Amazon Virtual Private Cloud

VPC Peering

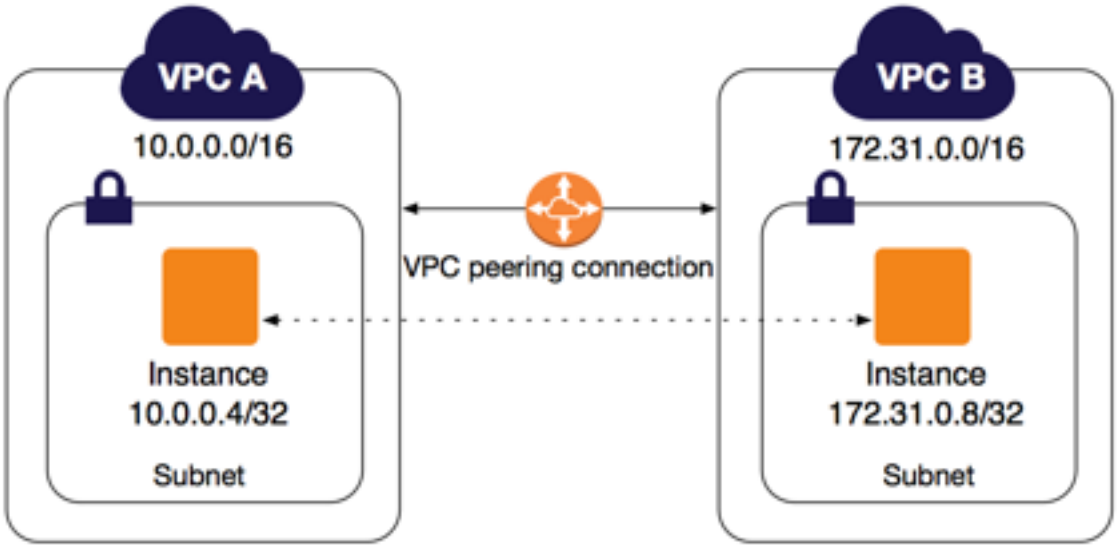
Amazon Virtual Private Cloud: VPC Peering

Table of Contents

What is VPC Peering?

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined.

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. The VPCs can be in different regions (also known as an inter-region VPC peering connection).



AWS uses the existing infrastructure of a VPC to create a VPC peering connection; it is neither a gateway nor a VPN connection, and does not rely on a separate piece of physical hardware. There is no single point of failure for communication or a bandwidth bottleneck.

A VPC peering connection helps you to facilitate the transfer of data. For example, if you have more than one AWS account, you can peer the VPCs across those accounts to create a file sharing network. You can also use a VPC peering connection to allow other VPCs to access resources you have in one of your VPCs.

You can establish peering relationships between VPCs across different AWS Regions (also called Inter-Region VPC Peering). This allows VPC resources including EC2 instances, Amazon RDS databases and Lambda functions that run in different AWS Regions to communicate with each other using private IP addresses, without requiring gateways, VPN connections, or separate network appliances. The traffic remains in the private IP space. All inter-region traffic is encrypted with no single point of failure, or bandwidth bottleneck. Traffic always stays on the global AWS backbone, and never traverses the public internet, which reduces threats, such as common exploits, and DDoS attacks. Inter-Region VPC Peering provides a simple and cost-effective way to share resources between regions or replicate data for geographic redundancy.

For more information, see the following:

- VPC Peering Basics (p. 2)

VPC Peering Basics

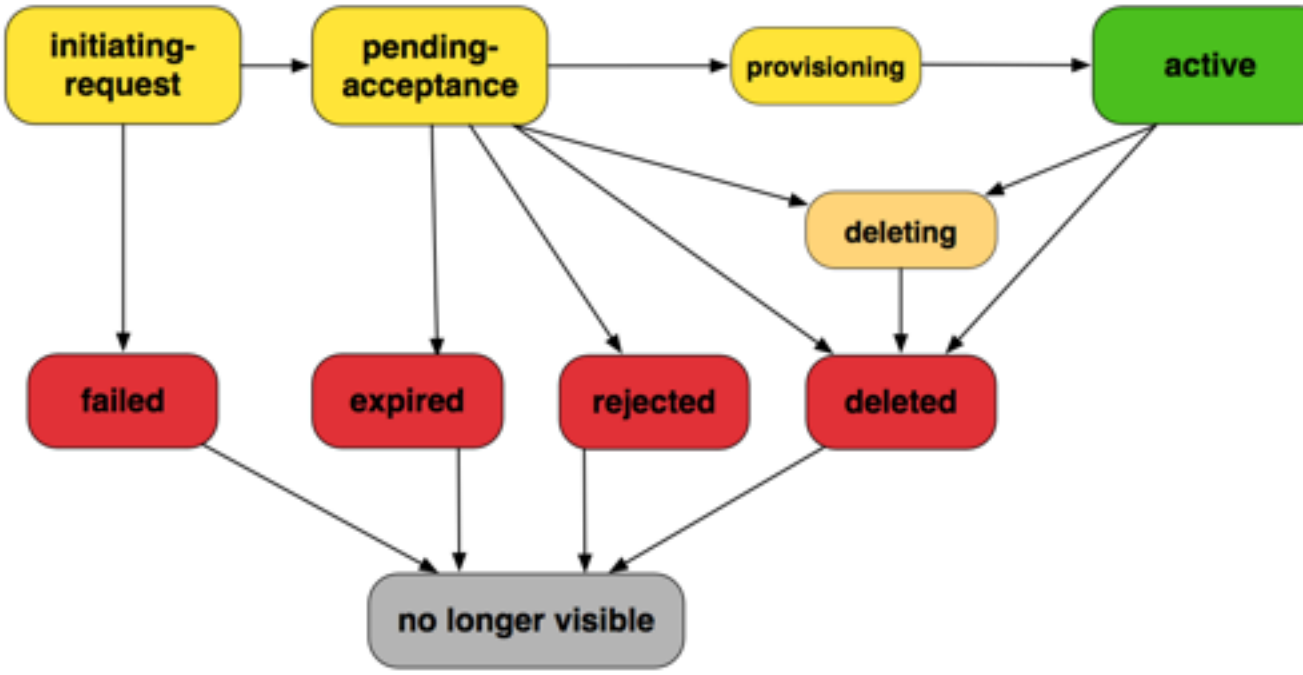
To establish a VPC peering connection, you do the following:

1. The owner of the *requester VPC* sends a request to the owner of the *accepter VPC* to create the VPC peering connection. The accepter VPC can be owned by you, or another AWS account, and cannot have a CIDR block that overlaps with the requester VPC's CIDR block.
2. The owner of the accepter VPC accepts the VPC peering connection request to activate the VPC peering connection.
3. To enable the flow of traffic between the VPCs using private IP addresses, the owner of each VPC in the VPC peering connection must manually add a route to one or more of their VPC route tables that points to the IP address range of the other VPC (the peer VPC).
4. If required, update the security group rules that are associated with your instance to ensure that traffic to and from the peer VPC is not restricted. If both VPCs are in the same region, you can reference a security group from the peer VPC as a source or destination for ingress or egress rules in your security group rules.
5. By default, if instances on either side of a VPC peering connection address each other using a public DNS hostname, the hostname resolves to the instance's public IP address. To change this behavior, enable DNS hostname resolution for your VPC connection. After enabling DNS hostname resolution, if instances on either side of the VPC peering connection address each other using a public DNS hostname, the hostname resolves to the private IP address of the instance.

For more information, see Working with VPC Peering Connections (p. 6).

VPC Peering Connection Lifecycle

A VPC peering connection goes through various stages starting from when the request is initiated. At each stage, there may be actions that you can take, and at the end of its lifecycle, the VPC peering connection remains visible in the Amazon VPC console and API or command line output for a period of time.



- **Initiating-request**: A request for a VPC peering connection has been initiated. At this stage, the peering connection may fail or may go to `pending-acceptance`.
- **Failed**: The request for the VPC peering connection has failed. During this state, it cannot be accepted, rejected, or deleted. The failed VPC peering connection remains visible to the requester for 2 hours.
- **Pending-acceptance**: The VPC peering connection request is awaiting acceptance from the owner of the accepter VPC. During this state, the owner of the requester VPC can delete the request, and the owner of the accepter VPC can accept or reject the request. If no action is taken on the request, it expires after 7 days.
- **Expired**: The VPC peering connection request has expired, and no action can be taken on it by either VPC owner. The expired VPC peering connection remains visible to both VPC owners for 2 days.
- **Rejected**: The owner of the accepter VPC has rejected a `pending-acceptance` VPC peering connection request. During this state, the request cannot be accepted. The rejected VPC peering connection remains visible to the owner of the requester VPC for 2 days, and visible to the owner of the accepter VPC for 2 hours. If the request was created within the same AWS account, the rejected request remains visible for 2 hours.
- **Provisioning**: The VPC peering connection request has been accepted, and will soon be in the `active` state.
- **Active**: The VPC peering connection is active, and traffic can flow between the VPCs (provided that your security groups and route tables allow the flow of traffic). During this state, either of the VPC owners can delete the VPC peering connection, but cannot reject it.

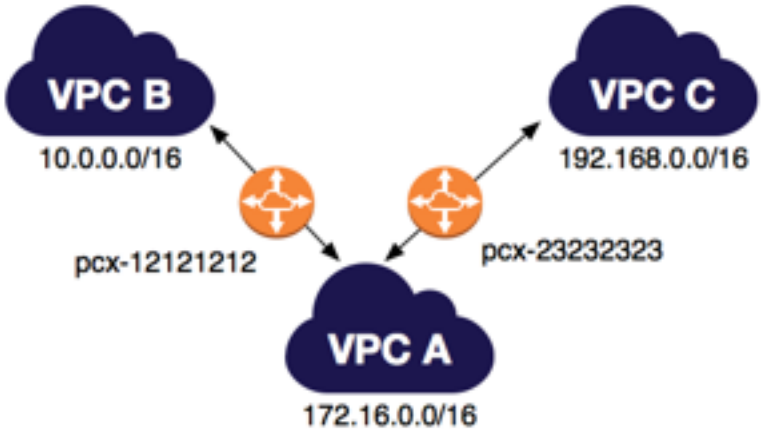
  Note
  If an event in a region in which a VPC resides prevents the flow of traffic, the status of the VPC peering connection remains `Active`.
- **Deleting**: Applies to an inter-region VPC peering connection that is in the process of being deleted. The owner of either VPC has submitted a request to delete an `active` VPC peering connection, or the owner of the requester VPC has submitted a request to delete a `pending-acceptance` VPC peering connection request.
- **Deleted**: An `active` VPC peering connection has been deleted by either of the VPC owners, or a `pending-acceptance` VPC peering connection request has been deleted by the owner of the

requester VPC. During this state, the VPC peering connection cannot be accepted or rejected. The VPC peering connection remains visible to the party that deleted it for 2 hours, and visible to the other party for 2 days. If the VPC peering connection was created within the same AWS account, the deleted request remains visible for 2 hours.

Multiple VPC Peering Connections

A VPC peering connection is a one to one relationship between two VPCs. You can create multiple VPC peering connections for each VPC that you own, but transitive peering relationships are not supported. You do not have any peering relationship with VPCs that your VPC is not directly peered with.

The following diagram is an example of one VPC peered to two different VPCs. There are two VPC peering connections: VPC A is peered with both VPC B and VPC C. VPC B and VPC C are not peered, and you cannot use VPC A as a transit point for peering between VPC B and VPC C. If you want to enable routing of traffic between VPC B and VPC C, you must create a unique VPC peering connection between them.



Pricing for a VPC Peering Connection

If the VPCs in the VPC peering connection are within the same region, the charges for transferring data within the VPC peering connection are the same as the charges for transferring data across Availability Zones. If the VPCs are in different regions, inter-region data transfer costs apply.

For more information, see Amazon EC2 Pricing.

VPC Peering Limitations

To create a VPC peering connection with another VPC, be aware of the following limitations and rules:

- You cannot create a VPC peering connection between VPCs that have matching or overlapping IPv4 or IPv6 CIDR blocks. Amazon always assigns your VPC a unique IPv6 CIDR block. If your IPv6 CIDR blocks are unique but your IPv4 blocks are not, you cannot create the peering connection.
- You have a quota on the number of active and pending VPC peering connections that you can have per VPC. For more information, see Amazon VPC Quotas in the *Amazon VPC User Guide*
- VPC peering does not support transitive peering relationships. In a VPC peering connection, your VPC does not have access to any other VPCs with which the peer VPC may be peered. This includes VPC peering connections that are established entirely within your own AWS account. For more information

about unsupported peering relationships, see Unsupported VPC Peering Configurations (p. 58). For examples of supported peering relationships, see VPC Peering Scenarios (p. 17).

- You cannot have more than one VPC peering connection between the same two VPCs at the same time.
- Unicast reverse path forwarding in VPC peering connections is not supported. For more information, see Routing for Response Traffic (p. 44).
- You can enable resources on either side of a VPC peering connection to communicate with each other over IPv6; however, IPv6 communication is not automatic. You must associate an IPv6 CIDR block with each VPC, enable the instances in the VPCs for IPv6 communication, and add routes to your route tables that route IPv6 traffic intended for the peer VPC to the VPC peering connection. For more information, see Your VPC and Subnets in the *Amazon VPC User Guide*.
- Any tags that you create for your VPC peering connection are only applied in the account or region in which you create them.
- If the IPv4 CIDR block of a VPC in a VPC peering connection falls outside of the private IPv4 address ranges specified by RFC 1918, private DNS hostnames for that VPC cannot be resolved to private IP addresses. To resolve private DNS hostnames to private IP addresses, you can enable DNS resolution support for the VPC peering connection. For more information, see Enabling DNS Resolution Support for a VPC Peering Connection (p. 15).
- You cannot connect to or query the Amazon DNS server in a peer VPC.

An inter-region VPC peering connection has additional limitations:

- You cannot create a security group rule that references a peer VPC security group.
- You cannot enable support for an EC2-Classic instance that's linked to a VPC via ClassicLink to communicate with the peer VPC.
- The Maximum Transmission Unit (MTU) across the VPC peering connection is 1500 bytes (jumbo frames are not supported).
- You must enable DNS resolution support for the VPC peering connection to resolve private DNS hostnames of the peered VPC to private IP addresses, even if the IPv4 CIDR for the VPC falls into the private IPv4 address ranges specified by RFC 1918.
- Inter-region peering in China is only allowed between the China (Beijing) Region, operated by SINNET and the China (Ningxia) Region, operated by NWCD.

Working with VPC Peering Connections

You can use the Amazon VPC console to create and work with VPC peering connections.

Tasks

Creating and Accepting a VPC Peering Connection

To create a VPC peering connection, first create a request to peer with another VPC. You can request a VPC peering connection with another VPC in your account, or with a VPC in a different AWS account. For an inter-region VPC peering connection where the VPCs are in different regions, the request must be made from the region of the requester VPC.

To activate the request, the owner of the accepter VPC must accept the request. For an inter-region VPC peering connection, the request must be accepted in the region of the accepter VPC.

Before you begin, ensure that you are aware of the limitations and rules (p. 4) for a VPC peering connection.

Tasks

Creating a VPC Peering Connection with Another VPC in Your Account

To request a VPC peering connection with a VPC in your account, ensure that you have the IDs of the VPCs with which you are creating the VPC peering connection. You must both create and accept the VPC peering connection request yourself to activate it.

You can create a VPC peering connection with a VPC in the same region, or a different region.

> **Important**
> Ensure that your VPCs do not have overlapping IPv4 CIDR blocks. If they do, the status of the VPC peering connection immediately goes to `failed`. This limitation applies even if the VPCs have unique IPv6 CIDR blocks.

To create a VPC peering connection with a VPC in the same region

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the following information, and choose **Create Peering Connection** when you are done:
   - **Peering connection name tag**: You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify.
   - **VPC (Requester)**: Select the VPC in your account with which you want to create the VPC peering connection.
   - Under **Select another VPC to peer with**: Ensure **My account** is selected, and select another of your VPCs.
4. In the confirmation dialog box, choose **OK**.
5. Select the VPC peering connection that you've created, and choose **Actions**, **Accept Request**.
6. In the confirmation dialog, choose **Yes, Accept**. A second confirmation dialog displays; choose **Modify my route tables now** to go directly to the route tables page, or choose **Close** to do this later.

To create a VPC peering connection with a VPC in a different region

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the following information, and choose **Create Peering Connection** when you are done:
   - **Peering connection name tag**: You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify.
   - **VPC (Requester)**: Select the requester VPC in your account with which to request the VPC peering connection.
   - **Account**: Ensure **My account** is selected.
   - **Region**: Choose **Another region**, select the region in which the accepter VPC resides.
   - **VPC (Accepter)**: Enter the ID of the accepter VPC.
4. In the confirmation dialog box, choose **OK**.
5. In the region selector, select the region of the accepter VPC.
6. In the navigation pane, choose **Peering Connections**. Select the VPC peering connection that you've created, and choose **Actions**, **Accept Request**.
7. In the confirmation dialog, choose **Yes, Accept**. A second confirmation dialog displays; choose **Modify my route tables now** to go directly to the route tables page, or choose **Close** to do this later.

Now that your VPC peering connection is active, you must add an entry to your VPC route tables to enable traffic to be directed between the peered VPCs. For more information, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

Creating a VPC Peering Connection with a VPC in Another AWS Account

You can request a VPC peering connection with a VPC that's in another AWS account. Before you begin, ensure that you have the AWS account number and VPC ID of the VPC to peer with. After you've created the request, the owner of the accepter VPC must accept the VPC peering connection to activate it.

You can create a VPC peering connection with a VPC in the same region, or a different region.

Important
If the VPCs have overlapping IPv4 CIDR blocks, or if the account ID and VPC ID are incorrect or do not correspond with each other, the status of the VPC peering connection immediately goes to `failed`.

To request a VPC peering connection with a VPC in another account in the same region

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the information as follows, and choose **Create Peering Connection** when you are done:
   - **Peering connection name tag**: You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify. This tag is only visible to you; the owner of the peer VPC can create their own tags for the VPC peering connection.
   - **VPC (Requester)**: Select the VPC in your account with which to create the VPC peering connection.
   - **Account**: Choose **Another account**.
   - **Account ID**: Enter the AWS account ID of the owner of the accepter VPC.
   - **VPC (Accepter)**: Enter the ID of the VPC with which to create the VPC peering connection.
4. In the confirmation dialog box, choose **OK**.

