



Amazon Web Services

Course: CS656 – Internet Higher Layer Protocol

Term Project (Fall 2021)

Submitted to:

Prof. Moshiur Rahman

Moshiur.rahman@njit.edu

TA Vishnu Nandan Chinta

vc368@njit.edu

Submitted by:

Karthik Chandrashekar

kc69@njit.edu

Sacheth Ranganathasetty Tenugondlu

st66@njit.edu

Sriraksha Sarathi Tayi

st64@njit.edu

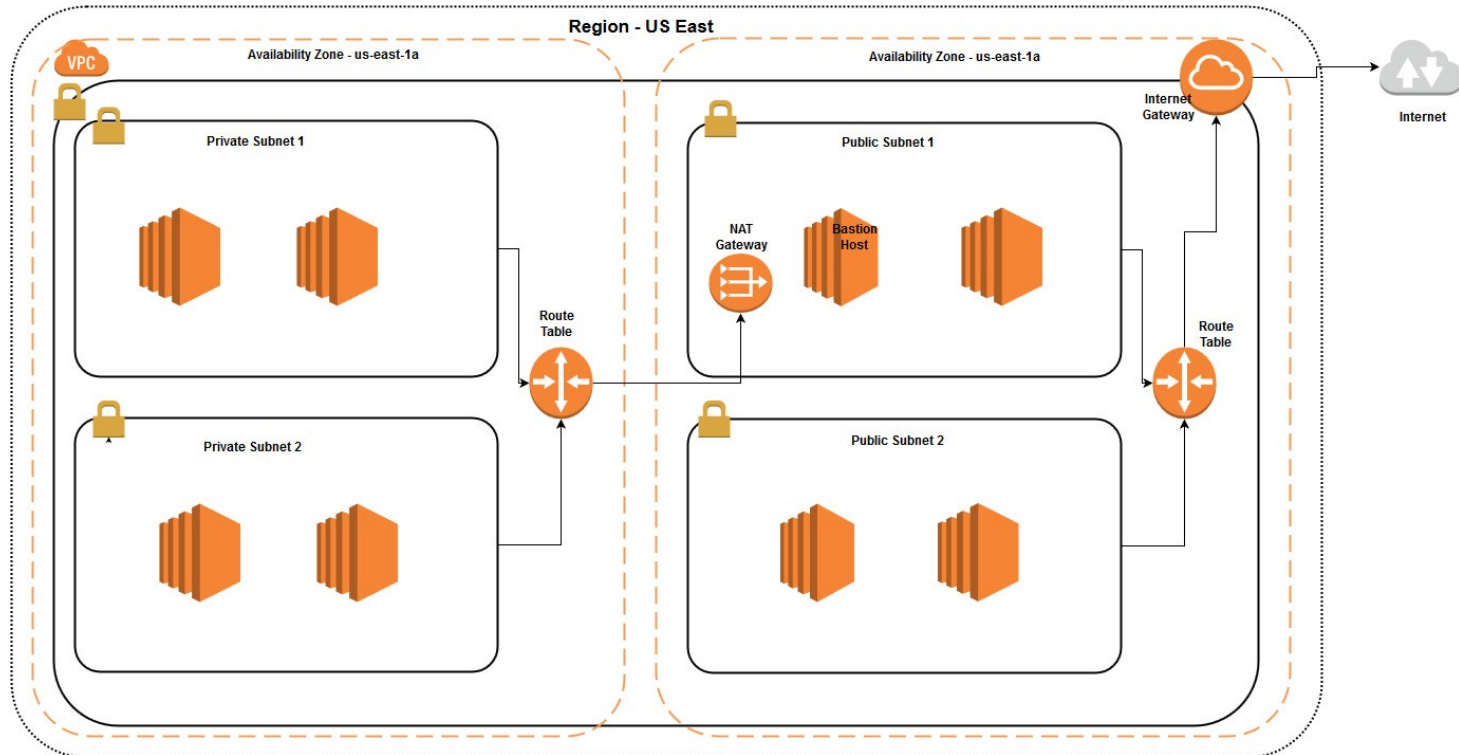


Table of Contents

What is Amazon VPC?	1
Amazon VPC Concepts	1
VPCs and Subnets	1
Supported Platforms	2
Accessing the Internet	2
Accessing a Corporate or Home Network	4
How to Get Started with Amazon VPC	4
Services that Support Amazon VPC	5
Accessing Amazon VPC	5
Pricing for Amazon VPC	6
Amazon VPC Limits	6
Scenarios for Amazon VPC	7
Scenario 1: VPC with a Public Subnet Only	7
Configuration for Scenario 1	8
Basic Components for Scenario 1	8
Routing for Scenario 1	9
Security for Scenario 1	9
Implementing Scenario 1	10
Scenario 2: VPC with Public and Private Subnets	12
Configuration for Scenario 2	13
Basic Components for Scenario 2	13
Routing for Scenario 2	14
Security for Scenario 2	15
Implementing Scenario 2	17
Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access	22
Configuration for Scenario 3	22
Basic Configuration for Scenario 3	23
Routing for Scenario 3	23
Security for Scenario 3	25
Implementing Scenario 3	27
Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access	31
Configuration for Scenario 4	31
Basic Components for Scenario 4	32
Routing for Scenario 4	33
Security for Scenario 4	33
Implementing Scenario 4	34
Your VPC and Subnets	37
Your VPC	37
Your New VPC	37
VPC Sizing	38
Connections Between Your VPC and Your Corporate or Home Network	39
Creating a VPC	39
Deleting Your VPC	40
Subnets in Your VPC	40
Your VPC with Subnets	40
Subnet Sizing	42
Subnet Routing	42
Subnet Security	42
Adding a Subnet to Your VPC	43
Launching an Instance into Your Subnet	43
Deleting Your Subnet	44
CLI Overview	44
Your Default VPC and Subnets	46
Default VPC Basics	46
Availability	46

Components.....	47
Default Subnets	48
Detecting Your Supported Platforms and Whether You Have a Default VPC	48
Detecting Platform Support Using the Console.....	48
Detecting Platform Support Using the Command Line.....	49
Launching an EC2 Instance into Your Default VPC.....	49
Launching an EC2 Instance Using the Console	49
Launching an EC2 Instance Using the Command Line.....	49
Deleting Your Default VPC	50
Security in Your VPC	51
Comparison of Security Groups and Network ACLs.....	51
Security Groups	53
Security Group Basics	53
Default Security Group for Your VPC	53
Security Group Rules	54
Differences Between Security Groups for EC2-Classic and EC2-VPC	55
Working with Security Groups	56
API and Command Overview	99
Network ACLs	59
Network ACL Basics	59
Network ACL Rules	60
Default Network ACL	60
Example Custom Network ACL	60
Ephemeral Ports	62
Working with Network ACLs	62
API and Command Overview	99
Recommended Network ACL Rules for Your VPC	67
Recommended Rules for Scenario 1	67
Recommended Rules for Scenario 2	69
Recommended Rules for Scenario 3.....	71
Recommended Rules for Scenario 4.....	74
Controlling Access.....	75
Example Policies for Amazon VPC.....	76
Networking in Your VPC	85
IP Addressing.....	85
Public and Private IP Addresses.....	85
Assigning a Public IP Address During Launch.....	86
Modifying Your Subnet's Public IP Addressing Behavior	87
Elastic IP Addresses.....	87
Network Interfaces	90
Route Tables	91
Route Table Basics	91
Main Route Tables.....	92
Custom Route Tables.....	92
Route Table Association	93
Route Tables for VPC Peering Connections.....	94
Working with Route Tables.....	95
API and Command Overview	99
Internet Gateways	101
Creating a Subnet.....	102
Attaching an Internet Gateway.....	102
Creating a Custom Route Table	103
Updating the Security Group Rules.....	103
Adding Elastic IP Addresses	104
Detaching an Internet Gateway from Your VPC.....	104
Deleting an Internet Gateway	104
API and Command Overview	105
NAT Instances.....	105

NAT Instance Basics.....	106
Setting up the NAT Instance.....	106
Creating the NATSG Security Group	108
Disabling Source/Destination Checks	109
Updating the Main Route Table	110
Testing Your NAT Instance Configuration	110
DHCP Options Sets.....	112
Overview of DHCP Options Sets	112
Amazon DNS Server	113
Changing DHCP Options	113
Working with DHCP Options Sets	113
API and Command Overview	115
DNS.....	116
Viewing DNS Hostnames for Your EC2 Instance	116
Updating DNS Support for Your VPC	117
Adding a Hardware Virtual Private Gateway to Your VPC	119
Components of Your VPN.....	119
Virtual Private Gateway	120
Customer Gateway	120
VPN Configuration Examples.....	120
Single VPN Connection	120
Multiple VPN connections	121
VPN Routing Options	121
What You Need for a VPN Connection	121
Configuring Two VPN Tunnels for Your VPN Connection	122
Using Redundant VPN Connections to Provide Failover	123
Setting Up the VPN Connection	124
Step 1: Create a Customer Gateway	124
Step 2: Create a Virtual Private Gateway	125
Step 3: Update Your Route Tables and Enable Route Propagation.....	125
Step 4: Update Your Security Group to Enable Inbound SSH, RDP and ICMP Access	126
Step 5: Create a VPN Connection and Configure the Customer Gateway.....	126
Step 6: Launch an Instance Into Your Subnet	126
Testing the End-to-End Connectivity of Your Instance	127
Replacing Compromised Credentials.....	128
Deleting a VPN Connection.....	128
Providing Secure Communication Between Sites Using VPN CloudHub	130
Dedicated Instances	133
Dedicated Instance Basics	133
Dedicated Instances Limitations.....	134
Amazon EBS with Dedicated Instances.....	134
Reserved Instances with Dedicated Tenancy	134
Auto Scaling of Dedicated Instances	134
Pricing for Dedicated Instances.....	134
Working with Dedicated Instances	134
Creating a VPC with an Instance Tenancy of Dedicated.....	135
Launching Dedicated Instances into a VPC.....	135
Displaying Tenancy Information	135
API and Command Overview.....	136
VPC Peering.....	138
VPC Peering Basics.....	138
VPC Peering Connection Lifecycle.....	139
VPC Peering Limitations	140
Working with VPC Peering Connections	140
Creating a VPC Peering Connection	141
Accepting a VPC Peering Connection	142
Rejecting a VPC Peering Connection	143
Updating Route Tables for Your VPC Peering Connection.....	143

Amazon Virtual Private Cloud User Guide

Describing Your VPC Peering Connections.....	144
Deleting a VPC Peering Connection.....	144
API and CLI Overview	145
Controlling Access to VPC Peering Connections	146
Amazon VPC Limits	147
Document History	149
AWS Glossary	151

What is Amazon VPC?

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (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.

Topics

- [Amazon VPC Concepts \(p. 1\)](#)
- [How to Get Started with Amazon VPC \(p. 4\)](#)
- [Services that Support Amazon VPC \(p. 5\)](#)
- [Accessing Amazon VPC \(p. 5\)](#)
- [Pricing for Amazon VPC \(p. 6\)](#)
- [Amazon VPC Limits \(p. 6\)](#)

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 Elastic Compute Cloud User Guide* to get a brief overview.

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 configure your VPC; you can select its IP address range, create subnets, and configure route tables, network gateways, and security settings.

A *subnet* is a range of IP addresses in your VPC. You can launch AWS resources into a subnet that you select. 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.

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 in Your VPC \(p. 51\)](#).

Supported Platforms

There are two supported platforms into which you can launch instances: EC2-Classic and EC2-VPC. For more information, see [Supported Platforms](#) in the *Amazon Elastic Compute Cloud User Guide*.

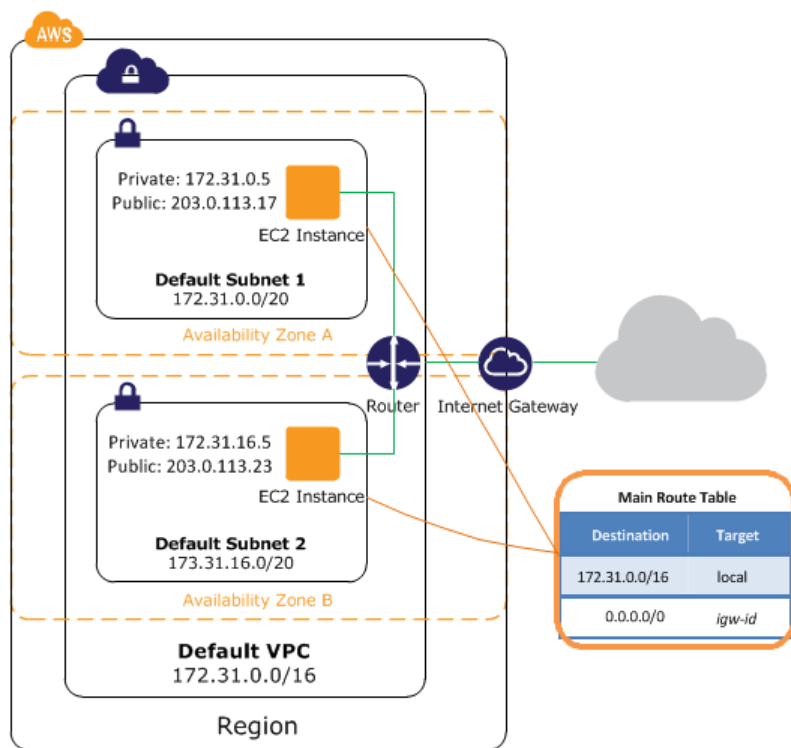
A default VPC combines the benefits of the advanced features provided by EC2-VPC with the ease of use of EC2-Classic. If you have a default VPC and don't specify a subnet when you launch an instance, the instance is launched into your default VPC. You can launch instances into your default VPC without needing to know anything about Amazon VPC.

For more information, see [Your Default VPC and Subnets \(p. 46\)](#).

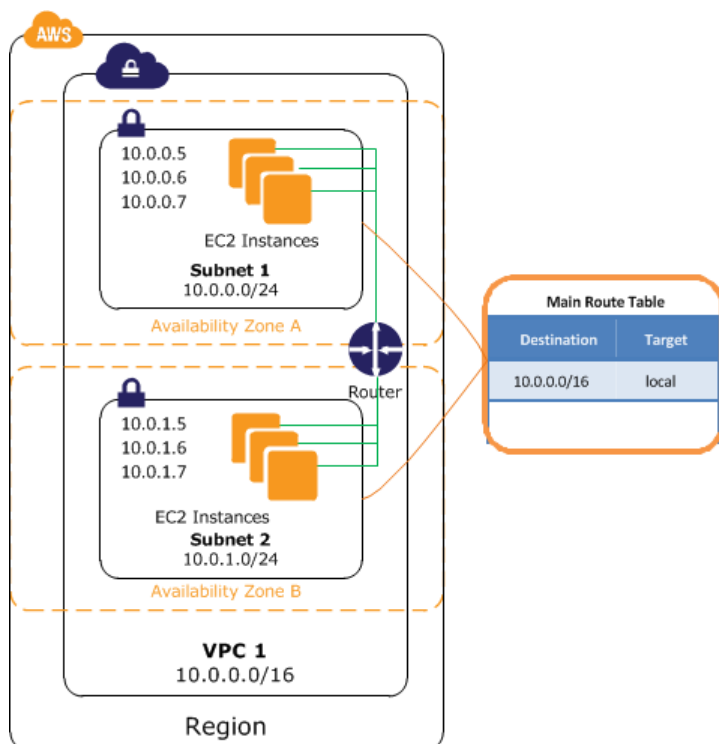
Accessing the Internet

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

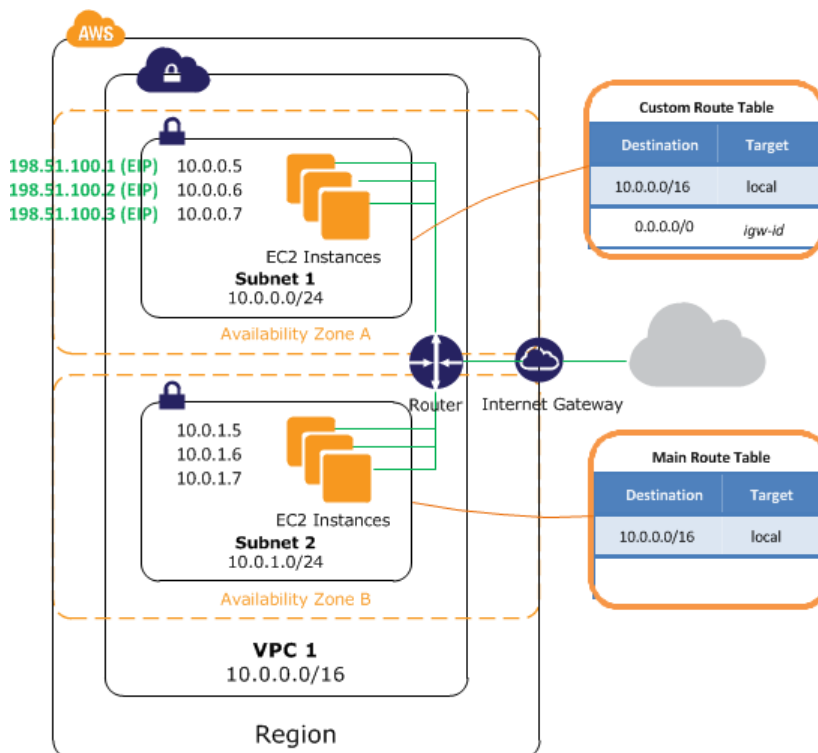
Each instance that you launch into a default subnet has a private IP address and a public IP address. These instances can communicate with the Internet through an Internet gateway. An Internet gateway enables your instances to connect to the Internet through the Amazon EC2 network edge. Your default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



Each instance that you launch into a nondefault subnet has a private IP address, but no public IP 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 or other AWS products, such as Amazon Simple Storage Service (Amazon S3).



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.



