**What Is Amazon VPC?**

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

**Amazon VPC Concepts**

As you get started with Amazon VPC, you should understand the key concepts of this virtual network, and how it is similar to or different from your own networks. This section provides a brief description of the key concepts for Amazon VPC.

Amazon VPC is the networking layer for Amazon EC2. If you're new to Amazon EC2, see What is Amazon EC2? in the*Amazon EC2 User Guide for Linux Instances* to get a brief overview.

**Contents**

* VPCs and Subnets
* Supported Platforms
* Default and Nondefault VPCs
* Accessing the Internet
* Accessing a Corporate or Home Network
* Accessing Services Through AWS PrivateLink

**VPCs and Subnets**

A *virtual private cloud* (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can specify an IP address range for the VPC, add subnets, associate security groups, and configure route tables.

A *subnet* is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. Use a public subnet for resources that must be connected to the internet, and a private subnet for resources that won't be connected to the internet. For more information about public and private subnets, see VPC and Subnet Basics.

To protect the AWS resources in each subnet, you can use multiple layers of security, including security groups and network access control lists (ACL). For more information, see Security.

**Supported Platforms**

The original release of Amazon EC2 supported a single, flat network that's shared with other customers called the *EC2-Classic* platform. Earlier AWS accounts still support this platform, and can launch instances into either EC2-Classic or a VPC. Accounts created after 2013-12-04 support EC2-VPC only. For more information, see Detecting Your Supported Platforms and Whether You Have a Default VPC.

By launching your instances into a VPC instead of EC2-Classic, you gain the ability to:

* Assign static private IPv4 addresses to your instances that persist across starts and stops
* Optionally associate an IPv6 CIDR block to your VPC and assign IPv6 addresses to your instances
* Assign multiple IP addresses to your instances
* Define network interfaces, and attach one or more network interfaces to your instances
* Change security group membership for your instances while they're running
* Control the outbound traffic from your instances (egress filtering) in addition to controlling the inbound traffic to them (ingress filtering)
* Add an additional layer of access control to your instances in the form of network access control lists (ACL)
* Run your instances on single-tenant hardware

**Default and Nondefault VPCs**

If your account supports the EC2-VPC platform only, it comes with a *default VPC* that has a *default subnet* in each Availability Zone. A default VPC has the benefits of the advanced features provided by EC2-VPC, and is ready for you to use. If you have a default VPC and don't specify a subnet when you launch an instance, the instance is launched into your default VPC. You can launch instances into your default VPC without needing to know anything about Amazon VPC.

Regardless of which platforms your account supports, you can create your own VPC, and configure it as you need. This is known as a *nondefault VPC*. Subnets that you create in your nondefault VPC and additional subnets that you create in your default VPC are called *nondefault subnets*.

**Accessing the Internet**

You control how the instances that you launch into a VPC access resources outside the VPC.

Your default VPC includes an internet gateway, and each default subnet is a public subnet. Each instance that you launch into a default subnet has a private IPv4 address and a public IPv4 address. These instances can communicate with the internet through the internet gateway. An internet gateway enables your instances to connect to the internet through the Amazon EC2 network edge.


      Using a default VPC
     

By default, each instance that you launch into a nondefault subnet has a private IPv4 address, but no public IPv4 address, unless you specifically assign one at launch, or you modify the subnet's public IP address attribute. These instances can communicate with each other, but can't access the internet.


      Using a nondefault VPC
     

You can enable internet access for an instance launched into a nondefault subnet by attaching an internet gateway to its VPC (if its VPC is not a default VPC) and associating an Elastic IP address with the instance.


     Using an internet gateway
    

Alternatively, to allow an instance in your VPC to initiate outbound connections to the internet but prevent unsolicited inbound connections from the internet, you can use a network address translation (NAT) device for IPv4 traffic. NAT maps multiple private IPv4 addresses to a single public IPv4 address. A NAT device has an Elastic IP address and is connected to the internet through an internet gateway. You can connect an instance in a private subnet to the internet through the NAT device, which routes traffic from the instance to the internet gateway, and routes any responses to the instance.

For more information, see NAT.

You can optionally associate an Amazon-provided IPv6 CIDR block with your VPC and assign IPv6 addresses to your instances. Instances can connect to the internet over IPv6 through an internet gateway. Alternatively, instances can initiate outbound connections to the internet over IPv6 using an egress-only internet gateway. For more information, see Egress-Only Internet Gateways. IPv6 traffic is separate from IPv4 traffic; your route tables must include separate routes for IPv6 traffic.

**Accessing a Corporate or Home Network**

You can optionally connect your VPC to your own corporate data center using an IPsec AWS managed VPN connection, making the AWS Cloud an extension of your data center.

A VPN connection consists of a virtual private gateway attached to your VPC and a customer gateway located in your data center. A virtual private gateway is the VPN concentrator on the Amazon side of the VPN connection. A customer gateway is a physical device or software appliance on your side of the VPN connection.


      Using a virtual private gateway
     

For more information, see AWS Managed VPN Connections.

**Accessing Services Through AWS PrivateLink**

AWS PrivateLink is a highly available, scalable technology that enables you to privately connect your VPC to supported AWS services, services hosted by other AWS accounts (VPC endpoint services), and supported AWS Marketplace partner services. You do not require an internet gateway, NAT device, public IP address, AWS Direct Connect connection, or VPN connection to communicate with the service. Traffic between your VPC and the service does not leave the Amazon network.

To use AWS PrivateLink, create an interface VPC endpoint for a service in your VPC. This creates an elastic network interface in your subnet with a private IP address that serves as an entry point for traffic destined to the service. For more information, see VPC Endpoints.


     Using an interface endpoint to access an AWS service
    

You can create your own AWS PrivateLink-powered service (endpoint service) and enable other AWS customers to access your service. For more information, see VPC Endpoint Services (AWS PrivateLink).

**How to Get Started with Amazon VPC**

To get a hands-on introduction to Amazon VPC, complete the exercise Getting Started. The exercise guides you through the steps to create a nondefault VPC with a public subnet, and to launch an instance into your subnet.

If you have a default VPC, and you want to get started launching instances into your VPC without performing any additional configuration on your VPC, see Launching an EC2 Instance into Your Default VPC.

To learn about the basic scenarios for Amazon VPC, see Scenarios and Examples. You can configure your VPC and subnets in other ways to suit your needs.

The following table lists related resources that you might find useful as you work with this service.

|  |  |
| --- | --- |
| **Resource** | **Description** |
| Amazon Virtual Private Cloud Connectivity Options | A whitepaper that provides an overview of the options for network connectivity. |
| Amazon VPC forum | A community-based forum for discussing technical questions related to Amazon VPC. |
| AWS Developer Resources | A central starting point to find documentation, code samples, release notes, and other information to help you create innovative applications with AWS. |
| AWS Support Center | The home page for AWS Support. |
| Contact Us | A central contact point for inquiries concerning AWS billing, accounts, and events. |

**Using Amazon VPC with Other AWS Services**

Amazon VPC integrates with many other AWS services; furthermore, some services require a VPC in your account to carry out certain functions. Below are examples of services that use Amazon VPC.

|  |  |
| --- | --- |
| **Service** | **Relevant Topic** |
| AWS Data Pipeline | Launching Resources for Your Pipeline into a VPC |
| Amazon EC2 | Amazon EC2 and Amazon VPC |
| Auto Scaling | Auto Scaling and Amazon VPC |
| Elastic Beanstalk | Using AWS Elastic Beanstalk with Amazon VPC |
| Elastic Load Balancing | Setting Up Elastic Load Balancing |
| Amazon ElastiCache | Using ElastiCache with Amazon VPC |
| Amazon EMR | Select a Subnet for the Cluster |
| AWS OpsWorks | Running a Stack in a VPC |
| Amazon RDS | Amazon RDS and Amazon VPC |
| Amazon Redshift | Managing Clusters in a VPC |
| Route 53 | Working with Private Hosted Zones |
| Amazon WorkSpaces | Create and Configure Your VPC |

To get a detailed view of the VPCs, subnets, and other VPC resources in your account and their relation to each other, you can use the AWS Config service. For more information, see the *AWS Config Developer Guide*.

**Accessing Amazon VPC**

Amazon VPC provides a web-based user interface, the Amazon VPC console. If you've signed up for an AWS account, you can access the Amazon VPC console by signing into the AWS Management Console and choosing **VPC**.

