Gateway Protocol (BGP) to advertise its routes to the virtual private gateway
- Specify *dynamic routing* if your customer gateway device supports BGP*

*We recommend that you use BGP-capable devices because the BGP protocol offers robust liveness detection checks.

When you create a Site-to-Site VPN connection, you must specify the type of routing that you plan to use and you must update the route table for your subnet.

AWS Site-to-Site VPN supports two types of routing. The type of routing that you select depends on the make and model of your VPN devices:

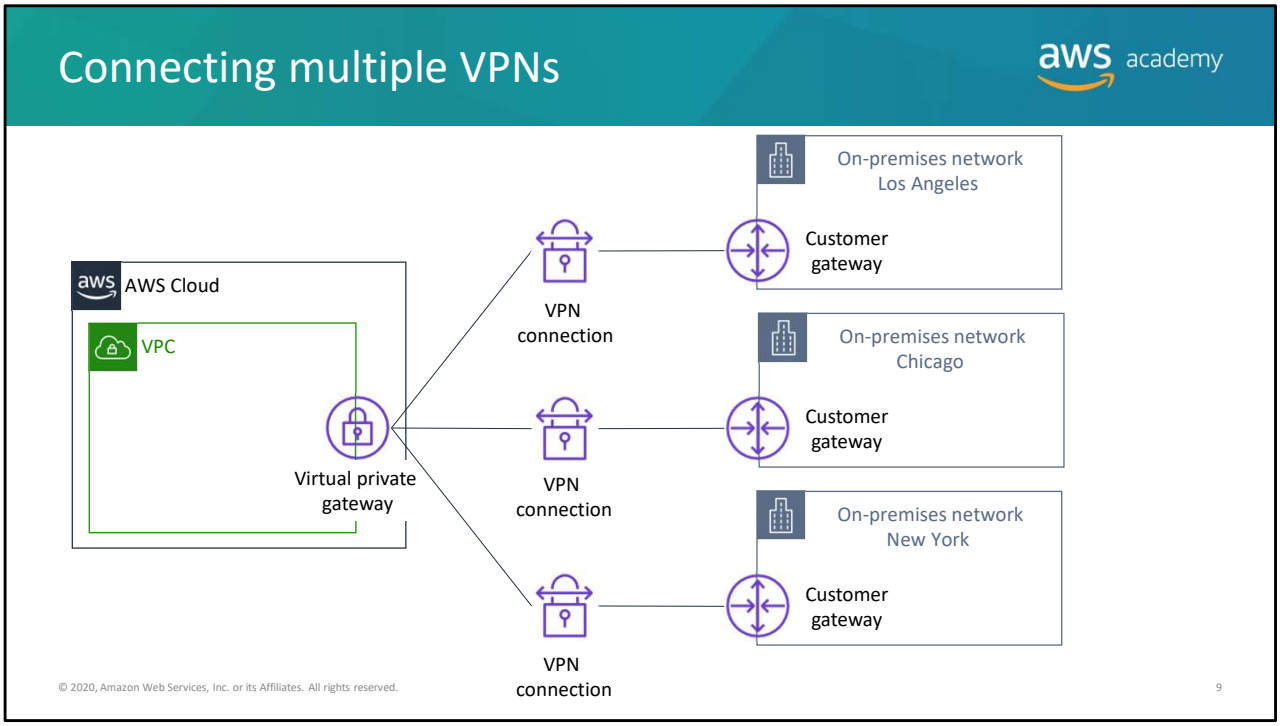
- If your VPN device supports Border Gateway Protocol (BGP), specify *dynamic routing* when you configure your Site-to-Site VPN connection. *Dynamic routing* uses the BGP to advertise routes to the virtual private gateway. Dynamic routing supports a maximum of 100 propagated routes per route table. (For current limits, see AWS Site-to-Site VPN limits.)
- If your VPN device does not support BGP, specify *static routing*. *Static routing* requires that you specify the routes (that is, IP prefixes) for your network that should be communicated to the virtual private gateway. Static routing supports 50 non-propagated routes per route table by default, up to a maximum of 1,000 non-propagated routes. (For current limits, see AWS Site-to-Site VPN limits.)

We recommend that you use BGP-capable devices because the BGP protocol offers robust liveness detection checks that can assist failover to the second VPN tunnel if the first tunnel

goes down. Devices that don't support BGP may also perform health checks to assist failover to the second tunnel when needed.

For a list of static and dynamic routing devices that have been tested with Amazon VPC, see Customer Gateway Devices We've Tested in the *AWS Site-to-Site VPN Network Administrator Guide*.

For more information about site-to-site VPN routing options, see Static and Dynamic Routing Options.



To maintain high availability of your customer gateway, you can set up redundant customer gateway devices. If you have redundant customer gateway devices, each device advertises the same prefix (for example, *0.0.0.0/0*) to the virtual private gateway. AWS uses BGP routing to determine the path for traffic. If one customer gateway device fails, the virtual private gateway directs all traffic to the working customer gateway device.

You can establish multiple VPN connections from multiple customer gateway devices to a single virtual private gateway using AWS VPN CloudHub. This configuration can be used in different ways to implement redundancy and failover on your side of the VPN connection.

AWS VPN CloudHub operates on a hub-and-spoke model to enable multiple sites to access your VPC or to securely access each other. You can use it with or without a VPC. You configure each customer gateway device to advertise a site-specific prefix (such as *10.0.0.0/24*, *10.0.1.0/24*) to the virtual private gateway. The virtual private gateway routes traffic to the appropriate site and advertises the reachability of one site to all the other sites.

For more information about using AWS Site-to-Site VPN, see the following resources:

- Site-to-Site VPN single and multiple connection examples
- Using redundant Site-to-Site VPN connections to provide failover

Section 2 key takeaways

- AWS Site-to-Site VPN is a highly available solution that enables you to securely connect your on-premises network or branch office site to your VPC
- AWS Site-to-Site VPN supports both static and dynamic routing
- You can establish multiple VPN connections from multiple customer gateway devices to a single virtual private gateway

Some key takeaways from this section of the module include:

- AWS Site-to-Site VPN is a highly available solution that enables you to securely connect your on-premises network or branch office site to your VPC
- AWS Site-to-Site VPN supports both static and dynamic routing
- You can establish multiple VPN connections from multiple customer gateway devices to a single virtual private gateway

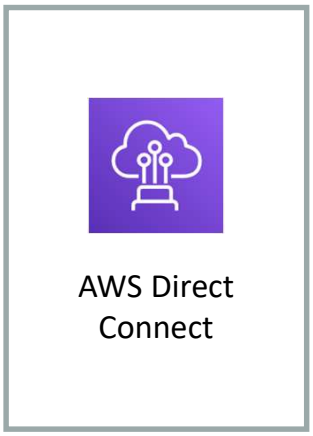
Module 7: Connecting Networks

Section 3: Connecting to your remote network with AWS Direct Connect

aws academy

Introducing Section 3: Connecting to your remote network with AWS Direct Connect.

AWS Direct Connect (DX)



AWS Direct Connect (which is also known as DX) provides you with a dedicated, private network connection capacity of either 1 Gbps or 10 Gbps.

Reduces data transfer costs

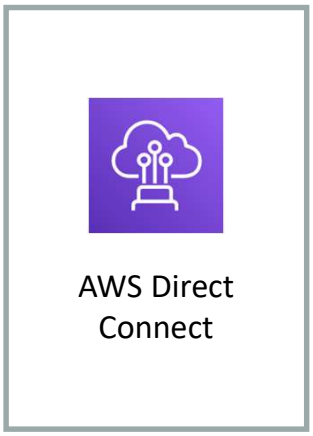
Improves application performance with predictable metrics

As you learned, AWS Site-to-Site VPN is one option for connecting your on-premises network to the AWS global network. With this option, your data is transferred through encrypted tunnels over the public internet.

AWS Direct Connect (or DX) is another solution that goes beyond simple connectivity over the internet. DX uses open standard 802.1q virtual local area networks (VLANs) so you can establish a dedicated, private network connection from your premises to AWS. This private connection can reduce network costs, increase bandwidth throughput, and provide a more consistent network experience than internet-based connections.

Dedicated connections are available with 1-Gbps and 10-Gbps capacity.

DX use cases

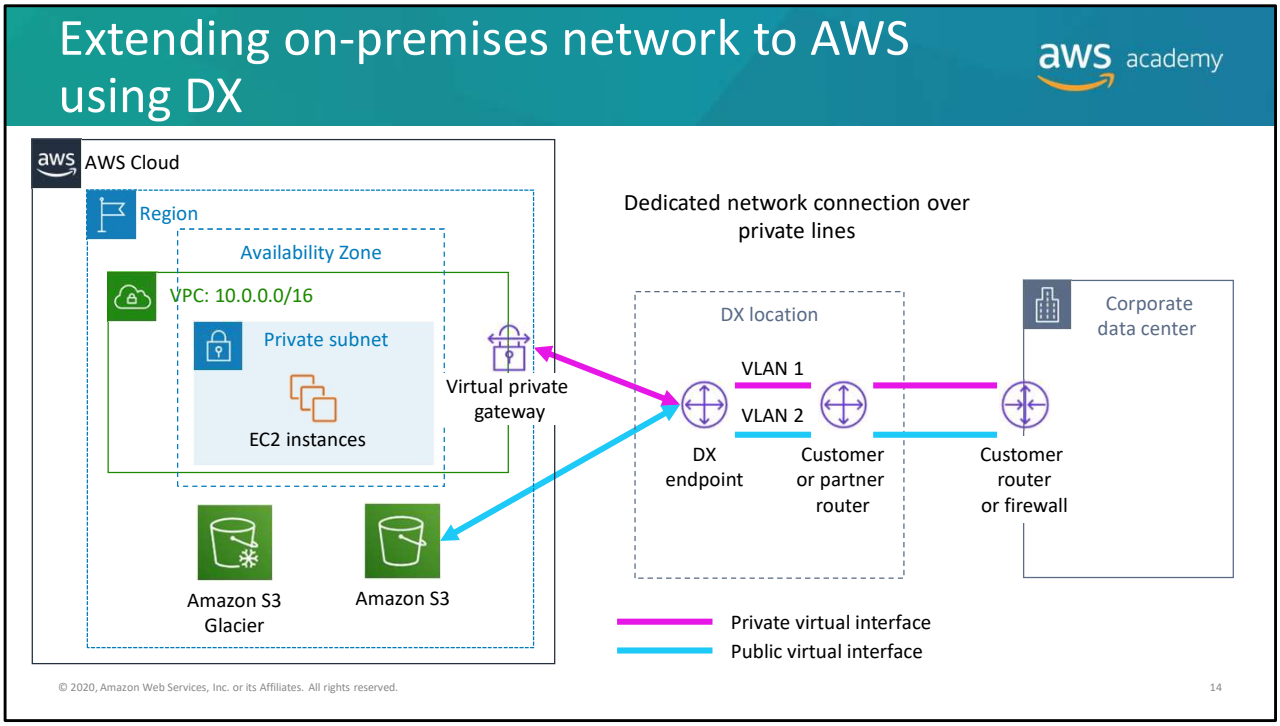


- Hybrid environments
- Transferring large datasets
- Network performance predictability
- Security and compliance

DX is useful for several scenarios, for example:

- *Hybrid environments* – For applications that require access to existing data center equipment, such as an on-premises database, DX enables you to create a hybrid environment that allows you to take advantage of the elasticity and economic benefits of AWS.
- *Transferring large datasets* – For applications that operate on large datasets, such as high performance computing (HPC) applications, transferring large datasets over the internet between your data center and the AWS Cloud can be time consuming and expensive. For such applications, connecting to the AWS Cloud using DX is a good solution because:
    - Network transfers will not compete for internet bandwidth at your data center.
    - The high-bandwidth link reduces the potential for network congestion and degraded application performance.
    - By limiting the internet bandwidth used by your application, you can reduce network fees that you pay to your internet service provider (ISP) and avoid having to pay for increased internet bandwidth commitments or new contracts. In addition, all data that is transferred over DX is charged at the reduced DX data transfer rate instead of internet data transfer rates, which can reduce your network costs.