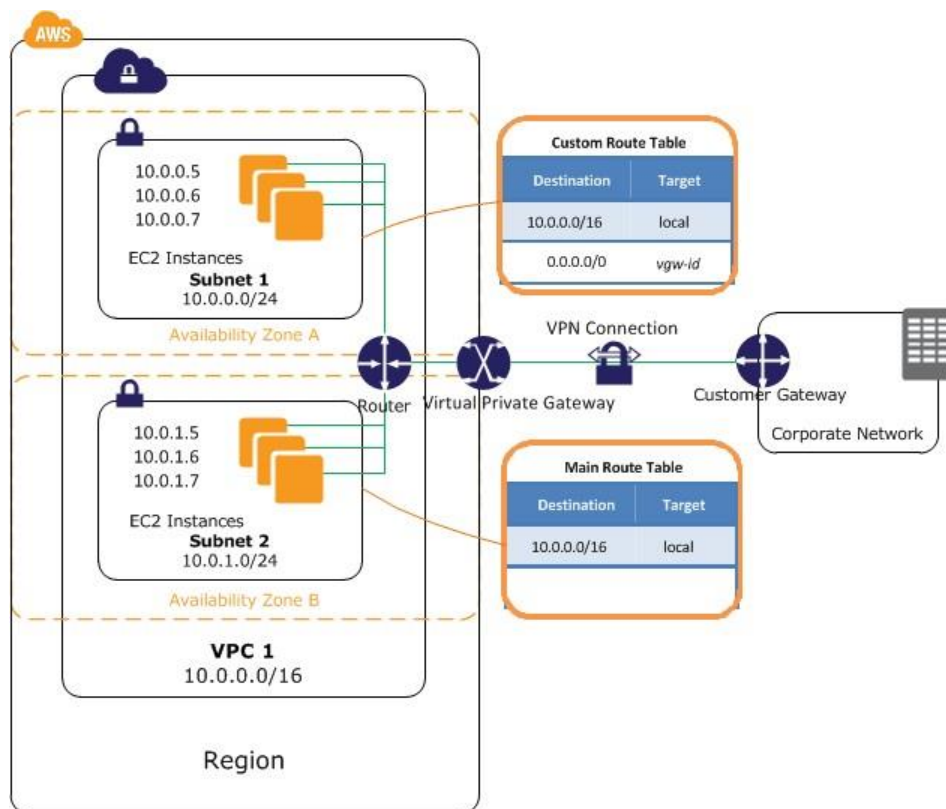
Alternatively, to allow an instance in your VPC to initiate outbound connections to the Internet but prevent unsolicited inbound connections from the Internet, you can use a network address translation (NAT) instance. NAT maps multiple private IP addresses to a single public IP address. A NAT instance 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 instance, which routes traffic from the instance to the Internet gateway, and routes any responses to the instance.

For more information about routing and NAT in your VPC, see [Route Tables \(p. 91\)](#) and [NAT Instances \(p. 105\)](#).

Accessing a Corporate or Home Network

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

A VPN connection consists of a VPG attached to your VPC and a customer gateway located in your data center. A VPG 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.



For more information, see [Adding a Hardware Virtual Private Gateway to Your VPC \(p. 119\)](#).

How to Get Started with Amazon VPC

To get a hands-on introduction to Amazon VPC, complete the tutorial [Getting Started with Amazon VPC](#) in the *Amazon Virtual Private Cloud Getting Started Guide*.

To learn about the basic scenarios for Amazon VPC, see [Scenarios for Amazon VPC \(p. 7\)](#). You can configure your VPC and subnets in other ways to suit your needs. For more information about other scenarios, see [Amazon Virtual Private Cloud Connectivity Options](#).

The following table lists related resources that you'll 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.

Services that Support Amazon VPC

To learn about using Amazon VPC with other AWS products, see the following documentation.

Service	Relevant Topic
Amazon EC2	Amazon EC2 and Amazon VPC
Amazon ElastiCache	Using ElastiCache with Amazon VPC
Amazon EMR	Select a Subnet for the Cluster
Amazon RDS	Amazon RDS and Amazon VPC
Amazon Redshift	Managing Clusters in a VPC
Auto Scaling	Auto Scaling and Amazon VPC
AWS Data Pipeline	Launching Resources for Your Pipeline into a VPC
Elastic Beanstalk	Using AWS Elastic Beanstalk with Amazon VPC
Elastic Load Balancing	Elastic Load Balancing and Amazon VPC

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 selecting **VPC** from the console home page.

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

AWS Command Line Interface (CLI)

Provides commands for a broad set of AWS products, and is supported on Windows, Mac, and Linux/UNIX. To get started, see [AWS Command Line Interface User Guide](#). For more information about the commands for Amazon VPC, see [ec2](#).

Amazon EC2 Command Line Interface (CLI) Tools

Provides commands for Amazon EC2, Amazon EBS, and Amazon VPC, and is supported on Windows, Mac, and Linux/UNIX. For more information about the commands, see [Commands \(CLI Tools\)](#) in the *Amazon Elastic Compute Cloud Command Line Reference*.

AWS Tools for Windows PowerShell

Provides commands for a broad set of AWS products 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 about the API actions for Amazon VPC, see [Actions](#) in the *Amazon Elastic Compute Cloud API Reference*.

If you prefer 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, so that it is easier for you to get started. For more information about downloading the AWS SDKs, 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. If you choose to create a hardware VPN connection, you pay for each hour that the VPN is connected to your VPC. 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 in these limits. For more information about these limits, and how to request an increase, see [Amazon VPC Limits \(p. 147\)](#).

Scenarios for Amazon VPC

This section describes the basic scenarios for using Amazon VPC. We provide the following for each scenario:

- A diagram showing the basic components
- Information about the VPC and subnets
- Information about the routing tables for the subnet
- Information about the recommended security group rules
- Step-by-step directions to implement the scenario

The following table describes the basic scenarios.

Scenario	Usage
Scenario 1: VPC with a Public Subnet Only (p. 7)	Run a single-tier, public-facing web application such as a blog or simple web site.
Scenario 2: VPC with Public and Private Subnets (p. 12)	Run a public-facing web application, while still maintaining non-publicly accessible back-end servers in a second subnet.
Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 22)	Extend your data center into the cloud, and also directly access the Internet from your VPC.
Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access (p. 31)	Extend your data center into the cloud, and leverage Amazon's infrastructure without exposing your network to the Internet.

Scenario 1: VPC with a Public Subnet Only

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.

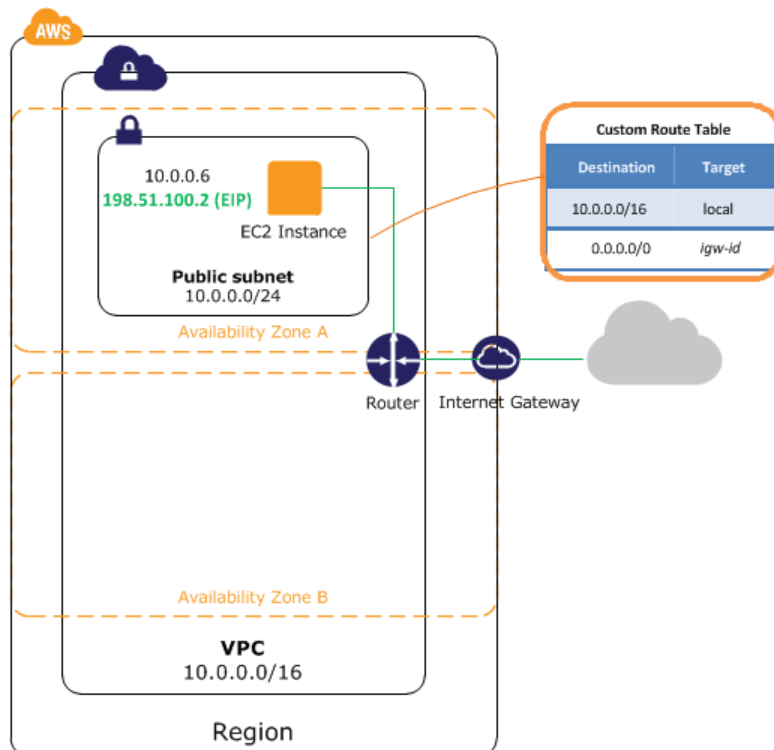
Topics

- [Configuration for Scenario 1 \(p. 8\)](#)

- [Basic Components for Scenario 1](#) (p. 8)
- [Routing for Scenario 1](#) (p. 9)
- [Security for Scenario 1](#) (p. 9)
- [Implementing Scenario 1](#) (p. 10)

Configuration for Scenario 1

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



Note

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

Basic Components for Scenario 1

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- An Internet gateway. This connects the VPC to the Internet and to other AWS products, such as Amazon Simple Storage Service (Amazon S3).
- An instance with a private IP 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 IP address (example: 198.51.100.2), which enables the instance to be reached from the Internet.

- A route table entry that enables instances in the subnet to communicate with other instances in the VPC, and a route table entry that enables instances in the subnet to communicate directly over the Internet.

For more information about subnets, see [Your VPC and Subnets \(p. 37\)](#) and [IP Addressing in Your VPC \(p. 85\)](#). For more information about Internet gateways, see [Adding an Internet Gateway to Your VPC \(p. 101\)](#).

Tip

If you'd like instances in your VPC to communicate over the Internet without having to assign each instance an Elastic IP address, you can use a NAT instance. For more information about configuring a NAT instances, see [Scenario 2: VPC with Public and Private Subnets \(p. 12\)](#) or [NAT Instances \(p. 105\)](#).

Routing for Scenario 1

Your VPC has an implied router (shown in the configuration diagram for this scenario.) For this scenario, the VPC wizard creates a 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. Otherwise, you'd need to create and associate the route table yourself.

The following table shows what the route table looks like for the example addresses used in the configuration diagram for this scenario. The first row shows the entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second row shows the entry for routing all other subnet traffic to the Internet gateway, which is specified using its AWS-assigned identifier.

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-xxxxxxx

Security for Scenario 1

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see [Security in Your VPC \(p. 51\)](#).

For scenario 1, you'll use a security group but not network ACLs. If you'd like to use a network ACL, see [Recommended Rules for Scenario 1 \(p. 67\)](#).

Recommended Security Group Rules

Your VPC comes with a default security group whose initial settings allow all outbound traffic and allow all traffic between instances assigned to the security group. If you don't specify a security group when you launch an instance, the instance is automatically assigned to the default security group for the VPC. We could modify the rules for the default security group, but the rules that you need for your web servers might not work for other instances that you launch into the VPC. Therefore, we recommend that you create a security group to use with the web servers in your public subnet.

You'll create a security group named `WebServerSG`, modify the rules as needed, and then specify the security group when you launch instances into the VPC. By default, new security groups start with only an outbound rule that allows all traffic to leave the instances. You must add rules to enable any inbound traffic or to restrict the outbound traffic.

The following table describes the inbound rules for the `WebServerSG` group. Because the web server doesn't initiate outbound communication, we'll remove the default outbound rule.

Inbound			
Source	Protocol	Port Range	Comments
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
Public IP address range of your network	TCP	22	(Linux instances) Allow inbound SSH access from your network
Public IP address range of your network	TCP	3389	(Windows instances) Allow inbound RDP access from your network

Tip

You can also get the public IP address of your local computer using a service. To locate a service that provides your IP address, use the search phrase "what is my IP address". If you are connecting through an ISP or from behind a firewall without a static IP address, you need to find the range of IP addresses used by client computers.

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 between the instances in your VPC, you must add a rule like the following to your security groups.

Inbound			
Source	Protocol	Port Range	Comments
The security group ID (sg-xxxxxxx)	All	All	Allow inbound traffic from other instances assigned to this security group

Implementing Scenario 1

Use the following process to implement the scenario using the VPC wizard.

Tip

The [Amazon Virtual Private Cloud Getting Started Guide](#) describes the same steps, but provides additional details for some of these steps.

To implement scenario 1 using the VPC wizard

1. Set up the VPC, subnet, and Internet gateway:
 - a. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 - b. In the navigation pane, click **VPC Dashboard**.

- c. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
 - d. Select the first option, **VPC with a Single Public Subnet**, and then click **Select**.
 - e. The confirmation page shows the CIDR ranges and settings that you've chosen. Make any changes that you need, and then click **Create VPC** to create your VPC, subnet, Internet gateway, and route table.
2. Create the WebServerSG security group and add rules:
 - a. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 - b. In the navigation pane, click **Security Groups**.
 - c. Click the **Create Security Group** button.
 - d. Specify `WebServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** menu, and then click **Yes, Create**.
 - e. 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.
 - f. On the **Inbound Rules** tab, click **Edit**, and then do the following:
 - Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
 - Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
 - Click **Add another rule**, then select **SSH** 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'll authorize only a specific IP address or range of addresses to access your instance.)
- Tip**
If your company uses both Linux and Windows instances, you can add access for both SSH and RDP.
- Click **Save**.

Type	Protocol	Port Range	Source	Remove
HTTP (80)	TCP (6)	80	0.0.0.0/0	
HTTPS (443)	TCP (6)	443	0.0.0.0/0	
SSH (22)	TCP (6)	22	192.0.2.0/24	
RDP (3389)	TCP (6)	3389	192.0.2.0/24	

Add another rule

- g. On the **Outbound Rules** tab, click **Edit**. Locate the default rule that enables all outbound traffic, click **Remove**, and then click **Save**.
3. Launch an instance into the VPC:
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. From the dashboard, click the **Launch Instance** button.

- c. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
 - d. On the **Configure Instance Details** page, select the VPC that you created in step 1 from the **Network** list, and then specify a subnet.
 - e. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IP address. To be able to connect to your instance, you can assign a public IP 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 the **Public IP** check box is selected.
- Note**
You can only assign a public IP address to a single, new network interface with the device index of eth0. For more information, see [Assigning a Public IP Address During Launch](#) (p. 86).
- f. 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. Click **Review and Launch**.
 - g. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.
4. If you did not assign a public IP address to your instance as part of step 3, you will not be able to connect to it. Assign an Elastic IP address to the instance:
 - a. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 - b. In the navigation pane, click **Elastic IPs**.
 - c. Click the **Allocate New Address** button.
 - d. From the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.
 - e. Select the Elastic IP address from the list, and then click the **Associate Address** button.
 - f. In the **Associate Address** dialog box, select the instance to associate the address with, and then click **Yes, 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 Elastic Compute Cloud User Guide*. For information about how to connect to a Windows instance, see [Connect to Your Windows Instance](#) in the *Amazon Elastic Compute Cloud Microsoft Windows Guide*.

Scenario 2: VPC with Public and Private Subnets

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.

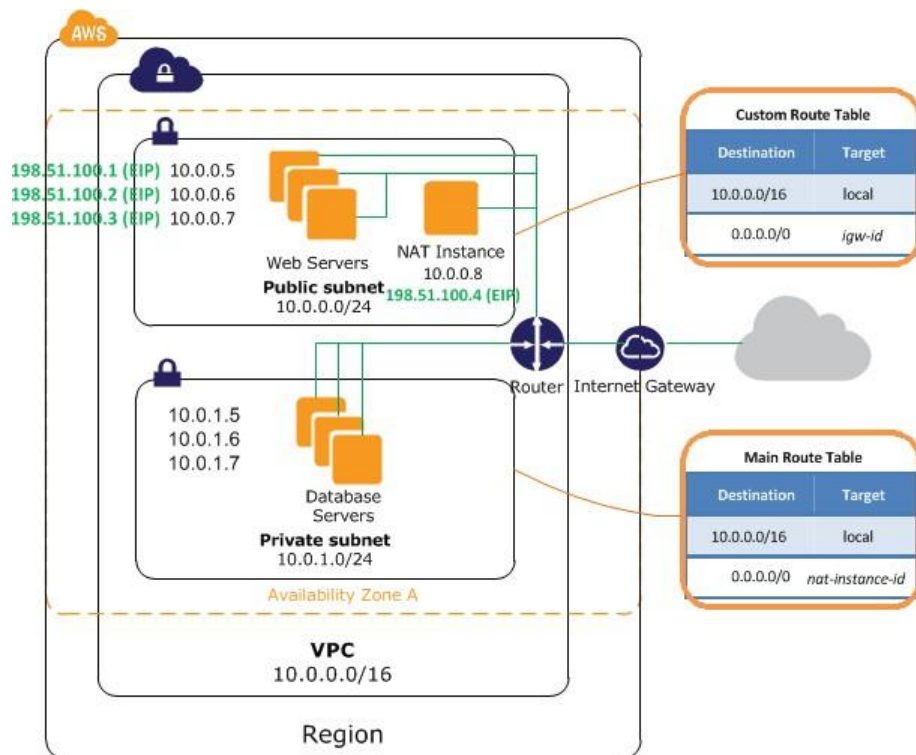
The instances in the public subnet can receive inbound traffic directly from the Internet, whereas the instances in the private subnet can't. The instances in the public subnet can send outbound traffic directly to the Internet, whereas the instances in the private subnet can't. Instead, the instances in the private subnet can access the Internet by using a network address translation (NAT) instance that you launch into the public subnet.

Topics

- [Configuration for Scenario 2 \(p. 13\)](#)
- [Basic Components for Scenario 2 \(p. 13\)](#)
- [Routing for Scenario 2 \(p. 14\)](#)
- [Security for Scenario 2 \(p. 15\)](#)
- [Implementing Scenario 2 \(p. 17\)](#)

Configuration for Scenario 2

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



Basic Components for Scenario 2

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A public subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- A private subnet of size /24 (example CIDR: 10.0.1.0/24). This provides 256 private IP addresses.
- An Internet gateway. This connects the VPC to the Internet and to other AWS products, such as Amazon Simple Storage Service (Amazon S3).
- Instances with private IP addresses in the subnet range (examples: 10.0.0.5, 10.0.1.5), which enables them to communicate with each other and other instances in the VPC. Instances in the public subnet also have Elastic IP addresses (example: 198.51.100.1), which enable them to be reached from the Internet. Instances in the private subnet are back-end servers that don't need to accept incoming traffic from the Internet; however, they can send requests to the Internet using the NAT instance (see the next bullet).

- A network address translation (NAT) instance with its own Elastic IP address. This enables instances in the private subnet to send requests to the Internet (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, and an entry that enables instances in the subnet to communicate directly with the Internet.
- 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, and an entry that enables instances in the subnet to communicate with the Internet through the NAT instance.

For more information about subnets, see [Your VPC and Subnets \(p. 37\)](#) and [IP Addressing in Your VPC \(p. 85\)](#). For more information about Internet gateways, see [Adding an Internet Gateway to Your VPC \(p. 101\)](#). For more information about NAT, see [NAT Instances \(p. 105\)](#).