If you prefer to use a command line interface, you have the following options:

AWS Command Line Interface (AWS CLI)

Provides commands for a broad set of AWS services, and is supported on Windows, macOS, and Linux/Unix. To get started, see AWS Command Line Interface User Guide. For more information about the commands for Amazon VPC, see ec2.

AWS Tools for Windows PowerShell

Provides commands for a broad set of AWS services for those who script in the PowerShell environment. To get started, see AWS Tools for Windows PowerShell User Guide.

Amazon VPC provides a Query API. These requests are HTTP or HTTPS requests that use the HTTP verbs GET or POST and a Query parameter named Action. For more information, see Actions in the *Amazon EC2 API Reference*.

To build applications using language-specific APIs instead of submitting a request over HTTP or HTTPS, AWS provides libraries, sample code, tutorials, and other resources for software developers. These libraries provide basic functions that automatically take care of tasks such as cryptographically signing your requests, retrying requests, and handling error responses. For more information, see AWS SDKs and Tools.

**Pricing for Amazon VPC**

There's no additional charge for using Amazon VPC. You pay the standard rates for the instances and other Amazon EC2 features that you use. There are charges for using an AWS managed VPN connection and using a NAT gateway. For more information, see Amazon VPC Pricing and Amazon EC2 Pricing.

**Amazon VPC Limits**

There are limits to the number of Amazon VPC components that you can provision. You can request an increase for some of these limits. For more information, see Amazon VPC Limits.

**PCI DSS Compliance**

Amazon VPC supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see PCI DSS Level 1.

# Getting Started with IPv6 for Amazon VPC

In this exercise, you create a VPC with an IPv6 CIDR block, a subnet with an IPv6 CIDR block, and launch a public-facing instance into your subnet. Your instance will be able to communicate with the Internet over IPv6, and you'll be able to access your instance over IPv6 from your local computer using SSH (if it's a Linux instance) or Remote Desktop (if it's a Windows instance). In your real world environment, you can use this scenario to create a public-facing web server, for example, to host a blog.

To complete this exercise, do the following:

* Create a nondefault VPC with an IPv6 CIDR block and a single public subnet. Subnets enable you to group instances based on your security and operational needs. A public subnet is a subnet that has access to the Internet through an Internet gateway.
* Create a security group for your instance that allows traffic only through specific ports.
* Launch an Amazon EC2 instance into your subnet, and associate an IPv6 address with your instance during launch. An IPv6 address is globally unique, and allows your instance to communicate with the Internet.

**Scenario 1: VPC with a Single Public Subnet**

The configuration for this scenario includes a virtual private cloud (VPC) with a single public subnet, and an Internet gateway to enable communication over the Internet. We recommend this configuration if you need to run a single-tier, public-facing web application, such as a blog or a simple website.

This topic assumes that you'll use the VPC wizard in the Amazon VPC console to create the VPC.

This scenario can also be optionally configured for IPv6—you can use the VPC wizard to create a VPC and subnet with associated IPv6 CIDR blocks. Instances launched into the public subnet can receive IPv6 addresses, and communicate using IPv6. For more information about IPv4 and IPv6 addressing, see IP Addressing in Your VPC.

**Topics**

* Overview
* Routing
* Security
* Implementing Scenario 1

**Overview**

The following diagram shows the key components of the configuration for this scenario.


     Diagram for scenario 1: VPC with a public subnet
    

**Note**

If you completed the exercise Getting Started, then you've already implemented this scenario using the VPC wizard in the Amazon VPC console.

The configuration for this scenario includes the following:

* A virtual private cloud (VPC) with a size /16 IPv4 CIDR block (example: 10.0.0.0/16). This provides 65,536 private IPv4 addresses.
* A subnet with a size /24 IPv4 CIDR block (example: 10.0.0.0/24). This provides 256 private IPv4 addresses.
* An Internet gateway. This connects the VPC to the Internet and to other AWS services.
* An instance with a private IPv4 address in the subnet range (example: 10.0.0.6), which enables the instance to communicate with other instances in the VPC, and an Elastic IPv4 address (example: 198.51.100.2), which is a public IPv4 address that enables the instance to be reached from the Internet.
* A custom route table associated with the subnet. The route table entries enable instances in the subnet to use IPv4 to communicate with other instances in the VPC, and to communicate directly over the Internet. A subnet that's associated with a route table that has a route to an Internet gateway is known as a *public subnet*.

For more information about subnets, see VPCs and Subnets. For more information about Internet gateways, see Internet Gateways.

**Overview for IPv6**

You can optionally enable IPv6 for this scenario. In addition to the components listed above, the configuration includes the following:

* A size /56 IPv6 CIDR block associated with the VPC (example: 2001:db8:1234:1a00::/56). Amazon automatically assigns the CIDR; you cannot choose the range yourself.
* A size /64 IPv6 CIDR block associated with the public subnet (example: 2001:db8:1234:1a00::/64). You can choose the range for your subnet from the range allocated to the VPC. You cannot choose the size of the subnet IPv6 CIDR block.
* An IPv6 address assigned to the instance from the subnet range (example: 2001:db8:1234:1a00::123).
* Route table entries in the custom route table that enable instances in the VPC to use IPv6 to communicate with each other, and directly over the Internet.


      IPv6-enabled VPC with a public subnet
     

**Routing**

Your VPC has an implied router (shown in the configuration diagram above). In this scenario, the VPC wizard creates a custom route table that routes all traffic destined for an address outside the VPC to the Internet gateway, and associates this route table with the subnet.

The following table shows the route table for the example in the configuration diagram above. The first entry is the default entry for local IPv4 routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second entry routes all other IPv4 subnet traffic to the Internet gateway (for example, igw-1a2b3c4d).

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 0.0.0.0/0 | *igw-id* |

**Routing for IPv6**

If you associate an IPv6 CIDR block with your VPC and subnet, your route table must include separate routes for IPv6 traffic. The following table shows the custom route table for this scenario if you choose to enable IPv6 communication in your VPC. The second entry is the default route that's automatically added for local routing in the VPC over IPv6. The fourth entry routes all other IPv6 subnet traffic to the Internet gateway.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 2001:db8:1234:1a00::/56 | local |
| 0.0.0.0/0 | *igw-id* |
| ::/0 | *igw-id* |

**Security**

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Security groups control inbound and outbound traffic for your instances, and network ACLs control inbound and outbound traffic for your subnets. In most cases, security groups can meet your needs; however, you can also use network ACLs if you want an additional layer of security for your VPC. For more information, see Security.

For this scenario, you use a security group but not a network ACL. If you'd like to use a network ACL, see Recommended Rules for Scenario 1.

Your VPC comes with a default security group. An instance that's launched into the VPC is automatically associated with the default security group if you don't specify a different security group during launch. You can add rules to the default security group, but the rules may not be suitable for other instances that you launch into the VPC. Instead, we recommend that you create a custom security group for your web server.

For this scenario, create a security group named WebServerSG. When you create a security group, it has a single outbound rule that allows all traffic to leave the instances. You must modify the rules to enable inbound traffic and restrict the outbound traffic as needed. You specify this security group when you launch instances into the VPC.

The following are the inbound and outbound rules for IPv4 traffic for the WebServerSG security group.

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| 0.0.0.0/0 | TCP | 80 | Allow inbound HTTP access to the web servers from any IPv4 address. |
| 0.0.0.0/0 | TCP | 443 | Allow inbound HTTPS access to the web servers from any IPv4 address |
| Public IPv4 address range of your network | TCP | 22 | (Linux instances) Allow inbound SSH access from your network over IPv4. You can get the public IPv4 address of your local computer using a service such as http://checkip.amazonaws.com or https://checkip.amazonaws.com. If you are connecting through an ISP or from behind your firewall without a static IP address, you need to find out the range of IP addresses used by client computers. |
| Public IPv4 address range of your network | TCP | 3389 | (Windows instances) Allow inbound RDP access from your network over IPv4. |
| The security group ID (sg-xxxxxxxx) | All | All | (Optional) Allow inbound traffic from other instances associated with this security group. This rule is automatically added to the default security group for the VPC; for any custom security group you create, you must manually add the rule to allow this type of communication. |
| **Outbound** (Optional) | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| 0.0.0.0/0 | All | All | Default rule to allow all outbound access to any IPv4 address. If you want your web server to initiate outbound traffic, for example, to get software updates, you can leave the default outbound rule. Otherwise, you can remove this rule. |

**Security for IPv6**

If you associate an IPv6 CIDR block with your VPC and subnet, you must add separate rules to your security group to control inbound and outbound IPv6 traffic for your web server instance. In this scenario, the web server will be able to receive all Internet traffic over IPv6, and SSH or RDP traffic from your local network over IPv6.