To request a VPC peering connection with a VPC in another account in a different region

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the information as follows, and choose **Create Peering Connection** when you are done:
   - **Peering connection name tag**: You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify. This tag is only visible to you; the owner of the peer VPC can create their own tags for the VPC peering connection.
   - **VPC (Requester)**: Select the VPC in your account with which to create the VPC peering connection.
   - **Account**: Choose **Another account**.
   - **Account ID**: Enter the AWS account ID of the owner of the accepter VPC.
   - **Region**: Choose **Another region**, select the region in which the accepter VPC resides.
   - **VPC (Accepter)**: Enter the ID of the VPC with which to create the VPC peering connection.
4. In the confirmation dialog box, choose **OK**.

The VPC peering connection that you've created is not active. To activate it, the owner of the accepter VPC must accept the VPC peering connection request. To enable traffic to be directed to the peer VPC, update your VPC route table. For more information, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

To create a VPC peering connection using the command line or an API

- create-vpc-peering-connection (AWS CLI)
- New-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- CreateVpcPeeringConnection (Amazon EC2 Query API)

Accepting a VPC Peering Connection

A VPC peering connection that's in the `pending-acceptance` state must be accepted by the owner of the accepter VPC to be activated. You cannot accept a VPC peering connection request that you've sent

to another AWS account. If you are creating a VPC peering connection in the same AWS account, you must both create and accept the request yourself.

If the VPCs are in different regions, the request must be accepted in the region of the accepter VPC.

> **Important**
> Do not accept VPC peering connections from unknown AWS accounts. A malicious user may have sent you a VPC peering connection request to gain unauthorized network access to your VPC. This is known as peer phishing. You can safely reject unwanted VPC peering connection requests without any risk of the requester gaining access to any information about your AWS account or your VPC. For more information, see Rejecting a VPC Peering Connection (p. 10). You can also ignore the request and let it expire; by default, requests expire after 7 days.

To accept a VPC peering connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. Use the region selector to choose the region of the accepter VPC.
3. In the navigation pane, choose **Peering Connections**.
4. Select the pending VPC peering connection (the status is `pending-acceptance`), and choose **Actions**, **Accept Request**.

   > **Note**
   > If you cannot see the pending VPC peering connection, check the region. An inter-region peering request must be accepted in the region of the accepter VPC.
5. In the confirmation dialog box, choose **Yes, Accept**. A second confirmation dialog displays; choose **Modify my route tables now** to go directly to the route tables page, or choose **Close** to do this later.

Now that your VPC peering connection is active, you must add an entry to your VPC route table to enable traffic to be directed to the peer VPC. For more information, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

To accept a VPC peering connection using the command line or an API

- accept-vpc-peering-connection (AWS CLI)
- Approve-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- AcceptVpcPeeringConnection (Amazon EC2 Query API)

Viewing Your VPC Peering Connections

You can view all of your VPC peering connections in the Amazon VPC console. By default, the console displays all VPC peering connections in different states, including those that may have been recently deleted or rejected. For more information about the lifecycle of a VPC peering connection, see VPC Peering Connection Lifecycle (p. 2).

To view your VPC peering connections

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**.
3. All of your VPC peering connections are listed. Use the filter search bar to narrow your results.

To describe a VPC peering connection using the command line or an API

- describe-vpc-peering-connections (AWS CLI)
- Get-EC2VpcPeeringConnections (AWS Tools for Windows PowerShell)

- DescribeVpcPeeringConnections (Amazon EC2 Query API)

Rejecting a VPC Peering Connection

You can reject any VPC peering connection request that you've received that's in the `pending-acceptance` state. You should only accept VPC peering connections from AWS accounts that you know and trust; you can reject any unwanted requests.

To reject a VPC peering connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**.
3. Select the VPC peering connection, and choose **Actions**, **Reject Request**.
4. In the confirmation dialog box, choose **Yes, Reject**.

To reject a VPC peering connection using the command line or an API

- reject-vpc-peering-connection (AWS CLI)
- Deny-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- RejectVpcPeeringConnection (Amazon EC2 Query API)

Updating Your Route Tables for a VPC Peering Connection

To send private IPv4 traffic from your instance to an instance in a peer VPC, you must add a route to the route table that's associated with your subnet in which your instance resides. The route points to the CIDR block (or portion of the CIDR block) of the peer VPC in the VPC peering connection, and specifies the VPC peering connection as the target.

Similarly, if the VPCs in the VPC peering connection have associated IPv6 CIDR blocks, you can add a route to your route table to enable communication with the peer VPC over IPv6.

If a subnet is not explicitly associated with a route table, it uses the main route table by default. For more information, see Route Tables in the *Amazon VPC User Guide*.

You have a quota on the number of entries you can add per route table. If the number of VPC peering connections in your VPC exceeds the route table entry quota for a single route table, consider using multiple subnets that are each associated with a custom route table.

For more information about supported route table configurations for VPC peering connections, see VPC Peering Configurations (p. 19).

You can add a route for a VPC peering connection that's in the `pending-acceptance` state. However, the route will have a state of `blackhole` and have no effect until the VPC peering connection is in the `active` state.

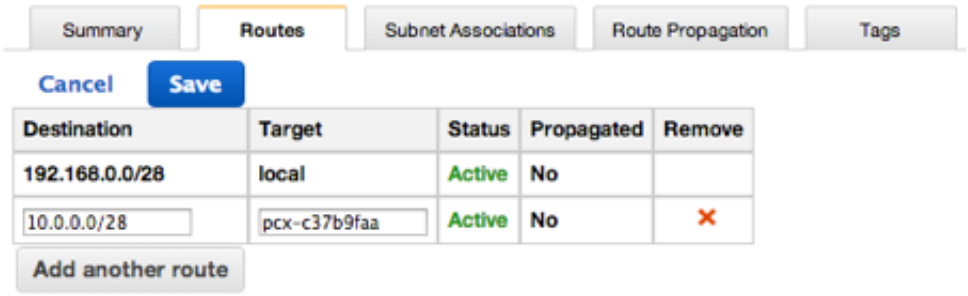
Warning
If you have a VPC peered with multiple VPCs that have overlapping or matching IPv4 CIDR blocks, ensure that your route tables are configured to avoid sending response traffic from your VPC to the incorrect VPC. AWS currently does not support unicast reverse path forwarding in

VPC peering connections that checks the source IP of packets and routes reply packets back to the source. For more information, see Routing for Response Traffic (p. 44).

To add an IPv4 route for a VPC peering connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Route Tables**.
3. Select the route table that's associated with the subnet in which your instance resides.

   Note
   If you do not have a route table associated with that subnet, select the main route table for the VPC, as the subnet then uses this route table by default.
4. Choose **Routes**, **Edit**, **Add Route**.
5. For **Destination**, enter the IPv4 address range to which the network traffic in the VPC peering connection must be directed. You can specify the entire IPv4 CIDR block of the peer VPC, a specific range, or an individual IPv4 address, such as the IP address of the instance with which to communicate. For example, if the CIDR block of the peer VPC is `10.0.0.0/16`, you can specify a portion `10.0.0.0/28`, or a specific IP address `10.0.0.7/32`.
6. Select the VPC peering connection from **Target**, and then choose **Save**.



The owner of the peer VPC must also complete these steps to add a route to direct traffic back to your VPC through the VPC peering connection.

If both VPCs in the VPC peering connection are in the same region, have IPv6 CIDR blocks, and the resources in the VPC are enabled to use IPv6, you can also add a route for IPv6 communication.

To add an IPv6 route for a VPC peering connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Route Tables** and select the route table that's associated with your subnet.
3. On the **Routes** tab, choose **Edit**, **Add another route**.
4. For **Destination**, enter the IPv6 address range for the peer VPC. You can specify the entire IPv6 CIDR block of the peer VPC, a specific range, or an individual IPv6 address. For example, if the CIDR block of the peer VPC is `2001:db8:1234:1a00::/56`, you can specify a portion `2001:db8:1234:1a00::/64`, or a specific IP address `2001:db8:1234:1a00::123/128`.
5. Select the VPC peering connection from **Target** and choose **Save**.

For more information, see Route Tables in the *Amazon VPC User Guide*.

To add or replace a route using the command line or an API

- create-route (AWS CLI)
- New-EC2Route (AWS Tools for Windows PowerShell)

- CreateRoute (Amazon EC2 Query API)
- replace-route (AWS CLI)
- Set-EC2Route (AWS Tools for Windows PowerShell)
- ReplaceRoute (Amazon EC2 Query API)

Updating Your Security Groups to Reference Peer VPC Groups

You can update the inbound or outbound rules for your VPC security groups to reference security groups in the peered VPC. Doing so allows traffic to flow to and from instances that are associated with the referenced security group in the peered VPC.

Requirements

- The peer VPC can be a VPC in your account, or a VPC in another AWS account. To reference a security group in another AWS account, include the account number in **Source** or **Destination** field; for example, `123456789012/sg-1a2b3c4d`.
- You cannot reference the security group of a peer VPC that's in a different region. Instead, use the CIDR block of the peer VPC.
- To reference a security group in a peer VPC, the VPC peering connection must be in the `active` state.

To update your security group rules using the console

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Security Groups**.
3. Select the security group, and choose **Inbound Rules** to modify the inbound rules or **Outbound Rules** to modify the outbound rules.
4. Choose **Edit**, **Add another rule**.
5. Specify the type, protocol, and port range as required. For **Source** (or **Destination** for an outbound rule), type the ID of the security group in the peer VPC if it is in the same region or the CIDR block of the peer VPC if it is in a different region.

   Note
   Security groups in a peer VPC are not automatically displayed.
6. Choose **Save**.

To update inbound rules using the command line

- authorize-security-group-ingress (AWS CLI)
- Grant-EC2SecurityGroupIngress (AWS Tools for Windows PowerShell)
- Revoke-EC2SecurityGroupIngress (AWS Tools for Windows PowerShell)
- revoke-security-group-ingress (AWS CLI)

To update outbound rules using the command line

- authorize-security-group-egress (AWS CLI)
- Grant-EC2SecurityGroupEgress (AWS Tools for Windows PowerShell)
- Revoke-EC2SecurityGroupEgress (AWS Tools for Windows PowerShell)
- revoke-security-group-egress (AWS CLI)

For example, to update your security group `sg-aaaa1111` to allow inbound access over HTTP from `sg-bbbb2222` that's in a peer VPC, you can use the following AWS CLI command:

```
aws ec2 authorize-security-group-ingress --group-id sg-aaaa1111 --protocol tcp --port 80 --source-group sg-bbbb2222
```

After you've updated the security group rules, use the describe-security-groups command to view the referenced security group in your security group rules.

Identifying Your Referenced Security Groups

To determine if your security group is being referenced in the rules of a security group in a peer VPC, use one of the following commands for one or more security groups in your account.

- describe-security-group-references (AWS CLI)
- Get-EC2SecurityGroupReference (AWS Tools for Windows PowerShell)
- DescribeSecurityGroupReferences (Amazon EC2 Query API)

In the following example, the response indicates that security group `sg-bbbb2222` is being referenced by a security group in VPC `vpc-aaaaaaaa`:

```
aws ec2 describe-security-group-references --group-id sg-bbbb2222
```

```
{
  "SecurityGroupsReferenceSet": [
    {
      "ReferencingVpcId": "vpc-aaaaaaaa",
      "GroupId": "sg-bbbb2222",
      "VpcPeeringConnectionId": "pcx-b04deed9"
    }
  ]
}
```

If the VPC peering connection is deleted, or if the owner of the peer VPC deletes the referenced security group, the security group rule becomes stale.

Working with Stale Security Group Rules

A stale security group rule is a rule that references a security group in a peer VPC where the VPC peering connection has been deleted or the security group in the peer VPC has been deleted. When a security group rule becomes stale, it's not automatically removed from your security group—you must manually remove it. If a security group rule was stale because the VPC peering connection was deleted and you then create a new VPC peering connection with the same VPCs, it will no longer be marked as stale.

You can view and delete the stale security group rules for a VPC using the Amazon VPC console.

To view and delete stale security group rules

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Security Groups**.
3. Choose **View your stale rules** in the notification icon on the right (this icon only displays if you have stale security group rules).
4. To delete a stale rule, choose **Edit**, and then delete the rule. Choose **Save Rules**. You can check for stale rules in another VPC by entering the VPC ID in the **VPC** field.
5. When you are done, choose **Close**.

To describe your stale security group rules using the command line or an API

- describe-stale-security-groups (AWS CLI)
- Get-EC2StaleSecurityGroup (AWS Tools for Windows PowerShell)
- DescribeStaleSecurityGroups (Amazon EC2 Query API)

In the following example, VPC A (`vpc-aaaaaaaa`) and VPC B were peered, and the VPC peering connection was deleted. Your security group `sg-aaaa1111` in VPC A references `sg-bbbb2222` in VPC B. When you run the `describe-stale-security-groups` command for your VPC, the response indicates that security group `sg-aaaa1111` has a stale SSH rule that references `sg-bbbb2222`.

```
aws ec2 describe-stale-security-groups --vpc-id vpc-aaaaaaaa
```

```
{
    "StaleSecurityGroupSet": [
        {
            "VpcId": "vpc-aaaaaaaa",
            "StaleIpPermissionsEgress": [],
            "GroupName": "Access1",
            "StaleIpPermissions": [
                {
                    "ToPort": 22,
                    "FromPort": 22,
                    "UserIdGroupPairs": [
                        {
                            "VpcId": "vpc-bbbbbbbb",
                            "PeeringStatus": "deleted",
                            "UserId": "123456789101",
                            "GroupName": "Prod1",
                            "VpcPeeringConnectionId": "pcx-b04deed9",
                            "GroupId": "sg-bbbb2222"
                        }
                    ],
                    "IpProtocol": "tcp"
                }
            ],
            "GroupId": "sg-aaaa1111",
            "Description": "Reference remote SG"
        }
    ]
}
```

After you've identified the stale security group rules, you can delete them using the revoke-security-group-ingress or revoke-security-group-egress commands.

Modifying VPC Peering Connection Options

You can modify a VPC peering connection to do the following:

- Enable one or more EC2-Classic instances that are linked to your VPC via ClassicLink to communicate with instances in the peer VPC, or to enable instances in your VPC to communicate with linked EC2-Classic instances in the peer VPC. For more information, see Configurations with ClassicLink (p. 51). You cannot enable EC2-Classic instances to communicate with instances in a peer VPC over IPv6.
- Enable a VPC to resolve public IPv4 DNS hostnames to private IPv4 addresses when queried from instances in the peer VPC. For more information, see Enabling DNS Resolution Support for a VPC Peering Connection (p. 15).

Enabling DNS Resolution Support for a VPC Peering Connection

To enable a VPC to resolve public IPv4 DNS hostnames to private IPv4 addresses when queried from instances in the peer VPC, you must modify the peering connection.

Both VPCs must be enabled for DNS hostnames and DNS resolution.

To enable DNS resolution support for the peering connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**.
3. Select the VPC peering connection, and choose **Actions**, **Edit DNS Settings**.
4. To ensure that queries from the peer VPC resolve to private IP addresses in your local VPC, choose the option to enable DNS resolution for queries from the peer VPC. This option is **Requester DNS resolution** or **Accepter DNS resolution**, depending on whether the VPC is the requester or accepter VPC.
5. If the peer VPC is in the same AWS account, you can enable DNS resolution for both VPCs in the peering connection.
6. Choose **Save**.
7. If the peer VPC is in a different AWS account or a different region, the owner of the peer VPC must sign into the VPC console, perform steps 2 through 4, and choose **Save**.

To enable DNS resolution using the command line or an API

- modify-vpc-peering-connection-options (AWS CLI)
- Edit-EC2VpcPeeringConnectionOption (AWS Tools for Windows PowerShell)
- ModifyVpcPeeringConnectionOptions (Amazon EC2 Query API)

You must modify the requester VPC peering options if you are the requester of the VPC peering connection, and you must modify the accepter VPC peering options if you are the accepter of the VPC peering connection. You can use the describe-vpc-peering-connections or Get-EC2VpcPeeringConnections commands to verify which VPC is the accepter and the requester for a VPC peering connection. For inter-region peering connections, you must use the region for the requester VPC to modify the requester VPC peering options and the region for the accepter VPC to modify the accepter VPC peering options.