- *Improved application performance* – Applications that require predictable network performance can also benefit from DX. Examples include applications that operate on real-time data feeds, such as audio or video streams. In such cases, a dedicated network connection can provide more consistent network performance than standard internet connectivity.
- *Security and compliance* – Enterprise security or regulatory policies sometimes require that applications hosted on the AWS Cloud can be accessed through private network circuits only. DX is a natural solution to this requirement because traffic between your data center and your application flows through the dedicated private network connection.

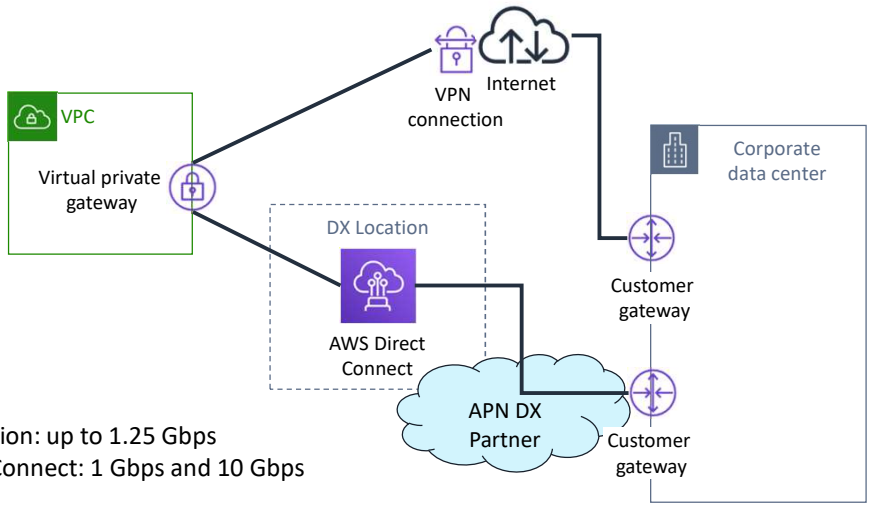


DX links your internal network to a DX location over a standard Ethernet fiber-optic cable. One end of the cable is connected to your router. The other end is connected to a DX router. With this connection, you can create *virtual interfaces* that enable direct access to AWS services. A public virtual interface enables access to public AWS services, such as Amazon Simple Storage Service (Amazon S3). A private virtual interface enables access to your VPC.

You can access any VPC or public AWS service in any Region (except China) from any supported DX location. If you do not have equipment at a DX location, you can access DX with the assistance of a DX AWS Partner Network (APN) Partner.

For information about DX, see What is AWS Direct Connect?

Enabling high availability: DX with backup VPN connection



- VPN connection: up to 1.25 Gbps
- AWS Direct Connect: 1 Gbps and 10 Gbps

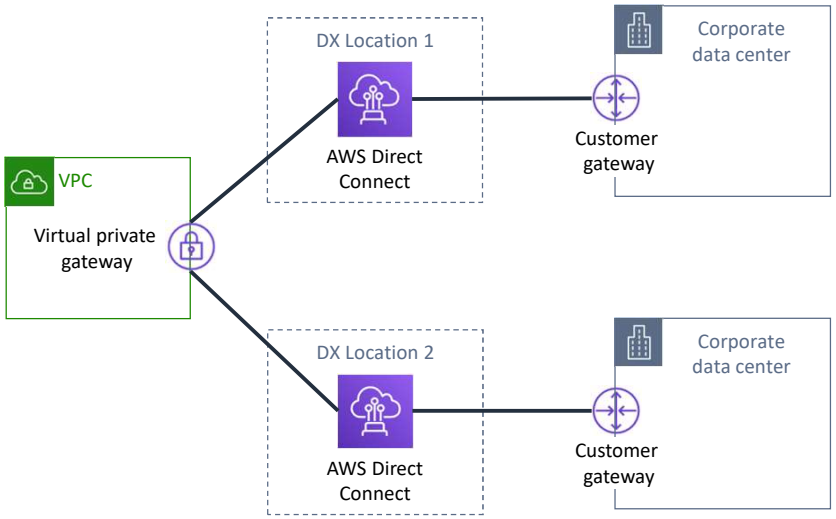
You can implement highly available connectivity between your data centers and your VPC by coupling one or more DX connections that you use for primary connectivity with a lower-cost backup VPN connection.

In this example, the configuration consists of two dynamically routed connections—one that uses DX and another that uses a VPN connection—from two different customer devices. AWS provides example router configurations to help you establish both DX and dynamically routed VPN connections. By default, AWS will always prefer to send traffic over your DX connection, so you do not need additional configuration specific to AWS to define primary and backup connections. However, you should configure DX and VPN-specific internal-route propagation to ensure that internal systems select the appropriate paths.

This approach enables you to choose the primary network path and network provider for your AWS traffic, with the option of using a different provider for a backup VPN connection. Choose network providers and DX locations that align with your organization's risk tolerance, financial expectations, and data center connectivity policies.

Finally, you can use multiple DX circuits and multiple VPN tunnels between separately deployed private IP address spaces. You can also use multiple DX locations for high availability. If you use multiple AWS Regions, you will also need multiple DX locations in at least two Regions. You might want to evaluate AWS Marketplace appliances that terminate VPNs.

Enabling high resiliency for critical workloads with DX



Highly resilient, fault-tolerant network connections are key to a well-architected system. AWS recommends connecting from multiple data centers for physical location redundancy. When you design remote connections, consider using redundant hardware and telecommunications providers.

Additionally, it is a best practice to use dynamically routed, active/active connections for automatic load balancing and failover across redundant network connections. Provision sufficient network capacity to ensure that the failure of one network connection does not overwhelm and degrade redundant connections.

For critical production workloads that require high resiliency, AWS recommends that you have one connection at multiple locations. As shown in the architecture diagram, such a topology ensures resilience against connectivity failures due to a hardware failure or a complete location failure. You can use Direct Connect Gateway to access any AWS Region (except AWS Regions in China) from any DX location.

To learn more about additional topology guidelines to keep in mind when you connect to AWS, see AWS Direct Connect Resiliency Recommendations.

Section 3 key takeaways

- AWS Direct Connect uses open standard 802.1q VLANs that enable you to establish a dedicated, private network connection from your premises to AWS
- You can access any VPC or public AWS service in any Region (except China) from any supported DX location
- You can implement highly available connectivity between your data centers and your VPC by coupling one or more DX connections that you use for primary connectivity with a lower-cost, backup VPN connection
- To implement a highly resilient, fault-tolerant architecture, connect to your AWS network from multiple data centers so you can have physical location redundancy

Some key takeaways from this section of the module include:

- AWS Direct Connect uses open standard 802.1q VLANs to let you establish a dedicated, private network connection from your premises to AWS
- You can access any VPC or public AWS service in any Region (except China) from any supported DX location
- You can implement highly available connectivity between your data centers and your VPC by coupling one or more DX connections that you use for primary connectivity with a lower-cost, backup VPN connection
- To implement a highly resilient, fault-tolerant architecture, connect to your AWS network from multiple data centers so you can have physical location redundancy

Module 7: Connecting Networks

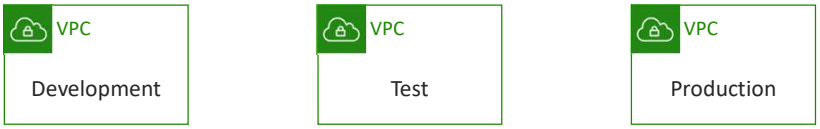
Section 4: Connecting VPCs in AWS with VPC peering

aws academy

Introducing Section 4: Connecting VPCs in AWS with VPC peering.

Connecting VPCs

- Isolating some of your workloads is generally a good practice
- However, you might need to transfer data between two or more VPCs



Isolating your workloads in individual VPCs is generally a good practice. For example, when your business or architecture becomes large enough, you might need to separate logical elements for security, for architectural purposes, or for simplicity. However, it can be highly desirable to have connectivity between your VPCs for situations when you need to transfer data between them.

VPC peering

- One-to-one networking connection between two VPCs
- No gateways, VPN connections, and separate network appliances needed
- Highly available connections
- No single point of failure or bandwidth bottleneck
- Traffic always stays on the global AWS backbone

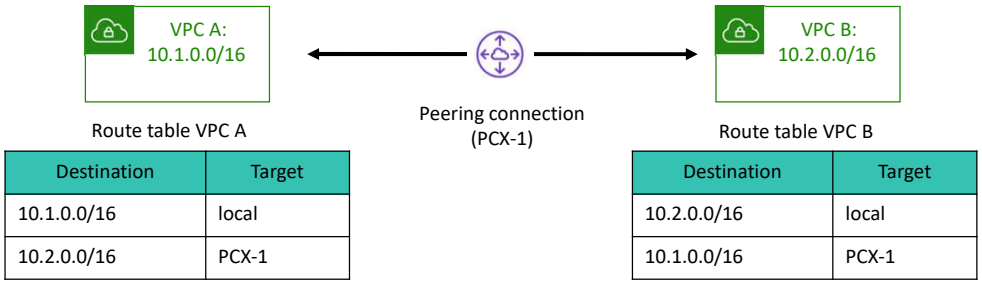
VPC peering is a one-to-one networking connection between two VPCs that enables you to route traffic between them privately. Instances in either VPC can communicate with each other like they are in the same network. You can create a VPC peering connection between your own VPCs, with a VPC in another AWS account, or with a VPC in a different AWS Region.