Tip

To help manage the instances in the private subnet, you can set up bastion servers in the public subnet to act as proxies. For example, you can set up SSH port forwarders or RDP gateways in the public subnet to proxy the traffic going to your database servers from your own network.

Routing for Scenario 2

Your VPC has an implied router (shown in the configuration diagram for this scenario). For 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. Otherwise, you'd need to create and associate the route tables yourself.

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 instance 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'd like 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 row describes the entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second row describes the entry that sends all other subnet traffic to the NAT instance, which is specified using its AWS-assigned identifiers (for example, network interface `eni-1a2b3c4d` and instance `i-1a2b3c4d`).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	eni-xxxxxxx / i-xxxxxxx

Custom Route Table

The first row describes the entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic to the Internet over the Internet gateway, which is specified using its AWS-assigned identifier (for example, `igw-1a2b3d4d`).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-XXXXXXX

Security for Scenario 2

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see [Security in Your VPC \(p. 51\)](#).

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 \(p. 69\)](#).

Recommended Security Groups

Your VPC comes with a default security group whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances assigned to the group. If you don't specify a security group when you launch an instance, the instance is automatically assigned to this default security group.

For this scenario, we recommend that you create the following security groups instead of modifying the default security group:

- **WebServerSG**—For the web servers in the public subnet
- **NATSG**—For the NAT instance in the public subnet
- **DBServerSG**—For 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 WebServerSG security group is the security group that you'll specify when you launch your web servers into your public subnet. The following table describes the recommended rules for this 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. Because the web server doesn't initiate outbound communication, we'll remove the default outbound rule.

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 anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
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			
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 DBServerSG
The ID of your DBServerSG security group	TCP	3306	Allow outbound MySQL access to the database servers assigned to DBServerSG

The NATSG security group is the security group that you'll specify when you launch a NAT instance into your public subnet. The following table describes the recommended rules for this security group, which allow 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.

NATSG: Recommended Rules

Inbound			
Source	Protocol	Port Range	Comments
10.0.1.0/24	TCP	80	Allow inbound HTTP traffic from database servers in the private subnet
10.0.1.0/24	TCP	443	Allow inbound HTTPS traffic from database servers 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)

The DBServerSG security group is the security group that you'll specify when you launch your database servers into your private subnet. The following table describes the recommended rules for this security group, which allow read or write database requests from the web servers. The database servers can also initiate traffic bound for the Internet (your route table sends that traffic to the NAT instance, 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 web servers assigned to WebServerSG Microsoft SQL Server access to database servers assigned to DBServerSG
The ID of your WebServerSG security group	TCP	3306	Allow web servers assigned to WebServerSG MySQL access to database servers assigned to DBServerSG
Outbound			
Destination	Protocol	Port Range	Comments
0.0.0.0/0	TCP	80	Allow outbound HTTP access to the Internet (for example, for software updates)
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (for example, for software updates)

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 between instances in your VPC when you use a different security group, you must add a rule like the following to your security groups.

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

Implementing Scenario 2

Use the following process to implement scenario 2 using the VPC wizard.

To implement scenario 2 using the VPC wizard

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **VPC Dashboard**.

3. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
4. Select the second option, **VPC with Public and Private Subnets**, and then click **Select**.
5. Verify the information on the confirmation page. Make any changes that you need, and then click **Create VPC** to create your VPC, subnets, Internet gateway, and route tables, and launch a NAT instance into the public subnet.

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, NATSG, and DBServerSG security groups

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Security Groups**.
3. Click the **Create Security Group** button.
4. In the **Create Security Group** dialog box, specify `WebServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
5. Click the **Create Security Group** button again.
6. In the **Create Security Group** dialog box, specify `NATSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
7. Click the **Create Security Group** button again.
8. In the **Create Security Group** dialog box, specify `DBServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **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, click **Edit** and add rules for inbound traffic as follows:
 - a. Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
 - b. Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
 - c. Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
 - d. Click **Add another rule**, then select **RDP** from the **Type** list. Enter your network's public IP address range in the **Source** field.
 - e. Click **Save**.

Type	Protocol	Port Range	Source	Remove
HTTP (80)	TCP (6)	80	0.0.0.0/0	
HTTPS (443)	TCP (6)	443	0.0.0.0/0	
SSH (22)	TCP (6)	22	192.0.2.0/24	
RDP (3389)	TCP (6)	3389	192.0.2.0/24	

Add another rule

3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:
 - a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
 - b. Select **MS SQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
 - c. Click **Add another rule**, then select **MySQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
 - d. Click **Save**.

Type	Protocol	Port Range	Destination	Remove
MS SQL (1433)	TCP (6)	1433	sg-1a2b3c4d	
MySQL (3306)	TCP (6)	3306	sg-1a2b3c4d	

To add the recommended rules to the NATSG security group

1. 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.
2. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:
 - a. Select **HTTP** from the **Type** list, and enter the IP address range of your private subnet in the **Source** field.
 - b. Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter the IP address range of your private subnet in the **Source** field.
 - c. Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
 - d. Click **Save**.
3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:
 - a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
 - b. Select **HTTP** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
 - c. Click **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
 - d. Click **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, click **Edit** and add rules for inbound traffic as follows:
 - a. Select **MS SQL** from the **Type** list, and specify the ID of your WebServerSG security group in the **Source** field.

- b. Click **Add another rule**, then select **MYSQL** from the **Type** list, and specify the ID of your WebServerSG security group in the **Source** field.
 - c. Click **Save**.
3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:
 - a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
 - b. Select **HTTP** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
 - c. Click **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
 - d. Click **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.

To change the security group of the NAT instance

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, click **Network Interfaces**.
3. Select the network interface for the NAT instance from the list, and then select **Change Security Groups** from the **Actions** list.
4. In the **Change Security Groups** dialog box, select the NATSG security group that you created (see [Security for Scenario 2 \(p. 15\)](#)) from the **Security groups** list, and then click **Save**.

You can launch instances into your VPC. If you're already familiar with launching instances outside a VPC, then you already know most of what you need to know to launch an instance into a VPC.

To launch an instance (web server or database server)

1. Create the WebServerSG and DBServerSG security groups if you haven't done so already (see [Security for Scenario 2 \(p. 15\)](#)). You'll specify one of these security groups when you launch the instance.
2. Start the launch wizard:
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. Click the **Launch Instance** button from the dashboard.
3. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
4. 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.
5. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IP address. To be able to connect to your instance, you can assign a public IP 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 the **Public IP** check box is selected.

Note

You can only assign a public IP address to a single, new network interface with the device index of eth0. For more information, see [Assigning a Public IP Address During Launch \(p. 86\)](#).

6. 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 a security group for the instance (**WebServerSG** for a web server or **DBServerSG** for a database server). Click **Review and Launch**.
7. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.

If you did not assign a public IP address to your instance 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, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. From the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.
5. Select the Elastic IP address from the list, and then click the **Associate Address** button.
6. In the **Associate Address** dialog box, select the network interface or instance. Select the address to associate the Elastic IP address with from the corresponding **Private IP address** list, and then click **Yes, 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 Elastic Compute Cloud User Guide*. For information about how to connect to a Windows instance, see [Connect to Your Windows Instance](#) in the *Amazon Elastic Compute Cloud Microsoft Windows Guide*.

Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access

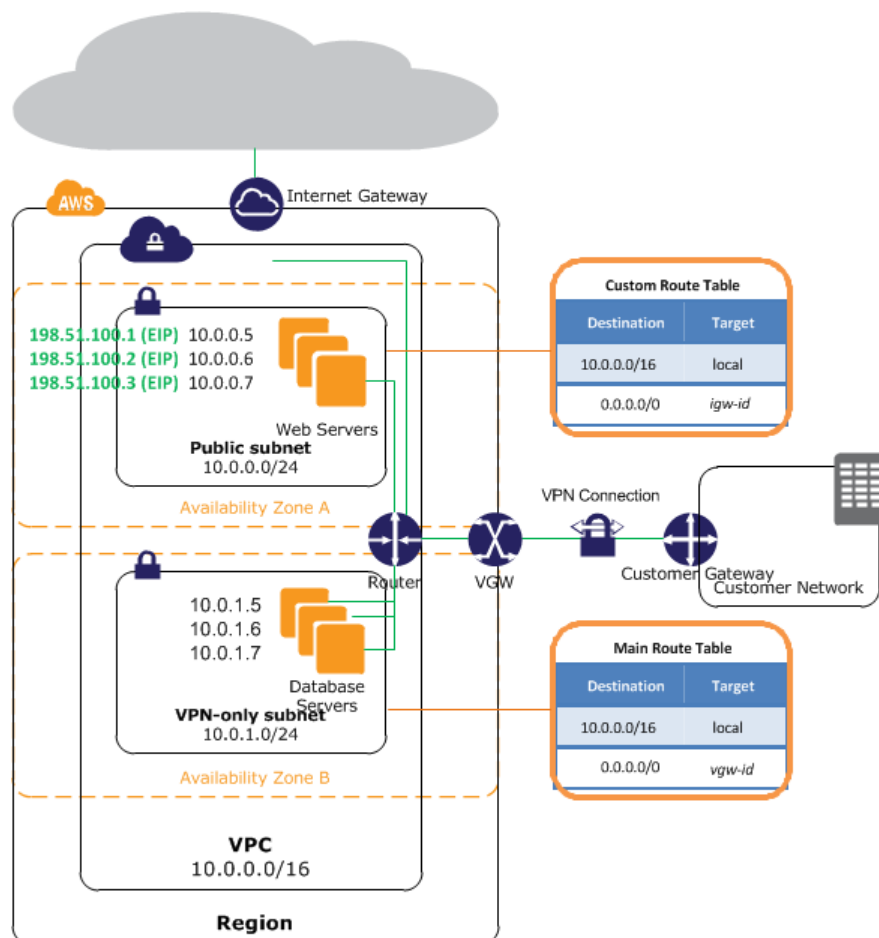
The configuration for this scenario includes a virtual private cloud (VPC) with a public subnet and a private subnet, and a virtual private gateway to enable communication with your own network over an IPsec VPN tunnel. We recommend this scenario if you want to extend your network into the cloud and 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.

Topics

- [Configuration for Scenario 3 \(p. 22\)](#)
- [Basic Configuration for Scenario 3 \(p. 23\)](#)
- [Routing for Scenario 3 \(p. 23\)](#)
- [Security for Scenario 3 \(p. 25\)](#)
- [Implementing Scenario 3 \(p. 27\)](#)

Configuration for Scenario 3

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



Important

For this scenario, the [Amazon Virtual Private Cloud 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.

Basic Configuration for Scenario 3

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A public subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- A VPN-only subnet of size /24 (example CIDR: 10.0.1.0/24). This provides 256 private IP addresses.
- An Internet gateway. This connects the VPC to the Internet and to other AWS products, such as Amazon Simple Storage Service (Amazon S3).
- 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 IP 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 also have Elastic IP addresses (example: 198.51.100.1), which enables them to be reached from the Internet. 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 [Your VPC and Subnets \(p. 37\)](#) and [IP Addressing in Your VPC \(p. 85\)](#). For more information about Internet gateways, see [Adding an Internet Gateway to Your VPC \(p. 101\)](#). For more information about your VPN connection, see [Adding a Hardware Virtual Private Gateway to Your VPC \(p. 119\)](#). For more information about configuring a customer gateway, see the [Amazon Virtual Private Cloud Network Administrator Guide](#).

Routing for Scenario 3

Your VPC has an implied router (shown in the configuration diagram for this scenario). For 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. Otherwise, you'd need to create and associate the route tables yourself.

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.

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 row describes the entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic from the private subnet to your network over the virtual private gateway, which is specified using its AWS-assigned identifier (for example, `vgw-1a2b3c4d`).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	vgw-XXXXXXX

Custom Route Table

The first row describes the entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic from the public subnet to the Internet over the Internet gateway, which is specified using its AWS-assigned identifier (for example, `igw-1a2b3c4d`).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-XXXXXXX

Alternate Routing

Alternatively, if you want instances in the private subnet to access the Internet, you could set up the routing so that the Internet-bound traffic for the subnet goes to a network address translation (NAT) instance in the public subnet. The NAT instance enables the instances in the VPN-only subnet to send requests over the Internet gateway (for example, for software updates). To enable the private subnet's Internet-bound traffic to go to the NAT instance, you must update the main route table as follows.

Main Route Table

The first row describes the entry for local routing in the VPC. The second row describes the entry for routing the subnet traffic bound for your network to the virtual private gateway, which is specified using its AWS-assigned identifier (for example, `vgw-1a2b3c4d`). The third row sends all other subnet traffic to the NAT instance, which is specified by its AWS-assigned identifier (for example, `i-1a2b3c4d`).

Destination	Target
10.0.0.0/16	local
172.16.0.0/12	vgw-XXXXXXX
0.0.0.0/0	i-XXXXXXX

For information about setting up a NAT instance manually, see [NAT Instances \(p. 105\)](#). For information about using the VPC wizard to set up a NAT instance, see [Scenario 2: VPC with Public and Private Subnets \(p. 12\)](#).

Security for Scenario 3

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see [Security in Your VPC \(p. 51\)](#).

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 \(p. 71\)](#).

Topics

- [Recommended Security Groups \(p. 25\)](#)

Recommended Security Groups

Your VPC comes with a default security group whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances assigned to the security group. If you don't specify a security group when you launch an instance, the instance is automatically assigned to this default security group.

For this scenario, we recommend that you create the following security groups instead of modifying the default security group:

- **WebServerSG**—For the web servers in the public subnet
- **DBServerSG**—For the 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 WebServerSG security group is the security group that you'll specify when you launch your web servers into your public subnet. The following table describes the recommended rules for this 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 server instances in the VPN-only subnet.

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 anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
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

The DBServerSG security group is the security group that you'll specify when you launch your database servers into your VPN-only subnet. The following table describes the recommended rules for this 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 web servers assigned to WebServerSG Microsoft SQL Server access to database servers assigned to DBServerSG
The ID of your WebServerSG security group	TCP	3306	Allow web servers assigned to WebServerSG MySQL access to database servers assigned to DBServerSG
Your network's IP address range	TCP	22	Allow inbound SSH traffic to Linux instances from your network (over the virtual private gateway)
Your network's IP 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 HTTP access to the Internet (for example, for software updates) over the virtual private gateway
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (for example, for software updates) over the virtual private gateway

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 between instances in your VPC when you use a different security group, you must add a rule like the following to your security groups.

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

Implementing Scenario 3

Use the following process to implement scenario 3 using the VPC wizard.

To prepare your customer gateway

1. Determine the appliance 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 Virtual Private Cloud Network Administrator Guide](#).
2. Obtain the Internet-routable IP address for the customer gateway's external interface. The address must be static and can't be behind a device performing network address translation (NAT).
3. Gather the list of internal IP ranges (in CIDR notation) that should be advertised across the VPN connection to the virtual private gateway (if you are using a statically routed VPN connection). For more information, see [VPN Routing Options \(p. 121\)](#).

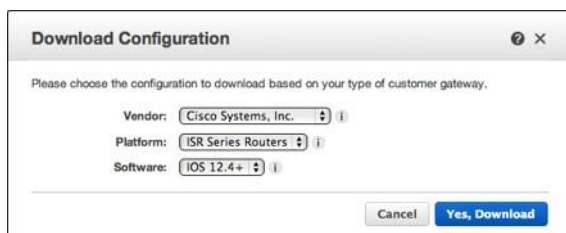
To implement scenario 3 using the VPC wizard

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **VPC Dashboard**.
3. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
4. Select the third option, **VPC with Public and Private Subnets and Hardware VPN Access**, and then click **Select**.
5. On the first page of the wizard, confirm the details for your VPC, public and private subnets, and then click **Next**.
6. On the **Configure your VPN** page, do the following, and then click **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, click **Static**. In **IP Prefix**, add each IP prefix for your network.

For more information about which option to choose, see [Amazon Virtual Private Cloud FAQs](#). For more information about dynamic versus static routing, see [VPN Routing Options \(p. 121\)](#).

7. When the wizard is done, click **VPN Connections** in the navigation pane. Select the VPN connection that the wizard created, and click **Download Configuration**. In the dialog box, select the vendor for the customer gateway, the platform, and the software version, and then click **Yes, Download**.



8. Save the text file containing the VPN configuration and give it to the network administrator along with this guide: [Amazon Virtual Private Cloud Network Administrator Guide](#). The VPN won't work until the network administrator configures the customer gateway.

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, click **Security Groups**.
3. Click the **Create Security Group** button.
4. In the **Create Security Group** dialog box, specify `WebServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
5. Click the **Create Security Group** button again.
6. In the **Create Security Group** dialog box, specify `DBServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.

To add the recommended 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, click **Edit** and add rules for inbound traffic as follows:
 - a. Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
 - b. Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
 - c. Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
 - d. Click **Add another rule**, then select **RDP** from the **Type** list. Enter your network's public IP address range in the **Source** field.
 - e. Click **Save**.

Type	Protocol	Port Range	Source	Remove
HTTP (80)	TCP (6)	80	0.0.0.0/0	
HTTPS (443)	TCP (6)	443	0.0.0.0/0	
SSH (22)	TCP (6)	22	192.0.2.0/24	
RDP (3389)	TCP (6)	3389	192.0.2.0/24	

Add another rule

3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:
 - a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
 - b. Select **MS SQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
 - c. Click **Add another rule**, then select **MySQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
 - d. Click **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, click **Edit** and add rules for inbound traffic as follows:
 - a. Select **SSH** from the **Type** list, and enter the IP address range of your network in the **Source** field.
 - b. Click **Add another rule**, then select **RDP** from the **Type** list, and enter the IP address range of your network in the **Source** field.
 - c. Click **Add another rule**, then select **MS SQL** from the **Type** list. Specify the ID of your WebServerSG security group in the **Source** field.
 - d. Click **Add another rule**, then select **MYSQL** from the **Type** list. Specify the ID of your WebServerSG security group in the **Source** field.
 - e. Click **Save**.
3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:
 - a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
 - b. Select **HTTP** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
 - c. Click **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter 0.0.0.0/0.
 - d. Click **Save**.