The following are the IPv6-specific rules for the WebServerSG security group (which are in addition to the rules listed above).

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | TCP | 80 | Allow inbound HTTP access to the web servers from any IPv6 address. |
| ::/0 | TCP | 443 | Allow inbound HTTPS access to the web servers from any IPv6 address. |
| IPv6 address range of your network | TCP | 22 | (Linux instances) Allow inbound SSH access over IPv6 from your network. |
| IPv6 address range of your network | TCP | 3389 | (Windows instances) Allow inbound RDP access over IPv6 from your network |
| **Outbound** (Optional) | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | All | All | Default rule to allow all outbound access to any IPv6 address. If you want your web server to initiate outbound traffic, for example, to get software updates, you can leave the default outbound rule. Otherwise, you can remove this rule. |

**Implementing Scenario 1**

To implement scenario 1, create a VPC using the VPC wizard, create and configure the WebServerSG security group, and then launch an instance into your VPC.

These procedures include optional steps for enabling and configuring IPv6 communication for your VPC. You do not have to perform these steps if you do not want to use IPv6 in your VPC.

**To create a VPC**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the dashboard, choose **Start VPC Wizard**.
3. Select the first option, **VPC with a Single Public Subnet**, and then choose **Select**.
4. For **VPC name** and **Subnet name**, you can name your VPC and subnet to help you to identify them later in the console. You can specify your own IPv4 CIDR block range for the VPC and subnet, or you can leave the default values (10.0.0.0/16 and 10.0.0.0/24 respectively).
5. (Optional, IPv6-only) For **IPv6 CIDR block**, choose **Amazon-provided IPv6 CIDR block**. For **Public subnet's IPv6 CIDR**, choose **Specify a custom IPv6 CIDR** and specify the hexadecimal pair value for your subnet, or leave the default value (00).
6. You can leave the rest of the default settings, and choose **Create VPC**.

**To create the WebServerSG security group**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Security Groups**.
3. Choose **Create Security Group**.
4. Provide a name and description for the security group. In this topic, the name WebServerSG is used as an example. Select the ID of your VPC from the **VPC** menu, and then choose **Yes, Create**.
5. Select the WebServerSG security group that you just created. The details pane include a tab for information about the security group, plus tabs for working with its inbound rules and outbound rules.
6. On the **Inbound Rules** tab, choose **Edit**, and then do the following:
   * Select **HTTP** from the **Type** list, and enter 0.0.0.0/0 in the **Source** field.
   * Choose **Add another rule**, then select **HTTPS** from the **Type** list, and enter 0.0.0.0/0 in the **Source** field.
   * Choose **Add another rule**, then select **SSH** (for Linux) or **RDP** (for Windows) from the **Type** list. Enter your network's public IP address range in the **Source** field. (If you don't know this address range, you can use 0.0.0.0/0 for testing purposes; in production, you authorize only a specific IP address or range of addresses to access your instance.)
   * (Optional) Choose **Add another rule**, then select **ALL traffic** from the **Type** list. In the **Source** field, enter the ID of the WebServerSG security group.
   * (Optional, IPv6-only) Choose **Add another rule**, select **HTTP** from the **Type** list, and enter ::/0 in the **Source** field.
   * (Optional, IPv6-only) Choose **Add another rule**, select **HTTPS** from the **Type** list, and enter ::/0 in the **Source** field.
   * (Optional, IPv6-only) Choose **Add another rule**, select **SSH** (for Linux) or **RDP** (for Windows) from the **Type**list. Enter your network's IPv6 address range in the **Source** field. (If you don't know this address range, you can use ::/0 for testing purposes; in production, you authorize only a specific IPv6 address or range of addresses to access your instance.)
7. Choose **Save**.
8. (Optional) On the **Outbound Rules** tab, choose **Edit**. Locate the default rule that enables all outbound traffic, choose **Remove**, and then choose **Save**.

**To launch an instance into the VPC**

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the dashboard, choose **Launch Instance**.
3. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then choose **Next: Configure Instance Details**.

**Note**

If you intend to use your instance for IPv6 communication, you must choose a supported instance type; for example, T2. For more information, see Amazon EC2 Instance Types.

1. On the **Configure Instance Details** page, select the VPC that you created in step 1 from the **Network** list, and then specify a subnet.
2. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IPv4 address. To be able to connect to your instance, you can assign a public IPv4 address now, or allocate an Elastic IP address and assign it to your instance after it's launched. To assign a public IPv4 address now, ensure that you select **Enable**from the **Auto-assign Public IP** list.

**Note**

You can only use the auto-assign public IP feature for a single, new network interface with the device index of eth0. For more information, see Assigning a Public IPv4 Address During Instance Launch.

1. (Optional, IPv6-only) You can auto-assign an IPv6 address to your instance from the subnet range. For **Auto-assign IPv6 IP**, choose **Enable**.
2. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, select the **Select an existing security group** option, and select the **WebServerSG** security group that you created in step 2. Choose **Review and Launch**.
3. Review the settings that you've chosen. Make any changes that you need, and then choose **Launch** to choose a key pair and launch your instance.
4. If you did not assign a public IPv4 address to your instance in step 5, you will not be able to connect to it over IPv4. Assign an Elastic IP address to the instance:
   1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
   2. In the navigation pane, choose **Elastic IPs**.
   3. Choose **Allocate new address**.
   4. Choose **Allocate**.

**Note**

If your account supports EC2-Classic, first choose **VPC**.

* 1. Select the Elastic IP address from the list, choose **Actions**, and then choose **Associate address**.
  2. Select the instance to associate the address with, and then choose **Associate**.

**Scenario 2: VPC with Public and Private Subnets (NAT)**

The configuration for this scenario includes a virtual private cloud (VPC) with a public subnet and a private subnet. We recommend this scenario if you want to run a public-facing web application, while maintaining back-end servers that aren't publicly accessible. A common example is a multi-tier website, with the web servers in a public subnet and the database servers in a private subnet. You can set up security and routing so that the web servers can communicate with the database servers.

The instances in the public subnet can send outbound traffic directly to the Internet, whereas the instances in the private subnet can't. Instead, the instances in the private subnet can access the Internet by using a network address translation (NAT) gateway that resides in the public subnet. The database servers can connect to the Internet for software updates using the NAT gateway, but the Internet cannot establish connections to the database servers.

**Note**

You can also use the VPC wizard to configure a VPC with a NAT instance; however, we recommend that you use a NAT gateway. For more information, see NAT Gateways.

This topic assumes that you'll use the VPC wizard in the Amazon VPC console to create the VPC and NAT gateway.

This scenario can also be optionally configured for IPv6—you can use the VPC wizard to create a VPC and subnets with associated IPv6 CIDR blocks. Instances launched into the subnets can receive IPv6 addresses, and communicate using IPv6. Instances in the private subnet can use an egress-only Internet gateway to connect to the Internet over IPv6, but the Internet cannot establish connections to the private instances over IPv6. For more information about IPv4 and IPv6 addressing, see IP Addressing in Your VPC.

**Topics**

* Overview
* Routing
* Security
* Implementing Scenario 2
* Implementing Scenario 2 with a NAT Instance

**Overview**

The following diagram shows the key components of the configuration for this scenario.


     Diagram for scenario 2: VPC with public and private subnets
    

The configuration for this scenario includes the following:

* A VPC with a size /16 IPv4 CIDR block (example: 10.0.0.0/16). This provides 65,536 private IPv4 addresses.
* A public subnet with a size /24 IPv4 CIDR block (example: 10.0.0.0/24). This provides 256 private IPv4 addresses. A public subnet is a subnet that's associated with a route table that has a route to an Internet gateway.
* A private subnet with a size /24 IPv4 CIDR block (example: 10.0.1.0/24). This provides 256 private IPv4 addresses.
* An Internet gateway. This connects the VPC to the Internet and to other AWS services.
* Instances with private IPv4 addresses in the subnet range (examples: 10.0.0.5, 10.0.1.5). This enables them to communicate with each other and other instances in the VPC.
* Instances in the public subnet with Elastic IPv4 addresses (example: 198.51.100.1), which are public IPv4 addresses that enable them to be reached from the Internet. The instances can have public IP addresses assigned at launch instead of Elastic IP addresses. Instances in the private subnet are back-end servers that don't need to accept incoming traffic from the Internet and therefore do not have public IP addresses; however, they can send requests to the Internet using the NAT gateway (see the next bullet).
* A NAT gateway with its own Elastic IPv4 address. Instances in the private subnet can send requests to the Internet through the NAT gateway over IPv4 (for example, for software updates).
* A custom route table associated with the public subnet. This route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC over IPv4, and an entry that enables instances in the subnet to communicate directly with the Internet over IPv4.
* The main route table associated with the private subnet. The route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC over IPv4, and an entry that enables instances in the subnet to communicate with the Internet through the NAT gateway over IPv4.