In this example, you are the requester of the VPC peering connection, therefore modify the peering connection options using the AWS CLI as follows:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-aaaabbbb --
requester-peering-connection-options AllowDnsResolutionFromRemoteVpc=true
```

Deleting a VPC Peering Connection

Either owner of a VPC in a peering connection can delete the VPC peering connection at any time. You can also delete a VPC peering connection that you've requested that is still in the `pending-acceptance` state.

Deleting a VPC in the Amazon VPC console that's part of an active VPC peering connection also deletes the VPC peering connection. If you have requested a VPC peering connection with a VPC in another

account, and you delete your VPC before the other party has accepted the request, the VPC peering connection is also deleted. You cannot delete a VPC for which you have a `pending-acceptance` request from a VPC in another account. You must first reject the VPC peering connection request.

When you delete a peering connection, the status is set to `Deleted`. When the connection is in this state, you cannot accept, reject, or edit the DNS settings.

To delete a VPC peering connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**.
3. Select the VPC peering connection, and choose **Actions**, **Delete VPC Peering Connection**.
4. In the confirmation dialog box, choose **Yes, Delete**.

To delete a VPC peering connection using the command line or an API

- delete-vpc-peering-connection (AWS CLI)
- Remove-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- DeleteVpcPeeringConnection (Amazon EC2 Query API)

VPC Peering Scenarios

There are a number of reasons you might need to set up a VPC peering connection between your VPCs, or between a VPC that you own and a VPC in a different AWS account. The following scenarios can help you determine which configuration is best suited to your networking requirements.

Scenarios

Peering Two or More VPCs to Provide Full Access to Resources

In this scenario, you have two or more VPCs that you want to peer to enable full sharing of resources between all VPCs. The following are some examples:

- Your company has a VPC for the finance department, and another VPC for the accounting department. The finance department requires access to all resources that are in the accounting department, and the accounting department requires access to all resources in the finance department.
- Your company has multiple IT departments, each with their own VPC. Some VPCs are located within the same AWS account, and others in a different AWS account. You want to peer together all VPCs to enable the IT departments to have full access to each others' resources.

For more information about how to set up the VPC peering connection configuration and route tables for this scenario, see the following documentation:

- Two VPCs Peered Together (p. 19)
- Three VPCs Peered Together (p. 23)
- Multiple VPCs Peered Together (p. 29)

For more information about creating and working with VPC peering connections in the Amazon VPC console, see Working with VPC Peering Connections (p. 6).

Peering to One VPC to Access Centralized Resources

In this scenario, you have a central VPC that contains resources that you want to share with other VPCs. Your central VPC may require full or partial access to the peer VPCs, and similarly, the peer VPCs may require full or partial access to the central VPC. The following are some examples:

- Your company's IT department has a VPC for file sharing. You want to peer other VPCs to that central VPC, however, you do not want the other VPCs to send traffic to each other.

- Your company has a VPC that you want to share with your customers. Each customer can create a VPC peering connection with your VPC, however, your customers cannot route traffic to other VPCs that are peered to yours, nor are they aware of the other customers' routes.
- You have a central VPC that is used for Active Directory services. Specific instances in peer VPCs send requests to the Active Directory servers and require full access to the central VPC. The central VPC does not require full access to the peer VPCs; it only needs to route response traffic to the specific instances.

For more information about how to set up the VPC peering connection configuration and route tables for this scenario, see the following topics:

- One VPC Peered with Two VPCs (p. 21)
- One VPC Peered with Multiple VPCs (p. 25)
- Two VPCs Peered to Two Subnets in One VPC (p. 38)
- One VPC Peered to Specific Subnets in Two VPCs (p. 42)
- Instances in One VPC Peered to Instances in Two VPCs (p. 46)
- One VPC Peered with Two VPCs Using Longest Prefix Match (p. 48)

For more information about creating and working with VPC peering connections in the Amazon VPC console, see Working with VPC Peering Connections (p. 6).

Peering with ClassicLink

You can modify a VPC peering connection to enable one or more EC2-Classic instances that are linked to your VPC via ClassicLink to communicate with instances in the peer VPC. Similarly, you can modify a VPC peering connection to enable instances in your VPC to communicate with linked EC2-Classic instances in the peer VPC.

For more information about how to set up the VPC peering connection configuration and route tables for this scenario, see Configurations with ClassicLink (p. 51).

VPC Peering Configurations

The following documentation describes the supported VPC peering configurations. You might require a configuration that allows routing between the entire CIDR block of each VPC, or a configuration that limits routing to specific subnets or IP addresses.

Configurations

Configurations with Routes to an Entire CIDR Block

You can configure VPC peering connections so that your route tables have access to the entire CIDR block of the peer VPC. For more information about scenarios in which you might need a specific VPC peering connection configuration, see VPC Peering Scenarios (p. 17). For more information about creating and working with VPC peering connections in the Amazon VPC console, see Working with VPC Peering Connections (p. 6).

Configurations

Two VPCs Peered Together

You have a VPC peering connection (`pcx-11112222`) between VPC A and VPC B, which are in the same AWS account, and do not have overlapping CIDR blocks.

You may want to use this kind of configuration when you have two VPCs that require access to each others' resources. For example, you set up VPC A for your accounting records, and VPC B for your financial records, and now you want each VPC to be able to access each others' resources without restriction.

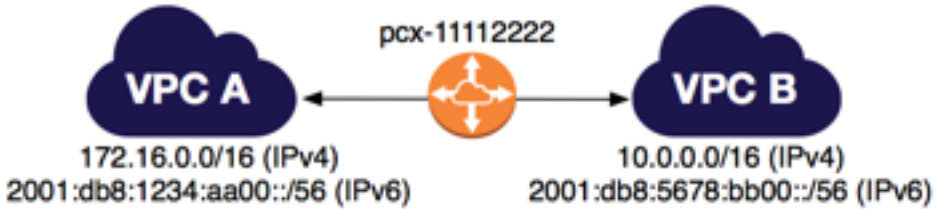
The route tables for each VPC point to the relevant VPC peering connection to access the entire CIDR block of the peer VPC.

Route table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-11112222
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-11112222

For more information about updating your route tables, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

Two VPCs Peered Together for IPv6

You have the same two VPCs in the VPC peering configuration as above. In this example, VPC A and VPC B both have associated IPv6 CIDR blocks.



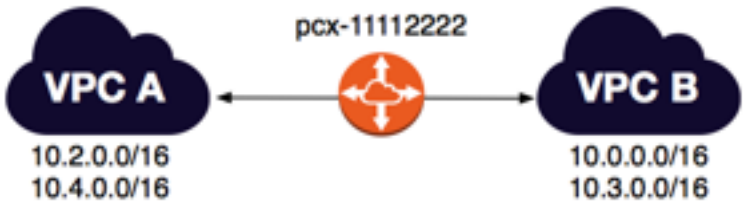
The route tables for each VPC point to the VPC peering connection to access the entire IPv6 CIDR block of the peer VPC.

Route table	Destination	Target
VPC A	172.16.0.0/16	Local
	2001:db8:1234:aa00::/56	Local
	10.0.0.0/16	pcx-11112222
	2001:db8:5678:bb00::/56	pcx-11112222
VPC B	10.0.0.0/16	Local
	2001:db8:5678:bb00::/56	Local
	172.16.0.0/16	pcx-11112222
	2001:db8:1234:aa00::/56	pcx-11112222

For more information about IPv6 in your VPC, see Your VPC and Subnets in the *Amazon VPC User Guide*.

Two VPCs with Multiple CIDRs Peered Together

You can add IPv4 CIDR blocks to your VPC. In this example, VPC A and VPC B have multiple IPv4 CIDR blocks.



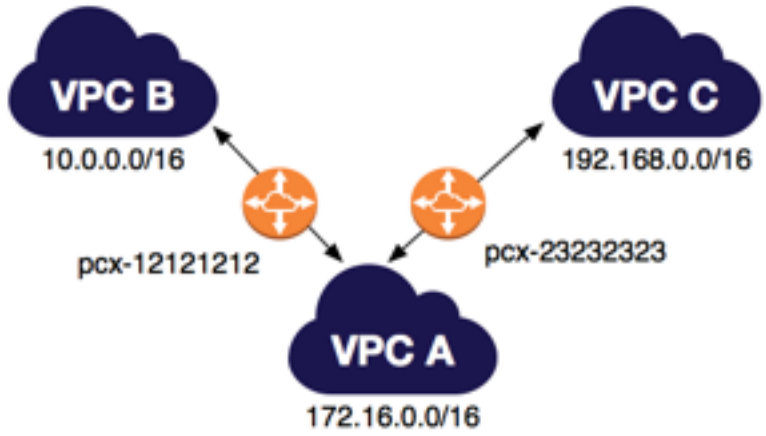
The route tables for each VPC point to the VPC peering connection to access all the IPv4 CIDR blocks of the peer VPC.

Route table	Destination	Target
VPC A	10.2.0.0/16	Local
	10.4.0.0/16	Local
	10.0.0.0/16	pcx-11112222
	10.3.0.0/16	pcx-11112222
VPC B	10.0.0.0/16	Local
	10.3.0.0/16	Local
	10.2.0.0/16	pcx-11112222
	10.4.0.0/16	pcx-11112222

For more information, see Adding IPv4 CIDR Blocks to a VPC in the *Amazon VPC User Guide*.

One VPC Peered with Two VPCs

You have a central VPC (VPC A), and you have a VPC peering connection between VPC A and VPC B (`pcx-12121212`), and between VPC A and VPC C (`pcx-23232323`). The VPCs are in the same AWS account, and do not have overlapping CIDR blocks.



You may want to use this 'flying V' configuration when you have resources on a central VPC, such as a repository of services, that other VPCs need to access. The other VPCs do not need access to each others' resources; they only need access to resources on the central VPC.