You can establish peering relationships between VPCs across different AWS Regions. Inter-Region VPC peering provides a simple and cost-effective way to share resources between Regions or replicate data for geographic redundancy. Data that is transferred across inter-Region VPC peering connections is charged at the standard inter-Region data transfer rates.

Inter-Region VPC peering enables VPC resources to communicate with each other using private IP addresses without requiring gateways, VPN connections, or separate network appliances. Some examples of VPC resources include Amazon Elastic Compute Cloud (Amazon EC2) instances, Amazon Relational Database Service (Amazon RDS) databases, and AWS Lambda functions that run in different Regions.

Traffic remains in the private IP address space. All inter-Region traffic is encrypted with no single point of failure or bandwidth bottleneck. Traffic always stays on the global AWS backbone. Traffic never traverses the public internet, which reduces threats, such as common exploits and distributed denial of service (DDoS) attacks.

Establishing VPC peering



Route table VPC A

Destination	Target
10.1.0.0/16	local
10.2.0.0/16	PCX-1

Route table VPC B

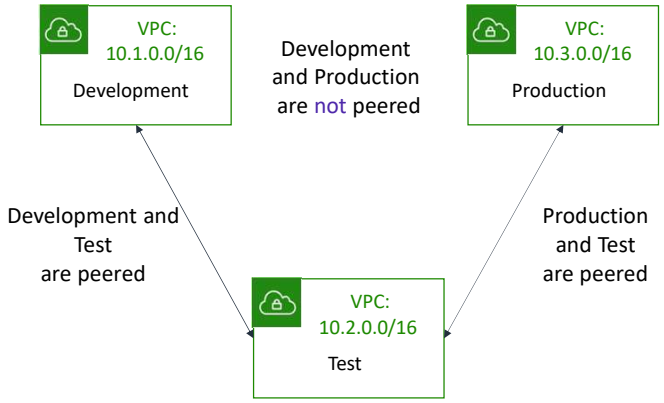
Destination	Target
10.2.0.0/16	local
10.1.0.0/16	PCX-1

To establish a VPC peering connection, the owner of the requester VPC (or local VPC) sends a request to the owner of the VPC peer. To activate the connection, the owner of the VPC peer must accept the VPC peering connection request.

To enable the flow of the traffic between the VPC peers by using private IP addresses, you must add a route to one or more of your VPC's route tables. This route must point to the IP address range of the VPC peer. The owner of the VPC peer adds a route to one of their VPC's route tables that points to the IP address range of your VPC.

You might also need to update the security group rules that are associated with your instance so that traffic to and from the peer VPC is not restricted.

VPC peering connection restrictions



- Use private IP addresses
- Can be established between different AWS accounts
- Cannot have overlapping CIDR blocks
- Can have only one peering resource between any two VPCs
- Do not support transitive peering relationships

There are some restrictions you should be aware of when you establish VPC peering connections:

- VPC peering connections use private IP addresses.
- VPC peering connections can be established between different AWS accounts. The CIDR block of the VPC peer cannot overlap with the CIDR block of the requester.
- You can have only one peering resource between any two VPCs.
- Transitive peering is not supported. For example, in the diagram, the Development and Test VPCs are peered, and the Production and Test VPCs are peered. However, this does not mean that the Production VPC is connected to the Development VPC. By default, VPC peering does not allow the Production VPC to connect to the Development VPC unless they are *explicitly established as peers.* Therefore, you control which VPCs can communicate with each other.

To learn more about VPC peering connection restrictions, see VPC peering limitations.

Considerations for peering multiple VPCs

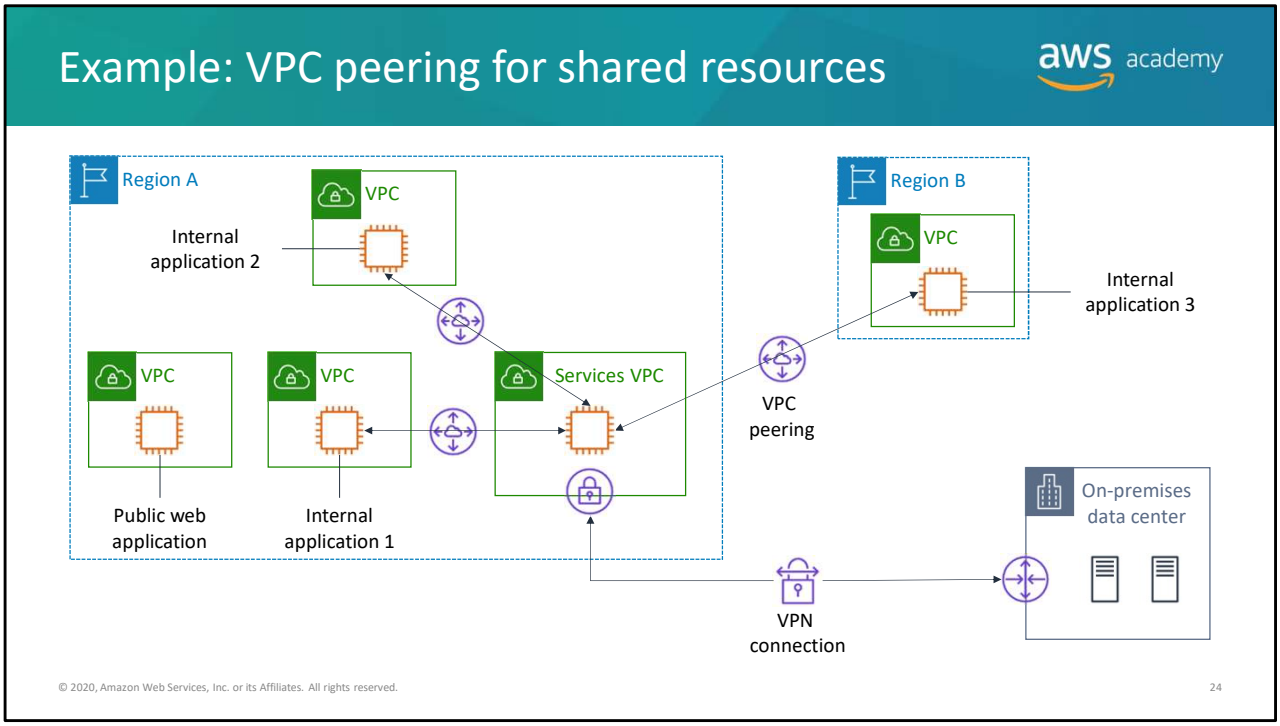
When you connect multiple VPCs, consider these network design principles:

Only connect essential VPCs

Make sure your solution can scale

When you connect multiple VPCs in a single AWS Region, consider these network design principles:

- Connect only those VPCs that truly must communicate with each other
- Make sure that the solution you choose can scale according to your current and future VPC connectivity needs



Here is an example of how you can use VPC peering for shared resources.

In this example, each department VPC in a company peers with a shared *Services VPC*. This VPC contains connections to Microsoft Active Directory, security scanning tools, monitoring and logging tools, and various other capabilities. It also provides a proxy through which the department VPCs can access some on-premises resources. VPC peering enables company applications that are in different VPCs to access the shared Services VPC but remain isolated from each other. In this example, also notice that a VPC peering connection was established between VPCs in different Regions.

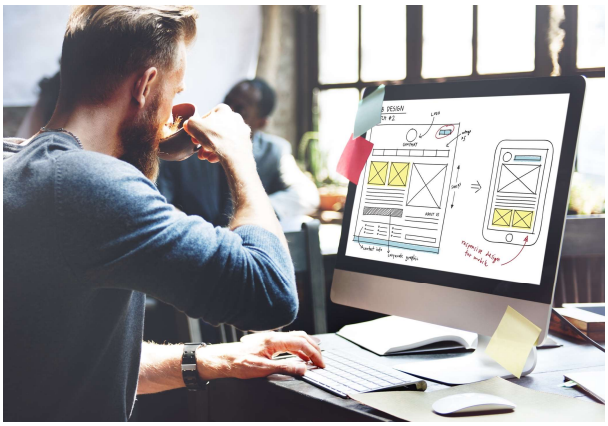
Section 4 key takeaways

- VPC peering is a one-to-one networking connection between two VPCs that enables you to route traffic between them privately
- You can establish peering relationships between VPCs across different AWS Regions
- VPC peering connections –
  - Use private IP addresses
  - Can be established between different AWS accounts
  - Cannot have overlapping CIDR blocks
  - Can have only one peering resource between any two VPCs
  - Do not support transitive peering relationships

Some key takeaways from this section of the module include:

- VPC peering is a one-to-one networking connection between two VPCs that enables you to route traffic between them privately
- You can establish peering relationships between VPCs across different AWS Regions
- VPC peering connections –
  - Use private IP addresses
  - Can be established between different AWS accounts
  - Cannot have overlapping CIDR blocks
  - Can have only one peering resource between any two VPCs
  - Do not support transitive peering relationships

Module 7 – Guided Lab: Creating a VPC Peering Connection



You will now complete Module 7 – Guided Lab: Creating a VPC Peering Connection.

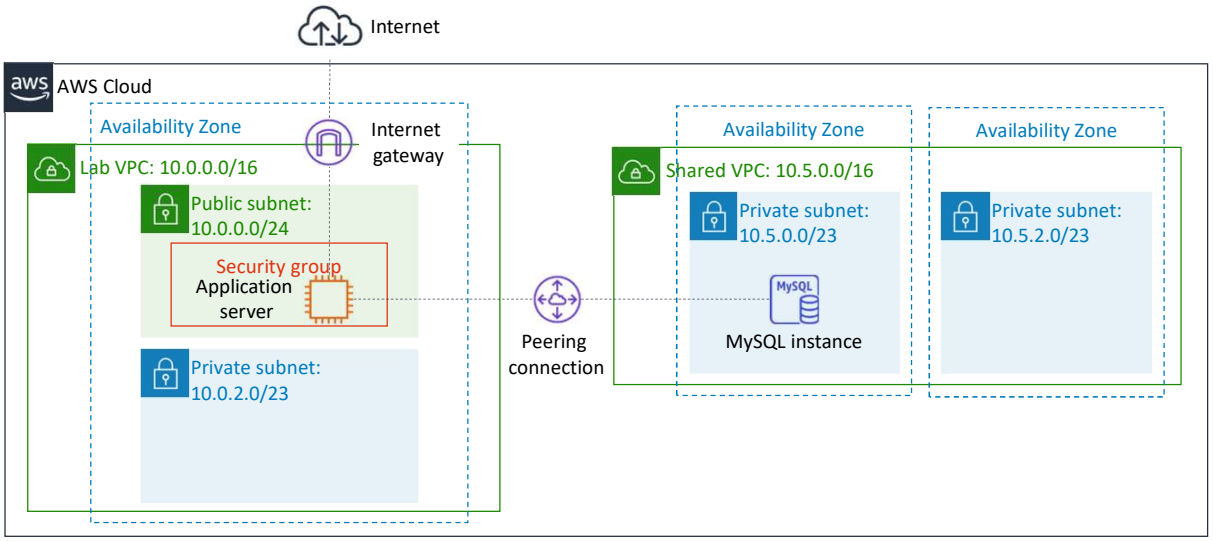
Guided lab: Tasks

1. Create a peering connection between two VPCs
2. Configure route tables to send traffic to the peering connection
3. Test the peering connection

In this guided lab, you will complete the following tasks:

1. Create a peering connection between two VPCs
2. Configure route tables to send traffic to the peering connection
3. Test the peering connection

Guided lab: Final product



The diagram summarizes what you will have built after you complete the lab.