For more information about subnets, see VPCs and Subnets. For more information about Internet gateways, seeInternet Gateways. For more information about NAT gateways, see NAT Gateways.

**Overview for IPv6**

You can optionally enable IPv6 for this scenario. In addition to the components listed above, the configuration includes the following:

* A size /56 IPv6 CIDR block associated with the VPC (example: 2001:db8:1234:1a00::/56). Amazon automatically assigns the CIDR; you cannot choose the range yourself.
* A size /64 IPv6 CIDR block associated with the public subnet (example: 2001:db8:1234:1a00::/64). You can choose the range for your subnet from the range allocated to the VPC. You cannot choose the size of the VPC IPv6 CIDR block.
* A size /64 IPv6 CIDR block associated with the private subnet (example: 2001:db8:1234:1a01::/64). You can choose the range for your subnet from the range allocated to the VPC. You cannot choose the size of the subnet IPv6 CIDR block.
* IPv6 addresses assigned to the instances from the subnet range (example: 2001:db8:1234:1a00::1a).
* An egress-only Internet gateway. This enables instances in the private subnet to send requests to the Internet over IPv6 (for example, for software updates). An egress-only Internet gateway is necessary if you want instances in the private subnet to be able to initiate communication with the Internet over IPv6. For more information, see Egress-Only Internet Gateways.
* Route table entries in the custom route table that enable instances in the public subnet to use IPv6 to communicate with each other, and directly over the Internet.
* Route table entries in the main route table that enable instances in the private subnet to use IPv6 to communicate with each other, and to communicate with the Internet through an egress-only Internet gateway.


      IPv6-enabled VPC with a public and private subnet
     

**Routing**

In this scenario, the VPC wizard updates the main route table used with the private subnet, and creates a custom route table and associates it with the public subnet.

In this scenario, all traffic from each subnet that is bound for AWS (for example, to the Amazon EC2 or Amazon S3 endpoints) goes over the Internet gateway. The database servers in the private subnet can't receive traffic from the Internet directly because they don't have Elastic IP addresses. However, the database servers can send and receive Internet traffic through the NAT device in the public subnet.

Any additional subnets that you create use the main route table by default, which means that they are private subnets by default. If you want to make a subnet public, you can always change the route table that it's associated with.

The following tables describe the route tables for this scenario.

**Main Route Table**

The first entry is the default entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second entry sends all other subnet traffic to the NAT gateway (for example, nat-12345678901234567).

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 0.0.0.0/0 | *nat-gateway-id* |

**Custom Route Table**

The first entry is the default entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second entry routes all other subnet traffic to the Internet over the Internet gateway (for example, igw-1a2b3d4d).

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 0.0.0.0/0 | *igw-id* |

**Routing for IPv6**

If you associate an IPv6 CIDR block with your VPC and subnets, your route tables must include separate routes for IPv6 traffic. The following tables show the route tables for this scenario if you choose to enable IPv6 communication in your VPC.

**Main Route Table**

The second entry is the default route that's automatically added for local routing in the VPC over IPv6. The fourth entry routes all other IPv6 subnet traffic to the egress-only Internet gateway.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 2001:db8:1234:1a00::/56 | local |
| 0.0.0.0/0 | *nat-gateway-id* |
| ::/0 | *egress-only-igw-id* |

**Custom Route Table**

The second entry is the default route that's automatically added for local routing in the VPC over IPv6. The fourth entry routes all other IPv6 subnet traffic to the Internet gateway.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 2001:db8:1234:1a00::/56 | local |
| 0.0.0.0/0 | *igw-id* |
| ::/0 | *igw-id* |

**Security**

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Security groups control inbound and outbound traffic for your instances, and network ACLs control inbound and outbound traffic for your subnets. In most cases, security groups can meet your needs; however, you can also use network ACLs if you want an additional layer of security for your VPC. For more information, see Security.

For scenario 2, you'll use security groups but not network ACLs. If you'd like to use a network ACL, see Recommended Rules for Scenario 2.

Your VPC comes with a default security group. An instance that's launched into the VPC is automatically associated with the default security group if you don't specify a different security group during launch. For this scenario, we recommend that you create the following security groups instead of using the default security group:

* **WebServerSG**: Specify this security group when you launch the web servers in the public subnet.
* **DBServerSG**: Specify this security group when you launch the database servers in the private subnet.

The instances assigned to a security group can be in different subnets. However, in this scenario, each security group corresponds to the type of role an instance plays, and each role requires the instance to be in a particular subnet. Therefore, in this scenario, all instances assigned to a security group are in the same subnet.

The following table describes the recommended rules for the WebServerSG security group, which allow the web servers to receive Internet traffic, as well as SSH and RDP traffic from your network. The web servers can also initiate read and write requests to the database servers in the private subnet, and send traffic to the Internet; for example, to get software updates. Because the web server doesn't initiate any other outbound communication, the default outbound rule is removed.

**Note**

These recommendations include both SSH and RDP access, and both Microsoft SQL Server and MySQL access. For your situation, you might only need rules for Linux (SSH and MySQL) or Windows (RDP and Microsoft SQL Server).

**WebServerSG: Recommended Rules**

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| 0.0.0.0/0 | TCP | 80 | Allow inbound HTTP access to the web servers from any IPv4 address. |
| 0.0.0.0/0 | TCP | 443 | Allow inbound HTTPS access to the web servers from any IPv4 address. |
| Your home network's public IPv4 address range | TCP | 22 | Allow inbound SSH access to Linux instances from your home network (over the Internet gateway). You can get the public IPv4 address of your local computer using a service such as http://checkip.amazonaws.com orhttps://checkip.amazonaws.com. If you are connecting through an ISP or from behind your firewall without a static IP address, you need to find out the range of IP addresses used by client computers. |
| Your home network's public IPv4 address range | TCP | 3389 | Allow inbound RDP access to Windows instances from your home network (over the Internet gateway). |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| The ID of your DBServerSG security group | TCP | 1433 | Allow outbound Microsoft SQL Server access to the database servers assigned to the DBServerSG security group. |
| The ID of your DBServerSG security group | TCP | 3306 | Allow outbound MySQL access to the database servers assigned to the DBServerSG security group. |
| 0.0.0.0/0 | TCP | 80 | Allow outbound HTTP access to any IPv4 address. |
| 0.0.0.0/0 | TCP | 443 | Allow outbound HTTPS access to any IPv4 address. |

The following table describes the recommended rules for the DBServerSG security group, which allow read or write database requests from the web servers. The database servers can also initiate traffic bound for the Internet (the route table sends that traffic to the NAT gateway, which then forwards it to the Internet over the Internet gateway).

**DBServerSG: Recommended Rules**

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| The ID of your WebServerSG security group | TCP | 1433 | Allow inbound Microsoft SQL Server access from the web servers associated with the WebServerSG security group. |
| The ID of your WebServerSG security group | TCP | 3306 | Allow inbound MySQL Server access from the web servers associated with the WebServerSG security group. |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| 0.0.0.0/0 | TCP | 80 | Allow outbound HTTP access to the Internet over IPv4 (for example, for software updates). |
| 0.0.0.0/0 | TCP | 443 | Allow outbound HTTPS access to the Internet over IPv4 (for example, for software updates). |

(Optional) The default security group for a VPC has rules that automatically allow assigned instances to communicate with each other. To allow that type of communication for a custom security group, you must add the following rules:

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| The ID of the security group | All | All | Allow inbound traffic from other instances assigned to this security group. |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| The ID of the security group | All | All | Allow outbound traffic to other instances assigned to this security group. |

(Optional) If you launch a bastion host in your public subnet to use as a proxy for SSH or RDP traffic from your home network to your private subnet, add a rule to the DBServerSG security group that allows inbound SSH or RDP traffic from the bastion instance or its associated security group.