After your network administrator configures your customer gateway, you can launch instances into your VPC. If you're already familiar with launching instances outside a VPC, then you already know most of what you need to know to launch an instance into a VPC.

To launch an instance (web server or database server)

1. Create the WebServerSG and DBServerSG security groups if you haven't done so already (see [Security for Scenario 3 \(p. 25\)](#)). You'll specify one of these security groups when you launch the instance.
2. Start the launch wizard:
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. Click the **Launch Instance** button from the dashboard.
3. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
4. 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.
5. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IP address. To be able to connect to your instance, you can assign a public IP 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 the **Public IP** check box is selected.

Note

You can only assign a public IP address to a single, new network interface with the device index of eth0. For more information, see [Assigning a Public IP Address During Launch \(p. 86\)](#).

6. 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 a security group for the instance (**WebServerSG** for a web server or **DBServerSG** for a database server). Click **Review and Launch**.
7. Review the settings that you've chosen. Make any changes that you need, and then click **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 End-to-End Connectivity of Your Instance \(p. 127\)](#).

If you did not assign a public IP address to your instance 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, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. From the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.
5. Select the Elastic IP address from the list, and then click the **Associate Address** button.
6. In the **Associate Address** dialog box, select the network interface or instance. Select the address to associate the Elastic IP address with from the corresponding **Private IP address** list, and then click **Yes, 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, click **DHCP Options Sets**.
3. Click the **Create DHCP Options Set** button.
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, separated by a comma, and then click **Yes, Create**. In this example, your DNS server is 192.0.2.1.
5. In the navigation pane, click **Your VPCs**.
6. Select the VPC, and then click the **Edit** button in the **Summary** tab.
7. Select the ID of the new set of options from the **DHCP options set** list and then click **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.

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 Elastic Compute Cloud User Guide*. For information about how to connect to a Windows instance, see [Connect to Your Windows Instance](#) in the *Amazon Elastic Compute Cloud Microsoft Windows Guide*.

Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access

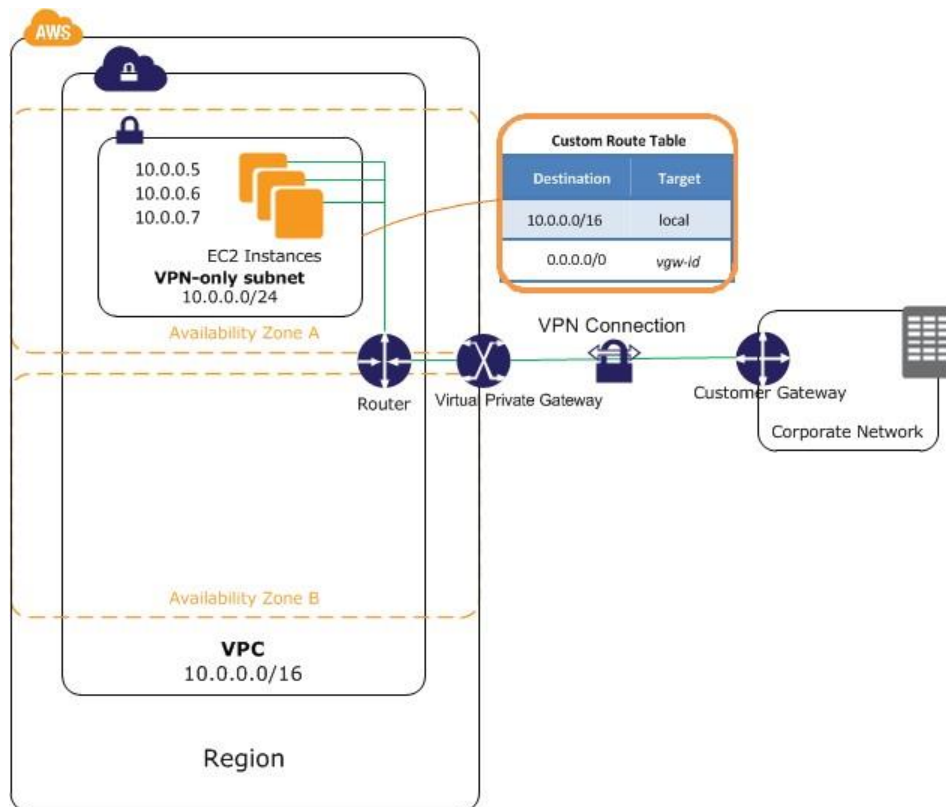
The configuration for this scenario includes a virtual private cloud (VPC) with a single private subnet, and a virtual private gateway to enable communication with your own network over an IPsec VPN tunnel. There is no Internet gateway to enable communication over the Internet. We recommend this scenario if you want to extend your network into [the cloud](#) using Amazon's infrastructure without exposing your network to the Internet.

Topics

- [Configuration for Scenario 4 \(p. 31\)](#)
- [Basic Components for Scenario 4 \(p. 32\)](#)
- [Routing for Scenario 4 \(p. 33\)](#)
- [Security for Scenario 4 \(p. 33\)](#)
- [Implementing Scenario 4 \(p. 34\)](#)

Configuration for Scenario 4

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



Important

For this scenario, the [Amazon Virtual Private Cloud 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.

Basic Components for Scenario 4

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A VPN-only subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- 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 IP addresses in the subnet range (examples: 10.0.0.5, 10.0.0.6, and 10.0.0.7), which enables the instances to communicate with each other and other instances in the VPC.
- A route table entry that enables instances in the subnet to communicate with other instances in the VPC, and a route table entry that enables instances in the subnet to communicate directly with your network.

For more information about subnets, see [Your VPC and Subnets \(p. 37\)](#) and [IP Addressing in Your VPC \(p. 85\)](#). For more information about your VPN connection, see [Adding a Hardware Virtual Private Gateway to Your VPC \(p. 119\)](#). For more information about configuring a customer gateway, see the [Amazon Virtual Private Cloud Network Administrator Guide](#).

Routing for Scenario 4

Your VPC has an implied router (shown in the configuration diagram for this scenario.) For this scenario, the VPC wizard creates a route table that routes all traffic destined for an address outside the VPC to the VPN connection, and associates the route table with the subnet. Otherwise, you'd need to create and associate the route table yourself.

The following table shows what the route table looks like for the example addresses used in the configuration diagram for this scenario. The first row describes the entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic to the virtual private gateway, which is specified using its AWS-assigned identifier (for example, `vgw-1a2b3c4d`).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	vgw-xxxxxxx

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 your VPC through BGP.

The instances in your VPC 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 Amazon S3 or Amazon EC2), the requests must go over the virtual private gateway to your network and then to the Internet before reaching AWS.

Security for Scenario 4

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see [Security in Your VPC \(p. 51\)](#).

For scenario 4, you'll use the default security group for your VPC but not network ACLs. If you'd like to use a network ACL, see [Recommended Rules for Scenario 4 \(p. 74\)](#).

Recommended Security Group Rules

Your VPC comes with a default security group whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between the instances assigned to the security group. We recommend that you add inbound rules to the default security group to allow SSH traffic (Linux) and Remote Desktop traffic (Windows) from your network.

Important

The default security group automatically allows assigned instances to communicate with each other, so you don't have to add a rule to allow this. If you use a different security group, you must add a rule to allow this.

The following table describes the inbound rules that you should add to the default security group for your VPC.

Default Security Group: Recommended Rules

Inbound			
Source	Protocol	Port Range	Comments
Private IP address range of your network	TCP	22	(Linux instances) Allow inbound SSH traffic from your network
Private IP address range of your network	TCP	3389	(Windows instances) Allow inbound RDP traffic from your network

Implementing Scenario 4

Use the following process to implement scenario 4 using the VPC wizard.

To prepare your customer gateway

1. Determine the appliance you'll use as your customer gateway. For 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 Virtual Private Cloud Network Administrator Guide](#).
2. Obtain the Internet-routable IP address for the customer gateway's external interface. The address must be static and can't be behind a device performing network address translation (NAT).
3. Gather the list of internal IP ranges (in CIDR notation) that should be advertised across the VPN connection to the virtual private gateway (if you are using a statically routed VPN connection). For more information, see [VPN Routing Options \(p. 121\)](#).

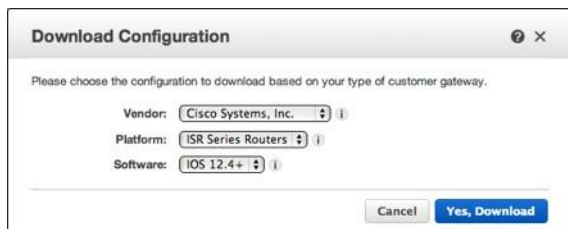
Next, use the VPC wizard as described in the following procedure to create your VPC and a VPN connection.

To implement scenario 4 using the VPC wizard

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **VPC Dashboard**.
3. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
4. Select the fourth option, **VPC with a Private Subnet Only and Hardware VPN Access**, and then click **Select**.
5. On the first page of the wizard, confirm the details for your VPC and private subnet, and then click **Next**.
6. On the **Configure your VPN** page, do the following, and then click **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, click **Static**. In **IP Prefix**, add each IP prefix for your network.

For more information about which option to choose, see [Amazon Virtual Private Cloud FAQs](#). For more information about dynamic versus static routing, see [VPN Routing Options](#) (p. 121).

- When the wizard is done, click **VPN Connections** in the navigation pane. Select the VPN connection that the wizard created, and click **Download Configuration**. In the dialog box, select the vendor for the customer gateway, the platform, and the software version, and then click **Yes, Download**.

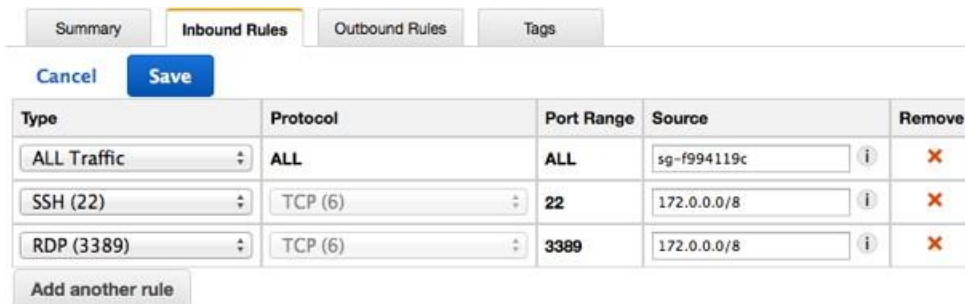


- Save the text file containing the VPN configuration and give it to the network administrator along with this guide: [Amazon Virtual Private Cloud Network Administrator Guide](#). The VPN won't work until the network administrator configures the customer gateway.

For this scenario, you need to update the default security group with new inbound rules that allow SSH and Remote Desktop (RDP) access from your network. If the instances won't initiate outbound communication, we can also remove the default outbound rule. Reminder: the initial settings of the default security group block all inbound traffic, allow all outbound traffic, and allow instances assigned to the group to communicate with each other.

To update the rules for the default security group

- Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
- Click **Security Groups** in the navigation pane, and then select the default security group for the VPC. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
- On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:
 - Select **SSH** from the **Type** list, and enter your network's private IP address range in the **Source** field.
 - Click **Add another rule**, then select **RDP** from the **Type** list, and enter your network's private IP address range in the **Source** field.
 - Click **Save**.



Type	Protocol	Port Range	Source	Remove
ALL Traffic	ALL	ALL	sg-f994119c	X
SSH (22)	TCP (6)	22	172.0.0.0/8	X
RDP (3389)	TCP (6)	3389	172.0.0.0/8	X

4. On the **Outbound Rules** tab, click **Edit**, locate the default rule that enables all outbound traffic, click **Remove**, and then click **Save**.

After your network administrator configures your customer gateway, you can launch instances into your VPC. If you're already familiar with launching instances outside a VPC, then you already know most of what you need to know to launch an instance into a VPC.

To launch an instance

1. Start the launch wizard:
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. Click the **Launch Instance** button from the dashboard.
2. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
3. On the **Configure Instance Details** page, select the VPC that you created earlier from the **Network** list, and then select a subnet. Click **Next: Add Storage**.
4. 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 default security group. Click **Review and Launch**.
5. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a keypair and launch your instance.

In scenario 4, you need a DNS server that enables your VPN-only subnet to communicate with servers in your network. You must create a new set of DHCP options that includes your DNS server and then configure the VPC to use that set of options.

Note

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. Scenario 4 doesn't have any public subnets, so you don't need this 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, click **DHCP Options Sets**.
3. Click the **Create DHCP Options Set** button.
4. In the **Create DHCP Options Set** dialog box, in the **Domain name servers** box, enter the address of your DNS server, and then click **Yes, Create**. In this example, your DNS server is 192.0.2.1.
5. In the navigation pane, click **Your VPCs**.
6. Select the VPC, and then click the **Edit** button in the **Summary** tab.
7. Select the ID of the new set of options from the **DHCP options set** list and then click **Save**.
8. (Optional) The VPC now uses this new set of DHCP options and therefore uses your DNS server. If you want, you can delete the original set of options that the VPC used.

You can now use SSH or RDP to connect to your instance in the VPC. For information about how to connect to a Linux instance, see [Connect to Your Linux Instance](#) in the *Amazon Elastic Compute Cloud User Guide*. For information about how to connect to a Windows instance, see [Connect to Your Windows Instance](#) in the *Amazon Elastic Compute Cloud Microsoft Windows Guide*.

Your VPC and Subnets

To get started with Amazon Virtual Private Cloud (Amazon VPC), you'll create a VPC and subnets. For a general overview of VPCs and subnets, see [What is Amazon VPC? \(p. 1\)](#).

Topics

- [Your VPC \(p. 37\)](#)
- [Subnets in Your VPC \(p. 40\)](#)
- [CLI Overview \(p. 44\)](#)

Your VPC

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. When you create a VPC, you specify the set of IP addresses for the VPC in the form of a Classless Inter-Domain Routing (CIDR) block (for example, `10.0.0.0/16`). For more information about CIDR notation and what `/16` means, see [Classless Inter-Domain Routing](#) on Wikipedia.

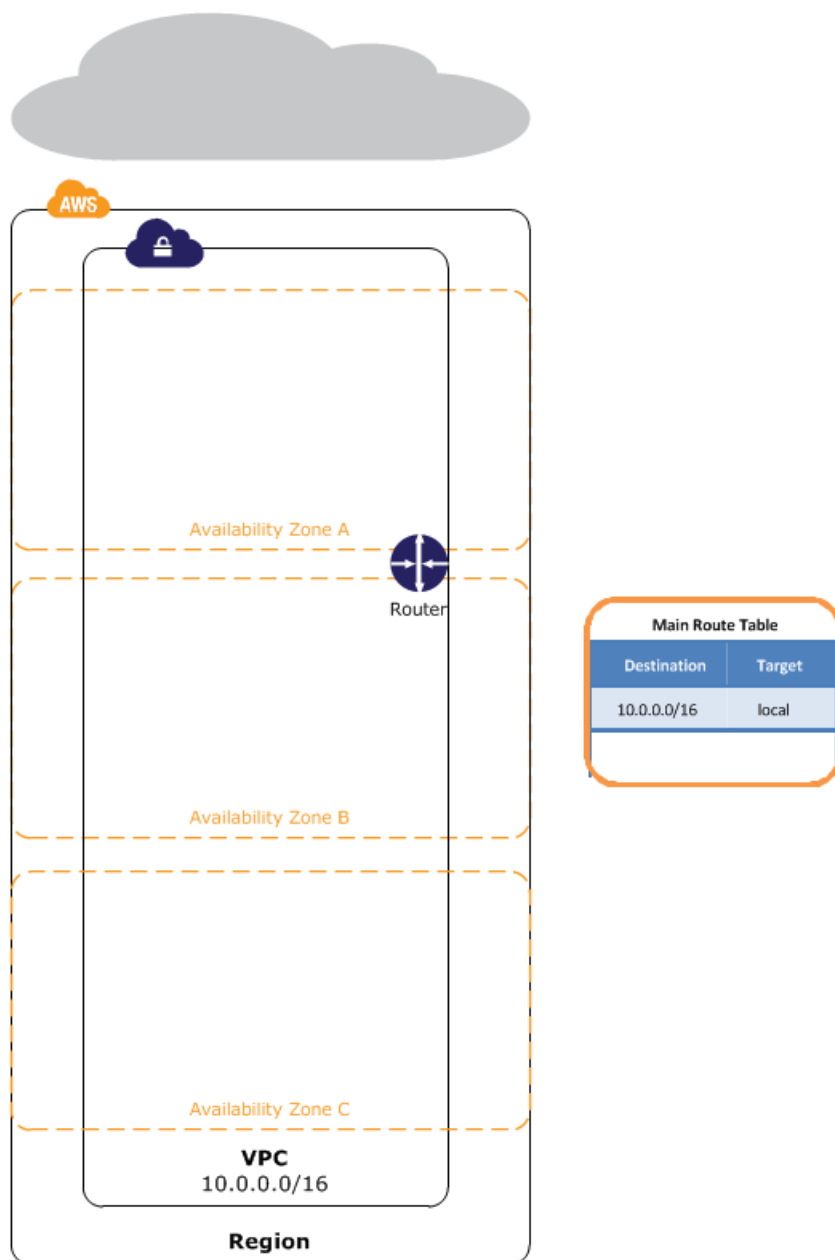
For information about the number of VPCs that you can create, see [Amazon VPC Limits \(p. 147\)](#).

Topics

- [Your New VPC \(p. 37\)](#)
- [VPC Sizing \(p. 38\)](#)
- [Connections Between Your VPC and Your Corporate or Home Network \(p. 39\)](#)
- [Creating a VPC \(p. 39\)](#)
- [Deleting Your VPC \(p. 40\)](#)

Your New VPC

The following diagram shows a new VPC with a default route table.



You need to add a subnet before you can launch an instance into your VPC.

VPC Sizing

You can assign a single CIDR block to a VPC. The allowed block size is between a /28 netmask and /16 netmask. In other words, the VPC can contain from 16 to 65,536 IP addresses. You can't change the size of a VPC after you create it. If your VPC is too small to meet your needs, you must terminate all the instances in the VPC, delete the VPC, and then create a new, larger VPC. For more information, see [Deleting Your VPC](#) (p. 40).