~ 20 minutes

It is now time to start the guided lab.

Guided lab debrief: Key takeaways

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

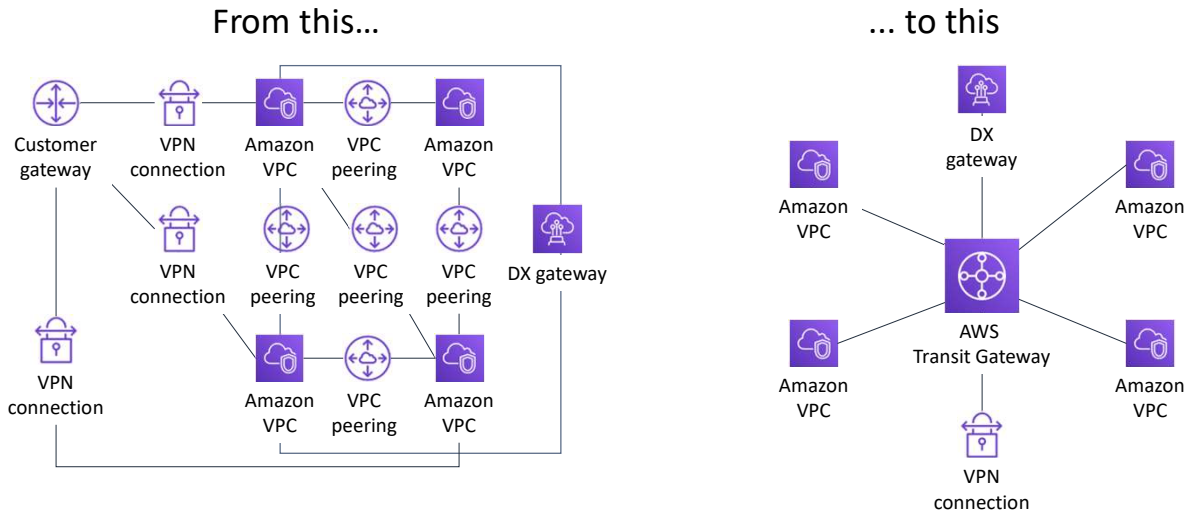
Module 7: Connecting Networks

Section 5: Scaling your VPC network with AWS Transit Gateway

aws academy

Introducing Section 5: Scaling your VPC network with AWS Transit Gateway.

Need to scale networks across multiple VPCs



As you grow the number of workloads that run on AWS, you must be able to scale your networks across multiple accounts and VPCs to keep up with the growth. You can use VPC peering to connect pairs of VPCs. However, it can be operationally costly and difficult to manage point-to-point connectivity across many VPCs if you cannot centrally manage the connectivity policies. For on-premises connectivity, you must attach your VPN to each individual VPC. This solution can be time-consuming to build and difficult to manage when the number of VPCs grows into the hundreds.

It's important to consider how large your environment might become over time, how well it will scale, and how you will organize your VPCs. To solve this problem, you can use AWS Transit Gateway to simplify your networking model.

AWS Transit Gateway

AWS Transit Gateway

AWS Transit Gateway is a service that enables you to connect your VPCs and on-premises networks to a single gateway.

- Fully managed, highly available, flexible routing service
- Acts as a hub for all traffic to flow through between your networks
- Connects up to 5,000 VPCs and on-premises environments with a single gateway

AWS Transit Gateway is a service that enables you to connect your VPCs and on-premises networks to a single gateway (called a transit gateway). With AWS Transit Gateway, you only need to create and manage a single connection from the central gateway into each VPC, on-premises data center, or remote office across your network.

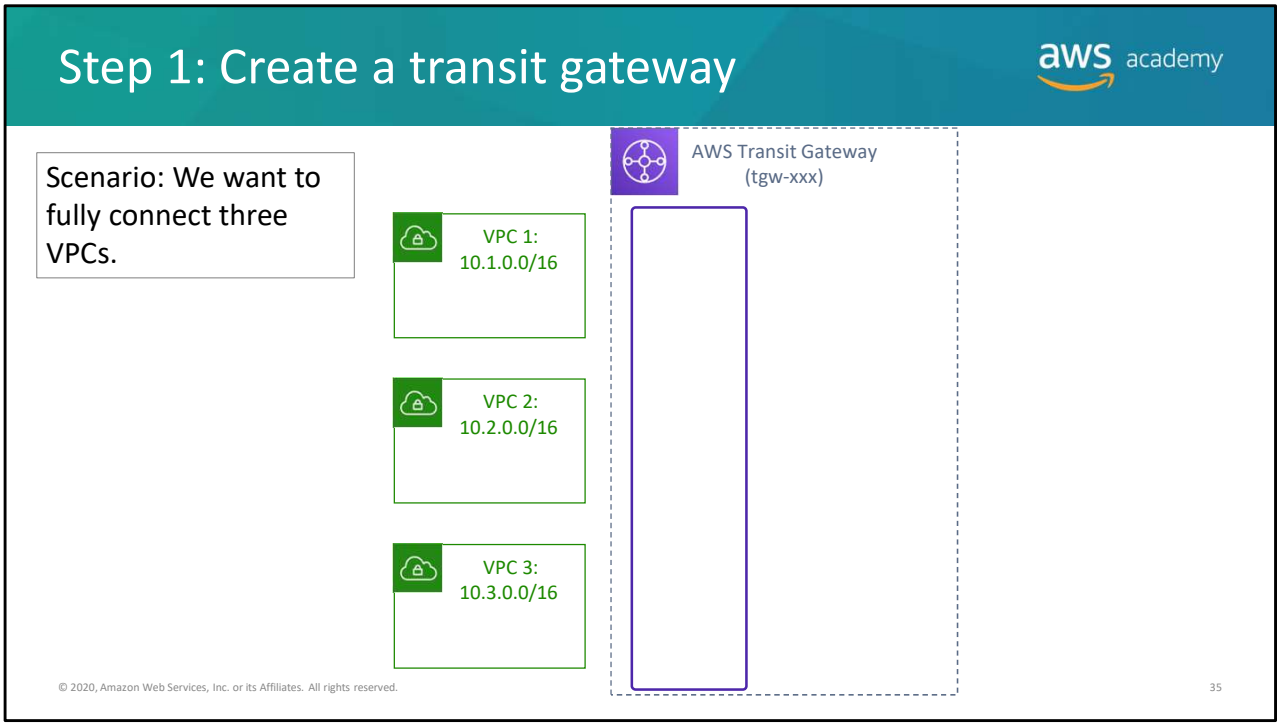
AWS Transit Gateway uses a hub-and-spoke model. This model significantly simplifies management and reduces operational costs because each network only needs to connect to the transit gateway and not to every other network. Any new VPC is connected to the transit gateway, and is then automatically available to every other network that is connected to the transit gateway. This ease of connectivity makes it easier to scale your network as you grow.

You can use AWS Transit Gateway to connect up to 5,000 VPCs and on-premises networks.

Connecting multiple VPCs

Scenario: We want to fully connect three VPCs.

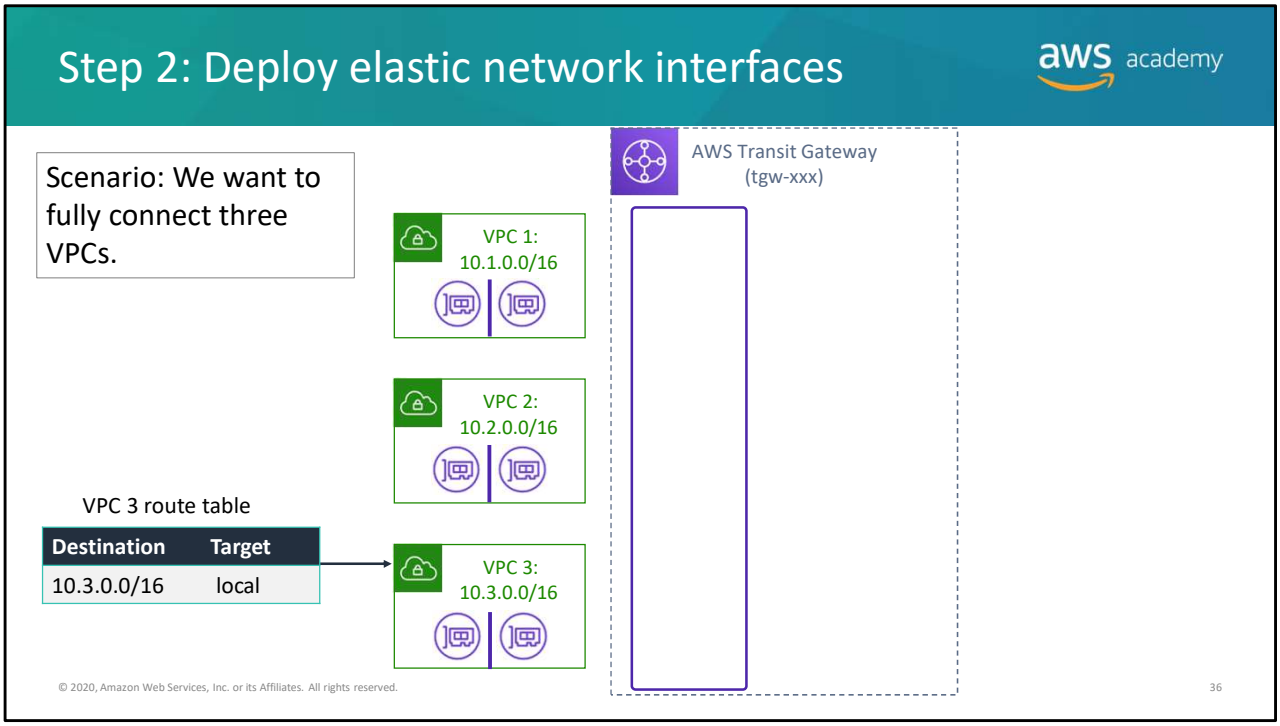
To understand how to connect multiple VPCs by using AWS Transit Gateway, consider this scenario. You want to fully connect three VPCs in your network. In this scenario, you learn how to deploy AWS Transit Gateway and three VPCs with non-overlapping IP address space in a single Region. You then attach the transit gateway to these VPCs.