**Security for IPv6**

If you associate an IPv6 CIDR block with your VPC and subnets, you must add separate rules to your WebServerSG and DBServerSG security groups to control inbound and outbound IPv6 traffic for your instances. In this scenario, the web servers will be able to receive all Internet traffic over IPv6, and SSH or RDP traffic from your local network over IPv6. They can also initiate outbound IPv6 traffic to the Internet. The database servers can initiate outbound IPv6 traffic to the Internet.

The following are the IPv6-specific rules for the WebServerSG security group (which are in addition to the rules listed above).

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | TCP | 80 | Allow inbound HTTP access to the web servers from any IPv6 address. |
| ::/0 | TCP | 443 | Allow inbound HTTPS access to the web servers from any IPv6 address. |
| IPv6 address range of your network | TCP | 22 | (Linux instances) Allow inbound SSH access over IPv6 from your network. |
| IPv6 address range of your network | TCP | 3389 | (Windows instances) Allow inbound RDP access over IPv6 from your network |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | TCP | HTTP | Allow outbound HTTP access to any IPv6 address. |
| ::/0 | TCP | HTTPS | Allow outbound HTTPS access to any IPv6 address. |

The following are the IPv6-specific rules for the DBServerSG security group (which are in addition to the rules listed above).

|  |  |  |  |
| --- | --- | --- | --- |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | TCP | 80 | Allow outbound HTTP access to any IPv6 address. |
| ::/0 | TCP | 443 | Allow outbound HTTPS access to any IPv6 address. |

**Implementing Scenario 2**

You can use the VPC wizard to create the VPC, subnets, NAT gateway, and optionally, an egress-only Internet gateway. You must specify an Elastic IP address for your NAT gateway; if you don't have one, you must first allocate one to your account. If you want to use an existing Elastic IP address, ensure that it's not currently associated with another instance or network interface. The NAT gateway is automatically created in the public subnet of your VPC.

These procedures include optional steps for enabling and configuring IPv6 communication for your VPC. You do not have to perform these steps if you do not want to use IPv6 in your VPC.

**(Optional) To allocate an Elastic IP address for the NAT gateway (IPv4)**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Elastic IPs**.
3. Choose **Allocate new address**.
4. Choose **Allocate**.

**Note**

If your account supports EC2-Classic, first choose **VPC**.

**To create a VPC**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. On the VPC dashboard, choose **Start VPC Wizard**.
3. Choose the second option, **VPC with Public and Private Subnets**, and **Select**.
4. For **VPC name**, **Public subnet name** and **Private subnet name**, you can name your VPC and subnets to help you identify them later in the console. You can specify your own IPv4 CIDR block range for the VPC and subnets, or you can leave the default values.
5. (Optional, IPv6-only) For **IPv6 CIDR block**, choose **Amazon-provided IPv6 CIDR block**. For **Public subnet's IPv6 CIDR**, choose **Specify a custom IPv6 CIDR** and specify the hexadecimal pair value for your subnet, or leave the default value. For **Private subnet's IPv6 CIDR**, choose **Specify a custom IPv6 CIDR**. Specify the hexadecimal pair value for the IPv6 subnet or leave the default value.
6. In the **Specify the details of your NAT gateway** section, specify the allocation ID for an Elastic IP address in your account.
7. You can leave the rest of the default values on the page, and choose **Create VPC**.

Because the WebServerSG and DBServerSG security groups reference each other, create all the security groups required for this scenario before you add rules to them.

**To create the WebServerSG and DBServerSG security groups**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Security Groups**, **Create Security Group**.
3. Provide a name and description for the security group. In this topic, the name WebServerSG is used as an example. For **VPC**, select the ID of the VPC you created and choose **Yes, Create**.
4. Choose **Create Security Group** again.
5. Provide a name and description for the security group. In this topic, the name DBServerSG is used as an example. For **VPC**, select the ID of your VPC and choose **Yes, Create**.

**To add rules to the WebServerSG security group**

1. Select the WebServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, choose **Edit** and add rules for inbound traffic as follows:
   1. Choose **Type**, **HTTP**. For **Source**, enter 0.0.0.0/0.
   2. Choose **Add another rule**, **Type**, **HTTPS**. For **Source**, enter 0.0.0.0/0.
   3. Choose **Add another rule**, **Type**, **SSH**. For **Source**, enter your network's public IPv4 address range.
   4. Choose **Add another rule**, **Type**, **RDP**. For **Source**, enter your network's public IPv4 address range.
   5. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTP**. For **Source**, enter ::/0.
   6. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTPS**. For **Source**, enter ::/0.
   7. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **SSH** (for Linux) or **RDP** (for Windows). For **Source**, enter your network's IPv6 address range.
   8. Choose **Save**.
3. On the **Outbound Rules** tab, choose **Edit** and add rules for outbound traffic as follows:
   1. Locate the default rule that enables all outbound traffic and choose **Remove**.
   2. Choose **Type**, **MS SQL**. For **Destination**, specify the ID of the DBServerSG security group.
   3. Choose **Add another rule**, **Type**, **MySQL**. For **Destination**, specify the ID of the DBServerSG security group.
   4. Choose **Add another rule**, **Type**, **HTTPS**. For **Destination**, enter 0.0.0.0/0.
   5. Choose **Add another rule**, **Type**, **HTTP**. For **Destination**, enter 0.0.0.0/0.
   6. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTPS**. For **Destination**, enter ::/0.
   7. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTP**. For **Destination**, enter ::/0.
   8. Choose **Save**.

**To add the recommended rules to the DBServerSG security group**

1. Select the DBServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, choose **Edit** and add rules for inbound traffic as follows:
   1. Choose **Type**, **MS SQL**. For **Source**, specify the ID of your WebServerSG security group.
   2. Choose **Add another rule**, **Type**, **MYSQL**. For **Source**, specify the ID of your WebServerSG security group.
   3. Choose **Save**.
3. On the **Outbound Rules** tab, choose **Edit** and add rules for outbound traffic as follows:
   1. Locate the default rule that enables all outbound traffic and choose **Remove**.
   2. Choose **Type**, **HTTP**. For **Destination**, enter 0.0.0.0/0.
   3. Choose **Add another rule**, **Type**, **HTTPS**. For **Destination**, enter 0.0.0.0/0.
   4. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTP**. For **Destination**, enter ::/0.
   5. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTPS**. For **Destination**, enter ::/0.
   6. Choose **Save**.

You can now launch instances into your VPC.

**To launch an instance (web server or database server)**

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the dashboard, choose **Launch Instance**.
3. Select an AMI and an instance type and choose **Next: Configure Instance Details**.

**Note**

If you intend to use your instance for IPv6 communication, you must choose a supported instance type; for example, T2. For more information, see Amazon EC2 Instance Types.

1. On the **Configure Instance Details** page, for **Network**, select the VPC that you created earlier and then select a subnet. For example, launch a web server into the public subnet and the database server into the private subnet.
2. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IPv4 address. To be able to connect to your instance in the public subnet, you can assign a public IPv4 address now, or allocate an Elastic IP address and assign it to your instance after it's launched. To assign a public IPv4 address now, ensure that you choose **Enable** from the **Auto-assign Public IP** list. You do not need to assign a public IP address to an instance in the private subnet.

**Note**

You can only use the auto-assign public IPv4 feature for a single, new network interface with the device index of eth0. For more information, see Assigning a Public IPv4 Address During Instance Launch.

1. (Optional, IPv6-only) You can auto-assign an IPv6 address to your instance from the subnet range. For **Auto-assign IPv6 IP**, choose **Enable**.
2. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, choose the **Select an existing security group** option, and select one of the security groups you created earlier (**WebServerSG** for a web server or **DBServerSG** for a database server). Choose **Review and Launch**.
3. Review the settings that you've chosen. Make any changes that you need and choose **Launch** to choose a key pair and launch your instance.

If you did not assign a public IPv4 address to your instance in the public subnet in step 5, you will not be able to connect to it. Before you can access an instance in your public subnet, you must assign it an Elastic IP address.

**To allocate an Elastic IP address and assign it to an instance (IPv4)**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Elastic IPs**.
3. Choose **Allocate new address**.
4. Choose **Allocate**.

**Note**

If your account supports EC2-Classic, first choose **VPC**.