Connections Between Your VPC and Your Corporate or Home Network

You can optionally set up a connection between your VPC and your corporate or home network. If you have an IP address prefix in your VPC that overlaps with one of your networks' prefixes, any traffic to the network's prefix is dropped. For example, let's say that you have the following:

- A VPC with CIDR block `10.0.0.0/16`
- A subnet in that VPC with CIDR block `10.0.1.0/24`
- Instances running in that subnet with IP addresses `10.0.1.4` and `10.0.1.5`
- On-premises host networks using CIDR blocks `10.0.37.0/24` and `10.1.38.0/24`

When those instances in the VPC try to talk to hosts in the `10.0.37.0/24` address space, the traffic is dropped because `10.0.37.0/24` is part of the larger prefix assigned to the VPC (`10.0.0.0/16`). The instances can talk to hosts in the `10.1.38.0/24` space because that block isn't part of `10.0.0.0/16`.

We therefore recommend you create a VPC with a CIDR range large enough for expected future growth, but not one that overlaps with current or expected future subnets anywhere in your corporate or home network.

Creating a VPC

There are two ways to create a VPC using the Amazon VPC console: the **Create VPC** dialog box and the VPC wizard. The following procedure uses the **Create VPC** dialog box, which creates only the VPC; you'd need to subsequently add subnets, gateways, and routing tables. For information about using the VPC wizard to create a VPC plus its subnets, gateways, and routing tables in one step, see [Scenarios for Amazon VPC \(p. 7\)](#).

Note

(EC2-Classic) If you use the launch wizard in the Amazon EC2 console to launch a T2 instance type and you do not have any existing VPCs, the wizard creates a nondefault VPC for you, with a subnet in each Availability Zone, an Internet gateway, and a route table that routes all VPC traffic to the Internet gateway. For more information about T2 instance types, see [T2 Instances](#) in the *Amazon Elastic Compute Cloud User Guide*.

To create a VPC

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Your VPCs**.
3. Click **Create VPC**.
4. In the **Create VPC** dialog box, specify the following VPC details as necessary, then click **Yes, Create**.
 - Optionally provide a name for your VPC. Doing so creates a tag with a key of `Name` and a value that you specify.
 - Specify a CIDR block for the VPC.
 - Select a tenancy option, for example, a dedicated tenancy that ensures your instances run on single-tenant hardware. For more information about dedicated instances, see [Dedicated Instances \(p. 133\)](#).

Deleting Your VPC

You can delete your VPC at any time (for example, if you decide it's too small). However, you must terminate all instances in the VPC first. When you delete a VPC using the VPC console, we delete all its components, such as subnets, security groups, network ACLs, route tables, Internet gateways, VPC peering connections, and DHCP options.

If you have a VPN connection, you don't have to delete it or the other components related to the VPN (such as the customer gateway and virtual private gateway). If you plan to use the customer gateway with another VPC, we recommend you keep the VPN connection and the gateways. Otherwise, your network administrator must configure the customer gateway again after you create a new VPN connection.

To delete your VPC

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Terminate all instances in the VPC.
3. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
4. In the navigation pane, click **Your VPCs**.
5. Select the VPC to delete, and then click **Delete**.
6. If you need to delete the VPN connection, select the option to do so; otherwise, leave it unselected. Click **Yes, Delete**.

Subnets in Your VPC

You can create a VPC that spans multiple Availability Zones. For more information, see [Creating a VPC \(p. 39\)](#). After creating a VPC, you can add one or more subnets in each Availability Zone. Each subnet must reside entirely within one Availability Zone and cannot span zones. Availability Zones are distinct locations that are engineered to be isolated from failures in other Availability Zones. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location. AWS assigns a unique ID to each subnet.

For information about the number of subnets that you can create, see [Amazon VPC Limits \(p. 147\)](#).

Topics

- [Your VPC with Subnets \(p. 40\)](#)
- [Subnet Sizing \(p. 42\)](#)
- [Subnet Routing \(p. 42\)](#)
- [Subnet Security \(p. 42\)](#)
- [Adding a Subnet to Your VPC \(p. 43\)](#)
- [Launching an Instance into Your Subnet \(p. 43\)](#)
- [Deleting Your Subnet \(p. 44\)](#)

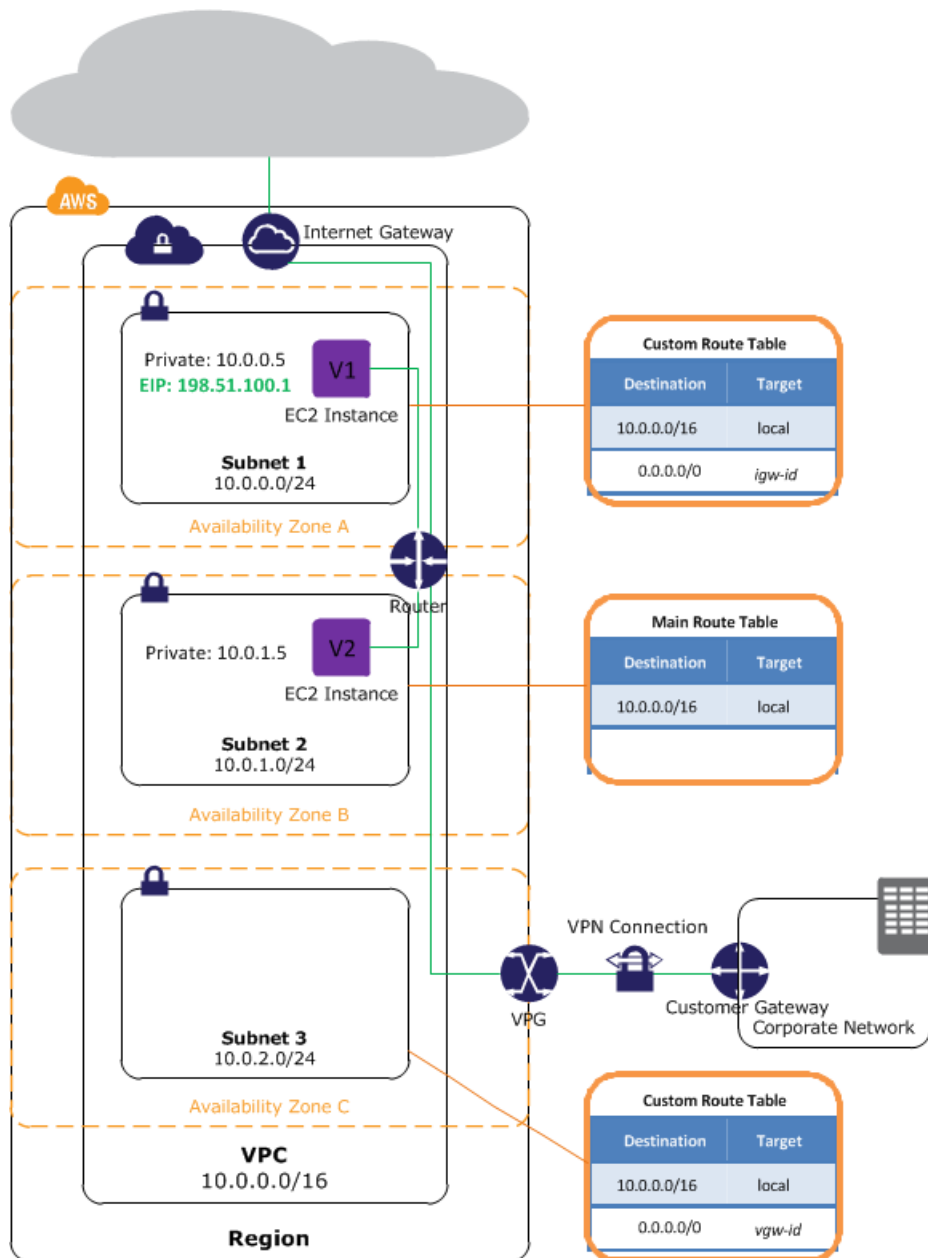
Your VPC with Subnets

The following diagram shows a VPC that has been configured with subnets in multiple Availability Zones. You can optionally add an Internet gateway to enable communication over the Internet, or a virtual private network (VPN) connection to enable communication with your network, as shown in the diagram.

If a subnet's traffic is routed to an Internet gateway, the subnet is known as a *public subnet*. In this diagram, subnet 1 is a public subnet.

If a subnet doesn't have a route to the Internet gateway, the subnet is known as a *private subnet*. In this diagram, subnet 2 is a private subnet.

If a subnet doesn't have a route to the Internet gateway, but has its traffic routed to a virtual private gateway, the subnet is known as a *VPN-only subnet*. In this diagram, subnet 3 is a VPN-only subnet.



For more information, see [Scenarios for Amazon VPC \(p. 7\)](#), [Adding an Internet Gateway to Your VPC \(p. 101\)](#), or [Adding a Hardware Virtual Private Gateway to Your VPC \(p. 119\)](#).

Instances that you launch into a nondefault subnet do not receive a public IP address by default. However, you can change your subnet's default public IP addressing behavior. For more information, see [Modifying Your Subnet's Public IP Addressing Behavior \(p. 87\)](#).

Subnet Sizing

When you create a subnet, you specify the CIDR block for the subnet. The CIDR block of a subnet can be the same as the CIDR block for the VPC (for a single subnet in the VPC), or a subset (to enable multiple subnets). If you create more than one subnet in a VPC, the CIDR blocks of the subnets must not overlap.

For example, if you create a VPC with CIDR block `10.0.0.0/24`, it supports 256 IP addresses. You can break this CIDR block into two subnets, each supporting 128 IP addresses. One subnet uses CIDR block `10.0.0.0/25` (for addresses `10.0.0.0 - 10.0.0.127`) and the other uses CIDR block `10.0.0.128/25` (for addresses `10.0.0.128 - 10.0.0.255`).

Important

AWS reserves both the first four IP addresses and the last IP address in each subnet CIDR block. They're not available for you to use.

There are many tools available to help you calculate subnet CIDR blocks. For information about a commonly used tool, see <http://www.subnet-calculator.com/cidr.php>. Also, your network engineering group can help you determine the CIDR blocks to specify for your subnets.

Subnet Routing

By design, each subnet must be associated with a route table, which specifies the allowed routes for outbound traffic leaving the subnet. Every subnet that you create is automatically associated with the main route table for the VPC. You can change the association, and you can change the contents of the main route table. For more information, see [Route Tables \(p. 91\)](#).

In the previous diagram, the route table associated with subnet 1 routes all traffic (`0.0.0.0/0`) to an Internet gateway (for example, `igw-1a2b3c4d`). Because instance V1 has an Elastic IP address, it can be reached from the Internet.

Note

The Elastic IP address or public IP address that's associated with your instance is accessed through the Internet gateway of your VPC. Traffic that goes through a VPN connection between your instance and another network traverses a virtual private gateway, not the Internet gateway, and therefore does not access the Elastic IP address or public IP address.

The instance V2 can't reach the Internet, but can reach other instances in the VPC. You can allow an instance in your VPC to initiate outbound connections to the Internet but prevent unsolicited inbound connections from the Internet using a network address translation (NAT) instance. Because you can allocate a limited number of Elastic IP addresses, we recommend that you use a NAT instance if you have more instances that require a static public IP address. For more information, see [NAT Instances \(p. 105\)](#).

The route table associated with subnet 3 routes all traffic (`0.0.0.0/0`) to a virtual private gateway (for example, `vgw-1a2b3c4d`).

Subnet Security

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see [Security in Your VPC \(p. 51\)](#).

By design, each subnet must be associated with a network ACL. Every subnet that you create is automatically associated with the VPC's default network ACL. You can change the association, and you can change the contents of the default network ACL. For more information, see [Network ACLs \(p. 59\)](#).

Adding a Subnet to Your VPC

When you add a new subnet to your VPC, you must set up the routing and security that you want for the subnet. You can do this manually, as described in this section, or let the VPC wizard set things up for you, as described in [Scenarios for Amazon VPC \(p. 7\)](#).

To add a subnet to your VPC

1. Create the subnet.
 - a. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 - b. In the navigation pane, click **Subnets**.
 - c. Click **Create Subnet**.
 - d. In the **Create Subnet** dialog box, optionally name your subnet, and then select the VPC, select the Availability Zone, specify the CIDR range for the subnet, and then click **Yes, Create**.
2. Set up routing for the subnet. For example, you can add a route to an Internet gateway or a NAT instance. For more information, see [Route Tables \(p. 91\)](#).
3. (Optional) Create or modify your security groups as needed. For more information, see [Security Groups for Your VPC \(p. 53\)](#).
4. (Optional) Create or modify your network ACLs as needed. For more information about network ACLs, see [Network ACLs \(p. 59\)](#).

Launching an Instance into Your Subnet

To launch an instance into your subnet

1. Start the launch wizard:
 - a. Open the Amazon EC2 console.
 - b. On the dashboard, click **Launch Instance**.
2. Follow the directions in the wizard. Select an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
3. On the **Configure Instance Details** page, ensure you have selected the required VPC in the **Network** list, then select the subnet to launch the instance into. Keep the other default settings on this page and click **Next: Add Storage**.
4. On the next pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, choose from any existing security group that you own, or follow the wizard directions to create a new security group. Click **Review and Launch** when you're done.
5. Review your settings and click **Launch**.
6. Choose an existing key pair that you own, or create a new one, then click **Launch Instances** when you're done.

Deleting Your Subnet

You must terminate any instances in the subnet first.

To delete your subnet

1. Open the Amazon EC2 console.
2. Terminate all instances in the subnet.
3. Open the Amazon VPC console.
4. In the navigation pane, click **Subnets**.
5. Select the subnet to delete and click **Delete**.
6. In the **Delete Subnet** dialog box, click **Yes, Delete**.

CLI Overview

You can perform the tasks described on this page using a command line interface (CLI). For more information, including a list of available API actions, see [Accessing Amazon VPC \(p. 5\)](#).

Create a VPC

- [create-vpc](#) (AWS CLI)
- [ec2-create-vpc](#) (Amazon EC2 CLI)
- [New-EC2Vpc](#) (AWS Tools for Windows PowerShell)

Create a Subnet

- [create-subnet](#) (AWS CLI)
- [ec2-create-subnet](#) (Amazon EC2 CLI)
- [New-EC2Subnet](#) (AWS Tools for Windows PowerShell)

Describe a VPC

- [describe-vpcs](#) (AWS CLI)
- [ec2-describe-vpcs](#) (Amazon EC2 CLI)
- [Get-EC2Vpc](#) (AWS Tools for Windows PowerShell)

Describe a Subnet

- [describe-subnets](#) (AWS CLI)
- [ec2-describe-subnets](#) (Amazon EC2 CLI)
- [Get-EC2Subnet](#) (AWS Tools for Windows PowerShell)

Delete a VPC

- [delete-vpc](#) (AWS CLI)
- [ec2-delete-vpc](#) (Amazon EC2 CLI)
- [Remove-EC2Vpc](#) (AWS Tools for Windows PowerShell)

Delete a Subnet

- [delete-subnet](#) (AWS CLI)
- [ec2-delete-subnet](#) (Amazon EC2 CLI)
- [Remove-EC2Subnet](#) (AWS Tools for Windows PowerShell)

Your Default VPC and Subnets

A default VPC combines the benefits of the advanced networking features provided by the EC2-VPC platform with the ease of use of the EC2-Classic platform.

For more information about the EC2-Classic and EC2-VPC platforms, see [Supported Platforms](#).

Topics

- [Default VPC Basics \(p. 46\)](#)
- [Detecting Your Supported Platforms and Whether You Have a Default VPC \(p. 48\)](#)
- [Launching an EC2 Instance into Your Default VPC \(p. 49\)](#)
- [Deleting Your Default VPC \(p. 50\)](#)

Default VPC Basics

This section provides information about your default virtual private cloud (VPC) and its default subnets.

Availability

If you created your AWS account after 2013-12-04, it supports only EC2-VPC. In this case, we create a default VPC for you in each AWS region. Therefore, unless you create a nondefault VPC and specify it when you launch an instance, we launch your instances into your default VPC.

If you created your AWS account before 2013-03-18, it supports both EC2-Classic and EC2-VPC in regions that you've used before, and only EC2-VPC in regions that you haven't used. In this case, we create a default VPC in each region in which you haven't created any AWS resources. Therefore, unless you create a nondefault VPC and specify it when you launch an instance in a region that you haven't used before, we launch the instance into your default VPC for that region. However, if you launch an instance in a region that you've used before, we launch the instance into EC2-Classic.

If you created your AWS account between 2013-03-18 and 2013-12-04, it may support only EC2-VPC, or it may support both EC2-Classic and EC2-VPC in some of the regions that you've used. For information about detecting the platform support in each region for your AWS account, see [Detecting Your Supported Platforms and Whether You Have a Default VPC \(p. 48\)](#). For information about when each region was enabled for default VPCs, see [Announcement: Enabling regions for the default VPC feature set](#) in the AWS forum for Amazon VPC.

If an AWS account supports only EC2-VPC, any IAM accounts associated with this AWS account also support only EC2-VPC, and use the same default VPC as the AWS account.

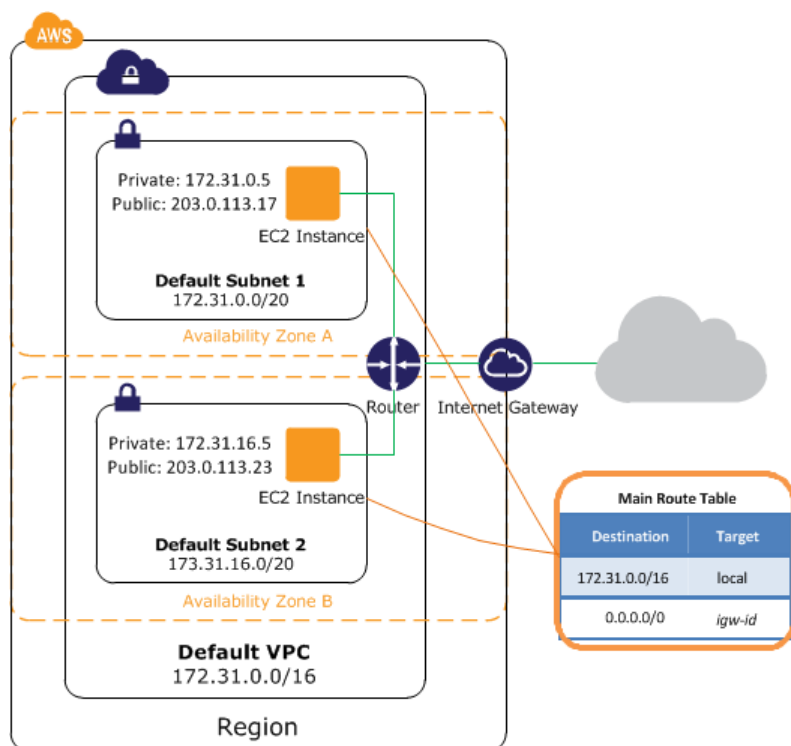
If your AWS account supports both EC2-Classic and EC2-VPC and you want the benefits of using EC2-VPC with the simplicity of launching instances into EC2-Classic, you can either create a new AWS account or launch your instances into a region that you haven't used before. If you'd prefer to add a default VPC to a region that doesn't have one, see "I really want a default VPC for my existing EC2 account. Is that possible?" in the [Default VPCs FAQ](#).