The first step for connecting multiple VPCs is to create a transit gateway. A *transit gateway* is a transit hub that you can use to interconnect your VPCs and on-premises networks. It acts as a Regional virtual router for traffic that flows between your VPCs and VPN connections. A transit gateway scales elastically based on the volume of network traffic.

You can set up a transit gateway through the Amazon VPC dashboard. Various charges apply for using AWS Transit Gateway, so make sure that your architecture and budget can support using a transit gateway.

For more information, see What Is a Transit Gateway?

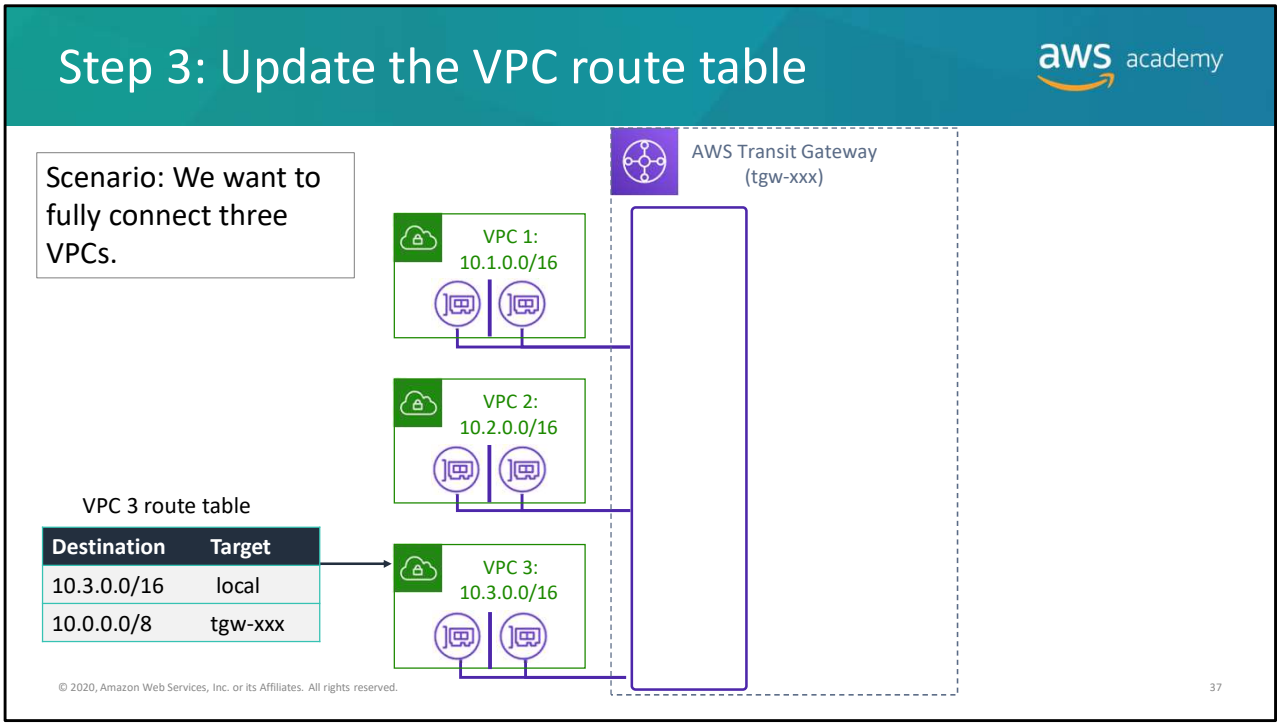


AWS Transit Gateway connects to a VPC though elastic network interfaces (or ENIs), which are deployed into subnets.

You must ensure that every Availability Zone that is part of the VPC has an ENI that connects the VPC to the transit gateway. You can do this by selecting at least one subnet from each Availability Zone for the ENI.

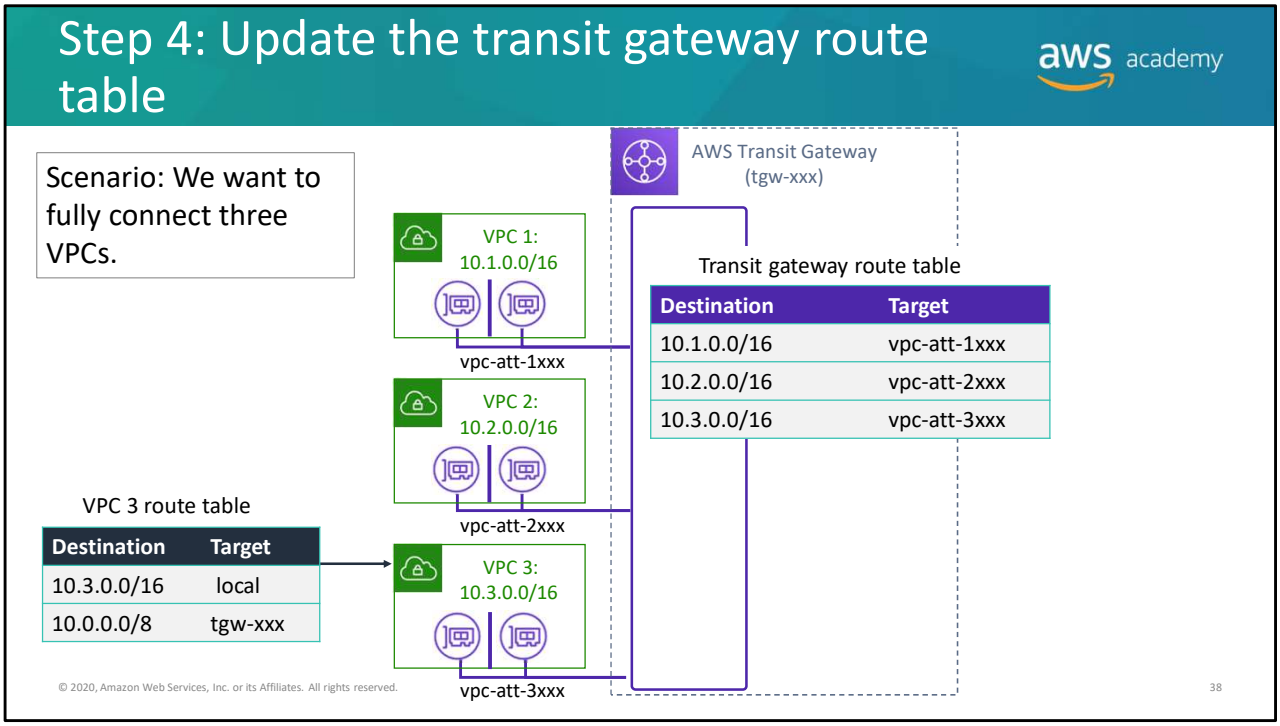
Notice in this example that the route table for VPC 3 has a single destination route that is local for VPC 3 using the 10.3.0.0/16 network.

Step 3: Update the VPC route table



After attaching the ENIs, the next step is to add a route in the VPC route table to send traffic that's destined for the other VPCs in the network to the transit gateway.

In this example, the second line of the VPC 3 route table shows that traffic destined for the 10.0.0.0/8 network is sent to the transit gateway. This route enables any traffic from VPC 3 going to either VPC 1 or VPC 2 to be sent to the transit gateway, because the CIDR block 10.0.0.0/8 includes the 10.X.0.0/16 CIDR blocks, which are used by the individual VPCs.

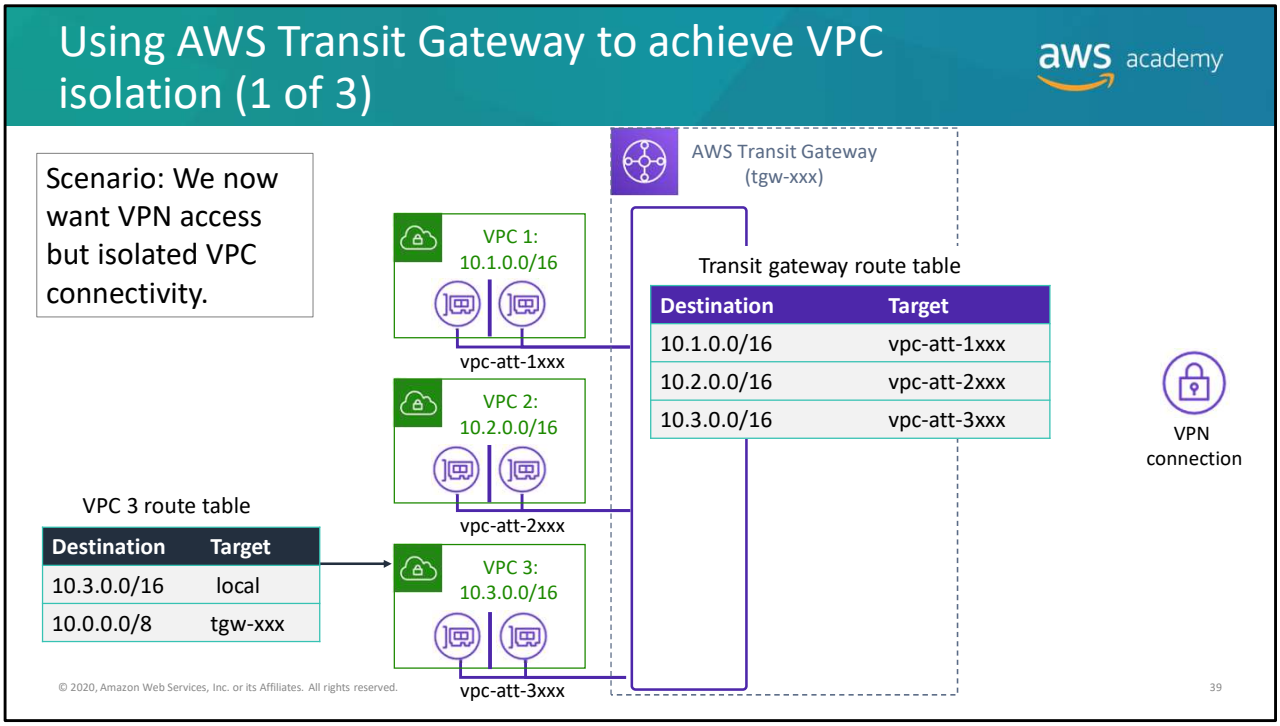


Next, you must configure the transit gateway route table to route traffic to the connected VPCs.