1. Select the Elastic IP address from the list and choose **Actions**, **Associate address**.
2. Select the network interface or instance. For **Private IP**, select the corresponding address to associate the Elastic IP address with and choose **Associate**.

You can now connect to your instances in the VPC. For information about how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Windows Instances*.

**Implementing Scenario 2 with a NAT Instance**

You can implement scenario 2 using a NAT instance instead of a NAT gateway. For more information about NAT instances, see NAT Instances.

You can follow the same procedures as above; however, in the NAT section of the VPC wizard, choose **Use a NAT instance instead** and specify the details for your NAT instance. You will also require a security group for your NAT instance (NATSG), which allows the NAT instance to receive Internet-bound traffic from instances in the private subnet, as well as SSH traffic from your network. The NAT instance can also send traffic to the Internet, so that instances in the private subnet can get software updates.

After you've created the VPC with the NAT instance, you must change the security group associated with the NAT instance to the new NATSG security group (by default, the NAT instance is launched using the default security group).

**NATSG: Recommended Rules**

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| 10.0.1.0/24 | TCP | 80 | Allow inbound HTTP traffic from database servers that are in the private subnet |
| 10.0.1.0/24 | TCP | 443 | Allow inbound HTTPS traffic from database servers that are in the private subnet |
| Your network's public IP address range | TCP | 22 | Allow inbound SSH access to the NAT instance from your network (over the Internet gateway) |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| 0.0.0.0/0 | TCP | 80 | Allow outbound HTTP access to the Internet (over the Internet gateway) |
| 0.0.0.0/0 | TCP | 443 | Allow outbound HTTPS access to the Internet (over the Internet gateway) |

**To create the NATSG security group**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Security Groups**, and the choose **Create Security Group**.
3. Specify a name and description for the security group. In this topic, the name NATSG is used as an example. For **VPC**, select the ID of your VPC and choose **Yes, Create**.
4. Select the NATSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
5. On the **Inbound Rules** tab, choose **Edit** and add rules for inbound traffic as follows:
   1. Choose **Type**, **HTTP** . For **Source**, enter the IP address range of your private subnet.
   2. Choose **Add another rule**, **Type**, **HTTPS**. For **Source**, enter the IP address range of your private subnet.
   3. Choose **Add another rule**, **Type**, **SSH**. For **Source**, enter your network's public IP address range.
   4. Choose **Save**.
6. On the **Outbound Rules** tab, choose **Edit** and add rules for outbound traffic as follows:
   1. Locate the default rule that enables all outbound traffic and choose **Remove**.
   2. Choose **Type**, **HTTP**. For **Destination**, enter 0.0.0.0/0.
   3. Choose **Add another rule**, **Type**, **HTTPS**. For **Destination**, enter 0.0.0.0/0.
   4. Choose **Save**.

When the VPC wizard launched the NAT instance, it used the default security group for the VPC. You need to associate the NAT instance with the NATSG security group instead.

**Scenario 3: VPC with Public and Private Subnets and AWS Managed VPN Access**

The configuration for this scenario includes a virtual private cloud (VPC) with a public subnet and a private subnet, and a virtual private gateway to enable communication with your own network over an IPsec VPN tunnel. We recommend this scenario if you want to extend your network into the cloud and also directly access the Internet from your VPC. This scenario enables you to run a multi-tiered application with a scalable web front end in a public subnet, and to house your data in a private subnet that is connected to your network by an IPsec VPN connection.

This topic assumes that you'll use the VPC wizard in the Amazon VPC console to create the VPC and the VPN connection.

This scenario can also be optionally configured for IPv6—you can use the VPC wizard to create a VPC and subnets with associated IPv6 CIDR blocks. Instances launched into the subnets can receive IPv6 addresses. Currently, we do not support IPv6 communication over a VPN connection; however, instances in the VPC can communicate with each other via IPv6, and instances in the public subnet can communicate over the Internet via IPv6. For more information about IPv4 and IPv6 addressing, see IP Addressing in Your VPC.

**Topics**

* Overview
* Routing
* Security
* Implementing Scenario 3

**Overview**

The following diagram shows the key components of the configuration for this scenario.


     Diagram for scenario 3: VPC with public and private subnets and
      VPN access
    

**Important**

For this scenario, the*Amazon VPC Network Administrator Guide* describes what your network administrator needs to do to configure the Amazon VPC customer gateway on your side of the VPN connection.

The configuration for this scenario includes the following:

* A virtual private cloud (VPC) with a size /16 IPv4 CIDR (example: 10.0.0.0/16). This provides 65,536 private IPv4 addresses.
* A public subnet with a size /24 IPv4 CIDR (example: 10.0.0.0/24). This provides 256 private IPv4 addresses. A public subnet is a subnet that's associated with a route table that has a route to an Internet gateway.
* A VPN-only subnet with a size /24 IPv4 CIDR (example: 10.0.1.0/24). This provides 256 private IPv4 addresses.
* An Internet gateway. This connects the VPC to the Internet and to other AWS products.
* A VPN connection between your VPC and your network. The VPN connection consists of a virtual private gateway located on the Amazon side of the VPN connection and a customer gateway located on your side of the VPN connection.
* Instances with private IPv4 addresses in the subnet range (examples: 10.0.0.5 and 10.0.1.5), which enables the instances to communicate with each other and other instances in the VPC.
* Instances in the public subnet with Elastic IP addresses (example: 198.51.100.1), which are public IPv4 addresses that enable them to be reached from the Internet. The instances can have public IPv4 addresses assigned at launch instead of Elastic IP addresses. Instances in the VPN-only subnet are back-end servers that don't need to accept incoming traffic from the Internet, but can send and receive traffic from your network.
* A custom route table associated with the public subnet. This route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC, and an entry that enables instances in the subnet to communicate directly with the Internet.
* The main route table associated with the VPN-only subnet. The route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC, and an entry that enables instances in the subnet to communicate directly with your network.

For more information about subnets, see VPCs and Subnets and IP Addressing in Your VPC. For more information about Internet gateways, see Internet Gateways. For more information about your VPN connection, see AWS Managed VPN Connections. For more information about configuring a customer gateway, see the *Amazon VPC Network Administrator Guide*.

**Overview for IPv6**

You can optionally enable IPv6 for this scenario. In addition to the components listed above, the configuration includes the following:

* A size /56 IPv6 CIDR block associated with the VPC (example: 2001:db8:1234:1a00::/56). AWS automatically assigns the CIDR; you cannot choose the range yourself.
* A size /64 IPv6 CIDR block associated with the public subnet (example: 2001:db8:1234:1a00::/64). You can choose the range for your subnet from the range allocated to the VPC. You cannot choose the size of the IPv6 CIDR.
* A size /64 IPv6 CIDR block associated with the VPN-only subnet (example: 2001:db8:1234:1a01::/64). You can choose the range for your subnet from the range allocated to the VPC. You cannot choose the size of the IPv6 CIDR.
* IPv6 addresses assigned to the instances from the subnet range (example: 2001:db8:1234:1a00::1a).
* Route table entries in the custom route table that enable instances in the public subnet to use IPv6 to communicate with each other, and directly over the Internet.
* A route table entry in the main route table that enable instances in the VPN-only subnet to use IPv6 to communicate with each other.


      IPv6-enabled VPC with a public and VPN-only subnet
     

**Routing**

Your VPC has an implied router (shown in the configuration diagram for this scenario). In this scenario, the VPC wizard updates the main route table used with the VPN-only subnet, and creates a custom route table and associates it with the public subnet.

The instances in the VPN-only subnet can't reach the Internet directly; any Internet-bound traffic must first traverse the virtual private gateway to your network, where the traffic is then subject to your firewall and corporate security policies. If the instances send any AWS-bound traffic (for example, requests to the Amazon S3 or Amazon EC2 APIs), the requests must go over the virtual private gateway to your network and then egress to the Internet before reaching AWS. Currently, we do not support IPv6 for VPN connections.

**Tip**

Any traffic from your network going to an Elastic IP address for an instance in the public subnet goes over the Internet, and not over the virtual private gateway. You could instead set up a route and security group rules that enable the traffic to come from your network over the virtual private gateway to the public subnet.

The VPN connection is configured either as a statically-routed VPN connection or as a dynamically-routed VPN connection (using BGP). If you select static routing, you'll be prompted to manually enter the IP prefix for your network when you create the VPN connection. If you select dynamic routing, the IP prefix is advertised automatically to the virtual private gateway for your VPC using BGP.