Components

When we create a default VPC, we do the following to set it up for you:

- Create a default subnet in each Availability Zone.
- Create an Internet gateway and connect it to your default VPC.
- Create a main route table for your default VPC with a rule that sends all traffic destined for the Internet to the Internet gateway.
- Create a default security group and associate it with your default VPC.
- Create a default network access control list (ACL) and associate it with your default VPC.
- Associate the default DHCP options set for your AWS account with your default VPC.

The following figure illustrates the key components that we set up for a default VPC.



Instances that you launch into a default subnet receive both a public IP address and a private IP address. Instances in a default subnet also receive both public and private DNS hostnames. Instances that you launch into a nondefault subnet in a default VPC don't receive a public IP address or a DNS hostname. You can change your subnet's default public IP addressing behavior. For more information, see [Modifying Your Subnet's Public IP Addressing Behavior](#) (p. 87).

You can use a default VPC as you would use any other VPC; you can add subnets, modify the main route table, add additional route tables, associate additional security groups, update the rules of the default security group, and add VPN connections. You can also create additional VPCs.

You can use a default subnet as you would use any other subnet; you can add custom route tables and set network ACLs. You can also specify a default subnet when you launch an EC2 instance.

Default Subnets

The CIDR block for a default VPC is always 172.31.0.0/16. This provides up to 65,536 private IP addresses. The netmask for a default subnet is always /20, which provides up to 4,096 addresses per subnet, a few of which are reserved for our use.

By default, a default subnet is a public subnet, because the main route table sends the subnet's traffic that is destined for the Internet to the Internet gateway. You can make a default subnet a private subnet by removing the route from the destination 0.0.0.0/0 to the Internet gateway. However, if you do this, any EC2 instance running in that subnet can't access the Internet or other AWS products, such as Amazon Simple Storage Service (Amazon S3).

Detecting Your Supported Platforms and Whether You Have a Default VPC

You can launch EC2 instances into a default VPC and use services such as Elastic Load Balancing, Amazon Relational Database Service (Amazon RDS), and Amazon Elastic MapReduce (Amazon EMR) without needing to know anything about Amazon VPC. Your experience with these services is the same whether you are using a default VPC or EC2-Classic. However, you can use the Amazon EC2 console or the command line to determine whether your AWS account supports both platforms and if you have a default VPC.

Detecting Platform Support Using the Console

The Amazon EC2 console indicates which platforms you can launch EC2 instances into, and whether you have a default VPC.

Verify that the region you'll use is selected in the navigation bar. On the Amazon EC2 console dashboard, look for **Supported Platforms** under **Account Attributes**. If there are two values, `EC2` and `VPC`, you can launch instances into either platform. If there is one value, `VPC`, you can launch instances only into EC2-VPC.

For example, the following indicates that the account supports the EC2-VPC platform only, and has a default VPC with the identifier `vpc-1a2b3c4d`.

```
Supported Platforms
VPC

Default VPC
vpc-1a2b3c4d
```

If you delete your default VPC, the **Default VPC** value displayed is `None`. For more information, see [Deleting Your Default VPC \(p. 50\)](#).

Detecting Platform Support Using the Command Line

The `supported-platforms` attribute indicates which platforms you can launch EC2 instances into. To get the value of this attribute for your account, use one of the following commands:

- `describe-account-attributes` (AWS CLI)
- `ec2-describe-account-attributes` (Amazon EC2 CLI)
- `Get-EC2AccountAttributes` (AWS Tools for Windows PowerShell)

Also, when you list your VPCs using the following commands, we indicate any default VPCs in the output:

- `describe-vpcs` (AWS CLI)
- `ec2-describe-vpcs` (Amazon EC2 CLI)
- `Get-EC2Vpc` (AWS Tools for Windows PowerShell)

Launching an EC2 Instance into Your Default VPC

When you launch an EC2 instance without specifying a subnet, it's automatically launched into a default subnet in your default VPC. By default, we select an Availability Zone for you and launch the instance into the corresponding subnet for that Availability Zone. Alternatively, you can select the Availability Zone for your instance by selecting its corresponding default subnet in the console, or by specifying the subnet or the Availability Zone in the CLI.

Launching an EC2 Instance Using the Console

To launch an EC2 instance into your default VPC

1. Open the Amazon EC2 console.
2. From the console dashboard, click **Launch Instance**.
3. Follow the directions in the wizard. Select an AMI, and choose an instance type. You can accept the default settings for the rest of the wizard by clicking **Review and Launch**. This takes you directly to the **Review Instance Launch** page.
4. Review your settings. In the **Instance Details** section, the default for **Subnet** is **No preference (default subnet in any Availability Zone)**. This means that the instance is launched into the default subnet of the Availability Zone that we select. Alternatively, you can click **Edit instance details** and select the default subnet for a particular Availability Zone.
5. Click **Launch** to choose a key pair and launch the instance.

Launching an EC2 Instance Using the Command Line

You can use one of the following commands to launch an EC2 instance:

- `run-instances` (AWS CLI)
- `ec2-run-instances` (Amazon EC2 CLI)

- [New-EC2Instance](#) (AWS Tools for Windows PowerShell)

To launch an EC2 instance into your default VPC, use these commands without specifying a subnet or an Availability Zone.

To launch an EC2 instance into a specific default subnet in your default VPC, specify its subnet ID or Availability Zone.

Deleting Your Default VPC

You can delete one or more of your default subnets just as you can delete any other subnet. However, after you've deleted a default subnet, it's gone. Now, you can't launch EC2 instances into that Availability Zone in your default VPC, unless you create a subnet in that Availability Zone and explicitly launch instances into that subnet. If you delete all default subnets for your default VPC, then you must specify a subnet in another VPC when you launch an EC2 instance, because you can't launch instances into EC2-Classic. For more information, see [Deleting Your Subnet \(p. 44\)](#).

If you try to delete your default subnet, the **Delete Subnet** dialog box displays a warning and requires you to acknowledge that you are aware that you are deleting a default subnet.

You can delete a default VPC just as you can delete any other VPC. However, after you've deleted your default VPC, it's gone. Now, you must specify a subnet in another VPC when you launch an EC2 instance, because you can't launch instances into EC2-Classic. If you try to delete your default VPC, the **Delete VPC** dialog box displays a warning and requires you to acknowledge that you are aware that you are deleting a default VPC. For more information, see [Deleting Your VPC \(p. 40\)](#).

If you delete your default VPC and then need to restore it, you can contact AWS Support to create a new default VPC for you.

Security in Your VPC

Amazon VPC provides two features that you can use to increase security for your VPC:

- Security groups—Act as a firewall for associated Amazon EC2 instances, controlling both inbound and outbound traffic at the instance level
- Network access control lists (ACLs)—Act as a firewall for associated subnets, controlling both inbound and outbound traffic at the subnet level

When you launch an instance in a VPC, you can associate one or more security groups that you've created. Each instance in your VPC could belong to a different set of security groups. If you don't specify a security group when you launch an instance, the instance automatically belongs to the default security group for the VPC. For more information about security groups, see [Security Groups for Your VPC \(p. 53\)](#)

You can secure your VPC instances using only security groups; however, you can add network ACLs as a second layer of defense. For more information about network ACLs, see [Network ACLs \(p. 59\)](#).

You can use AWS Identity and Access Management to control who in your organization has permission to create and manage security groups and network ACLs. For example, you can give only your network administrators that permission, but not personnel who only need to launch instances. For more information, see [Controlling Access to Amazon VPC Resources \(p. 75\)](#).

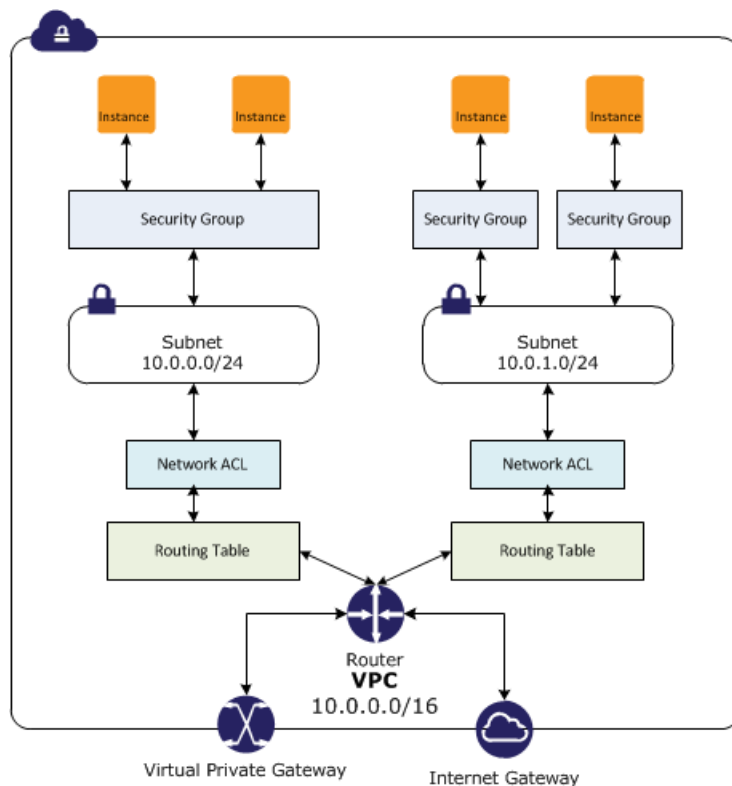
Amazon security groups and network ACLs don't filter traffic to or from link-local addresses (169.254.0.0/16) or AWS reserved addresses (the first four IP addresses and the last one in each subnet). These addresses support the services: Domain Name Services (DNS), Dynamic Host Configuration Protocol (DHCP), Amazon EC2 instance metadata, Key Management Server (KMS—license management for Windows instances), and routing in the subnet. You can implement additional firewall solutions in your instances to block network communication with link-local addresses.

Comparison of Security Groups and Network ACLs

The following table summarizes the basic differences between security groups and network ACLs.

Security Group	Network ACL
Operates at the instance level (first layer of defense)	Operates at the subnet level (second layer of defense)
Supports allow rules only	Supports allow rules and deny rules
Is stateful: Return traffic is automatically allowed, regardless of any rules	Is stateless: Return traffic must be explicitly allowed by rules
We evaluate all rules before deciding whether to allow traffic	We process rules in number order when deciding whether to allow traffic
Applies to an instance only if someone specifies the security group when launching the instance, or associates the security group with the instance later on	Automatically applies to all instances in the subnets it's associated with (backup layer of defense, so you don't have to rely on someone specifying the security group)

The following diagram illustrates the layers of security provided by security groups and network ACLs. For example, traffic from an Internet gateway is routed to the appropriate subnet using the routes in the routing table. The rules of the network ACL associated with the subnet control which traffic is allowed to the subnet. The rules of the security group associated with an instance control which traffic is allowed to the instance.



Security Groups for Your VPC

A *security group* acts as a virtual firewall for your instance to control inbound and outbound traffic. When you launch an instance in a VPC, you can assign the instance to up to five security groups. Security groups act at the instance level, not the subnet level. Therefore, each instance in a subnet in your VPC could be assigned to a different set of security groups. If you don't specify a particular group at launch time, the instance is automatically assigned to the default security group for the VPC.

For each security group, you add *rules* that control the inbound traffic to instances, and a separate set of rules that control the outbound traffic. This section describes the basic things you need to know about security groups for your VPC and their rules.

You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC. For more information about the differences between security groups and network ACLs, see [Comparison of Security Groups and Network ACLs \(p. 51\)](#).

Topics

- [Security Group Basics \(p. 53\)](#)
- [Default Security Group for Your VPC \(p. 53\)](#)
- [Security Group Rules \(p. 54\)](#)
- [Differences Between Security Groups for EC2-Classic and EC2-VPC \(p. 55\)](#)
- [Working with Security Groups \(p. 56\)](#)
- [API and Command Overview \(p. 99\)](#)

Security Group Basics

The following are the basic characteristics of security groups for your VPC:

- You can create up to 100 security groups per VPC. You can add up to 50 rules to each security group. If you need to apply more than 50 rules to an instance, you can associate up to 5 security groups with each network interface.
- You can specify allow rules, but not deny rules.
- You can specify separate rules for inbound and outbound traffic.
- By default, no inbound traffic is allowed until you add inbound rules to the security group.
- By default, all outbound traffic is allowed until you add outbound rules to the group (and then, you specify the outbound traffic that's allowed).
- Responses to allowed inbound traffic are allowed to flow outbound regardless of outbound rules, and vice versa (security groups are therefore stateful).
- Instances associated with a security group can't talk to each other unless you add rules allowing it (exception: the default security group has these rules by default).
- After you launch an instance, you can change which security groups the instance is associated with.

For information about increasing the limits related to security groups, see [Amazon VPC Limits \(p. 147\)](#).

Default Security Group for Your VPC

Your VPC automatically comes with a default security group. Each EC2 instance that you launch in your VPC is automatically associated with the default security group if you don't specify a different security group when you launch the instance.

The following table describes the default rules for a default security group.

Inbound			
Source	Protocol	Port Range	Comments
The security group ID (sg-xxxxxxx)	All	All	Allow inbound traffic from instances assigned to the same security group
Outbound			
Destination	Protocol	Port Range	Comments
0.0.0.0/0	All	All	Allow all outbound traffic

You can change the rules for the default security group.

Security Group Rules

You can add or remove rules for a security group (also referred to as *authorizing* or *revoking* inbound or outbound access). A rule applies either to inbound traffic (ingress) or outbound traffic (egress). You can grant access to a specific CIDR range, or to another security group in your VPC.

The following are the basic parts of a security group rule:

- (Inbound rules only) The source of the traffic (CIDR range or security group) and the destination port or port range
- (Outbound rules only) The destination for the traffic (CIDR range or security group) and the destination port or port range
- Any protocol that has a standard protocol number (for a list, see [Protocol Numbers](#))

If you specify ICMP as the protocol, you can specify any or all of the ICMP types and codes

When you specify a security group as the source for a rule, this allows instances associated with the source security group to access instances in the security group. (Note that this does not add rules from the source security group to this security group.)

When you add or remove rules, they are automatically applied to all instances associated with the security group.

Some systems for setting up firewalls let you filter on source ports. Security groups let you filter only on destination ports.

The following table describes example rules for a security group for web servers. The web servers can receive HTTP and HTTPS traffic, and send SQL or MySQL traffic to a database server.

Inbound			
Source	Protocol	Port Range	Comments
0.0.0.0/0	TCP	80	Allow inbound HTTP access from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access from anywhere

Amazon Virtual Private Cloud User Guide
Differences Between Security Groups for EC2-Classical
and EC2-VPC

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			
Destination	Protocol	Port Range	Comments
The ID of the security group for your database servers	TCP	1433	Allow outbound Microsoft SQL Server access to instances in the specified security group
The ID of the security group for your MySQL database servers	TCP	3306	Allow outbound MySQL access to instances in the specified security group

For step-by-step directions for creating security groups for web servers and database servers, see [Recommended Security Groups \(p. 25\)](#).

Differences Between Security Groups for EC2-Classical and EC2-VPC

If you're already an Amazon EC2 user, you're probably familiar with security groups. However, you can't use the security groups that you've created for use with EC2-Classical with instances in your VPC. You must create security groups specifically for use with instances in your VPC. The rules you create for use with a security group for a VPC can't reference a security group for EC2-Classical, and vice versa.

The following table summarizes the differences between security groups for use with EC2-Classical and those for use with EC2-VPC.

EC2-Classical	EC2-VPC
You can create up to 500 security groups per region.	You can create up to 100 security groups per VPC.
You can add up to 100 rules to a security group.	You can add up to 50 rules to a security group.
You can add rules for inbound traffic only.	You can add rules for inbound and outbound traffic.
You can assign an unlimited number of security groups to an instance.	You can assign up to 5 security groups to an instance.
You can reference security groups from other AWS accounts.	You can reference security groups for your VPC only.
After you launch an instance, you can't change the security groups assigned to it.	You can change the security groups assigned to an instance after it's launched.
When you add a rule to a security group, you don't have to specify a protocol, and only TCP, UDP, or ICMP are available.	When you add a rule to a security group, you must specify a protocol, and it can be any protocol with a standard protocol number, or all protocols (see Protocol Numbers).

EC2-Classic	EC2-VPC
When you add a rule to a security group, you must specify port numbers (for TCP or UDP).	When you add a rule to a security group, you can specify port numbers only if the rule is for TCP or UDP, and you can specify all port numbers.

Working with Security Groups

This section shows you how to work with security groups using the AWS Management Console.

Topics

- [Modifying the Default Security Group \(p. 56\)](#)
- [Creating a Security Group \(p. 56\)](#)
- [Adding and Removing Rules \(p. 56\)](#)
- [Changing an Instance's Security Groups \(p. 57\)](#)
- [Deleting a Security Group \(p. 57\)](#)
- [Deleting the 2009-07-15-default Security Group \(p. 57\)](#)

Modifying the Default Security Group

Your VPC includes a default security group whose initial rules are to deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances in the group. You can't delete this group; however, you can change the group's rules. The procedure is the same as modifying any other security group. For more information, see [Adding and Removing Rules \(p. 56\)](#).

Creating a Security Group

Although you can use the default security group for your instances, you might want to create your own groups to reflect the different roles that instances play in your system.