When you create a transit gateway, a default transit gateway route table is created. Each route in the transit gateway route table enables the transit gateway to send traffic destined for one of the VPCs to a corresponding attachment, which is a reference to the ENI that is attached to the VPC itself.

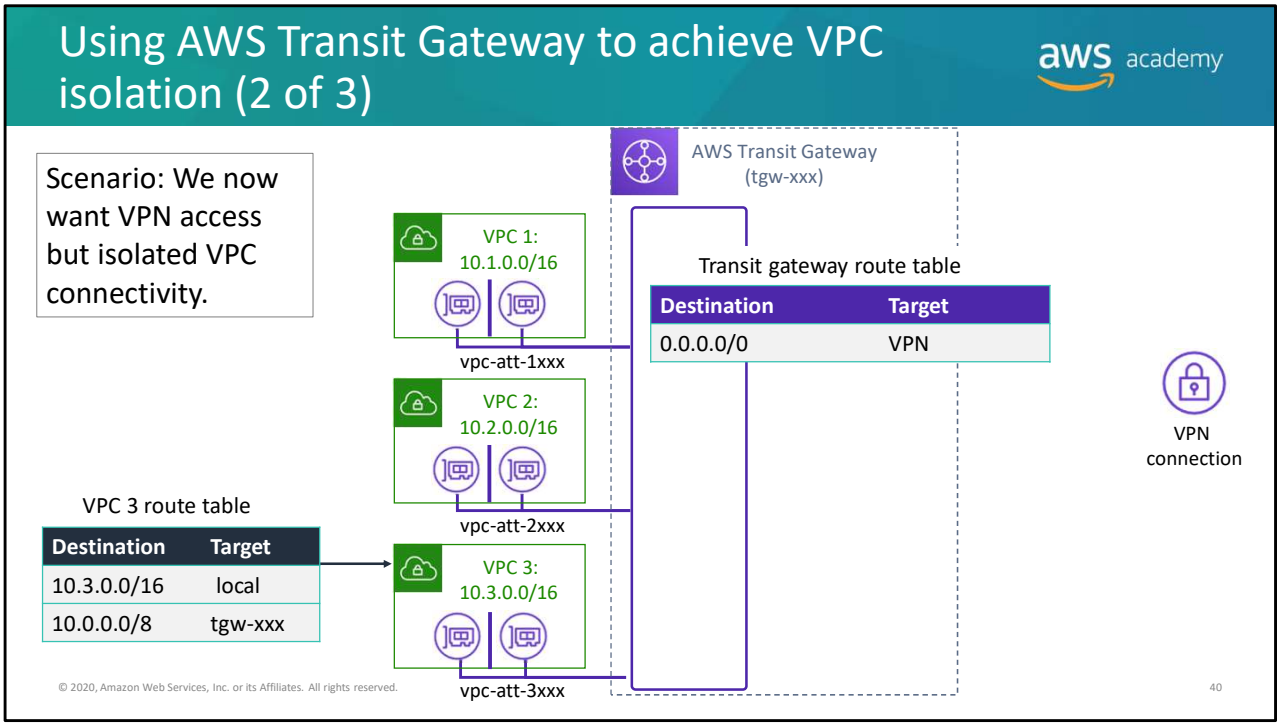
In this example, there is a route in the transit gateway route table that sends any traffic destined for the 10.1.0.0/16 network to vpc-att-1xxx, which is the attachment for VPC 1. Similarly, any traffic destined for the other VPC networks are sent to the corresponding attachments.

For more information about how to create connected environments using AWS Transit Gateway, see Getting Started with Transit Gateways.



Though you can use AWS Transit Gateway to connect multiple VPCs, you can also use it to achieve isolation in your VPC environment. In this scenario, you want to connect your VPN source to your VPC environment. You also want to prevent your VPCs from directly connecting to each other, leaving the VPN to decide if traffic from one VPC has to be forwarded to another.

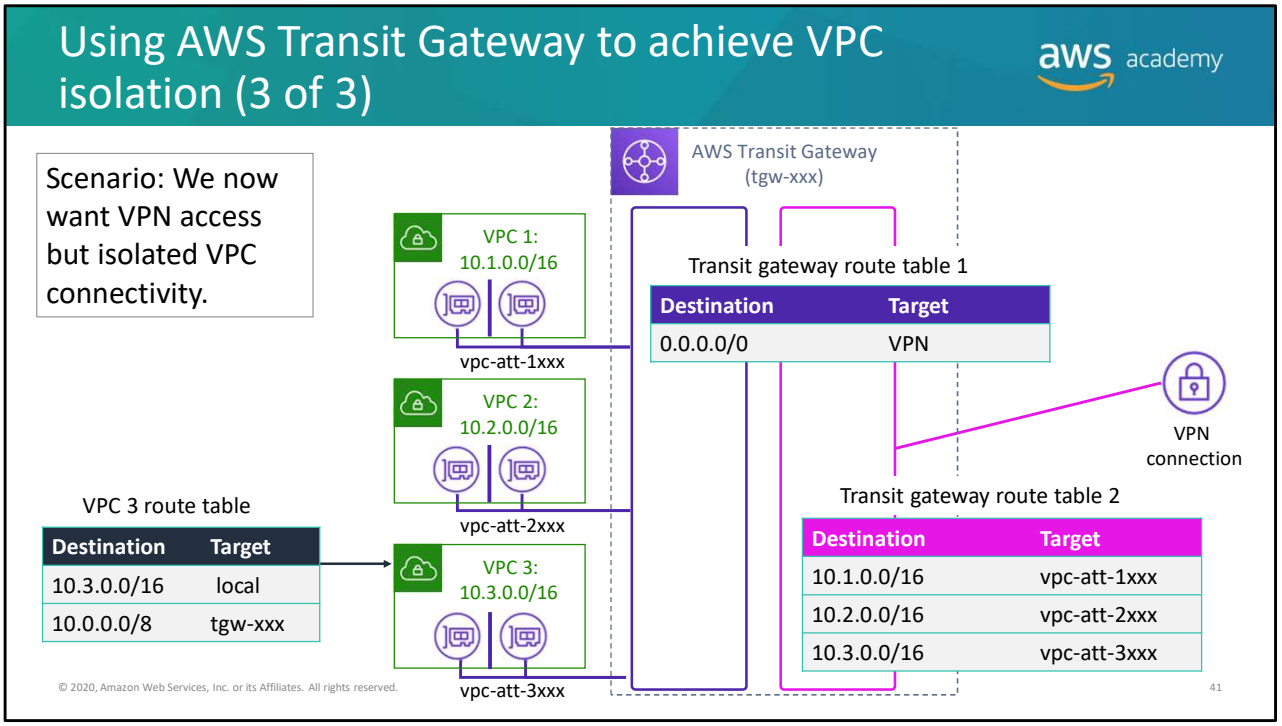
By setting up the route table appropriately for the transit gateway, you can prevent information sharing between the VPCs.



To implement this solution, update the route in the transit gateway route table to send all known traffic to the VPN connection.

In this example, when traffic for any of the VPCs in the 10.0.0.0/8 network is sent from VPC 3 to the transit gateway, the transit gateway will forward the traffic to the VPN as shown on the slide. The transit gateway will not send the traffic to any of the other VPCs because there are no routes pointing to any of the VPC attachments.

You now have isolated and secured VPN access to your VPC environment with no cross communication between the VPCs.



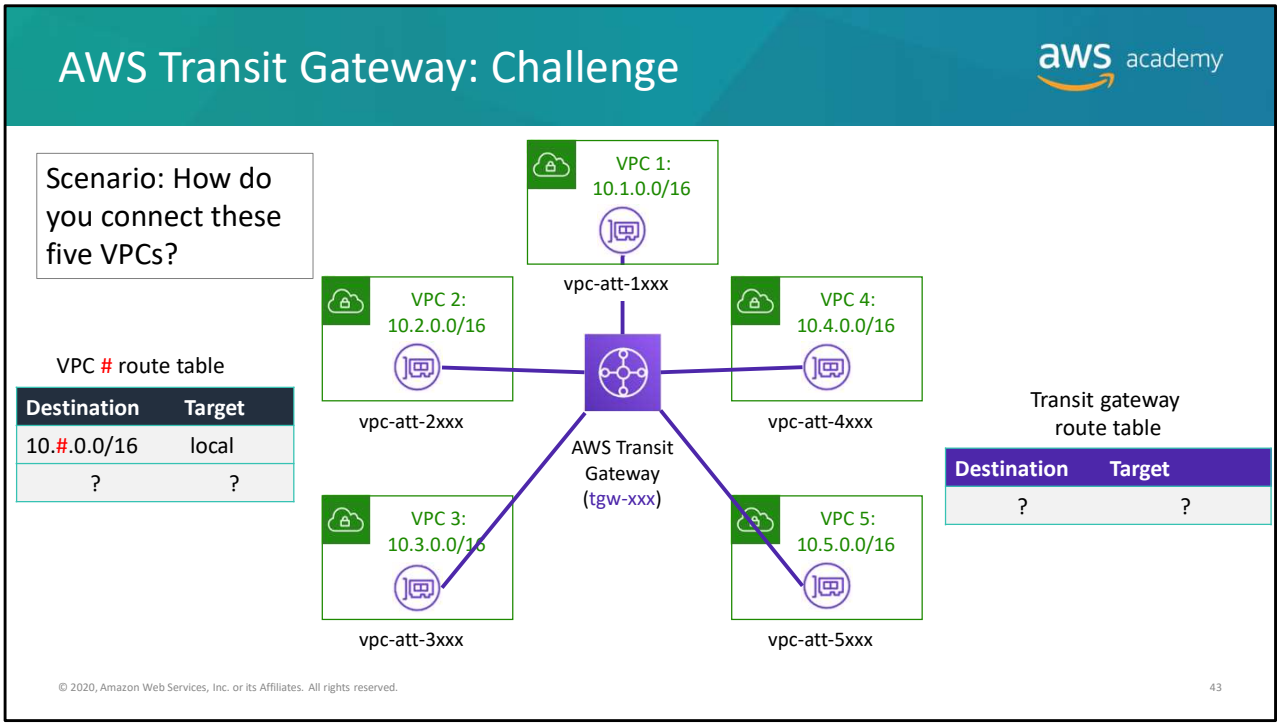
You can create multiple transit gateway route tables for specific interactions to direct traffic, as you see fit.