The following tables describe the route tables for this scenario.

**Main Route Table**

The first entry is the default entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other over IPv4. The second entry routes all other IPv4 subnet traffic from the private subnet to your network over the virtual private gateway (for example, vgw-1a2b3c4d).

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 0.0.0.0/0 | *vgw-id* |

**Custom Route Table**

The first entry is the default entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second entry routes all other IPv4 subnet traffic from the public subnet to the Internet over the Internet gateway (for example, igw-1a2b3c4d).

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 0.0.0.0/0 | *igw-id* |

**Alternate Routing**

Alternatively, if you want instances in the private subnet to access the Internet, you can create a network address translation (NAT) gateway or instance in the public subnet, and set up the routing so that the Internet-bound traffic for the subnet goes to the NAT device. This enables the instances in the VPN-only subnet to send requests over the Internet gateway (for example, for software updates).

For more information about setting up a NAT device manually, see NAT. For information about using the VPC wizard to set up a NAT device, see Scenario 2: VPC with Public and Private Subnets (NAT).

To enable the private subnet's Internet-bound traffic to go to the NAT device, you must update the main route table as follows.

The first entry is the default entry for local routing in the VPC. The second row entry for routes the subnet traffic bound for your customer network (in this case, assume your local network's IP address is 172.16.0.0/12) to the virtual private gateway. The third entry sends all other subnet traffic to a NAT gateway.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 172.16.0.0/12 | *vgw-id* |
| 0.0.0.0/0 | *nat-gateway-id* |

**Routing for IPv6**

If you associate an IPv6 CIDR block with your VPC and subnets, your route tables must include separate routes for IPv6 traffic. The following tables show the route tables for this scenario if you choose to enable IPv6 communication in your VPC.

**Main Route Table**

The second entry is the default route that's automatically added for local routing in the VPC over IPv6.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 2001:db8:1234:1a00::/56 | local |
| 0.0.0.0/0 | *vgw-id* |

**Custom Route Table**

The second entry is the default route that's automatically added for local routing in the VPC over IPv6. The fourth entry routes all other IPv6 subnet traffic to the Internet gateway.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | local |
| 2001:db8:1234:1a00::/56 | local |
| 0.0.0.0/0 | *igw-id* |
| ::/0 | *igw-id* |

**Security**

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Security groups control inbound and outbound traffic for your instances, and network ACLs control inbound and outbound traffic for your subnets. In most cases, security groups can meet your needs; however, you can also use network ACLs if you want an additional layer of security for your VPC. For more information, see Security.

For scenario 3, you'll use security groups but not network ACLs. If you'd like to use a network ACL, see Recommended Rules for Scenario 3.

Your VPC comes with a default security group. An instance that's launched into the VPC is automatically associated with the default security group if you don't specify a different security group during launch. For this scenario, we recommend that you create the following security groups instead of using the default security group:

* **WebServerSG**: Specify this security group when you launch web servers in the public subnet.
* **DBServerSG**: Specify this security group when you launch database servers in the VPN-only subnet.

The instances assigned to a security group can be in different subnets. However, in this scenario, each security group corresponds to the type of role an instance plays, and each role requires the instance to be in a particular subnet. Therefore, in this scenario, all instances assigned to a security group are in the same subnet.

The following table describes the recommended rules for the WebServerSG security group, which allow the web servers to receive Internet traffic, as well as SSH and RDP traffic from your network. The web servers can also initiate read and write requests to the database servers in the VPN-only subnet, and send traffic to the Internet; for example, to get software updates. Because the web server doesn't initiate any other outbound communication, the default outbound rule is removed.

**Note**

The group includes both SSH and RDP access, and both Microsoft SQL Server and MySQL access. For your situation, you might only need rules for Linux (SSH and MySQL) or Windows (RDP and Microsoft SQL Server).

**WebServerSG: Recommended Rules**

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| 0.0.0.0/0 | TCP | 80 | Allow inbound HTTP access to the web servers from any IPv4 address. |
| 0.0.0.0/0 | TCP | 443 | Allow inbound HTTPS access to the web servers from any IPv4 address. |
| Your network's public IP address range | TCP | 22 | Allow inbound SSH access to Linux instances from your network (over the Internet gateway). |
| Your network's public IP address range | TCP | 3389 | Allow inbound RDP access to Windows instances from your network (over the Internet gateway). |
| **Outbound** | | | |
| The ID of your DBServerSG security group | TCP | 1433 | Allow outbound Microsoft SQL Server access to the database servers assigned to DBServerSG. |
| The ID of your DBServerSG security group | TCP | 3306 | Allow outbound MySQL access to the database servers assigned to DBServerSG. |
| 0.0.0.0/0 | TCP | 80 | Allow outbound HTTP access to the Internet. |
| 0.0.0.0/0 | TCP | 443 | Allow outbound HTTPS access to the Internet. |

The following table describes the recommended rules for the DBServerSG security group, which allow Microsoft SQL Server and MySQL read and write requests from the web servers and SSH and RDP traffic from your network. The database servers can also initiate traffic bound for the Internet (your route table sends that traffic over the virtual private gateway).

**DBServerSG: Recommended Rules**

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port range** | **Comments** |
| The ID of your WebServerSG security group | TCP | 1433 | Allow inbound Microsoft SQL Server access from the web servers associated with the WebServerSG security group. |
| The ID of your WebServerSG security group | TCP | 3306 | Allow inbound MySQL Server access from the web servers associated with the WebServerSG security group. |
| Your network's IPv4 address range | TCP | 22 | Allow inbound SSH traffic to Linux instances from your network (over the virtual private gateway). |
| Your network's IPv4 address range | TCP | 3389 | Allow inbound RDP traffic to Windows instances from your network (over the virtual private gateway). |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port range** | **Comments** |
| 0.0.0.0/0 | TCP | 80 | Allow outbound IPv4 HTTP access to the Internet (for example, for software updates) over the virtual private gateway. |
| 0.0.0.0/0 | TCP | 443 | Allow outbound IPv4 HTTPS access to the Internet (for example, for software updates) over the virtual private gateway. |

(Optional) The default security group for a VPC has rules that automatically allow assigned instances to communicate with each other. To allow that type of communication for a custom security group, you must add the following rules:

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| The ID of the security group | All | All | Allow inbound traffic from other instances assigned to this security group. |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| The ID of the security group | All | All | Allow outbound traffic to other instances assigned to this security group. |

**Security for IPv6**

If you associate an IPv6 CIDR block with your VPC and subnets, you must add separate rules to your WebServerSG and DBServerSG security groups to control inbound and outbound IPv6 traffic for your instances. In this scenario, the web servers will be able to receive all Internet traffic over IPv6, and SSH or RDP traffic from your local network over IPv6. They can also initiate outbound IPv6 traffic to the Internet. The database servers cannot initiate outbound IPv6 traffic to the Internet, so they do not require any additional security group rules.

The following are the IPv6-specific rules for the WebServerSG security group (which are in addition to the rules listed above).

|  |  |  |  |
| --- | --- | --- | --- |
| **Inbound** | | | |
| **Source** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | TCP | 80 | Allow inbound HTTP access to the web servers from any IPv6 address. |
| ::/0 | TCP | 443 | Allow inbound HTTPS access to the web servers from any IPv6 address. |
| IPv6 address range of your network | TCP | 22 | (Linux instances) Allow inbound SSH access over IPv6 from your network. |
| IPv6 address range of your network | TCP | 3389 | (Windows instances) Allow inbound RDP access over IPv6 from your network |
| **Outbound** | | | |
| **Destination** | **Protocol** | **Port Range** | **Comments** |
| ::/0 | TCP | HTTP | Allow outbound HTTP access to any IPv6 address. |
| ::/0 | TCP | HTTPS | Allow outbound HTTPS access to any IPv6 address. |

**Implementing Scenario 3**

To implement scenario 3, get information about your customer gateway, and create the VPC using the VPC wizard. The VPC wizard creates a VPN connection for you with a customer gateway and virtual private gateway.

These procedures include optional steps for enabling and configuring IPv6 communication for your VPC. You do not have to perform these steps if you do not want to use IPv6 in your VPC.