To create a security group

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Security Groups**.
3. Click the **Create Security Group** button.
4. Enter a name of the security group (for example, `my-security-group`) and provide a description. Select the ID of your VPC from the **VPC** menu, and then click **Yes, Create**.

By default, new security groups start with only an outbound rule that allows all traffic to leave the instances. You must add rules to enable any inbound traffic or to restrict the outbound traffic.

Adding and Removing Rules

When you add or remove a rule, any instances already assigned to the security group are subject to the change. You can't modify rules; you can only add and delete rules.

Several of the scenarios presented in this guide include instructions for adding rules to security groups. For an example, see [Recommended Security Groups \(p. 25\)](#).

To add a rule

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

2. In the navigation pane, click **Security Groups**.
3. Select the security group to update. The details pane displays the details for the security group, plus tabs for working with its inbound rules and outbound rules.
4. On the **Inbound Rules** tab, click **Edit**. Select an option for a rule for inbound traffic from the **Type** list, and then fill in the required information. For example, select **HTTP** or **HTTPS** and specify the **Source** as `0.0.0.0/0`. Click **Save** when you are done.
5. You can also allow communication between all instances associated with this security group. On the **Inbound Rules** tab, select **All Traffic** from the **Type** list. Start typing the ID of the security group in the **Source** field; this provides you with a list of security groups. Select the security group from the list, and then click **Save**.
6. If you need to, you can use the **Outbound Rules** tab to add rules for outbound traffic.

To delete a rule

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Security Groups**.
3. Select the security group to update. The details pane displays the details for the security group, plus tabs for working with its inbound rules and outbound rules.
4. Click **Edit**, and then click the **Remove** button for rule you want to delete. Click **Save** when you're done.

Changing an Instance's Security Groups

You can change the security groups that an instance in a VPC is assigned to after the instance is launched. When you make this change, the instance can be either running or stopped.

To change an instance's security groups

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Click **Instances** in the navigation pane.
3. Right-click the instance, and then select **Change Security Groups**.
4. In the **Change Security Groups** dialog box, select one or more security groups from the list, and then click **Assign Security Groups**.

Deleting a Security Group

You can delete a security group only if there are no instances assigned to it (either running or stopped). You can assign the instances to another security group before you delete the security group (see [Changing an Instance's Security Groups \(p. 57\)](#)).

To delete a security group

1. Open the Amazon VPC console.
2. Click **Security Groups** in the navigation pane.
3. Select the security group, and then click **Delete**.
4. In the **Delete Security Group** dialog box, click **Yes, Delete**.

Deleting the 2009-07-15-default Security Group

Any VPC created using an API version older than 2011-01-01 has the `2009-07-15-default` security group. This security group exists in addition to the regular `default` security group that comes with every

VPC. You can't attach an Internet gateway to a VPC that has the `2009-07-15-default` security group. Therefore, you must delete this security group before you can attach an Internet gateway to the VPC.

Note

If you assigned this security group to any instances, you must assign these instances a different security group before you can delete the security group.

To delete the `2009-07-15-default` security group

1. Ensure that this security group is not assigned to any instances.
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. In the navigation pane, click **Network Interfaces**.
 - c. Select the network interface for the instance from the list, and then select **Change Security Groups** from the **Actions** list.
 - d. In the **Change Security Groups** dialog box, select a new security group from the list, and then click **Save**.

Tip

When changing an instance's security group, you can select multiple groups from the list. The security groups that you select replace the current security groups for the instance.

- e. Repeat the preceding steps for each instance.
2. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 3. In the navigation pane, click **Security Groups**.
 4. Select the `2009-07-15-default` security group, and then click the **Delete** button.
 5. In the **Delete Security Group** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Create a security group

- [create-security-group](#) (AWS CLI)
- [ec2-create-group](#) (Amazon EC2 CLI)
- [New-EC2SecurityGroup](#) (AWS Tools for Windows PowerShell)

Add a rule to a security group

- [authorize-security-group-ingress](#) and [authorize-security-group-egress](#) (AWS CLI)
- [ec2-authorize](#) (Amazon EC2 CLI)
- [Grant-EC2SecurityGroupIngress](#) and [Grant-EC2SecurityGroupEgress](#) (AWS Tools for Windows PowerShell)

Describe one or more security groups

- [describe-security-groups](#) (AWS CLI)
- [ec2-describe-group](#) (Amazon EC2 CLI)
- [Get-EC2SecurityGroup](#) (AWS Tools for Windows PowerShell)

Modify the security groups for an instance

- [modify-instance-attribute](#) (AWS CLI)
- [ec2-modify-instance-attribute](#) (Amazon EC2 CLI)
- [Edit-EC2InstanceAttribute](#) (AWS Tools for Windows PowerShell)

Remove a rule from a security group

- [revoke-security-group-ingress](#) and [revoke-security-group-egress](#) (AWS CLI)
- [ec2-revoke](#) (Amazon EC2 CLI)
- [Revoke-EC2SecurityGroupIngress](#) and [Revoke-EC2SecurityGroupEgress](#) (AWS Tools for Windows PowerShell)

Delete a security group

- [delete-security-group](#) (AWS CLI)
- [ec2-delete-group](#) (Amazon EC2 CLI)
- [Remove-EC2SecurityGroup](#) (AWS Tools for Windows PowerShell)

Network ACLs

A *network access control list (ACL)* is an optional layer of security that acts as a firewall for controlling traffic in and out of a subnet. You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC. For more information about the differences between security groups and network ACLs, see [Comparison of Security Groups and Network ACLs \(p. 51\)](#).

Topics

- [Network ACL Basics \(p. 59\)](#)
- [Network ACL Rules \(p. 60\)](#)
- [Default Network ACL \(p. 60\)](#)
- [Example Custom Network ACL \(p. 60\)](#)
- [Ephemeral Ports \(p. 62\)](#)
- [Working with Network ACLs \(p. 62\)](#)
- [API and Command Overview \(p. 99\)](#)

Network ACL Basics

The following are the basic things that you need to know about network ACLs:

- A network ACL is a numbered list of rules that we evaluate in order, starting with the lowest numbered rule, to determine whether traffic is allowed in or out of any subnet associated with the network ACL. The highest number that you can use for a rule is 32766. We suggest that you start by creating rules with rule numbers that are multiples of 100, so that you can insert new rules where you need to later on.
- A network ACL has separate inbound and outbound rules, and each rule can either allow or deny traffic.
- Your VPC automatically comes with a modifiable default network ACL; by default, it allows all inbound and outbound traffic.
- You can create custom network ACLs; each custom network ACL starts out closed (permits no traffic) until you add a rule.

- Each subnet must be associated with a network ACL; if you don't explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL.
- Network ACLs are stateless; responses to allowed inbound traffic are subject to the rules for outbound traffic (and vice versa).

For information about the number of network ACLs you can create, see [Amazon VPC Limits \(p. 147\)](#).

Network ACL Rules

You can add or remove rules from the default network ACL, or create additional network ACLs for your VPC. When you add or remove rules from a network ACL, the changes are automatically applied to the subnets it's associated with.

The following are the parts of a network ACL rule:

- Rule number. Rules are evaluated starting with the lowest numbered rule.
- Protocol. You can specify any protocol that has a standard protocol number. For more information, see [Protocol Numbers](#). If you specify ICMP as the protocol, you can specify any or all of the ICMP types and codes.
- [Inbound rules only] The source of the traffic (CIDR range) and the destination (listening) port or port range.
- [Outbound rules only] The destination for the traffic (CIDR range) and the destination port or port range.
- Choice of allow or deny.

Default Network ACL

To help you understand what ACL rules look like, here's what the default network ACL looks like in its initial state. It is configured to allow all traffic to flow in and out of each subnet. Each network ACL includes a rule whose rule number is an asterisk. This rule ensures that if a packet doesn't match any of the other rules, it's denied. You can't modify or remove this rule.

Inbound				
Rule #	Source IP	Protocol	Port	Allow/Deny
100	0.0.0.0/0	All	All	ALLOW
*	0.0.0.0/0	All	All	DENY
Outbound				
Rule #	Dest IP	Protocol	Port	Allow/Deny
100	0.0.0.0/0	all	all	ALLOW
*	0.0.0.0/0	all	all	DENY

Example Custom Network ACL

The following table shows an example of a custom network ACL. It includes rules that allow HTTP and HTTPS traffic in (inbound rules 100 and 110). There's a corresponding outbound rule that enables responses to that inbound traffic (outbound rule 120, which covers ephemeral ports 49152-65535). For more information about how to select the appropriate ephemeral port range, see [Ephemeral Ports \(p. 62\)](#).

The network ACL also includes inbound rules that allow SSH and RDP traffic into the subnet. The outbound rule 120 enables responses to egress the subnet.

The network ACL has outbound rules (100 and 110) that allow outbound HTTP and HTTPS traffic out of the subnet. There's a corresponding inbound rule that enables responses to that outbound traffic (inbound rule 140, which covers ephemeral ports 49152-65535).

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic from anywhere.
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic from anywhere.
120	192.0.2.0/24	TCP	22	ALLOW	Allows inbound SSH traffic from your home network's public IP address range (over the Internet gateway).
130	192.0.2.0/24	TCP	3389	ALLOW	Allows inbound RDP traffic to the web servers from your home network's public IP address range (over the Internet gateway).
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from the Internet (that is, for requests that originate in the subnet). For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 62) .
150	0.0.0.0/0	UDP	32768-61000	ALLOW	Allows inbound return UDP traffic. For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 62) .
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable).
Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet.

110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet.
120	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet). For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 62) .
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable).

As a packet comes to the subnet, we evaluate it against the ingress rules of the ACL the subnet is associated with (starting at the top of the list of rules, and moving to the bottom). Let's say the packet is destined for the SSL port (443). The packet doesn't match the first rule evaluated (rule 100). It does match the second rule (110), which allows the packet into the subnet. If the packet had been destined for port 139 (NetBIOS), the first two rules would not have matched, but the * rule ultimately would have denied the packet.

You might want to add a DENY rule in a situation where you legitimately need to open a wide range of ports, but there are certain ports within that range you want to deny. Just make sure to place the DENY rule earlier in the table than the rule that allows the wide range of port traffic.

Ephemeral Ports

The example network ACL in the preceding section uses an ephemeral port range of 49152-65535. However, you might want to use a different range for your network ACLs. This section explains why.

The client that initiates the request chooses the ephemeral port range. The range varies depending on the client's operating system. Many Linux kernels (including the Amazon Linux kernel) use ports 32768-61000. Requests originating from Elastic Load Balancing use ports 1024-65535. Windows operating systems through Windows Server 2003 use ports 1025-5000. Windows Server 2008 uses ports 49152-65535. Therefore, if a request comes in to a web server in your VPC from a Windows XP client on the Internet, your network ACL must have an outbound rule to enable traffic destined for ports 1025-5000.

If an EC2 instance in your VPC is the client initiating a request, your network ACL must have an inbound rule to enable traffic destined for the ephemeral ports specific to the type of instance (Amazon Linux, Windows Server 2008, and so on.).

In practice, to cover the different types of clients that might initiate traffic to public-facing instances in your VPC, you need to open ephemeral ports 1024-65535. However, you can also add rules to the ACL to deny traffic on any malicious ports within that range. Make sure to place the DENY rules earlier in the table than the rule that opens the wide range of ephemeral ports.

Working with Network ACLs

This section shows you how to work with network ACLs using the Amazon VPC console.

Important

The VPC console has been redesigned, and you can switch between the old and new interfaces by clicking the link in the preview message at the top of each console page. You can use the old interface during the trial period; however, this topic may refer to features of the new interface only.

Topics

- [Determining Which Network ACL a Subnet Is Associated With](#) (p. 63)
- [Determining Which Subnets Are Associated with a Network ACL](#) (p. 63)
- [Creating a Network ACL](#) (p. 63)
- [Adding and Deleting Rules](#) (p. 64)
- [Associating a Subnet with a Network ACL](#) (p. 65)
- [Disassociating a Network ACL from a Subnet](#) (p. 65)
- [Changing a Subnet's Network ACL](#) (p. 65)
- [Deleting a Network ACL](#) (p. 66)

Determining Which Network ACL a Subnet Is Associated With

To determine which network ACL a subnet is associated with

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Subnets**, and then select the subnet.

The network ACL associated with the subnet is included in the **Network ACL** tab, along with the network ACL's rules.

Determining Which Subnets Are Associated with a Network ACL

To determine which subnets are associated with a network ACL

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Network ACLs**.

The console displays your network ACLs. The **Associated With** column indicates the number of associated subnets.

3. Select a network ACL.
4. In the details pane, click the **Subnet Associations** tab to display the subnets associated with the network ACL.

Creating a Network ACL

To create a network ACL

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Network ACLs**.
3. Click the **Create Network ACL** button.
4. In the **Create Network ACL** dialog box, optionally name your network ACL, and then select the ID of your VPC from the **VPC** list, and click **Yes, Create**.

The initial settings for a network ACL block all inbound and outbound traffic. The network ACL has no rules except the * rule present in every ACL.

There are no subnets associated with a new ACL.

Adding and Deleting Rules

When you add or delete a rule from an ACL, any subnets associated with the ACL are subject to the change. You don't have to terminate and relaunch the instances in the subnet; the changes take effect after a short period.

You can't modify rules; you can only add and delete rules. If you need to change the order of a rule in the ACL, you must add a new rule with the new rule number, and then delete the original rule.

To add rules to a network ACL

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Network ACLs**.
3. In the details pane, select either the **Inbound Rules** or **Outbound Rules** tab, depending on the type of rule that you need to add.
4. Click **Edit**, and select a rule from the **Type** list. For example, to add a rule for HTTP, select the **HTTP** option. To add a rule to allow all TCP traffic, select **All TCP**. For some of these options (for example, HTTP), we fill in the port for you. To use a protocol that's not listed, select **Custom protocol rule**.
5. Provide the rule's details:
 - a. In **Rule #**, enter a rule number (for example, 100). The rule number must not already be used in the network ACL. We process the rules in order, starting with the lowest number.

Tip

We recommend that you leave gaps between the rule numbers (such as 100, 200, 300), rather than using sequential numbers (101, 102, 103). This makes it easier add a new rule where it belongs without having to renumber the existing rules.
 - b. (Optional) If you're creating a custom protocol rule, select the protocol's number (47) and name (GRE) from the **Protocol** list. For more information, see [IANA List of Protocol Numbers](#).
 - c. (Optional) If the protocol you've selected requires a port number, enter the port number or port range separated by a hyphen (for example, 49152-65535).
 - d. In the **Source** or **Destination** box (depending on whether this is an inbound or outbound rule), enter the CIDR range that the rule applies to.
6. From the **Allow/Deny** list, select **ALLOW** to allow the specified traffic or **DENY** to deny the specified traffic.
7. Click **Save**.

To delete a rule from a network ACL

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Network ACLs**, and then select the network ACL.
3. In the details pane, select either the **Inbound Rules** or **Outbound Rules** tab, and then click **Edit**. Click the **Remove** button for the rule you want to delete, and then click **Save**.

Associating a Subnet with a Network ACL

To apply the rules of a network ACL to a particular subnet, you must associate the subnet with the network ACL. You can associate a network ACL with multiple subnets; however, a subnet can be associated with only one network ACL. Any subnet not associated with a particular ACL is associated with the default network ACL by default.

To associate a subnet with a network ACL

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Network ACLs**, and then select the network ACL.
3. In the details pane, on the **Subnet Associations** tab, click **Edit**. Select the **Associate** check box for the subnet to associate with the table, and then click **Save**.

Disassociating a Network ACL from a Subnet

You might want to disassociate a subnet from its network ACL. For example, you might have a subnet that is associated with a custom network ACL, and you instead want it associated with the default network ACL. By disassociating the subnet from the custom network ACL, the subnet becomes associated with the default network ACL.

To disassociate a subnet from a network ACL

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Network ACLs**, and then select the network ACL.
3. In the details pane, click the **Subnet Associations** tab.
4. Click **Edit**, and then deselect the **Associate** check box for the subnet. Click **Save**.

Changing a Subnet's Network ACL

You can change which network ACL a subnet is associated with. For example, when you create a subnet, it is initially associated with the default network ACL. You might want to instead associate it with a custom network ACL that you've created.

After changing a subnet's network ACL, you don't have to terminate and relaunch the instances in the subnet; the changes take effect after a short period.

To change a subnet's network ACL association

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Subnets**, and then select the subnet.
3. Click the **Network ACL** tab, and then click **Edit**.
4. Select the network ACL to associate the subnet with from the **Network ACL** list, and then click **Save**.

subnet-e9a0a09d (10.0.0.0/28)

Summary Route Table **Network ACL** Tags

Cancel Save

Network ACL: acl-ac13f0c9 (default) :

Inbound:

Rule #	Type	Protocol	Port Range / ICMP Type	Source	Allow / Deny
100	ALL Traffic	ALL	ALL	0.0.0.0/0	ALLOW
*	ALL Traffic	ALL	ALL	0.0.0.0/0	DENY

Outbound:

Rule #	Type	Protocol	Port Range / ICMP Type	Source	Allow / Deny
--------	------	----------	------------------------	--------	--------------

Deleting a Network ACL

You can delete a network ACL only if there are no subnets associated with it. You can't delete the default network ACL.