In this example, the second route table will direct inbound traffic from the VPN to one of the corresponding VPCs attached to the transit gateway.

Activity: AWS Transit Gateway

You will now complete the following activity: AWS Transit Gateway.

AWS Transit Gateway: Challenge



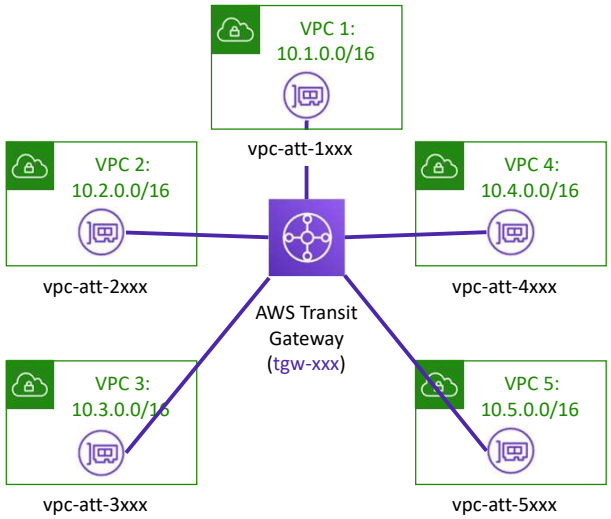
In this activity, you have five VPCs that you want connect to each other through AWS Transit Gateway.

Answer the following questions:

- What routes are necessary to add to each of the VPC route tables to enable full connectivity?
- What routes are necessary to add to the transit gateway route table to enable full connectivity?

AWS Transit Gateway activity: Solution

Scenario: How do you connect these five VPCs?



VPC 3 route table

Destination	Target
10.3.0.0/16	local
10.0.0.0/8	tgw-xxx

Transit gateway route table

Destination	Target
10.1.0.0/16	vpc-att-1xxx
10.2.0.0/16	vpc-att-2xxx
10.3.0.0/16	vpc-att-3xxx
10.4.0.0/16	vpc-att-4xxx
10.5.0.0/16	vpc-att-5xxx

What routes must you add in each VPC route table to enable full connectivity?

- See the provided solution for the VPC 3 route table. You update the other VPC route tables in a similar way.

What routes must you add to the transit gateway route table to enable full connectivity?

- Add a route for each VPC attachment to direct traffic to each VPC.

Section 5 key takeaways

- AWS Transit Gateway enables you to connect your VPCs and on-premises networks to a single gateway (called a transit gateway)
- AWS Transit Gateway uses a hub-and-spoke model to simplify VPC management and reduce operational costs

Some key takeaways from this section of the module include:

- AWS Transit Gateway enables you to connect your VPCs and on-premises networks to a single gateway (called a transit gateway)
- AWS Transit Gateway uses a hub-and-spoke model to simplify VPC management and reduce operational costs

Module 7: Connecting Networks

Section 6: Connecting your VPC to supported AWS services

aws academy

Introducing Section 6: Connecting your VPC to supported AWS services.

VPC endpoints

- Enable you to privately connect your VPC to supported AWS services and to VPC endpoint services that are powered by AWS PrivateLink
- Enable traffic between your VPC and the other service without leaving the Amazon network
- Do not require an internet gateway, VPN, network address translation (NAT) devices, or firewall proxies
- Are horizontally scaled, redundant, and highly available

A *VPC endpoint* enables you to privately connect your VPC to supported AWS services and to VPC endpoint services that are powered by AWS PrivateLink. VPC endpoint services that are powered by AWS PrivateLink include some AWS services, services hosted by other AWS customers and AWS Partner Network (APN) Partners in their own VPCs (which are referred to as *endpoint services*), and supported AWS Marketplace Partner services.

VPC endpoints do not require an internet gateway, NAT device, VPN connection, or DX connection. Instances in your VPC do not require public IP addresses to communicate with resources in the service. Traffic between your VPC and the other service does not leave the Amazon network.

Endpoints are virtual devices. They are horizontally scaled, redundant, and highly available VPC components. Endpoints allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic.

Two types of VPC endpoints

- Interface endpoint – An elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported service
- Powered by AWS PrivateLink
- Examples –
  - Amazon CloudWatch
  - Amazon EC2 API
  - Elastic Load Balancing

- Gateway endpoint – A gateway that you specify as a target for a route in your route table for traffic destined to a supported AWS service
- Supported AWS services –
  - Amazon S3
  - Amazon DynamoDB

There are two types of VPC endpoints:

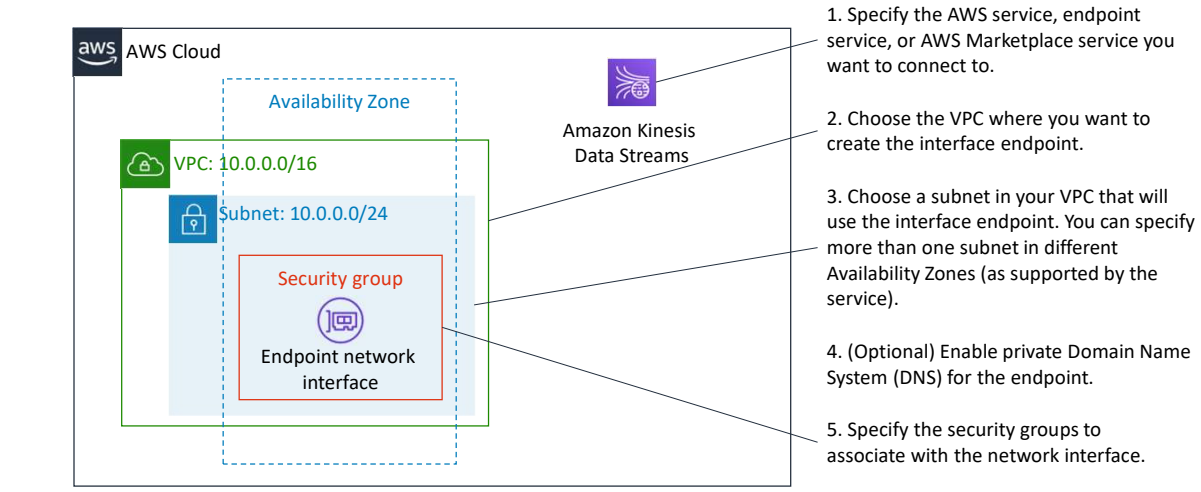
- An *interface endpoint* is an elastic network interface with a private IP address. This IP address serves as an entry point for traffic that is destined to a supported service. Interface endpoints enable you to connect to services that are powered by AWS PrivateLink. The owner of the service is the *service provider.* As the principal who creates the interface endpoint, you are the *service consumer*. For a complete list of services that are supported by interface endpoints, see VPC Endpoints – Interface Endpoints.

- A *gateway endpoint* is a gateway that you specify as a target for a route in your route table. The route is for traffic that is destined to a supported AWS service. Amazon S3 and Amazon DynamoDB are supported by gateway endpoints.

There are no data processing or hourly charges for using gateway VPC endpoints. However, you will be billed for each hour that your VPC endpoint remains provisioned in each Availability Zone, regardless of the state of its association with the service. This hourly billing for your VPC endpoint will stop when you delete it. Hourly billing will also stop if the endpoint service owner rejects your VPC endpoint's attachment to their service. That service is subsequently deleted. For more information about interface endpoint pricing, see AWS PrivateLink pricing.

To learn more about VPC endpoints, see VPC Endpoints in the AWS Documentation.

How to set up an interface endpoint



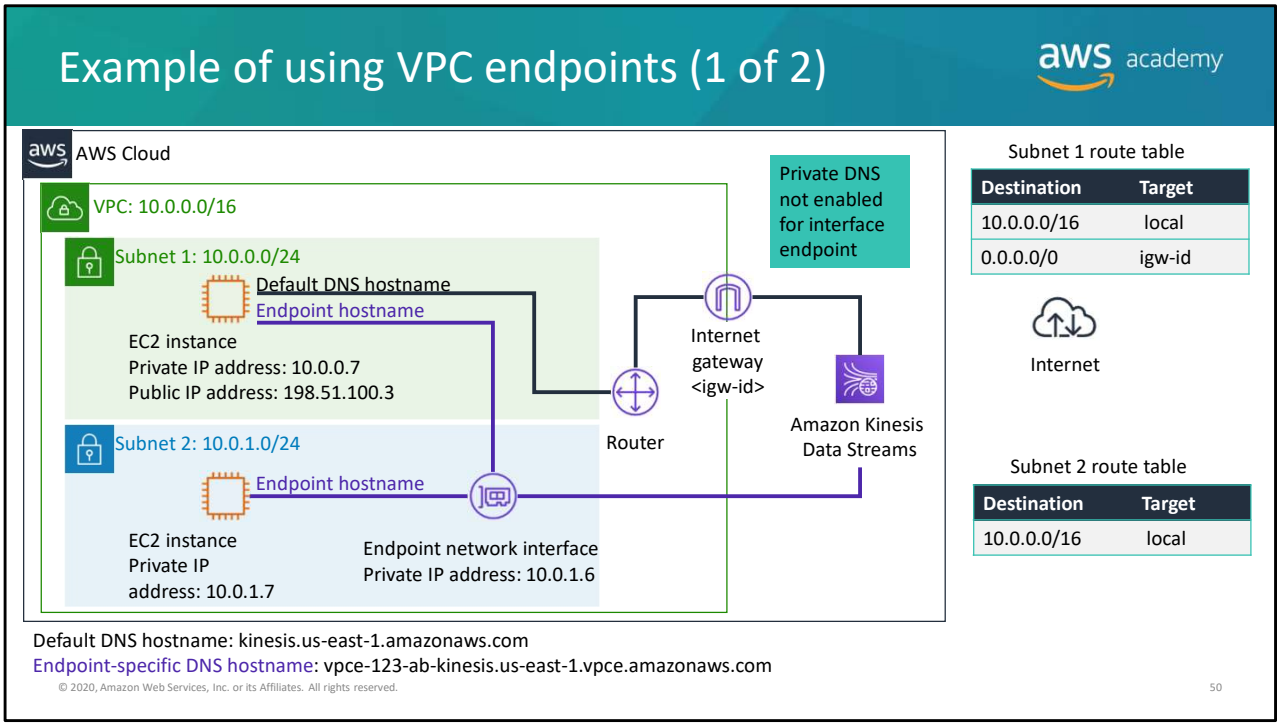
To set up an interface endpoint, follow these general steps from the Amazon VPC console:

1. Specify the name of the AWS service, endpoint service, or AWS Marketplace service you want to connect to.
2. Choose the VPC where you want to create the interface endpoint. You can specify more than one subnet in different Availability Zones (as supported by the service). Doing so helps ensure that your interface endpoint is resilient to Availability Zone failures. In that case, an endpoint network interface is created in each subnet that you specify.
3. Choose a subnet in your VPC that will use the interface endpoint. When you create an interface endpoint for a service in your VPC, an *endpoint network interface* is created in the selected subnet. The endpoint network interface has a private IP address that serves as an entry point for traffic destined to the service.
4. (Optional) Enable private Domain Name System (DNS) for the endpoint. Doing so enables you to make requests to the service by using its default DNS hostname (which is enabled by default for endpoints created for AWS services and AWS Marketplace Partner services).
5. Specify the security groups to associate with the network interface. The security group rules control the traffic to the endpoint network interface from resources in your VPC. If you do not specify a security group, the default security group for the VPC is used.