**To prepare your customer gateway**

1. Determine the device you'll use as your customer gateway. For more information about the devices that we've tested, see Amazon Virtual Private Cloud FAQs. For more information about the requirements for your customer gateway, see the Amazon VPC Network Administrator Guide.
2. Obtain the Internet-routable IP address for the customer gateway's external interface. The address must be static and may be behind a device performing network address translation (NAT).
3. If you want to create a statically-routed VPN connection, get the list of internal IP ranges (in CIDR notation) that should be advertised across the VPN connection to the virtual private gateway. For more information, see VPN Routing Options.

**To create a VPC using the VPC wizard**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. On the dashboard, choose **Start VPC Wizard**.
3. Select the third option, **VPC with Public and Private Subnets and Hardware VPN Access**, and then choose **Select**.
4. For **VPC name**, **Public subnet name** and **Private subnet name**, you can name your VPC and subnets to help you identify them later in the console. You can specify your own IPv4 CIDR block range for the VPC and subnets, or you can leave the default values.
5. (Optional, IPv6-only) For **IPv6 CIDR block**, choose **Amazon-provided IPv6 CIDR block**. For **Public subnet's IPv6 CIDR**, choose **Specify a custom IPv6 CIDR** and specify the hexadecimal pair value for your subnet, or leave the default value. For **Private subnet's IPv6 CIDR**, choose **Specify a custom IPv6 CIDR**. Specify the hexadecimal pair value for the IPv6 subnet or leave the default value.
6. Choose **Next**.
7. On the **Configure your VPN** page, do the following, and then choose **Create VPC**:
   * In **Customer Gateway IP**, specify the public IP address of your VPN router.
   * Optionally specify a name for your customer gateway and VPN connection.
   * In **Routing Type**, select one of the routing options as follows:
     + If your VPN router supports Border Gateway Protocol (BGP), select **Dynamic (requires BGP)**.
     + If your VPN router does not support BGP, choose **Static**. In **IP Prefix**, add each IP range for your network in CIDR notation.

For more information, see VPN Routing Options.

1. When the wizard is done, choose **VPN Connections** in the navigation pane. Select the VPN connection that the wizard created, and choose **Download Configuration**. In the dialog box, select the vendor for your customer gateway, the platform, and the software version, and then choose **Yes, Download**.
2. Save the text file containing the VPN configuration and give it to the network administrator along with this guide: Amazon VPC Network Administrator Guide. The VPN won't work until the network administrator configures the customer gateway.

Create the WebServerSG and DBServerSG security groups. These security groups will reference each other, therefore you must create them before you add rules to them.

**To create the WebServerSG and DBServerSG security groups**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Security Groups**.
3. Choose **Create Security Group**.
4. Provide a name and description for the security group. In this topic, the name WebServerSG is used as an example. Select the ID of your VPC from the **VPC** list, and then choose **Yes, Create**.
5. Choose **Create Security Group** again.
6. Provide a name and description for the security group. In this topic, the name DBServerSG is used as an example. Select the ID of your VPC from the **VPC** list, and then choose **Yes, Create**.

**To add rules to the WebServerSG security group**

1. Select the WebServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, choose **Edit** and add rules for inbound traffic as follows:
   1. Select **HTTP** from the **Type** list, and enter 0.0.0.0/0 in the **Source** field.
   2. Choose **Add another rule**, then select **HTTPS** from the **Type** list, and enter 0.0.0.0/0 in the **Source** field.
   3. Choose **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   4. Choose **Add another rule**, then select **RDP** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   5. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTP**. For **Source**, enter ::/0.
   6. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **HTTPS**. For **Source**, enter ::/0.
   7. (Optional, IPv6-only) Choose **Add another rule**, **Type**, **SSH** (for Linux) or **RDP** (for Windows). For **Source**, enter your network's IPv6 address range.
   8. Choose **Save**.
3. On the **Outbound Rules** tab, choose **Edit** and add rules for outbound traffic as follows:
   1. Locate the default rule that enables all outbound traffic, and then choose **Remove**.
   2. Select **MS SQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
   3. Choose **Add another rule**, then select **MySQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
   4. Choose **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
   5. Choose **Add another rule**, then select **HTTP** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
   6. Choose **Save**.

**To add the recommended rules to the DBServerSG security group**

1. Select the DBServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, choose **Edit** and add rules for inbound traffic as follows:
   1. Select **SSH** from the **Type** list, and enter the IP address range of your network in the **Source** field.
   2. Choose **Add another rule**, then select **RDP** from the **Type** list, and enter the IP address range of your network in the **Source** field.
   3. Choose **Add another rule**, then select **MS SQL** from the **Type** list. Specify the ID of your WebServerSG security group in the **Source** field.
   4. Choose **Add another rule**, then select **MYSQL** from the **Type** list. Specify the ID of your WebServerSG security group in the **Source** field.
   5. Choose **Save**.
3. On the **Outbound Rules** tab, choose **Edit** and add rules for outbound traffic as follows:
   1. Locate the default rule that enables all outbound traffic, and then choose **Remove**.
   2. Select **HTTP** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
   3. Choose **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
   4. Choose **Save**.

After your network administrator configures your customer gateway, you can launch instances into your VPC.

**To launch an instance (web server or database server)**

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Choose **Launch Instance** on the dashboard.
3. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then choose **Next: Configure Instance Details**.

**Note**

If you intend to use your instance for IPv6 communication, you must choose a supported instance type; for example, T2. For more information, see Amazon EC2 Instance Types.

1. On the **Configure Instance Details** page, select the VPC that you created earlier from the **Network** list, and then select a subnet. For example, launch a web server into the public subnet and the database server into the private subnet.
2. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IPv4 address. To be able to connect to your instance in the public subnet, you can assign a public IPv4 address now, or allocate an Elastic IP address and assign it to your instance after it's launched. To assign a public IP address now, ensure that you select **Enable** from the **Auto-assign Public IP** list. You do not need to assign a public IP address to an instance in the private subnet.

**Note**

You can only use the auto-assign public IP address feature with a single, new network interface with the device index of eth0. For more information, see Assigning a Public IPv4 Address During Instance Launch.

1. (Optional, IPv6-only) You can auto-assign an IPv6 address to your instance from the subnet range. For **Auto-assign IPv6 IP**, choose **Enable**.
2. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, select the **Select an existing security group** option, and select one of the security groups that you created (**WebServerSG** for a web server instance or **DBServerSG** for a database server instance). Choose **Review and Launch**.
3. Review the settings that you've chosen. Make any changes that you need, and then choose **Launch** to choose a key pair and launch your instance.

For the instances running in the VPN-only subnet, you can test their connectivity by pinging them from your network. For more information, see Testing the VPN Connection.

If you did not assign a public IPv4 address to your instance in the public subnet in step 5, you will not be able to connect to it. Before you can access an instance in your public subnet, you must assign it an Elastic IP address.

**To allocate an Elastic IP address and assign it to an instance using the console**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Elastic IPs**.
3. Choose **Allocate new address**.
4. Choose **Allocate**.

**Note**

If your account supports EC2-Classic, first choose **VPC**.

1. Select the Elastic IP address from the list, and choose **Actions**, **Associate address**.
2. Select the network interface or instance. Select the address to associate the Elastic IP address with from the corresponding **Private IP** list, and then choose **Associate**.

In scenario 3, you need a DNS server that enables your public subnet to communicate with servers on the Internet, and you need another DNS server that enables your VPN-only subnet to communicate with servers in your network.

Your VPC automatically has a set of DHCP options with domain-name-servers=AmazonProvidedDNS. This is a DNS server that Amazon provides to enable any public subnets in your VPC to communicate with the Internet over an Internet gateway. You must provide your own DNS server and add it to the list of DNS servers your VPC uses. Sets of DHCP options aren't modifiable, so you must create a set of DHCP options that includes both your DNS server and the Amazon DNS server, and update the VPC to use the new set of DHCP options.

**To update the DHCP options**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **DHCP Options Sets**.
3. Choose **Create DHCP options set**.
4. In the **Create DHCP options set** dialog box, in the **Domain name servers** box, specify the address of the Amazon DNS server (AmazonProvidedDNS) and the address of your DNS server (for example, 192.0.2.1), separated by a comma, and then choose **Yes, Create**.
5. In the navigation pane, choose **Your VPCs**.
6. Select the VPC, and then choose **Actions**, **Edit DHCP Options Set**.
7. Select the ID of the new set of options from the **DHCP options set** list and then choose **Save**.
8. (Optional) The VPC now uses this new set of DHCP options and therefore has access to both DNS servers. If you want, you can delete the original set of options that the VPC used.