> **Note**
> VPC B and VPC C cannot send traffic directly to each other through VPC A. VPC peering does not support transitive peering relationships, nor edge to edge routing. You must create a VPC peering connection between VPC B and VPC C in order to route traffic directly between them. For more information, see Three VPCs Peered Together (p. 23). For more information about unsupported peering scenarios, see Unsupported VPC Peering Configurations (p. 58).

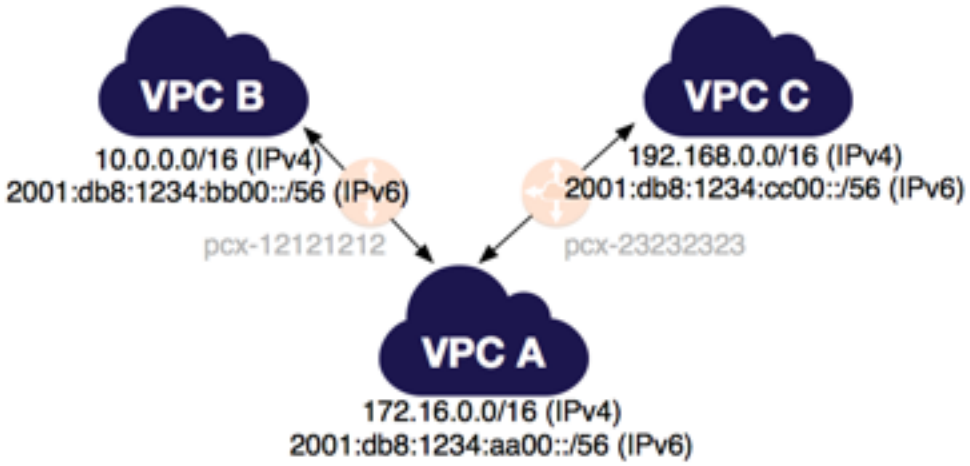
The route tables for each VPC point to the relevant VPC peering connection to access the entire CIDR block of the peer VPC.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-12121212
	192.168.0.0/16	pcx-23232323
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-12121212
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-23232323

For more information about updating your route tables, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

One VPC Peered with Two VPCs for IPv6

You have the same three VPCs in the VPC peering configuration as above. In this example, all three VPCs have associated IPv6 CIDR blocks.



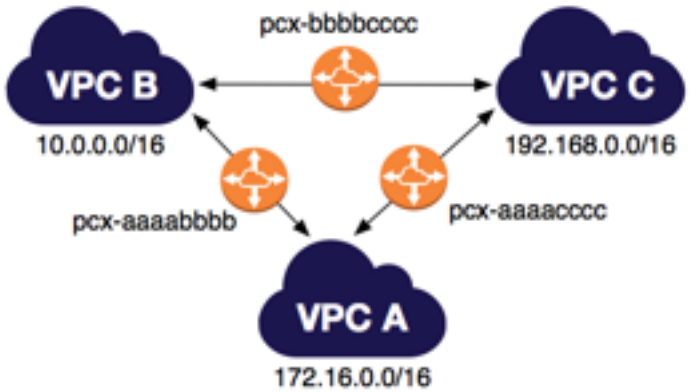
The route tables for each VPC point to the VPC peering connection to access the entire IPv6 CIDR block of the peer VPC.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	2001:db8:1234:aa00::/56	Local
	10.0.0.0/16	pcx-12121212
	192.168.0.0/16	pcx-23232323
	2001:db8:1234:bb00::/56	pcx-12121212
	2001:db8:1234:cc00::/56	pcx-23232323
VPC B	10.0.0.0/16	Local
	2001:db8:1234:bb00::/56	Local
	172.16.0.0/16	pcx-12121212
	2001:db8:1234:aa00::/56	pcx-12121212
VPC C	192.168.0.0/16	Local
	2001:db8:1234:cc00::/56	Local
	172.16.0.0/16	pcx-23232323
	2001:db8:1234:aa00::/56	pcx-23232323

Three VPCs Peered Together

You have peered three VPCs together in a full mesh configuration. The VPCs are in the same AWS account and do not have overlapping CIDR blocks:

- VPC A is peered to VPC B through VPC peering connection `pcx-aaaabbbb`
- VPC A is peered to VPC C through VPC peering connection `pcx-aaaacccc`
- VPC B is peered to VPC C through VPC peering connection `pcx-bbbbcccc`



You may want to use this full mesh configuration when you have separate VPCs that need to share resources with each other without restriction; for example, as a file sharing system.

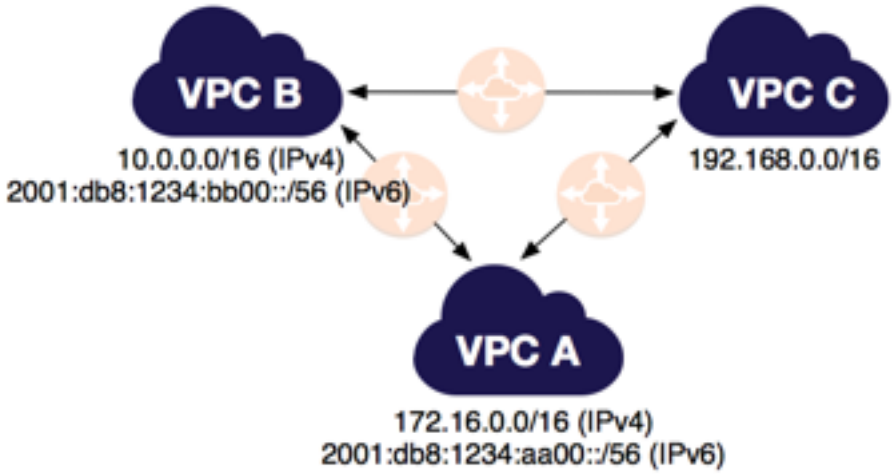
The route tables for each VPC point to the relevant VPC peering connection to access the entire CIDR block of the peer VPCs.

Route Tables	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaacccc
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-bbbbcccc
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc
	10.0.0.0/16	pcx-bbbbcccc

For more information about updating your route tables, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

Three VPCs Peered for IPv6

You have the same three VPCs in the VPC peering configuration as above. In this example, VPC A and VPC B both have associated IPv6 CIDR blocks. VPC C does not have an associated IPv6 CIDR block.



The route tables for VPC A and VPC B include routes that point to VPC peering connection `pcx-aaaabbbb` to access the entire IPv6 CIDR block of the peer VPC. VPC A and VPC B can communicate using IPv6 over the VPC peering connection. VPC C cannot communicate using IPv6 with either VPC A or VPC B.

Route Tables	Destination	Target
VPC A	172.16.0.0/16	Local
	2001:db8:1234:aa00::/56	Local
	10.0.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaacccc
	2001:db8:1234:bb00::/56	pcx-aaaabbbb
VPC B	10.0.0.0/16	Local
	2001:db8:1234:bb00::/56	Local
	172.16.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-bbbbcccc
	2001:db8:1234:aa00::/56	pcx-aaaabbbb
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc
	10.0.0.0/16	pcx-bbbbcccc

The owner of VPC C associates an IPv6 CIDR block with the VPC (`2001:db8:1234:cc00::/56`). VPC C can now communicate over IPv6 with both VPC A and VPC B using the existing VPC peering connection. To enable this, the following routes must be added to the existing route tables:

Route Tables	Destination	Target
VPC A	2001:db8:1234:cc00::/56	pcx-aaaacccc
VPC B	2001:db8:1234:cc00::/56	pcx-bbbbcccc
VPC C	2001:db8:1234:aa00::/56	pcx-aaaaacccc
	2001:db8:1234:bb00::/56	pcx-bbbbcccc

For more information about IPv6 in your VPC, see Your VPC and Subnets in the *Amazon VPC User Guide*.

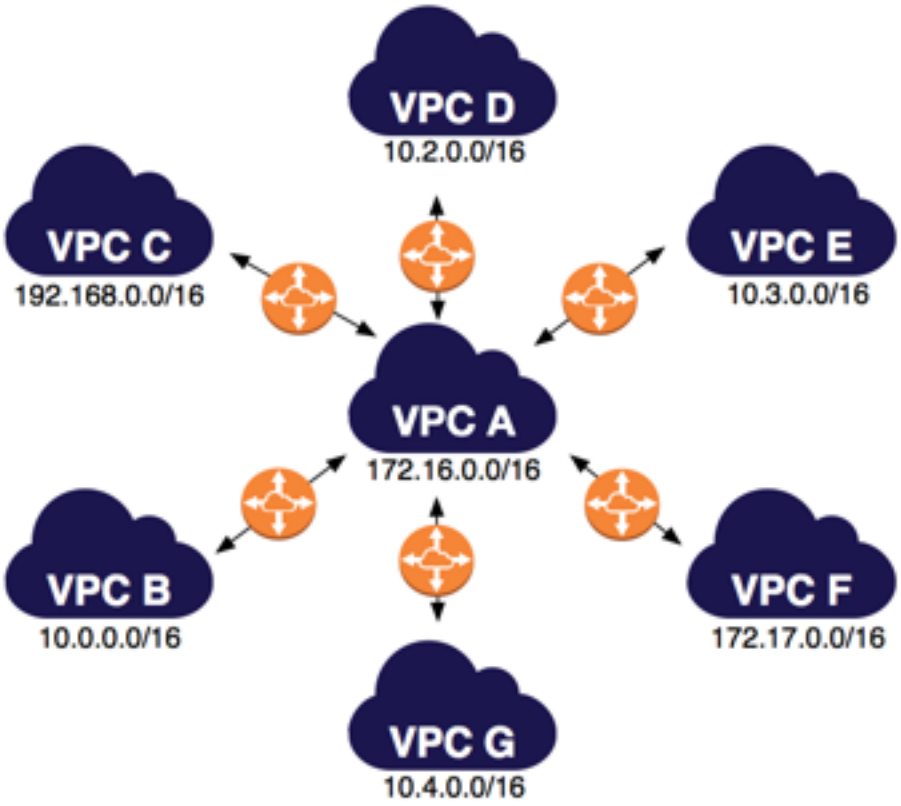
One VPC Peered with Multiple VPCs

You have a central VPC (VPC A) that's peered to the following VPCs:

- VPC B through `pcx-aaaabbbb`
- VPC C through `pcx-aaaacccc`
- VPC D through `pcx-aaaadddd`
- VPC E through `pcx-aaaaeeee`
- VPC F through `pcx-aaaaffff`
- VPC G through `pcx-aaaagggg`

VPC A is peered with all other VPCs, but the other VPCs are not peered to each other. The VPCs are in the same AWS account and do not have overlapping CIDR blocks.

Note
None of the other VPCs can send traffic directly to each other through VPC A. VPC peering does not support transitive peering relationships, nor edge to edge routing. You must create a VPC peering connection between the other VPCs in order to route traffic between them. For more information, see Multiple VPCs Peered Together (p. 29). For more information about unsupported peering scenarios, see Unsupported VPC Peering Configurations (p. 58).



You may want to use this spoke configuration when you have resources on a central VPC, such as a repository of services, that other VPCs need to access. The other VPCs do not need access to each others' resources; they only need access to resources on the central VPC.

The route tables for each VPC point to the relevant VPC peering connection to access the entire CIDR block of the peer VPC.

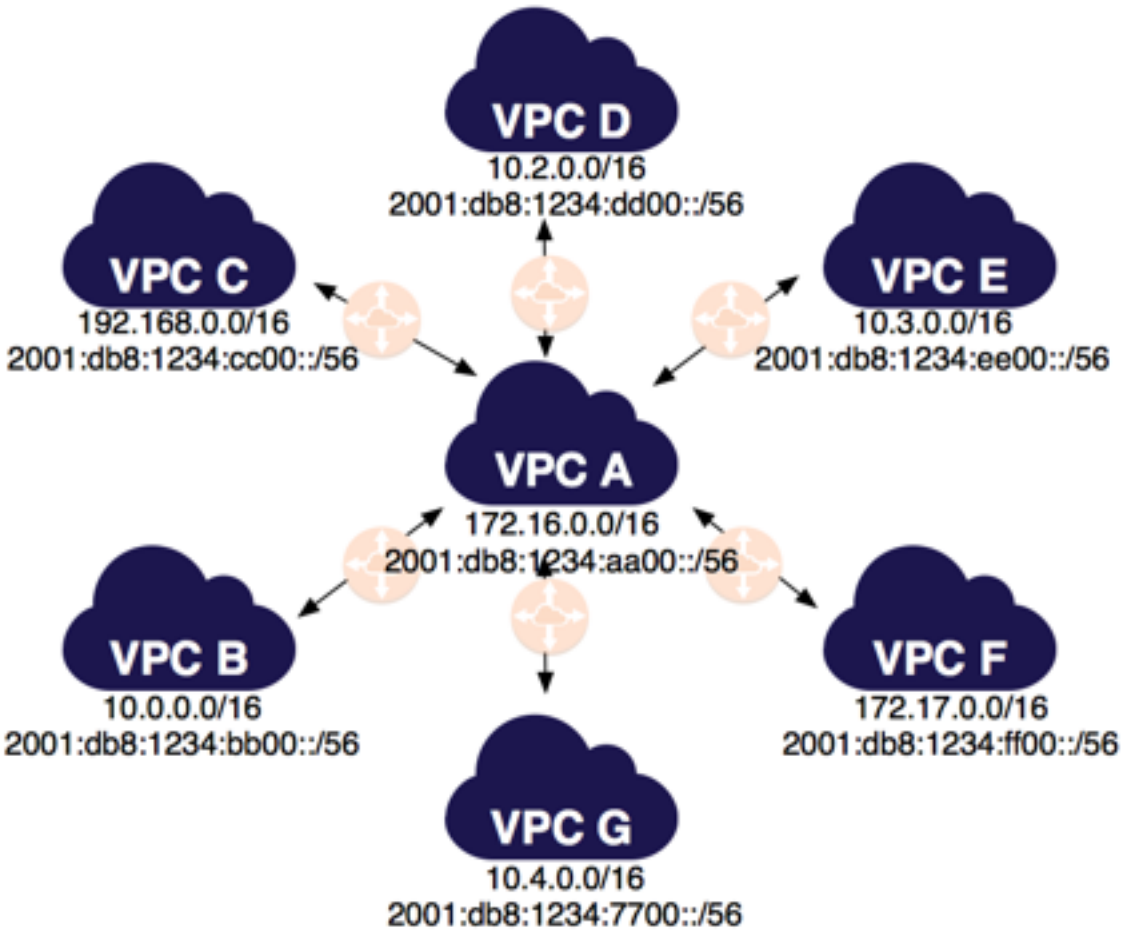
Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaacccc
	10.2.0.0/16	pcx-aaaadddd
	10.3.0.0/16	pcx-aaaaeeee

Route Table	Destination	Target
	172.17.0.0/16	pcx-aaaaffff
	10.4.0.0/16	pcx-aaaagggg
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc
VPC D	10.2.0.0/16	Local
	172.16.0.0/16	pcx-aaaadddd
VPC E	10.3.0.0/16	Local
	172.16.0.0/16	pcx-aaaaeeee
VPC F	172.17.0.0/16	Local
	172.16.0.0/16	pcx-aaaaffff
VPC G	10.4.0.0/16	Local
	172.16.0.0/16	pcx-aaaagggg

For more information about updating your route tables, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

One VPC Peered with Multiple VPCs for IPv6

You have the same VPCs in the VPC peering configuration as above. All VPCs have associated IPv6 CIDR blocks.



The route tables for each VPC point to the relevant VPC peering connection to access the entire IPv6 CIDR block of the peer VPC.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	2001:db8:1234:aa00::/56	Local
	10.0.0.0/16	pcx-aaaabbbb
	2001:db8:1234:bb00::/56	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaacccc
	2001:db8:1234:cc00::/56	pcx-aaaacccc
	10.2.0.0/16	pcx-aaaadddd
	2001:db8:1234:dd00::/56	pcx-aaaadddd
	10.3.0.0/16	pcx-aaaaeeee
	2001:db8:1234:ee00::/56	pcx-aaaaeeee

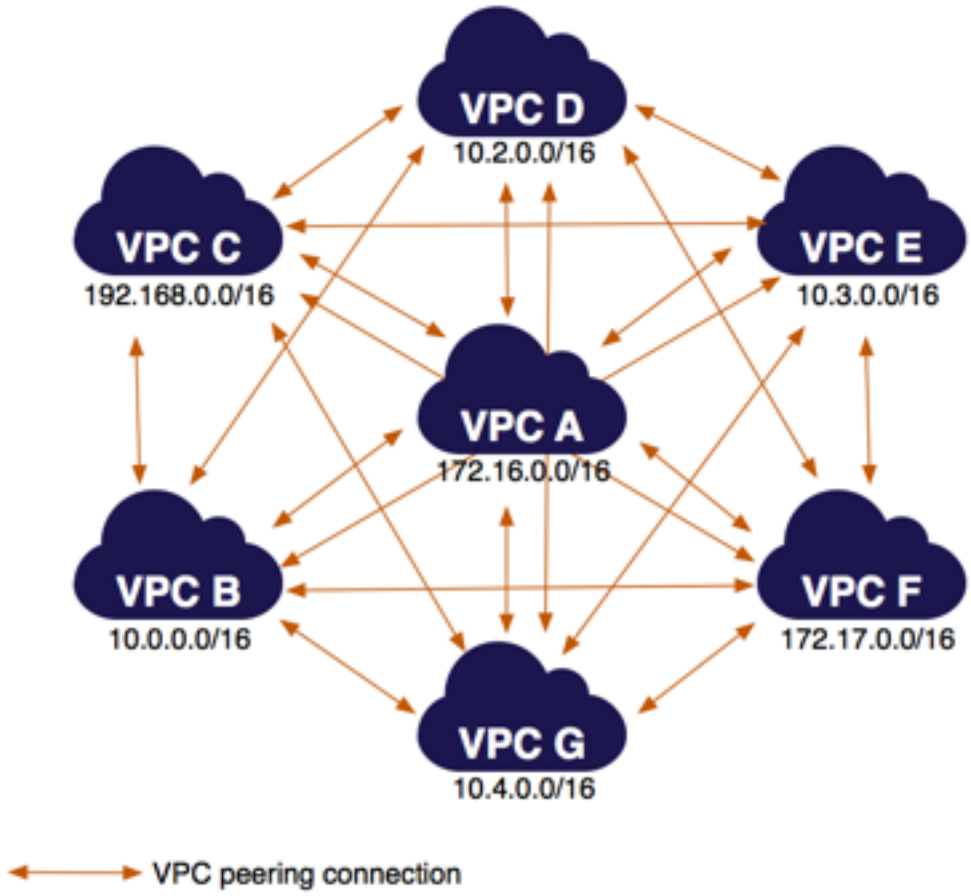
Route Table	Destination	Target
	172.17.0.0/16	pcx-aaaaffff
	2001:db8:1234:ff00::/56	pcx-aaaaffff
	10.4.0.0/16	pcx-aaaagggg
	2001:db8:1234:7700::/56	pcx-aaaagggg
VPC B	10.0.0.0/16	Local
	2001:db8:1234:bb00::/56	Local
	172.16.0.0/16	pcx-aaaabbbb
	2001:db8:1234:aa00::/56	pcx-aaaabbbb
VPC C	192.168.0.0/16	Local
	2001:db8:1234:cc00::/56	Local
	172.16.0.0/16	pcx-aaaacccc
	2001:db8:1234:aa00::/56	pcx-aaaacccc
VPC D	10.2.0.0/16	Local
	2001:db8:1234:dd00::/56	Local
	172.16.0.0/16	pcx-aaaadddd
	2001:db8:1234:aa00::/56	pcx-aaaadddd
VPC E	10.3.0.0/16	Local
	2001:db8:1234:ee00::/56	Local
	172.16.0.0/16	pcx-aaaaeeee
	2001:db8:1234:aa00::/56	pcx-aaaaeeee
VPC F	172.17.0.0/16	Local
	2001:db8:1234:ff00::/56	Local
	172.16.0.0/16	pcx-aaaaffff
	2001:db8:1234:aa00::/56	pcx-aaaaffff
VPC G	10.4.0.0/16	Local
	2001:db8:1234:7700::/56	Local
	172.16.0.0/16	pcx-aaaagggg
	2001:db8:1234:aa00::/56	pcx-aaaagggg

Multiple VPCs Peered Together

You have peered seven VPCs together in a full mesh configuration:

VPCs	VPC Peering Connection
A and B	pcx-aaaabbbb
A and C	pcx-aaaacccc
A and D	pcx-aaaadddd
A and E	pcx-aaaaeeee
A and F	pcx-aaaaffff
A and G	pcx-aaaagggg
B and C	pcx-bbbbcccc
B and D	pcx-bbbbdddd
B and E	pcx-bbbbeeee
B and F	pcx-bbbbffff
B and G	pcx-bbbbgggg
C and D	pcx-ccccdddd
C and E	pcx-cccceeee
C and F	pcx-ccccffff
C and G	pcx-ccccgggg
D and E	pcx-ddddeeee
D and F	pcx-ddddffff
D and G	pcx-ddddgggg
E and F	pcx-eeeeffff
E and G	pcx-eeeegggg
F and G	pcx-ffffgggg

The VPCs are in the same AWS account and do not have overlapping CIDR blocks.



You may want to use this full mesh configuration when you have multiple VPCs that must be able to access each others' resources without restriction; for example, as a file sharing network.

The route tables for each VPC point to the relevant VPC peering connection to access the entire CIDR block of the peer VPC.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaacccc
	10.2.0.0/16	pcx-aaaadddd
	10.3.0.0/16	pcx-aaaaeeee
	172.17.0.0/16	pcx-aaaaffff
	10.4.0.0/16	pcx-aaaagggg
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-bbbbcccc

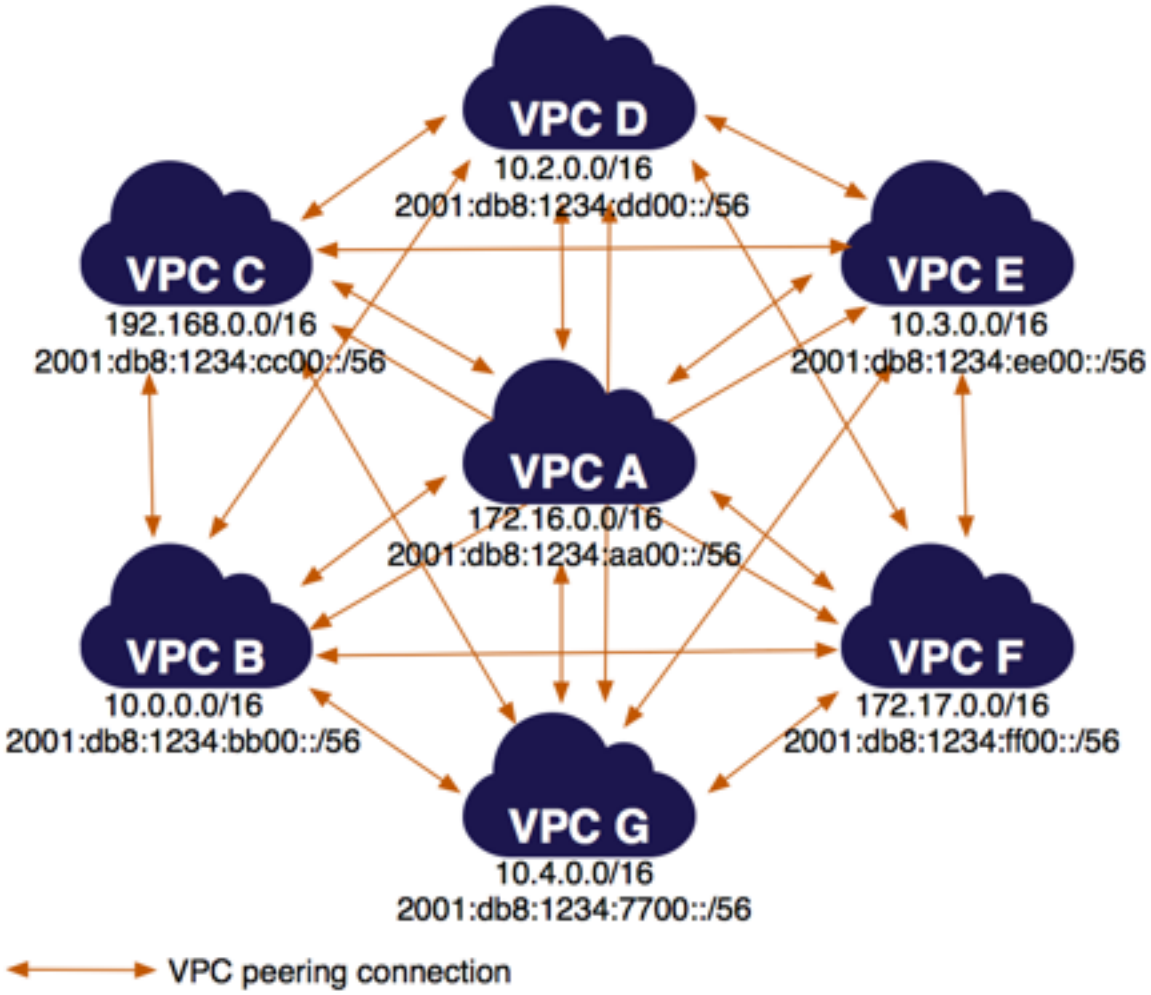
Route Table	Destination	Target
	10.2.0.0/16	pcx-bbbbdddd
	10.3.0.0/16	pcx-bbbbeeee
	172.17.0.0/16	pcx-bbbbffff
	10.4.0.0/16	pcx-bbbbgggg
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc
	10.0.0.0/16	pcx-bbbbcccc
	10.2.0.0/16	pcx-ccccdddd
	10.3.0.0/16	pcx-cccceeee
	172.17.0.0/16	pcx-ccccffff
	10.4.0.0/16	pcx-ccccgggg
VPC D	10.2.0.0/16	Local
	172.16.0.0/16	pcx-aaaadddd
	10.0.0.0/16	pcx-bbbbdddd
	192.168.0.0/16	pcx-ccccdddd
	10.3.0.0/16	pcx-ddddeeee
	172.17.0.0/16	pcx-ddddffff
	10.4.0.0/16	pcx-ddddgggg
VPC E	10.3.0.0/16	Local
	172.16.0.0/16	pcx-aaaaeeee
	10.0.0.0/16	pcx-bbbbeeee
	192.168.0.0/16	pcx-cccceeee
	10.2.0.0/16	pcx-ddddeeee
	172.17.0.0/16	pcx-eeeeffff
	10.4.0.0/16	pcx-eeeegggg
VPC F	172.17.0.0/16	Local
	172.16.0.0/16	pcx-aaaaffff
	10.0.0.0/16	pcx-bbbbffff
	192.168.0.0/16	pcx-ccccffff
	10.2.0.0/16	pcx-ddddffff
	10.3.0.0/16	pcx-eeeeffff

Route Table	Destination	Target
	10.4.0.0/16	pcx-ffffgggg
VPC G	10.4.0.0/16	Local
	172.16.0.0/16	pcx-aaaagggg
	10.0.0.0/16	pcx-bbbbgggg
	192.168.0.0/16	pcx-ccccgggg
	10.2.0.0/16	pcx-ddddgggg
	10.3.0.0/16	pcx-eeeegggg
	172.17.0.0/16	pcx-ffffgggg

For more information about updating route tables, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

Multiple VPCs Peered Together for IPv6

You have the same VPCs in the VPC peering configuration as above. All VPCs have associated IPv6 CIDR blocks.



The route tables for each VPC point to the VPC peering connection to access the entire IPv6 CIDR block of the peer VPC.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	2001:db8:1234:aa00::/56	Local
	10.0.0.0/16	pcx-aaaabbbb
	2001:db8:1234:bb00::/56	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaacccc
	2001:db8:1234:cc00::/56	pcx-aaaacccc
	10.2.0.0/16	pcx-aaaadddd
	2001:db8:1234:dd00::/56	pcx-aaaadddd
	10.3.0.0/16	pcx-aaaaeeee

Route Table	Destination	Target
	2001:db8:1234:ee00::/56	pcx-aaaaeeee
	172.17.0.0/16	pcx-aaaaffff
	2001:db8:1234:ff00::/56	pcx-aaaaffff
	10.4.0.0/16	pcx-aaaagggg
	2001:db8:1234:7700::/56	pcx-aaaagggg
VPC B	10.0.0.0/16	Local
	2001:db8:1234:bb00::/56	Local
	172.16.0.0/16	pcx-aaaabbbb
	2001:db8:1234:aa00::/56	pcx-aaaabbbb
	192.168.0.0/16	pcx-bbbbcccc
	2001:db8:1234:cc00::/56	pcx-bbbbcccc
	10.2.0.0/16	pcx-bbbbdddd
	2001:db8:1234:dd00::/56	pcx-bbbbdddd
	10.3.0.0/16	pcx-bbbbeeee
	2001:db8:1234:ee00::/56	pcx-bbbbeeee
	172.17.0.0/16	pcx-bbbbffff
	2001:db8:1234:ff00::/56	pcx-bbbbffff
	10.4.0.0/16	pcx-bbbbgggg
	2001:db8:1234:7700::/56	pcx-bbbbgggg
VPC C	192.168.0.0/16	Local
	2001:db8:1234:cc00::/56	Local
	172.16.0.0/16	pcx-aaaacccc
	2001:db8:1234:aa00::/56	pcx-aaaacccc
	10.0.0.0/16	pcx-bbbbcccc
	2001:db8:1234:bb00::/56	pcx-bbbbcccc
	10.2.0.0/16	pcx-ccccdddd
	2001:db8:1234:dd00::/56	pcx-ccccdddd
	10.3.0.0/16	pcx-ccceeeee
	2001:db8:1234:ee00::/56	pcx-ccceeeee
	172.17.0.0/16	pcx-ccccffff
	2001:db8:1234:ff00::/56	pcx-ccccffff

Route Table	Destination	Target
	10.4.0.0/16	pcx-ccccgggg
	2001:db8:1234:7700::/56	pcx-ccccgggg
VPC D	10.2.0.0/16	Local
	2001:db8:1234:dd00::/56	Local
	172.16.0.0/16	pcx-aaaadddd
	2001:db8:1234:aa00::/56	pcx-aaaadddd
	10.0.0.0/16	pcx-bbbbdddd
	2001:db8:1234:bb00::/56	pcx-bbbbdddd
	192.168.0.0/16	pcx-ccccdddd
	2001:db8:1234:cc00::/56	pcx-ccccdddd
	10.3.0.0/16	pcx-ddddeeee
	2001:db8:1234:ee00::/56	pcx-ddddeeee
	172.17.0.0/16	pcx-ddddffff
	2001:db8:1234:ff00::/56	pcx-ddddffff
	10.4.0.0/16	pcx-ddddgggg
	2001:db8:1234:7700::/56	pcx-ddddgggg
VPC E	10.3.0.0/16	Local
	2001:db8:1234:ee00::/56	Local
	172.16.0.0/16	pcx-aaaaeeee
	2001:db8:1234:aa00::/56	pcx-aaaaeeee
	10.0.0.0/16	pcx-bbbbeeee
	2001:db8:1234:bb00::/56	pcx-bbbbeeee
	192.168.0.0/16	pcx-cccceeee
	2001:db8:1234:cc00::/56	pcx-cccceeee
	10.2.0.0/16	pcx-ddddeeee
	2001:db8:1234:dd00::/56	pcx-ddddeeee
	172.17.0.0/16	pcx-eeeeffff
	2001:db8:1234:ff00::/56	pcx-eeeeffff
	10.4.0.0/16	pcx-eeeegggg
	2001:db8:1234:7700::/56	pcx-eeeegggg
VPC F	172.17.0.0/16	Local

Route Table	Destination	Target
	2001:db8:1234:ff00::/56	Local
	172.16.0.0/16	pcx-aaaaffff
	2001:db8:1234:aa00::/56	pcx-aaaaffff
	10.0.0.0/16	pcx-bbbbffff
	2001:db8:1234:bb00::/56	pcx-bbbbffff
	192.168.0.0/16	pcx-ccccffff
	2001:db8:1234:cc00::/56	pcx-ccccffff
	10.2.0.0/16	pcx-ddddffff
	2001:db8:1234:dd00::/56	pcx-ddddffff
	10.3.0.0/16	pcx-eeeeffff
	2001:db8:1234:ee00::/56	pcx-eeeeffff
	10.4.0.0/16	pcx-ffffgggg
	2001:db8:1234:7700::/56	pcx-ffffgggg
VPC G	10.4.0.0/16	Local
	2001:db8:1234:7700::/56	Local
	172.16.0.0/16	pcx-aaaagggg
	2001:db8:1234:aa00::/56	pcx-aaaagggg
	10.0.0.0/16	pcx-bbbbgggg
	2001:db8:1234:bb00::/56	pcx-bbbbgggg
	192.168.0.0/16	pcx-ccccgggg
	2001:db8:1234:cc00::/56	pcx-ccccgggg
	10.2.0.0/16	pcx-ddddgggg
	2001:db8:1234:dd00::/56	pcx-ddddgggg
	10.3.0.0/16	pcx-eeeegggg
	2001:db8:1234:ee00::/56	pcx-eeeegggg
	172.17.0.0/16	pcx-ffffgggg
	2001:db8:1234:ff00::/56	pcx-ffffgggg

Configurations with Specific Routes

You can configure a VPC peering connections to provide access to part of the CIDR block, a specific CIDR block (if the VPC has multiple CIDR blocks) or a specific instance within the peer VPC. In these examples, a central VPC is peered to two or more VPCs that have overlapping CIDR blocks. For examples

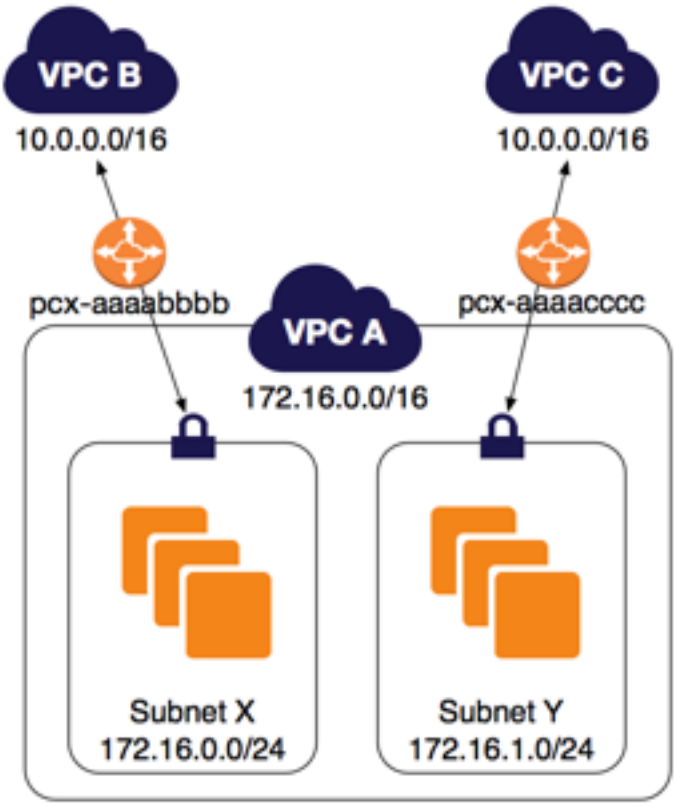
of scenarios in which you might need a specific VPC peering connection configuration, see VPC Peering Scenarios (p. 17). For more information about creating and working with VPC peering connections, see Working with VPC Peering Connections (p. 6). For more information about updating your route tables, see Updating Your Route Tables for a VPC Peering Connection (p. 10).

Configurations

- Two VPCs Peered to Two Subnets in One VPC (p. 38)
- Two VPCs Peered to a Specific CIDR Block in One VPC (p. 42)
- One VPC Peered to Specific Subnets in Two VPCs (p. 42)
- Instances in One VPC Peered to Instances in Two VPCs (p. 46)
- One VPC Peered with Two VPCs Using Longest Prefix Match (p. 48)
- Multiple VPC Configurations (p. 49)

Two VPCs Peered to Two Subnets in One VPC

You have a central VPC (VPC A), and you have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`), and between VPC A and VPC C (`pcx-aaaacccc`). VPC A has two subnets: one for each VPC peering connection.



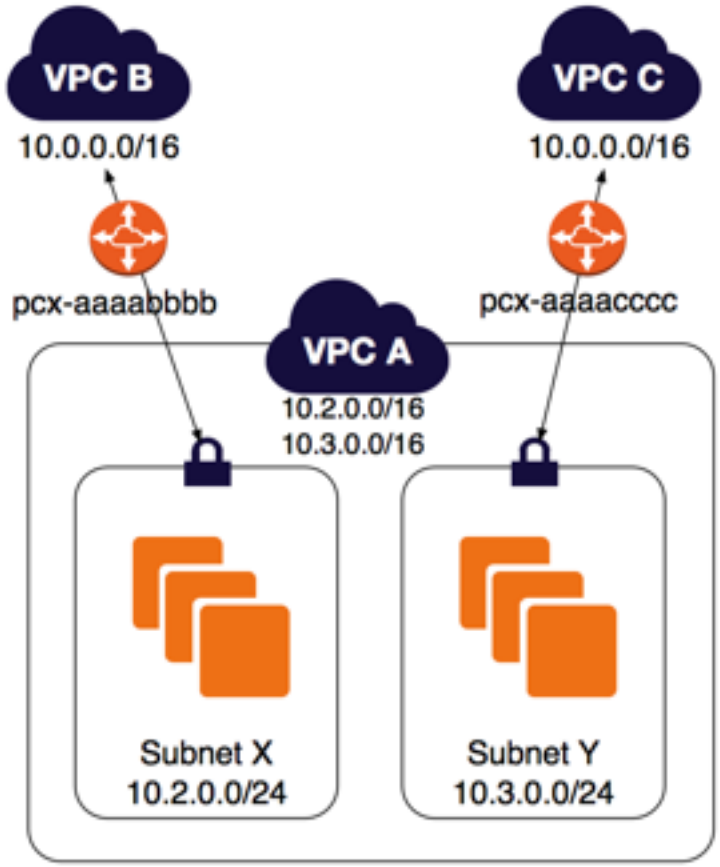
You may want to use this kind of configuration when you have a central VPC with separate sets of resources in different subnets. Other VPCs may require access to some of the resources, but not all of them.

The route table for subnet X points to VPC peering connection `pcx-aaaabbbb` to access the entire CIDR block of VPC B. VPC B's route table points to `pcx-aaaabbbb` to access the CIDR block of only subnet X in

VPC A. Similarly, the route table for subnet Y points to VPC peering connection `pcx-aaaacccc` to access the entire CIDR block of VPC C. VPC C's route table points to `pcx-aaaacccc` to access the CIDR block of only subnet Y in VPC A.

Route Table	Destination	Target
Subnet X in VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
Subnet Y in VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaacccc
VPC B	10.0.0.0/16	Local
	172.16.0.0/24	pcx-aaaabbbb
VPC C	10.0.0.0/16	Local
	172.16.1.0/24	pcx-aaaacccc

Similarly, the central VPC (VPC A) can have multiple CIDR blocks, and VPC B and VPC C can have a VPC peering connection to a subnet in each CIDR block.

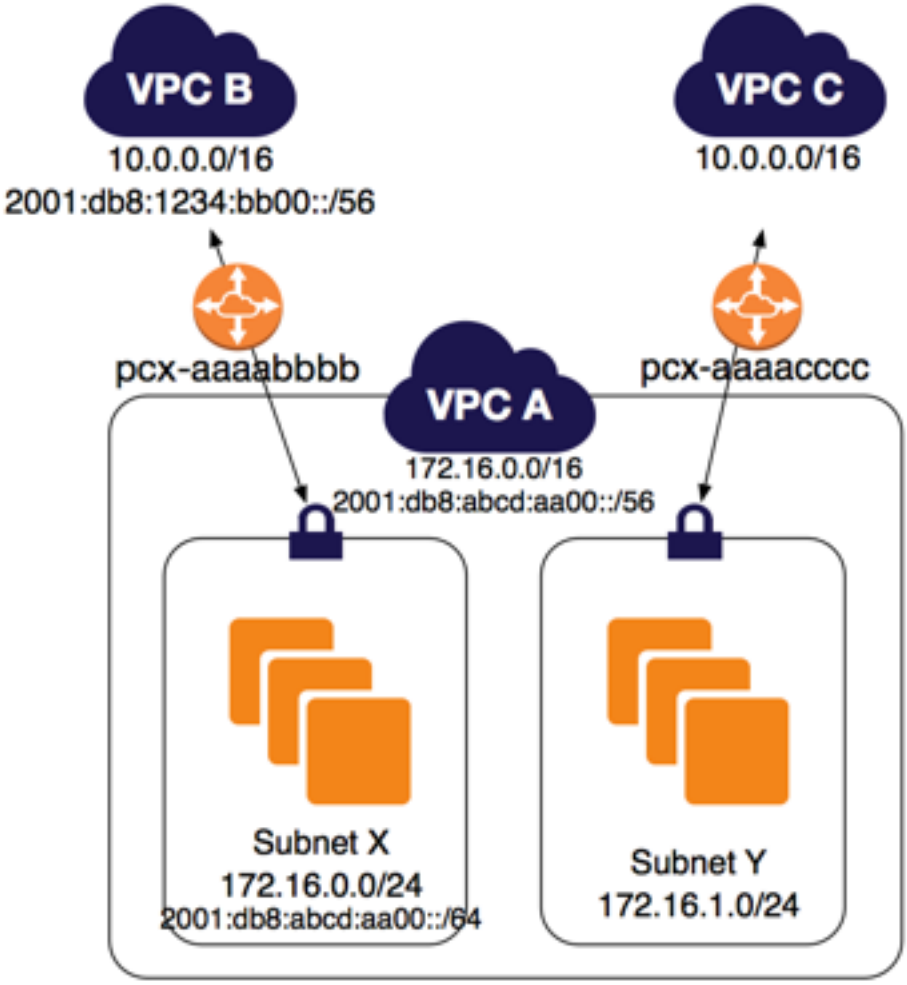


Route Table	Destination	Target
Subnet X in VPC A	10.2.0.0/16	Local
	10.3.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
Subnet Y in VPC A	10.2.0.0/16	Local
	10.3.0.0/16	Local
	10.0.0.0/16	pcx-aaaacccc
VPC B	10.0.0.0/16	Local
	10.2.0.0/24	pcx-aaaabbbb
VPC C	10.0.0.0/16	Local
	10.3.0.0/24	pcx-aaaacccc

For more information, see Adding IPv4 CIDR Blocks to a VPC in the *Amazon VPC User Guide*.

Two VPCs Peered to Two Subnets in One VPC for IPv6

You have the same VPC peering configuration as above. VPC A and VPC B are enabled for IPv6, therefore both VPCs have associated IPv6 CIDR blocks. Subnet X in VPC A has an associated IPv6 CIDR block.



You can enable VPC B to communicate with subnet X in VPC A over IPv6 using the VPC peering connection. To do this, add a route to the route table for VPC A with a destination of the IPv6 CIDR block for VPC B, and a route to the route table for VPC B with a destination of the IPv6 CIDR of subnet X in VPC A.

Route Table	Destination	Target	Notes
Subnet X in VPC A	172.16.0.0/16	Local	
	2001:db8:abcd:aa00::/56	Local	Local route that's automatically added for IPv6 communication within the VPC.
	10.0.0.0/16	pcx-aaaabbbb	
	2001:db8:1234:bb00::/56	pcx-aaaabbbb	Route to the IPv6 CIDR block of VPC B.
Subnet Y in VPC A	172.16.0.0/16	Local	

Route Table	Destination	Target	Notes
	2001:db8:abcd:aa00::/56	Local	Local route that's automatically added for IPv6 communication within the VPC.