Services cannot initiate requests to resources in your VPC through the endpoint. An endpoint only returns responses to traffic that are initiated from resources in your VPC.

For details about how to create interface endpoints, see:

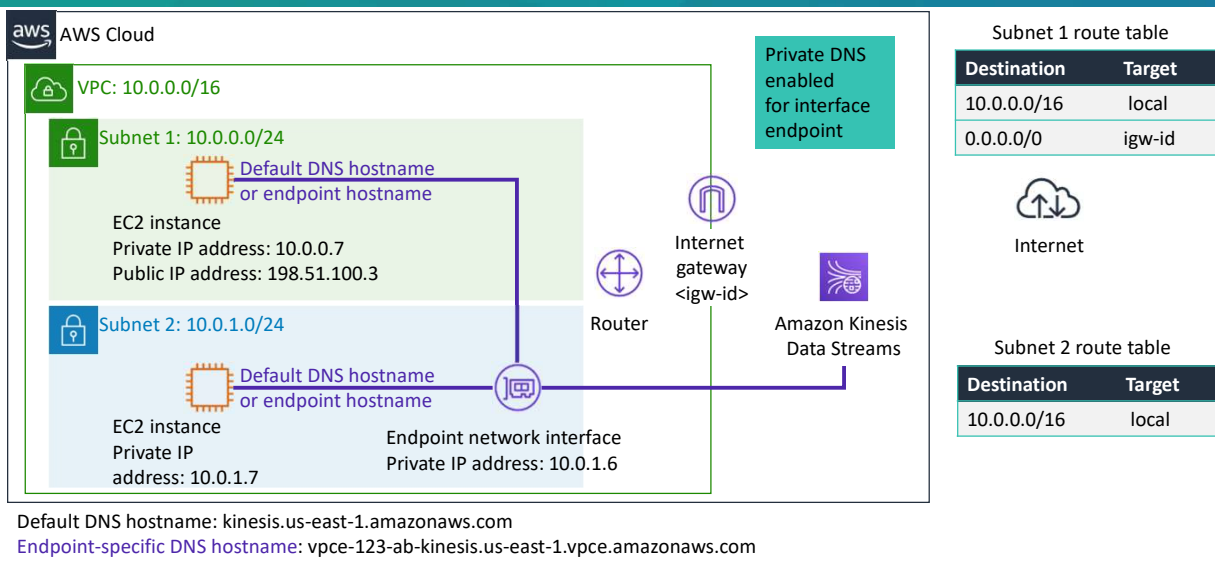
- Creating an Interface Endpoint
- What is an Interface VPC Endpoint and How Can I Create Interface Endpoints for My VPC?



When you create an interface endpoint, endpoint-specific DNS hostnames are generated that you can use to communicate with the service. For AWS services and AWS Marketplace Partner services, the private DNS option (enabled by default) associates a private hosted zone with your VPC. The hosted zone contains a record set for the default DNS name for the service (for example, kinesis.us-east-1.amazonaws.com) that resolves to the private IP addresses of the endpoint network interfaces in your VPC. This enables your application to make requests to the service using its default DNS hostname instead of the endpoint-specific DNS hostnames. This allows your existing applications to make requests to an AWS service through the interface endpoint without requiring any configuration changes.

In this example, there is an interface endpoint for Amazon Kinesis Data Streams and an endpoint network interface in subnet 2. Private DNS for the interface endpoint has **not** been enabled. Instances in either subnet can send requests to Amazon Kinesis Data Streams through the interface endpoint by using an endpoint-specific DNS hostname. Instances in subnet 1 can communicate with Amazon Kinesis Data Streams over the public IP address space in the AWS Region by using the service's default DNS hostname.

Example of using VPC endpoints (2 of 2)



Here, private DNS for the endpoint has been enabled. Instances in either subnet can use either the default DNS hostname or the endpoint-specific DNS hostname to send requests to Amazon Kinesis Data Streams through the interface endpoint.

For more information about this example, see Private DNS for interface endpoints.

Section 6 key takeaways

- A VPC endpoint enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by AWS PrivateLink
- VPC endpoints do not require an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection
- There are two types of VPC endpoints: interface endpoints and gateway endpoints

Some key takeaways from this section of the module include:

- A VPC endpoint enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by AWS PrivateLink
- VPC endpoints do not require an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection
- There are two types of VPC endpoints: interface endpoints and gateway endpoints

Module 7: Connecting Networks

Module wrap-up

aws academy

It’s now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Describe how to connect an on-premises network to the AWS Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud by using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services

In summary, in this module, you learned how to:

- Describe how to connect an on-premises network to the AWS Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services

Complete the knowledge check

It is now time to complete the knowledge check for this module.

Sample exam question

An application running on Amazon Elastic Compute Cloud (Amazon EC2) instances processes sensitive information stored on Amazon Simple Storage Service (Amazon S3). The information is accessed over the internet. The security team is concerned that the internet connectivity to Amazon S3 is a security risk.

Which solution will resolve the security concern?

A. Access the data through an internet gateway.
B. Access the data through a VPN connection.
C. Access the data through a NAT gateway.
D. Access the data through a VPC endpoint for Amazon S3.

Look at the answer choices, and rule them out based on the keywords that were previously highlighted.

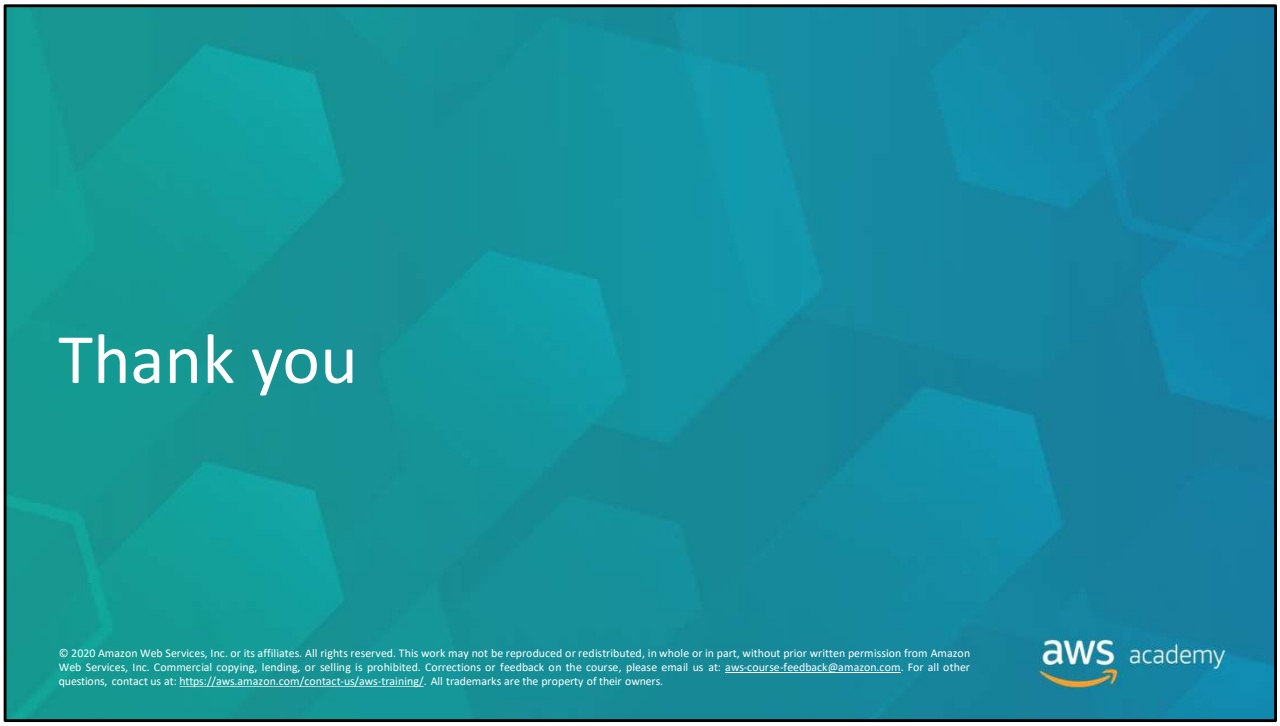
***The correct answer is D**: "Access the data through a VPC endpoint for Amazon S3."* Choice A ("Access the data through an internet gateway") can be eliminated because making the data stored in Amazon S3 public would be a security risk. Choice B ("Access the data through a VPN connection") can also be eliminated because you cannot connect to Amazon S3 by VPN. While choice C ("Access the data through a NAT gateway") is not wrong, you can pick only one correct answer. Choice D is more appropriate because there is no additional cost or limit to performance.

Additional resources

- AWS re:Invent 2018 video: AWS VPN Solutions
- AWS Knowledge Center video: How do I create a VPN with Amazon VPC?
- How do I configure a VPN over AWS Direct Connect?
- AWS re:Invent 2019 video: From one to many: Evolving Amazon VPC design
- Building a Scalable and Secure Multi-VPC AWS Network Infrastructure whitepaper
- AWS Knowledge Center video: What is AWS Peering?
- AWS re:Invent 2019 video: AWS Transit Gateway reference architectures for many VPCs
- AWS Knowledge Center video: What is an Interface VPC Endpoint and How Can I Create Interface Endpoints for my VPC?

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- AWS re:Invent 2018 video: AWS VPN Solutions
- AWS Knowledge Center video: How do I create a VPN with Amazon VPC?
- How do I configure a VPN over AWS Direct Connect?
- AWS re:Invent 2019 video: From one to many: Evolving Amazon VPC design
- Building a Scalable and Secure Multi-VPC AWS Network Infrastructure whitepaper
- AWS Knowledge Center video: What is AWS Peering?
- AWS re:Invent 2019 video: AWS Transit Gateway reference architectures for many VPCs
- AWS Knowledge Center video: What is an Interface VPC Endpoint and How Can I



Thank you for completing this module.