	10.0.0.0/16	pcx-aaaacccc	
VPC B	10.0.0.0/16	Local	
	2001:db8:1234:bb00::/56	Local	Local route that's automatically added for IPv6 communication within the VPC.
	172.16.0.0/24	pcx-aaaabbbb	
	2001:db8:abcd:aa00::/64	pcx-aaaabbbb	Route to the IPv6 CIDR block of VPC A.
VPC C	10.0.0.0/16	Local	
	172.16.1.0/24	pcx-aaaacccc	

Two VPCs Peered to a Specific CIDR Block in One VPC

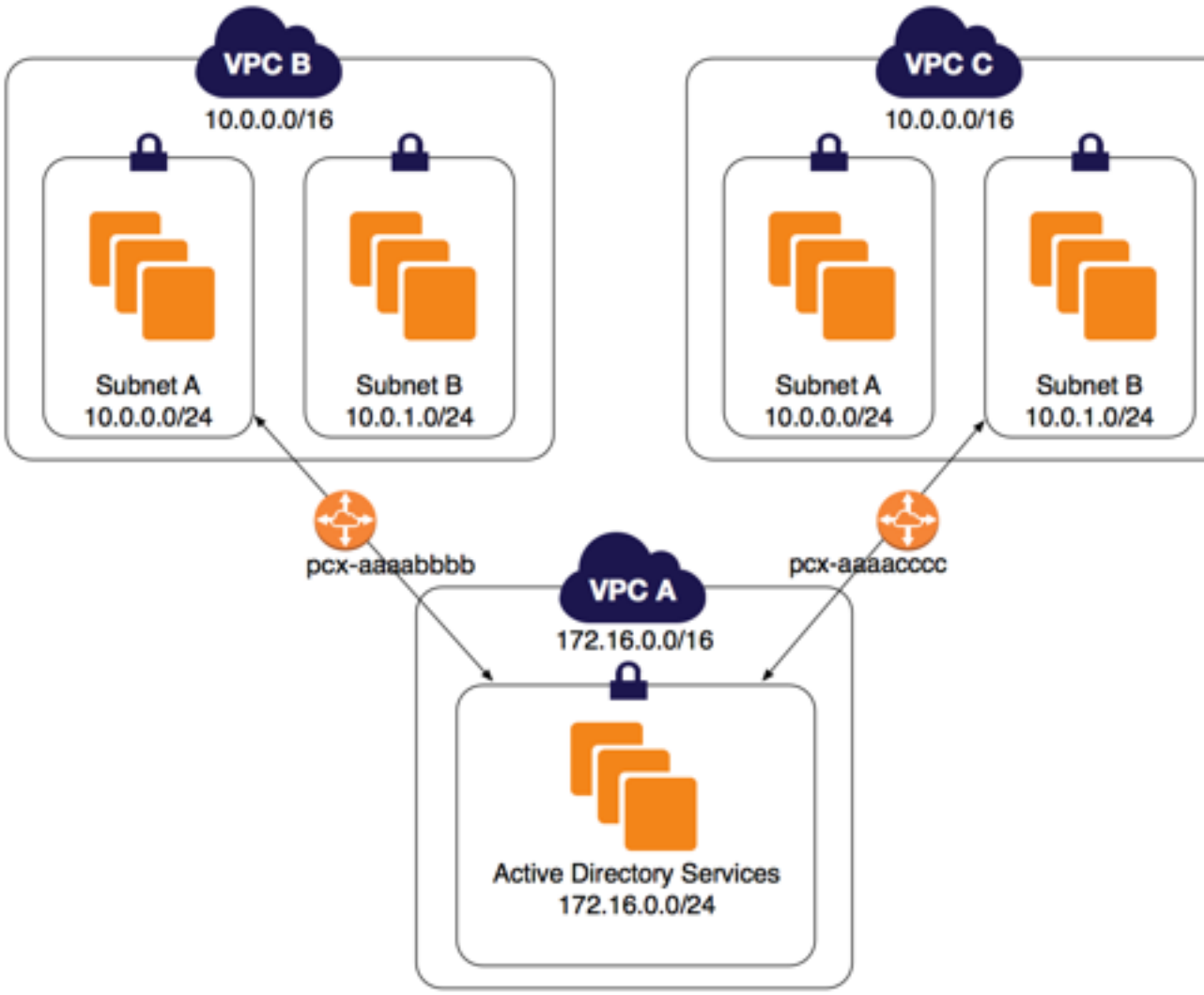
You have a central VPC (VPC A), and you have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`), and between VPC A and VPC C (`pcx-aaaacccc`). VPC A has two CIDR blocks; one for each VPC peering connection.

Route Table	Destination	Target
VPC A	10.0.0.0/16	Local
	10.2.0.0/16	Local
	172.31.0.0/16	pcx-aaaabbbb
	192.168.0.0/24	pcx-aaaacccc
VPC B	172.31.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
VPC C	192.168.0.0/24	Local
	10.2.0.0/16	pcx-aaaacccc

For more information, see Adding IPv4 CIDR Blocks to a VPC in the *Amazon VPC User Guide*.

One VPC Peered to Specific Subnets in Two VPCs

You have a central VPC (VPC A) with one subnet, and you have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`), and between VPC A and VPC C (`pcx-aaaacccc`). VPC B and VPC C each have two subnets, and only one in each is used for the peering connection with VPC A.



You may want to use this kind of configuration when you have a central VPC that has a single set of resources, such as Active Directory services, that other VPCs need to access. The central VPC does not require full access to the VPCs that it's peered with.

The route table for VPC A points to both VPC peering connections to access only specific subnets in VPC B and VPC C. The route tables for the subnets in VPC B and VPC C point to their VPC peering connections to access the VPC A subnet.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/24	pcx-aaaabbbb
	10.0.1.0/24	pcx-aaaacccc
Subnet A in VPC B	10.0.0.0/16	Local
	172.16.0.0/24	pcx-aaaabbbb

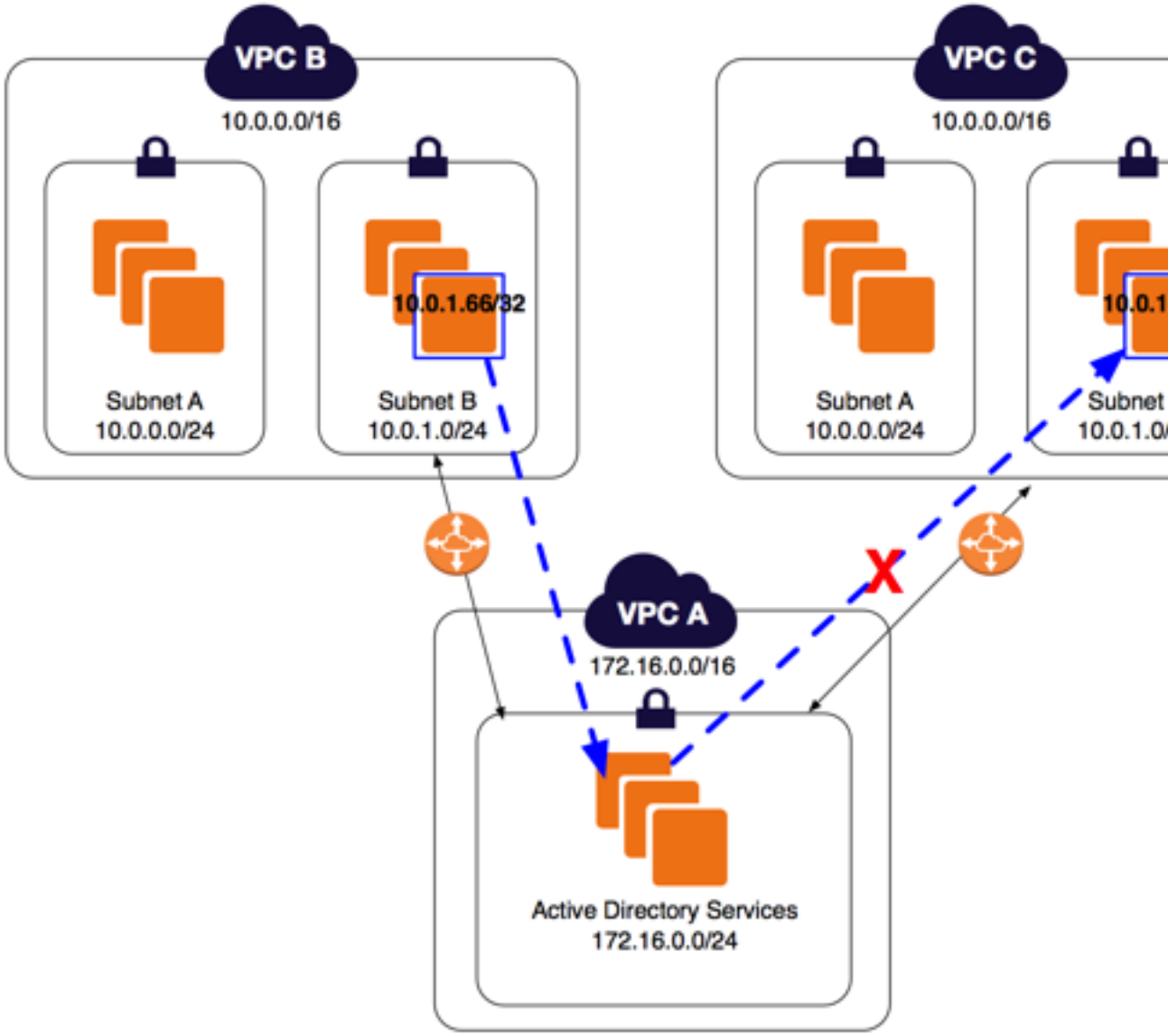
Route Table	Destination	Target
Subnet B in VPC C	10.0.0.0/16	Local
	172.16.0.0/24	pcx-aaaacccc

Routing for Response Traffic

If you have a VPC peered with multiple VPCs that have overlapping or matching CIDR blocks, ensure that your route tables are configured to avoid sending response traffic from your VPC to the incorrect VPC. AWS currently does not support unicast reverse path forwarding in VPC peering connections that checks the source IP of packets and routes reply packets back to the source.

For example, VPC A is peered with VPC B and VPC C. VPC B and VPC C have matching CIDR blocks, and their subnets have matching CIDR blocks. The route table for subnet B in VPC B points to the VPC peering connection `pcx-aaaabbbb` to access the VPC A subnet. The VPC A route table is configured to send `10.0.0.0/16` traffic to peering connection `pcx-aaaaccccc`.

Route Table	Destination	Target
Subnet B in VPC B	10.0.0.0/16	Local
	172.16.0.0/24	pcx-aaaabbbb
VPC A	172.16.0.0/24	Local
	10.0.0.0/16	pcx-aaaacccc



An instance in subnet B in VPC B with a private IP address of `10.0.1.66/32` sends traffic to the Active Directory server in VPC A using VPC peering connection `pcx-aaaabbbb`. VPC A sends the response traffic to `10.0.1.66/32`. However, the VPC A route table is configured to send all traffic within the `10.0.0.0/16` range of IP addresses to VPC peering connection `pcx-aaaacccc`. If subnet B in VPC C has an instance with an IP address of `10.0.1.66/32`, it receives the response traffic from VPC A. The instance in subnet B in VPC B does not receive a response to its request to VPC A.

To prevent this, you can add a specific route to VPC A's route table with a destination of `10.0.1.0/24` and a target of `pcx-aaaabbbb`. The route for `10.0.1.0/24` traffic is more specific, therefore traffic destined for the `10.0.1.0/24` IP address range goes via VPC peering connection `pcx-aaaabbbb`

Alternatively, in the following example, VPC A's route table has a route for each subnet for each VPC peering connection. VPC A can communicate with subnet B in VPC B and with subnet A in VPC C. This scenario is useful if you need to add another VPC peering connection with another subnet that falls within the `10.0.0.0/16` IP address range —you can simply add another route for that specific subnet.

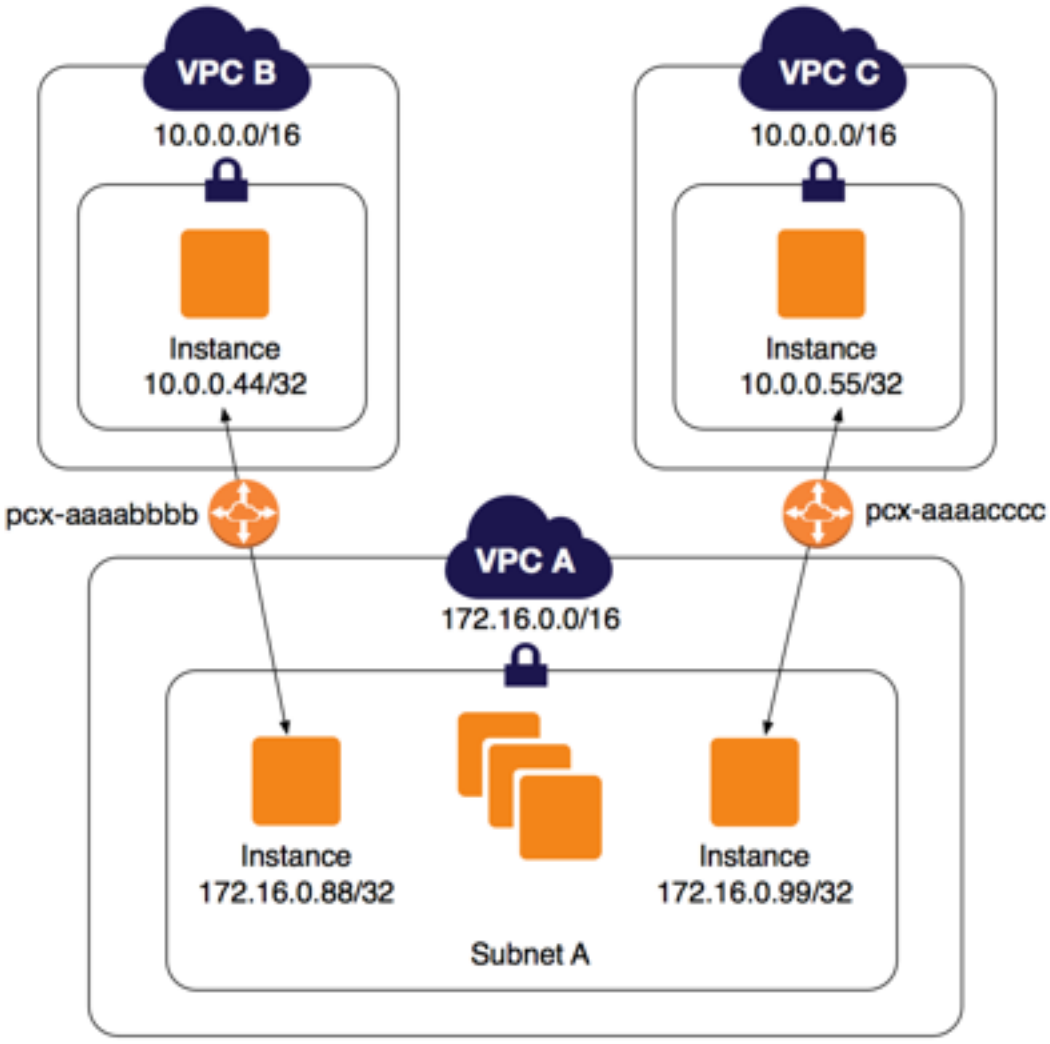
Destination	Target
172.16.0.0/16	Local
10.0.1.0/24	pcx-aaaabbbb
10.0.0.0/24	pcx-aaaacccc

Alternatively, depending on your use case, you can create a route to a specific IP address in VPC B to ensure that traffic routed back to the correct server (the route table uses longest prefix match to prioritize the routes):

Destination	Target
172.16.0.0/16	Local
10.0.1.66/32	pcx-aaaabbbb
10.0.0.0/16	pcx-aaaacccc

Instances in One VPC Peered to Instances in Two VPCs

You have a central VPC (VPC A) with one subnet, and you have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`), and between VPC A and VPC C (`pcx-aaaacccc`). VPC A has one subnet that has multiple instances; one for each of the VPCs that it's peered with. You may want to use this kind of configuration to limit peering traffic to specific instances.

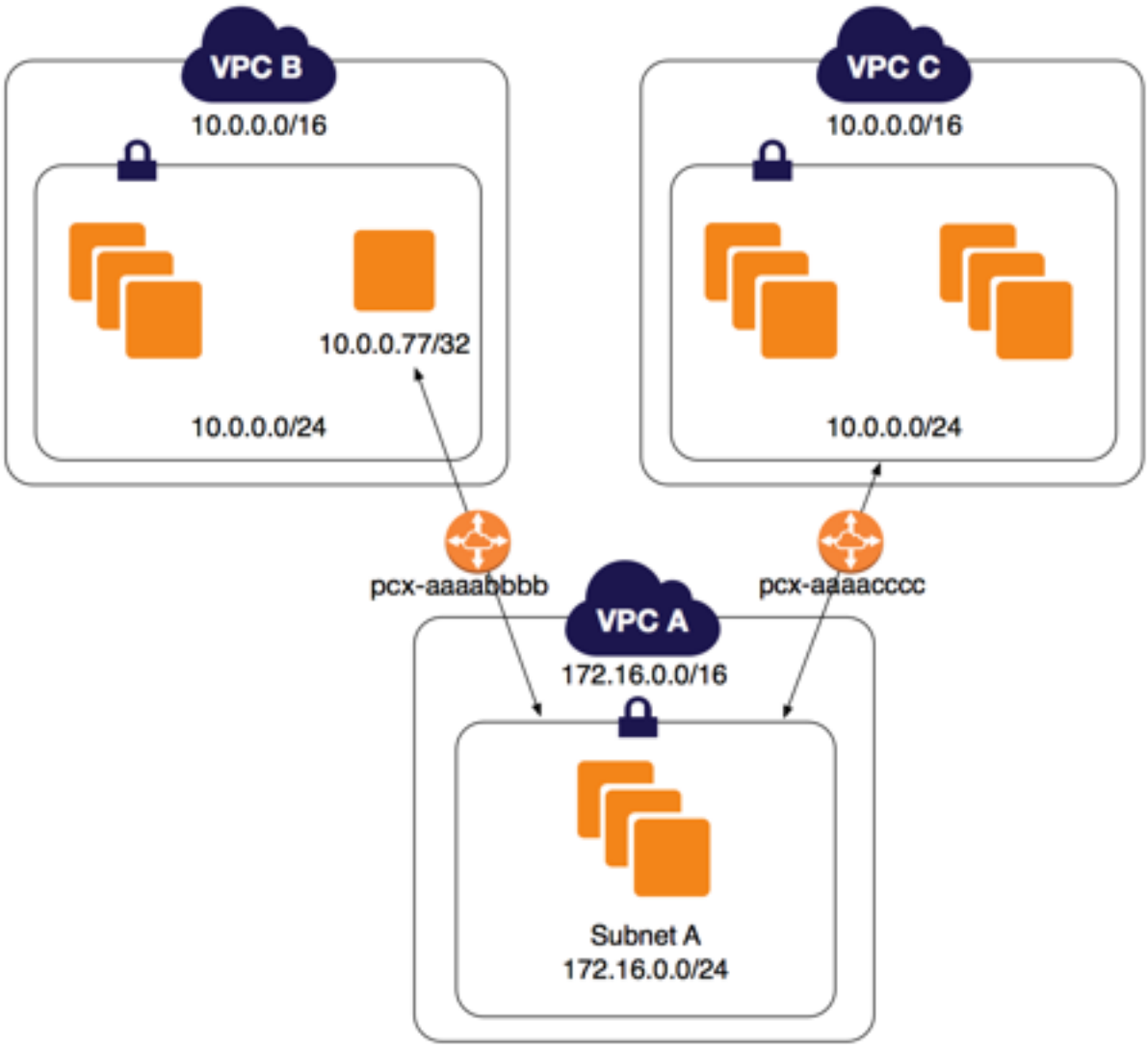


Each VPC route table points to the relevant VPC peering connection to access a single IP address (and therefore a specific instance) in the peer VPC.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.44/32	pcx-aaaabbbb
	10.0.0.55/32	pcx-aaaacccc
VPC B	10.0.0.0/16	Local
	172.16.0.88/32	pcx-aaaabbbb
VPC C	10.0.0.0/16	Local
	172.16.0.99/32	pcx-aaaacccc

One VPC Peered with Two VPCs Using Longest Prefix Match

You have a central VPC (VPC A) with one subnet, and you have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`), and between VPC A and VPC C (`pcx-aaaacccc`). VPC B and VPC C have matching CIDR blocks. You want to use VPC peering connection `pcx-aaaabbbb` to route traffic between VPC A and specific instance in VPC B. All other traffic destined for the `10.0.0.0/16` IP address range is routed through `pcx-aaaacccc` between VPC A and VPC C.



VPC route tables use longest prefix match to select the most specific route across the intended VPC peering connection. All other traffic is routed through the next matching route, in this case, across the VPC peering connection `pcx-aaaacccc`.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local

Route Table	Destination	Target
	10.0.0.77/32	pcx-aaaabbbb
	10.0.0.0/16	pcx-aaaacccc
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
VPC C	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc

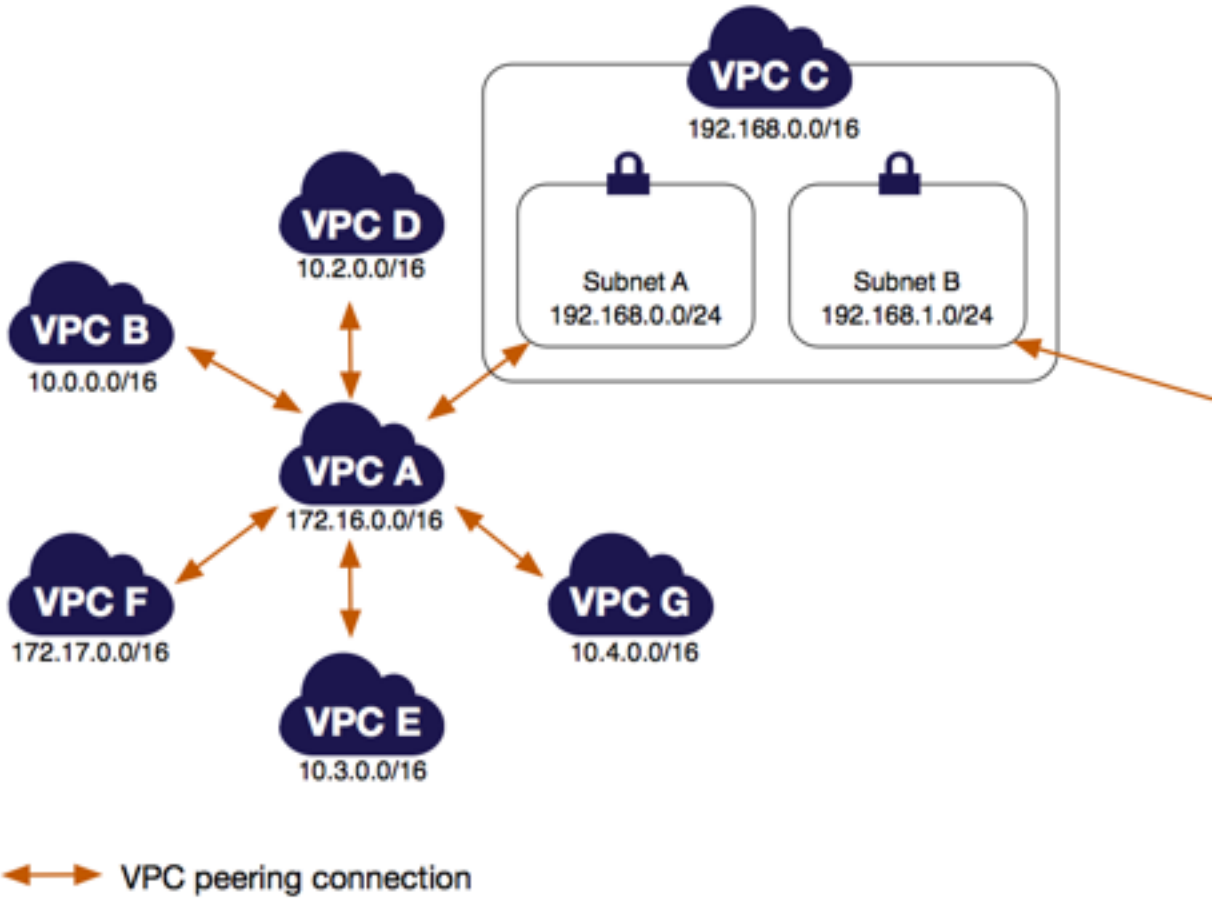
Important
If an instance other than `10.0.0.77/32` in VPC B sends traffic to VPC A, the response traffic may be routed to VPC C instead of VPC B. For more information, see Routing for Response Traffic (p. 44).

Multiple VPC Configurations

In this example, a central VPC (VPC A) is peered with multiple VPCs in a spoke configuration. For more information about this type of configuration, see One VPC Peered with Multiple VPCs (p. 25). You also have three VPCs (VPCs M, N, and P) peered together in a full mesh configuration. For more information about this type of configuration, see Three VPCs Peered Together (p. 23).

VPC C also has a VPC peering connection with VPC M (`pcx-ccccmmmm`). VPC A and VPC M have overlapping CIDR blocks. This means that peering traffic between VPC A and VPC C is limited to a specific subnet (subnet A) in VPC C. This is to ensure that if VPC C receives a request from VPC A or VPC M, it sends the response traffic to the correct VPC. AWS currently does not support unicast reverse path forwarding in VPC peering connections that checks the source IP of packets and routes reply packets back to the source. For more information, see Routing for Response Traffic (p. 44).

Similarly, VPC C and VPC P have overlapping CIDR blocks. Peering traffic between VPC M and VPC C is limited to subnet B in VPC C, and peering traffic between VPC M and VPC P is limited to subnet A in VPC P. This is to ensure that if VPC M receives peering traffic from VPC C or VPC P, it sends the response traffic back to the correct VPC.



The route tables for VPCs B, D, E, F, and G point to the relevant peering connections to access the full CIDR block for VPC A, and the VPC A route table points to the relevant peering connections for VPCs B, D, E, F, and G to access their full CIDR blocks. For peering connection `pcx-aaaacccc`, the VPC A route table routes traffic only to subnet A in VPC C (`192.168.0.0/24`) and the subnet A route table in VPC C points to the full CIDR block of VPC A.

The VPC N route table points to the relevant peering connections to access the full CIDR blocks of VPC M and VPC P, and the VPC P route table points to the relevant peering connection to access the full CIDR block of VPC N. The subnet A route table in VPC P points to the relevant peering connection to access the full CIDR block of VPC M. The VPC M route table points to the relevant peering connection to access subnet B in VPC C, and subnet A in VPC P.

Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
	192.168.0.0/24	pcx-aaaacccc
	10.2.0.0/16	pcx-aaaadddd
	10.3.0.0/16	pcx-aaaaeeee
	172.17.0.0/16	pcx-aaaaffff

Route Table	Destination	Target
	10.4.0.0/16	pcx-aaaagggg
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
Subnet Ain VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc
Subnet B in VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-ccccmmmm
VPC D	10.2.0.0/16	Local
	172.16.0.0/16	pcx-aaaadddd
VPC E	10.3.0.0/16	Local
	172.16.0.0/16	pcx-aaaaeeee
VPC F	172.17.0.0/16	Local
	172.16.0.0/16	pcx-aaaaaffff
VPC G	10.4.0.0/16	Local
	172.16.0.0/16	pcx-aaaagggg
VPC M	172.16.0.0/16	Local
	192.168.1.0/24	pcx-ccccmmmm
	10.0.0.0/16	pcx-mmmmnnnn
	192.168.0.0/24	pcx-mmmmpppp
VPC N	10.0.0.0/16	Local
	172.16.0.0/16	pcx-mmmmnnnn
	192.168.0.0/16	pcx-nnnnpppp
VPC P	192.168.0.0/16	Local
	10.0.0.0/16	pcx-nnnnpppp
	172.16.0.0/16	pcx-mmmmpppp

Configurations with ClassicLink

If you have a VPC peering connection between two VPCs, and there are one or more EC2-Classic instances that are linked to one or both of the VPCs using ClassicLink, you can extend the VPC peering connection to enable communication between the EC2-Classic instances and the instances in the VPC on the other side of the VPC peering connection. This enables the EC2-Classic instances and the instances in the VPC to communicate using private IP addresses. To do this, you enable a local VPC to communicate

with a linked EC2-Classic instance in a peer VPC, or you enable a local linked EC2-Classic instance to communicate with VPC instances in a peer VPC.

Communication over ClassicLink only works if both VPCs in the VPC peering connection are in the same region.

Important
EC2-Classic instances cannot be enabled for IPv6 communication. You can enable VPC instances on either side of a VPC peering connection to communicate with each other over IPv6; however, an EC2-Classic instance that's ClassicLinked with a VPC can communicate with VPC instances on either side of the VPC peering connection over IPv4 only.

To enable your VPC peering connection for communication with linked EC2-Classic instances, you must modify the requester VPC peering options if you are the requester of the VPC peering connection, and you must modify the accepter VPC peering options if you are the accepter of the VPC peering connection. You can use the describe-vpc-peering-connections command to verify which VPC is the accepter and the requester for a VPC peering connection.

You can modify the VPC peering connection options as follows:

- **Enable a local linked EC2-Classic instance to communicate with instances in a peer VPC**

  In this case, you modify the VPC peering connection options to enable outbound communication from the local ClassicLink connection to the peer VPC on the other side of the VPC peering connection. The owner of the peer VPC modifies the VPC peering connection options to enable outbound communication from their local VPC to the remote ClassicLink connection.
- **Enable a local VPC to communicate with a linked EC2-Classic instance in a peer VPC**

  In this case, you modify the VPC peering connection options to enable outbound communication from your local VPC to the remote ClassicLink connection on the other side of the VPC peering connection. The owner of the peer VPC with the linked EC2-Classic instance modifies the VPC peering connection options to enable outbound communication from their local ClassicLink connection to the remote VPC.

When you enable a local linked EC2-Classic instance to communicate with instances in a peer VPC, you must manually add a route to the main route table of your local VPC with a destination of the peer VPC CIDR block, and a target of the VPC peering connection. The linked EC2-Classic instance is not associated with any subnet in the VPC; it relies on the main route table for communication with the peer VPC.

Important
The route for the VPC peering connection must be added to the main route table, regardless of any custom route tables with existing routes to the peering connection. If not, the EC2-Classic instance cannot communicate with the peer VPC.

When you enable a local VPC for communication with a remote ClassicLink connection, a route is automatically added to all the local VPC route tables with a destination of `10.0.0.0/8` and a target of `Local`. This enables communication with the remote linked EC2-Classic instance. If your route table has an existing static route in the `10.0.0.0/8` IP address range (including VPC peering connection routes), you cannot enable the local VPC for communication with the remote ClassicLink connection.

Region Support

You can modify the VPC peering connection options in the following regions:

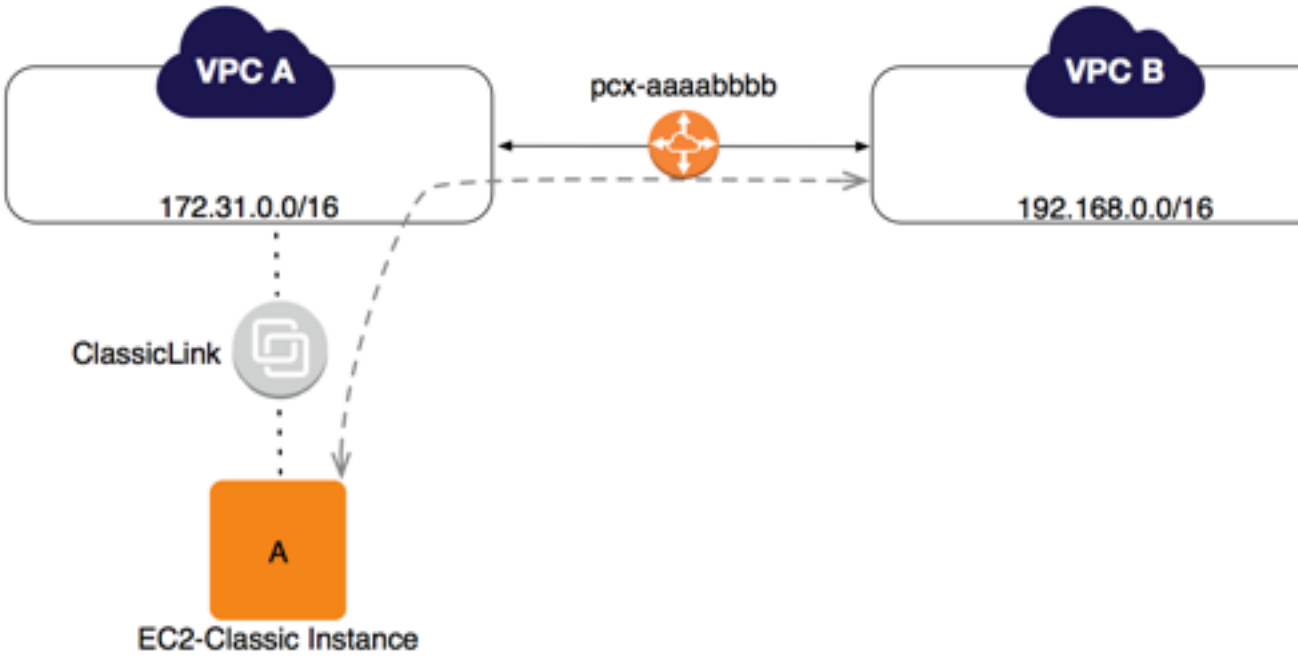
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Europe (Ireland)
- Asia Pacific (Tokyo)
- Asia Pacific (Singapore)

- South America (São Paulo)
- Asia Pacific (Sydney)

Enabling Communication Between a ClassicLink Instance and a Peer VPC

In the following scenario, VPC A is enabled for ClassicLink, and instance A is linked to VPC A using ClassicLink. VPC B is in a different AWS account, and is peered to VPC A using VPC peering connection `pcx-aaaabbbb`. The VPC peering connection was requested by VPC A and accepted by VPC B. You want instance A to communicate with instances in VPC B over private IP, and you want instances in VPC B to communicate with instance A over private IP.

Note
In this scenario, VPC B can either be a VPC in an account that supports EC2-Classic, or an account that supports EC2-VPC only.



The route tables for VPC A contain routes for local VPC traffic, and routes to enable communication to the linked EC2-Classic instance. The custom route table for VPC A contains a route that enables instances in the subnet to communicate with the peer VPC over the VPC peering connection, and a route to route all Internet traffic to the Internet gateway. The custom route table for VPC B contains a route to enable instances in the subnet to communicate with the peer VPC over the VPC peering connection.

Route Table	Destination	Target	Notes
VPC A custom	172.31.0.0/16	Local	Default local route for VPC A.
	192.168.0.0/16	pcx-aaaabbbb	Route manually added for the peering connection between VPC A and VPC B.

Route Table	Destination	Target	Notes
	10.0.0.0/8	Local	Route automatically added to enable ClassicLink communication (added when you linked the instance to VPC A).
	0.0.0.0/0	igw-11aa22bb	Route manually added to route Internet traffic to the Internet gateway.
VPC A main	172.31.0.0/16	Local	Default local route for VPC A.
	10.0.0.0/8	Local	Route automatically added to enable ClassicLink communication (added when you linked the instance to VPC A).
VPC B custom	192.168.0.0/16	Local	Default local route for VPC B.
	172.31.0.0/16	pcx-aaaabbbb	Route manually added for the peering connection between VPC A and VPC B.

The owner of VPC A must modify the VPC peering connection to enable instance A to communicate with VPC B, and update the main route table. The owner of VPC B must modify the VPC peering connection to enable VPC B to communicate with instance A.

> **Note**
> If one VPC is in a region with EC2-Classic and the other VPC is in a region without EC2-Classic, the option to update the ClassicLink settings is disabled in the console for the region without EC2-Classic. Therefore, you must use the AWS CLI instead.

Tasks

- Modifying the VPC Peering Connection for VPC A (p. 54)
- Modifying the VPC Peering Connection for VPC B (p. 55)
- Viewing VPC Peering Connection Options (p. 56)

Modifying the VPC Peering Connection for VPC A

To enable communication from the EC2-Classic instance to VPC B, the AWS account owner of VPC A must modify the VPC peering connection options to enable the local ClassicLink connection to send traffic to instances in the peer VPC.

To modify the VPC peering connection using the console

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.

   The owner of VPC A must sign in to the console.

2. In the navigation pane, choose **Peering Connections**.
3. Select the VPC peering connection, and choose **Actions**, **Edit ClassicLink Settings**.
4. Choose the option to allow local linked EC2-Classic instances to communicate with the peer VPC, and choose **Save**.

To modify the VPC peering connection using the AWS CLI

You can use the modify-vpc-peering-connection-options command. In this case, VPC A was the requester of the VPC peering connection; therefore, modify the requester options as follows:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-aaaabbbb --
requester-peering-connection-options AllowEgressFromLocalClassicLinkToRemoteVpc=true
```

To update the main route table

There are no changes to the route tables for VPC B or to the custom route table for VPC A. The owner of VPC A must manually add a route to the main route table that enables the linked EC2-Classic instance to communicate over the VPC peering connection.

Destination	Target
172.31.0.0/16	Local
10.0.0.0/8	Local
192.168.0.0/16	**pcx-aaaabbbb**

For more information about adding routes, see Adding and Removing Routes from a Route Table in the *Amazon VPC User Guide*.

Modifying the VPC Peering Connection for VPC B

Next, the AWS account owner of VPC B must modify the VPC peering connection options to enable VPC B to send traffic to EC2-Classic instance A.

To modify the VPC peering connection using the console

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.

   The owner of VPC B must sign in to the console.
2. In the navigation pane, choose **Peering Connections**.
3. Select the VPC peering connection, and choose **Actions**, **Edit ClassicLink Settings**.
4. Choose the option to allow local VPC instances to communicate with EC2-Classic instances in the peer VPC, and choose **Save**.

To modify the VPC peering connection using the AWS CLI

VPC B accepted the VPC peering connection; therefore, modify the accepter options as follows:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-aaaabbbb --
accepter-peering-connection-options AllowEgressFromLocalVpcToRemoteClassicLink=true
```

There are no changes to the route tables for VPC A. A new route is automatically added to the route table for VPC B that allows instances in VPC B to communicate with the linked EC2-Classic instance in VPC A.

Route Table	Destination	Target	Notes
VPC A custom	172.31.0.0/16	Local	Default local route for VPC A.
	192.168.0.0/16	pcx-aaaabbbb	Route manually added for the peering connection between VPC A and VPC B.
	10.0.0.0/8	Local	Route automatically added to enable ClassicLink communication (added when you linked the instance to VPC A).
	0.0.0.0/0	igw-11aa22bb	Route manually added to route Internet traffic to the Internet gateway.
VPC A main	172.31.0.0/16	Local	Default local route for VPC A.
	10.0.0.0/8	Local	Route automatically added to enable ClassicLink communication (added when you linked the instance to VPC A).
	192.168.0.0/16	pcx-aaaabbbb	Route manually added to enable the linked EC2-Classic instance to communicate with VPC B over the VPC peering connection.
VPC B custom	192.168.0.0/16	Local	Default local route for VPC B.
	172.31.0.0/16	pcx-aaaabbbb	Route manually added for the peering connection between VPC A and VPC B.
	10.0.0.0/8	Local	Route automatically added to enable ClassicLink communication with the linked instance in VPC A.

Viewing VPC Peering Connection Options

You can view the VPC peering connection options for the accepter VPC and requester VPC using the Amazon VPC console or the AWS CLI.

To view VPC peering connection options using the console

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Peering Connections**.
3. Select the VPC peering connection, and choose **ClassicLink**. Information about the enabled or disabled VPC peering connection options is displayed.

To view VPC peering connection options using the AWS CLI

You can use the describe-vpc-peering-connections command:

```
aws ec2 describe-vpc-peering-connections --vpc-peering-connection-id pcx-aaaabbbb
```

```
{
    "VpcPeeringConnections": [
        {
            "Status": {
                "Message": "Active",
                "Code": "active"
            },
            "Tags": [
                {
                    "Value": "MyPeeringConnection",
                    "Key": "Name"
                }
            ],
            "AccepterVpcInfo": {
                "PeeringOptions": {
                    "AllowEgressFromLocalVpcToRemoteClassicLink": true,
                    "AllowEgressFromLocalClassicLinkToRemoteVpc": false,
                    "AllowDnsResolutionFromRemoteVpc": false
                },
                "OwnerId": "123456789101",
                "VpcId": "vpc-80cb52e4",
                "CidrBlock": "172.31.0.0/16"
            },
            "VpcPeeringConnectionId": "pcx-aaaabbbb",
            "RequesterVpcInfo": {
                "PeeringOptions": {
                    "AllowEgressFromLocalVpcToRemoteClassicLink": false,
                    "AllowEgressFromLocalClassicLinkToRemoteVpc": true,
                    "AllowDnsResolutionFromRemoteVpc": false
                },
                "OwnerId": "111222333444",
                "VpcId": "vpc-f527be91",
                "CidrBlock": "192.168.0.0/16"
            }
        }
    ]
}
```

Unsupported VPC Peering Configurations

The following VPC peering connection configurations are not supported.

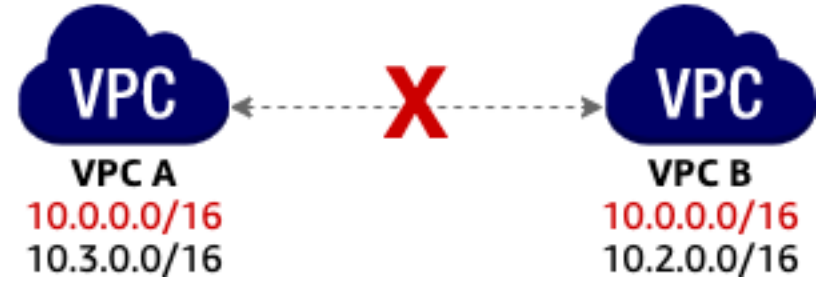
For more information about VPC peering limitations, see VPC Peering Limitations (p. 4).

Invalid Configurations

Overlapping CIDR Blocks

You cannot create a VPC peering connection between VPCs with matching or overlapping IPv4 CIDR blocks.

If the VPCs have multiple IPv4 CIDR blocks, you cannot create a VPC peering connection if any of the CIDR blocks overlap (regardless of whether you intend to use the VPC peering connection for communication between the non-overlapping CIDR blocks only).

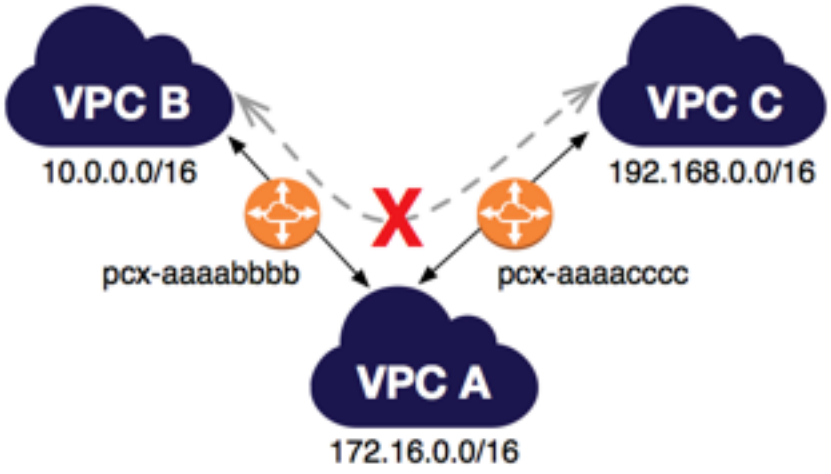


This limitation also applies to VPCs that have non-overlapping IPv6 CIDR blocks. Even if you intend to use the VPC peering connection for IPv6 communication only, you cannot create a VPC peering connection if the VPCs have matching or overlapping IPv4 CIDR blocks.

Transitive Peering

Instead of using VPC peering, you can use an AWS Transit Gateway that acts as a network transit hub, to interconnect your VPCs and on-premises networks. For more information about transit gateways, see What is a Transit Gateway in *Amazon VPC Transit Gateways*.

You have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`), and between VPC A and VPC C (`pcx-aaaacccc`). There is no VPC peering connection between VPC B and VPC C. You cannot route packets directly from VPC B to VPC C through VPC A.



To route packets directly between VPC B and VPC C, you can create a separate VPC peering connection between them (provided they do not have overlapping CIDR blocks). For more information, see Three VPCs Peered Together (p. 23).

Edge to Edge Routing Through a Gateway or Private Connection

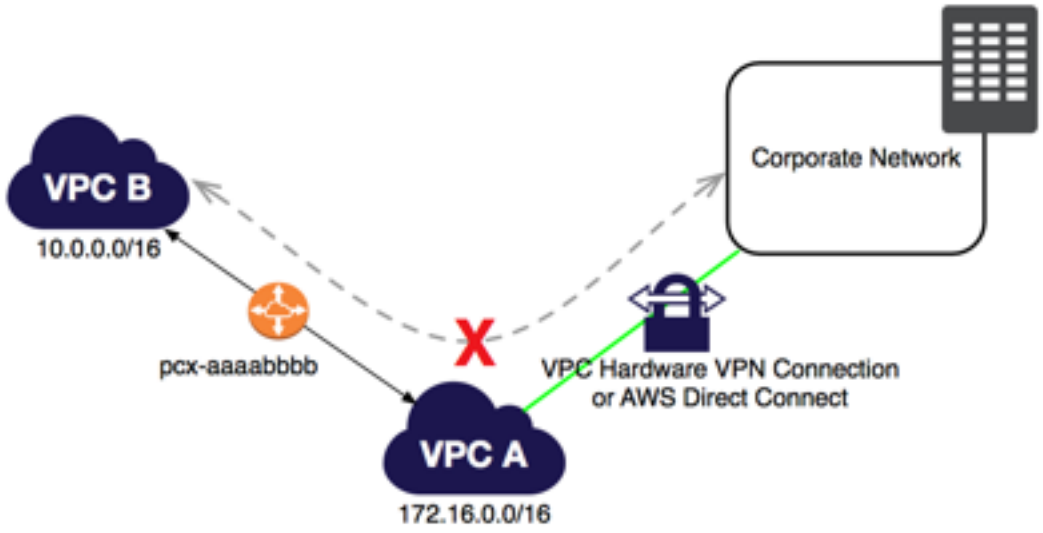
If either VPC in a peering relationship has one of the following connections, you cannot extend the peering relationship to that connection:

- A VPN connection or an AWS Direct Connect connection to a corporate network
- An internet connection through an internet gateway
- An internet connection in a private subnet through a NAT device
- A gateway VPC endpoint to an AWS service; for example, an endpoint to Amazon S3.
- (IPv6) A ClassicLink connection. You can enable IPv4 communication between a linked EC2-Classic instance and instances in a VPC on the other side of a VPC peering connection. However, IPv6 is not supported in EC2-Classic, so you cannot extend this connection for IPv6 communication.

For example, if VPC A and VPC B are peered, and VPC A has any of these connections, then instances in VPC B cannot use the connection to access resources on the other side of the connection. Similarly, resources on the other side of a connection cannot use the connection to access VPC B.

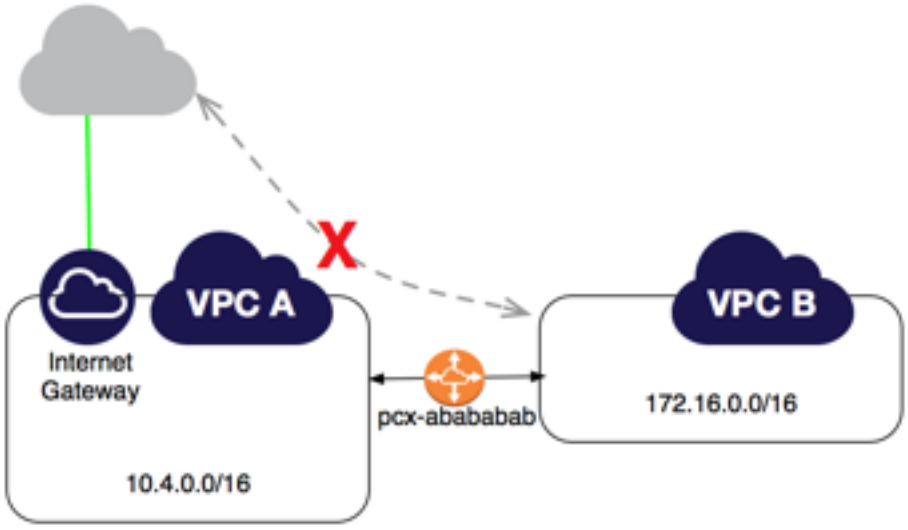
Example: Edge to Edge Routing Through a VPN Connection or an AWS Direct Connect Connection

You have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`). VPC A also has a VPN connection or an AWS Direct Connect connection to a corporate network. Edge to edge routing is not supported; you cannot use VPC A to extend the peering relationship to exist between VPC B and the corporate network. For example, traffic from the corporate network can't directly access VPC B by using the VPN connection or the AWS Direct Connect connection to VPC A.



Example: Edge to Edge Routing Through an Internet Gateway

You have a VPC peering connection between VPC A and VPC B (`pcx-abababab`). VPC A has an internet gateway; VPC B does not. Edge to edge routing is not supported; you cannot use VPC A to extend the peering relationship to exist between VPC B and the internet. For example, traffic from the internet can't directly access VPC B by using the internet gateway connection to VPC A.

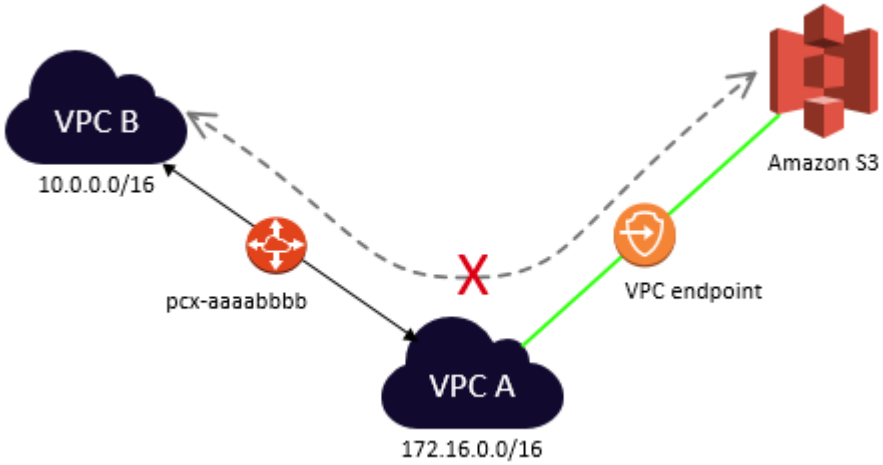


Similarly, if VPC A has a NAT device that provides internet access to instances in private subnets in VPC A, instances in VPC B cannot use the NAT device to access the internet.

Example: Edge to Edge Routing Through a VPC Gateway Endpoint

You have a VPC peering connection between VPC A and VPC B (`pcx-aaaabbbb`). VPC A has a VPC gateway endpoint that connects it to Amazon S3. Edge to edge routing is not supported; you cannot use

VPC A to extend the peering relationship to exist between VPC B and Amazon S3. For example, VPC B can't directly access Amazon S3 using the VPC gateway endpoint connection to VPC A.



Identity and Access Management for VPC Peering

By default, IAM users cannot create or modify VPC peering connections. To allow access to VPC peering resources, you create and attach an IAM policy either to:

- The IAM user or
- The group to which the IAM user belongs.

The following are example IAM policies for working with VPC peering connections.

For a list of Amazon VPC actions and supported resources and conditions keys for each action, see Actions, Resources, and Condition Keys for Amazon EC2 in the *IAM User Guide*.

Contents

Creating a VPC Peering Connection

The following policy allows users to create VPC peering connection requests using only VPCs that are tagged with `Purpose=Peering`. The first statement applies a condition key (`ec2:ResourceTag`) to the VPC resource. Note that the VPC resource for the `CreateVpcPeeringConnection` action is always the requester VPC.

The second statement grants users permissions to create the VPC peering connection resource, and therefore uses the * wildcard in place of a specific resource ID.

```
{
"Version": "2012-10-17",
"Statement":[{
 "Effect":"Allow",
 "Action": "ec2:CreateVpcPeeringConnection",
 "Resource": "arn:aws:ec2:region:account:vpc/*",
  "Condition": {
    "StringEquals": {
     "ec2:ResourceTag/Purpose": "Peering"
    }
   }
  },
  {
  "Effect": "Allow",
  "Action": "ec2:CreateVpcPeeringConnection",
  "Resource": "arn:aws:ec2:region:account:vpc-peering-connection/*"
  }
 ]
```

```
}
```

The following policy allows users in AWS account 333333333333 to create VPC peering connections using any VPC in the `us-east-1` region, but only if the VPC that will be accepting the peering connection is a specific VPC (`vpc-11223344556677889`) in a specific account (777788889999).

```
{
"Version": "2012-10-17",
"Statement": [{
 "Effect":"Allow",
 "Action": "ec2:CreateVpcPeeringConnection",
 "Resource": "arn:aws:ec2:us-east-1:333333333333:vpc/*"
 },
 {
  "Effect": "Allow",
  "Action": "ec2:CreateVpcPeeringConnection",
  "Resource": "arn:aws:ec2:region:333333333333:vpc-peering-connection/*",
    "Condition": {
     "ArnEquals": {
       "ec2:AccepterVpc": "arn:aws:ec2:region:777788889999:vpc/vpc-11223344556677889"
     }
    }
   }
 ]
}
```

Accepting a VPC Peering Connection

The following policy allows users to accept VPC peering connection requests from AWS account 444455556666 only. This helps to prevent users from accepting VPC peering connection requests from unknown accounts. The first statement uses the `ec2:RequesterVpc` condition key to enforce this.

The policy also grants users permissions to accept VPC peering requests only when your VPC has the tag `Purpose=Peering`.

```
{
"Version": "2012-10-17",
"Statement":[{
 "Effect":"Allow",
 "Action": "ec2:AcceptVpcPeeringConnection",
 "Resource": "arn:aws:ec2:region:account:vpc-peering-connection/*",
  "Condition": {
   "ArnEquals": {
    "ec2:RequesterVpc": "arn:aws:ec2:region:444455556666:vpc/*"
   }
  }
 },
 {
 "Effect": "Allow",
 "Action": "ec2:AcceptVpcPeeringConnection",
 "Resource": "arn:aws:ec2:region:account:vpc/*",
  "Condition": {
   "StringEquals": {
    "ec2:ResourceTag/Purpose": "Peering"
    }
   }
  }
 ]
}
```

Deleting a VPC Peering Connection

The following policy allows users in account 444455556666 to delete any VPC peering connection, except those that use the specified VPC `vpc-11223344556677889`, which is in the same account. The policy specifies both the `ec2:AccepterVpc` and `ec2:RequesterVpc` condition keys, as the VPC may have been the requester VPC or the peer VPC in the original VPC peering connection request.

```
{
"Version": "2012-10-17",
"Statement": [{
  "Effect":"Allow",
  "Action": "ec2:DeleteVpcPeeringConnection",
  "Resource": "arn:aws:ec2:region:444455556666:vpc-peering-connection/*",
   "Condition": {
    "ArnNotEquals": {
     "ec2:AccepterVpc": "arn:aws:ec2:region:444455556666:vpc/vpc-11223344556677889",
     "ec2:RequesterVpc": "arn:aws:ec2:region:444455556666:vpc/vpc-11223344556677889"
    }
   }
  }
 ]
}
```

Working Within a Specific Account

The following policy allows users to work with VPC peering connections entirely within a specific account. Users can view, create, accept, reject, and delete VPC peering connections, provided they are all within AWS account 333333333333.

The first statement allows users to view all VPC peering connections. The `Resource` element requires a * wildcard in this case, as this API action (`DescribeVpcPeeringConnections`) currently does not support resource-level permissions.

The second statement allows users to create VPC peering connections, and allows access to all VPCs in account 333333333333 in order to do so.

The third statement uses a * wildcard as part of the `Action` element to allow all VPC peering connection actions. The condition keys ensure that the actions can only be performed on VPC peering connections with VPCs that are part of account 333333333333. For example, a user is not allowed to delete a VPC peering connection if either the accepter or requester VPC is in a different account. A user cannot create a VPC peering connection with a VPC in a different account.

```
{
"Version": "2012-10-17",
"Statement": [{
 "Effect": "Allow",
 "Action": "ec2:DescribeVpcPeeringConnections",
 "Resource": "*"
  },
  {
  "Effect": "Allow",
  "Action": ["ec2:CreateVpcPeeringConnection","ec2:AcceptVpcPeeringConnection"],
  "Resource": "arn:aws:ec2:*:333333333333:vpc/*"
  },
  {
  "Effect": "Allow",
  "Action": "ec2:*VpcPeeringConnection",
  "Resource": "arn:aws:ec2:*:333333333333:vpc-peering-connection/*",
```

```
  "Condition": {
   "ArnEquals": {
    "ec2:AccepterVpc": "arn:aws:ec2:*:333333333333:vpc/*",
    "ec2:RequesterVpc": "arn:aws:ec2:*:333333333333:vpc/*"
    }
   }
  }
 ]
}
```

Managing VPC Peering Connections in the Console

To view VPC peering connections in the Amazon VPC console, users must have permission to use the `ec2:DescribeVpcPeeringConnections` action. To use the **Create Peering Connection** page, users must have permission to use the `ec2:DescribeVpcs` action. This allows them to view and select a VPC. You can apply resource-level permissions to all the `ec2:*PeeringConnection` actions, except `ec2:DescribeVpcPeeringConnections`.

The following policy allows users to view VPC peering connections, and to use the **Create VPC Peering Connection** dialog box to create a VPC peering connection using a specific requester VPC (`vpc-11223344556677889`) only. If users try to create a VPC peering connection with a different requester VPC, the request fails.

```
{
  "Version": "2012-10-17",
  "Statement":[{
   "Effect":"Allow",
   "Action": [
    "ec2:DescribeVpcPeeringConnections", "ec2:DescribeVpcs"
   ],
   "Resource": "*"
  },
  {
   "Effect":"Allow",
   "Action": "ec2:CreateVpcPeeringConnection",
   "Resource": [
    "arn:aws:ec2:*:*:vpc/vpc-11223344556677889",
    "arn:aws:ec2:*:*:vpc-peering-connection/*"
   ]
  }
  ]
}
```

Document History

For more information about the important changes in each release of the *Amazon VPC Peering Guide*, see Document History in the *Amazon VPC User Guide*.