To delete a network ACL

1. Open the Amazon VPC console.
2. Click **Network ACLs** in the navigation pane.
3. Select the network ACL, and then click the **Delete** button.
4. In the **Delete Network ACL** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Create a network ACL for your VPC

- [create-network-acl](#) (AWS CLI)
- [ec2-create-network-acl](#) (Amazon EC2 CLI)
- [New-EC2NetworkAcl](#) (AWS Tools for Windows PowerShell)

Describe one or more of your network ACLs

- [describe-network-acls](#) (AWS CLI)
- [ec2-describe-network-acls](#) (Amazon EC2 CLI)
- [Get-EC2NetworkAcl](#) (AWS Tools for Windows PowerShell)

Add a rule to a network ACL

- [create-network-acl-entry](#) (AWS CLI)
- [ec2-create-network-acl-entry](#) (Amazon EC2 CLI)
- [New-EC2NetworkAclEntry](#) (AWS Tools for Windows PowerShell)

Delete a rule from a network ACL

- [delete-network-acl-entry](#) (AWS CLI)
- [ec2-delete-network-acl-entry](#) (Amazon EC2 CLI)

- [Remove-EC2NetworkAclEntry](#) (AWS Tools for Windows PowerShell)

Replace an existing rule in a network ACL

- [replace-network-acl-entry](#) (AWS CLI)
- [ec2-replace-network-acl-entry](#) (Amazon EC2 CLI)
- [Set-EC2NetworkAclEntry](#) (AWS Tools for Windows PowerShell)

Replace a network ACL association

- [replace-network-acl-association](#) (AWS CLI)
- [ec2-replace-network-acl-association](#) (Amazon EC2 CLI)
- [Set-EC2NetworkAclAssociation](#) (AWS Tools for Windows PowerShell)

Delete a network ACL

- [delete-network-acl](#) (AWS CLI)
- [ec2-delete-network-acl](#) (Amazon EC2 CLI)
- [Remove-EC2NetworkAcl](#) (AWS Tools for Windows PowerShell)

Recommended Network ACL Rules for Your VPC

The VPC wizard helps you implement common scenarios for Amazon VPC. If you implement these scenarios as described in the documentation, you'll use the default network access control list (ACL), which allows all inbound and outbound traffic. If you need an additional layer of security, you can create a network ACL and add rules. We recommend the following rules for each scenario.

Topics

- [Recommended Rules for Scenario 1](#) (p. 67)
- [Recommended Rules for Scenario 2](#) (p. 69)
- [Recommended Rules for Scenario 3](#) (p. 71)
- [Recommended Rules for Scenario 4](#) (p. 74)

For more information about network ACLs and how to use them, see [Network ACLs](#) (p. 59).

Important

We use the ephemeral port range 49152-65535. You can select a different range. For more information, see [Ephemeral Ports](#) (p. 62).

Recommended Rules for Scenario 1

Scenario 1 is a single subnet with instances that can receive and send Internet traffic. For more information, see [Scenario 1: VPC with a Public Subnet Only](#) (p. 7).

The following table shows the rules we recommended. They block all traffic except that which is explicitly required.

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments

Amazon Virtual Private Cloud User Guide
Recommended Rules for Scenario 68

100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic from anywhere
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic from anywhere
120	Public IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic from your home network (over the Internet gateway)
130	Public IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic from your home network (over the Internet gateway)
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Recommended Rules for Scenario 2

Scenario 2 is a public subnet with instances that can receive and send Internet traffic, and a private subnet that can't receive traffic directly from the Internet. However, it can initiate traffic to the Internet (and receive responses) through a NAT instance in the public subnet. For more information, see [Scenario 2: VPC with Public and Private Subnets \(p. 12\)](#).

For this scenario you have a network ACL for the public subnet, and a separate one for the private subnet. The following table shows the rules we recommend for each ACL. They block all traffic except that which is explicitly required. They mostly mimic the security group rules for the scenario.

ACL Rules for the Public Subnet

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic from anywhere
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic from anywhere
120	Public IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic from your home network (over the Internet gateway)
130	Public IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic from your home network (over the Internet gateway)
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet

Amazon Virtual Private Cloud User Guide
Recommended Rules for Scenario 2

110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	10.0.1.0/24	TCP	1433	ALLOW	Allows outbound MS SQL access to database servers in the private subnet
130	10.0.1.0/24	TCP	3306	ALLOW	Allows outbound MySQL access to database servers in the private subnet
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

ACL Rules for the Private Subnet

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	10.0.0.0/24	TCP	1433	ALLOW	Allows web servers in the public subnet to read and write to MS SQL servers in the private subnet
110	10.0.0.0/24	TCP	3306	ALLOW	Allows web servers in the public subnet to read and write to MySQL servers in the private subnet
120	10.0.0.0/24	TCP	22	ALLOW	Allows inbound SSH traffic from the SSH bastion in the public subnet
130	10.0.0.0/24	TCP	3389	ALLOW	Allows inbound RDP traffic from the Microsoft Terminal Services gateway in the public subnet

140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from NAT instance in the public subnet for requests originating in the private subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	10.0.0.0/24	TCP	49152-65535	ALLOW	Allows outbound responses to the public subnet (for example, responses to web servers in the public subnet that are communicating with DB Servers in the private subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Recommended Rules for Scenario 3

Scenario 3 is a public subnet with instances that can receive and send Internet traffic, and a VPN-only subnet with instances that can communicate only with your home network over the VPN connection. For more information, see [Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access \(p. 22\)](#).

For this scenario you have a network ACL for the public subnet, and a separate one for the VPN-only subnet. The following table shows the rules we recommend for each ACL. They block all traffic except that which is explicitly required.

ACL Rules for the Public Subnet

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic to the web servers from anywhere
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic to the web servers from anywhere
120	Public IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic to the web servers from your home network (over the Internet gateway)
130	Public IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic to the web servers from your home network (over the Internet gateway)
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	10.0.1.0/24	TCP	1433	ALLOW	Allows outbound MS SQL access to database servers in the VPN-only subnet
130	10.0.1.0/24	TCP	3306	ALLOW	Allows outbound MySQL access to database servers in the VPN-only subnet

140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

ACL Settings for the VPN-Only Subnet

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	10.0.0.0/24	TCP	1433	ALLOW	Allows web servers in the public subnet to read and write to MS SQL servers in the VPN-only subnet
110	10.0.0.0/24	TCP	3306	ALLOW	Allows web servers in the public subnet to read and write to MySQL servers in the VPN-only subnet
120	Private IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic from the home network (over the virtual private gateway)
130	Private IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic from the home network (over the virtual private gateway)
140	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows inbound return traffic from clients in the home network (over the virtual private gateway) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)

Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	Private IP address range of your home network	All	All	ALLOW	Allows all outbound traffic from the subnet to your home network (over the virtual private gateway)
110	10.0.0.0/24	TCP	49152-65535	ALLOW	Allows outbound responses to the web servers in the public subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
120	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows outbound responses to clients in the home network (over the virtual private gateway) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Recommended Rules for Scenario 4

Scenario 4 is a single subnet with instances that can communicate only with your home network over a VPN connection. For a more information, see [Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access \(p. 31\)](#).

The following table shows the rules we recommended. They block all traffic except that which is explicitly required.

Inbound					
Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	Private IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic to the subnet from your home network

110	Private IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic to the subnet from your home network
120	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	Dest IP	Protocol	Port	Allow/Deny	Comments
100	Private IP address range of your home network	All	All	ALLOW	Allows all outbound traffic from the subnet to your home network
120	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows outbound responses to clients in the home network See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Controlling Access to Amazon VPC Resources

Do you want to control who can set up and manage your virtual private clouds (VPC)? Do you want to control who can attach an Internet gateway or define security groups and network ACLs? You can use AWS Identity and Access Management (IAM) to create and manage users and groups in your account, and control which API actions and Amazon VPC resources each user and group can use. Therefore, you can ensure that not everyone in your organization can make changes to the layout, routing, and security of your VPC.

Important

Currently, not all Amazon EC2 API actions support individual ARNs; we'll add support for additional API actions and ARNs for additional Amazon EC2 resources later. For information about which ARNs you can use with which Amazon EC2 API actions, as well as supported condition keys

for each ARN, see [Supported Resources and Conditions for Amazon EC2 API Actions](#) in the *Amazon Elastic Compute Cloud User Guide*.

For more information about creating IAM policies for Amazon EC2, supported resources for EC2 API actions, as well example policies for Amazon EC2, see [IAM Policies for Amazon EC2](#) in the *Amazon Elastic Compute Cloud User Guide*.

The following examples show policy statements that you could use to control the permissions that IAM users have to Amazon VPC.

- [1. Managing a VPC \(p. 76\)](#)
- [2. Read-Only Policy for Amazon VPC \(p. 77\)](#)
- [3. Custom Policy for Amazon VPC \(p. 77\)](#)
- [4. Launching instances into a specific subnet \(p. 78\)](#)
- [5. Launching instances into a specific VPC \(p. 79\)](#)
- [6. Managing security groups in a VPC \(p. 80\)](#)
- [7. Creating and managing VPC peering connections \(p. 81\)](#)

Example Policies for Amazon VPC

Example 1. Managing a VPC

The following policy grants users permission to create and manage your VPC. You might attach this policy to a group of network administrators. The `Action` element specifies the API actions related to VPCs, subnets, Internet gateways, customer gateways, virtual private gateways, VPN connections, route tables, Elastic IP addresses, security groups, network ACLs, and DHCP options sets. The policy also allows the group to run, stop, start, and terminate instances. It also allows the group to list Amazon EC2 resources.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:*Vpc*",
      "ec2:*Subnet*",
      "ec2:*Gateway*",
      "ec2:*Vpn*",
      "ec2:*Route*",
      "ec2:*Address*",
      "ec2:*SecurityGroup*",
      "ec2:*NetworkAcl*",
      "ec2:*DhcpOptions*",
      "ec2:RunInstances",
      "ec2:StopInstances",
      "ec2:StartInstances",
      "ec2:TerminateInstances",
      "ec2:Describe*"
    ],
    "Resource": "*"
  }]
}
```

The policy uses wildcards to specify all actions for each type of object (for example, `*SecurityGroup*`). Alternatively, you could list each action explicitly. If you use the wildcards, be aware that if we add new actions whose names include any of the wildcarded strings in the policy, the policy would automatically grant the group access to those new actions.

Example 2. Read-Only Policy for Amazon VPC

The following policy grants users permission to list your VPCs and their components. They can't create, update, or delete them.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:DescribeVpcs",
      "ec2:DescribeSubnets",
      "ec2:DescribeInternetGateways",
      "ec2:DescribeCustomerGateways",
      "ec2:DescribeVpnGateways",
      "ec2:DescribeVpnConnections",
      "ec2:DescribeRouteTables",
      "ec2:DescribeAddresses",
      "ec2:DescribeSecurityGroups",
      "ec2:DescribeNetworkAcls",
      "ec2:DescribeDhcpOptions",
      "ec2:DescribeTags",
      "ec2:DescribeInstances"
    ],
    "Resource": "*"
  }
]
```

Example 3. Custom Policy for Amazon VPC

The following policy grants users permission to launch instances, stop instances, start instances, terminate instances, and describe the available resources for Amazon EC2 and Amazon VPC.

The second statement in the policy protects against any other policy that might grant the user access to a wider range of API actions by explicitly denying permissions.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:RunInstances",
      "ec2:StopInstances",
      "ec2:StartInstances",
      "ec2:TerminateInstances",
      "ec2:Describe*"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Deny",
    "NotAction": [
      "ec2:RunInstances",
      "ec2:StopInstances",
      "ec2:StartInstances",
      "ec2:TerminateInstances",
      "ec2:Describe*"
    ],
    "Resource": "*"
  }
]
```

Example 4. Launching instances into a specific subnet

The following policy grants users permission to launch instances into a specific subnet, and to use a specific security group in the request. The policy does this by specifying the ARN for `subnet-1a2b3c4d`, and the ARN for `sg-123abc123`. If users attempt to launch an instance into a different subnet or using a different security group, the request will fail (unless another policy or statement grants users permission to do so).

The policy also grants permission to use the network interface resource. When launching into a subnet, the `RunInstances` request creates a primary network interface by default, so the user needs permission to create this resource when launching the instance.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": [
      "arn:aws:ec2:region::image/ami-*",
      "arn:aws:ec2:region:account:subnet/subnet-1a2b3c4d",
      "arn:aws:ec2:region:account:network-interface/*",
      "arn:aws:ec2:region:account:volume/*",
      "arn:aws:ec2:region:account:key-pair/*",
      "arn:aws:ec2:region:account:security-group/sg-123abc123"
    ]
  }]
}
```

Example 5. Launching instances into a specific VPC

The following policy grants users permission to launch instances into any subnet within a specific VPC. The policy does this by applying a condition key (`ec2:Vpc`) to the subnet resource.

The policy also grants users permission to launch instances using only AMIs that have the tag `"department=dev"`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "arn:aws:ec2:region:account:subnet/*",
    "Condition": {
      "StringEquals": {
        "ec2:Vpc": "arn:aws:ec2:region:account:vpc/vpc-1a2b3c4d"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "arn:aws:ec2:region::image/ami-*",
    "Condition": {
      "StringEquals": {
        "ec2:ResourceTag/department": "dev"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": [
      "arn:aws:ec2:region:account:instance/*",
      "arn:aws:ec2:region:account:volume/*",
      "arn:aws:ec2:region:account:network-interface/*",
      "arn:aws:ec2:region:account:key-pair/*",
      "arn:aws:ec2:region:account:security-group*"
    ]
  }
]
```

Example 6. Managing security groups in a VPC

The following policy grants users permission to create and delete inbound and outbound rules for any security group within a specific VPC. The policy does this by applying a condition key (`ec2:Vpc`) to the security group resource for the `Authorize` and `Revoke` actions.

The second part of the policy grants users permission to describe all security groups. This is necessary in order for users to be able to modify security group rules using the CLI.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:AuthorizeSecurityGroupIngress",
      "ec2:AuthorizeSecurityGroupEgress",
      "ec2:RevokeSecurityGroupIngress",
      "ec2:RevokeSecurityGroupEgress"],
    "Resource": "arn:aws:ec2:region:account:security-group/*",
    "Condition": {
      "StringEquals": {
        "ec2:Vpc": "arn:aws:ec2:region:account:vpc/vpc-1a2b3c4d"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": "ec2:DescribeSecurityGroups",
    "Resource": "*"
  }
]
```

Example 7. Creating and managing VPC peering connections

The following are examples of policies you can use to manage the creation and modification of VPC peering connections.

a. Create 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 part of the 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 part of the 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-aaa111bb`) 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-aaa111bb"
      }
    }
  }
]
```

```
]
}
```

b. Accept 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 part of the 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"
      }
    }
  }
]
```

c. 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-1a2b3c4d`, which is in the same account. The policy specifies both the `ec2:AcceptorVpc` 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:AcceptorVpc": "arn:aws:ec2:region:444455556666:vpc/vpc-1a2b3c4d",
        "ec2:RequesterVpc": "arn:aws:ec2:region:444455556666:vpc/vpc-1a2b3c4d"
      }
    }
  }]
}
```

d. 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 part of the 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 part of the statement allows users to create VPC peering connections, and allows access to all VPCs in account 333333333333 in order to do so.

The third part of the 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 acceptor 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:AcceptorVpc": "arn:aws:ec2:*:333333333333:vpc/*",
        "ec2:RequesterVpc": "arn:aws:ec2:*:333333333333:vpc/*"
      }
    }
  }
]
}
```

Networking in Your VPC

You can use the following components to configure networking in your VPC:

- [IP Addresses](#) (p. 85)
- [Network Interfaces](#) (p. 90)
- [Route Tables](#) (p. 91)
- [Internet Gateways](#) (p. 101)
- [NAT Instances](#) (p. 105)
- [DHCP Options Sets](#) (p. 112)
- [DNS](#) (p. 116)

IP Addressing in Your VPC

This topic describes the IP addresses available to your Amazon EC2 instances in your VPC.

Public and Private IP Addresses

We provide your instances in a VPC with IP addresses. You can use private IP addresses for communication between the instances in your VPC. You can use public IP addresses for communication between your instances and the Internet.

Note

To ensure that your instances can communicate with the Internet, you must also attach an Internet gateway to your VPC. For more information, see [Adding an Internet Gateway to Your VPC](#) (p. 101).

Each instance launched in a VPC has a default network interface that is assigned a primary private IP address in the address range of the subnet. If you don't specify a primary private IP address, we select an available IP address in the subnet range for you. For more information about network interfaces, see [Elastic Network Interfaces](#) in the *Amazon Elastic Compute Cloud User Guide*.

All subnets have an attribute that determines whether instances launched into that subnet receive a public IP address. By default, instances launched into a default subnet are assigned a public IP address, which is mapped to the primary private IP address through network address translation (NAT). However, you can control whether your instance in a default or nondefault subnet receives a public IP address. For

more information, see [Assigning a Public IP Address During Launch](#) (p. 86), and [Modifying Your Subnet's Public IP Addressing Behavior](#) (p. 87).

Note

T2 instance types can only be launched into a VPC. If you use the Amazon EC2 launch wizard to launch a T2 instance type in your EC2-Classical account, and you have no VPCs, the launch wizard creates a nondefault VPC for you, with a subnet in each Availability Zone. The wizard modifies the subnets' attributes to request a public IP address for your instance automatically. For more information about T2 instance types, see [T2 Instances](#).

A public IP address is assigned to your instance from Amazon's pool of public IP addresses; it's not associated with your account. When a public IP address is disassociated from your instance, it's released back into the pool, and is no longer available for you to use. You cannot manually associate or disassociate a public IP address. Instead, in certain cases, we release the public IP address from your instance, or assign it a new one. For more information, see [Public IP Addresses](#) in the *Amazon Elastic Compute Cloud User Guide*.

If you require a persistent public IP address that can be assigned to and removed from instances as you require, use an Elastic IP address (EIP) instead. To do this, you must allocate an Elastic IP address for use with the VPC, and then associate that EIP with a private IP address specified by the network interface attached to the instance. For more information, see [Elastic IP Addresses](#) (p. 87).

You can assign additional IP addresses, known as secondary private IP addresses, to instances that are running in a VPC. Unlike a primary private IP address, you can reassign a secondary private IP address from one network interface to another and from one instance to another. For more information about primary and secondary IP addresses, see [Multiple IP Addresses](#) in the *Amazon Elastic Compute Cloud User Guide*.

Assigning a Public IP Address During Launch

A public IP addressing feature is available for you to control whether your instance in a default or nondefault subnet is assigned a public IP address during launch. The public IP address is assigned to the eth0 network interface (the primary network interface). This feature depends on certain conditions at the time you launch your instance.

Important

You can't manually disassociate the public IP address from your instance after launch. Instead, it's automatically released in certain cases, after which you cannot reuse it. If you require a persistent public IP address that you can associate or disassociate at will, associate an Elastic IP address with the instance after launch instead. For more information, see [Elastic IP Addresses](#) (p. 87).

To access the public IP addressing feature when launching an instance

1. Open the Amazon EC2 console.
2. Click **Launch Instance**.
3. Choose an AMI and click its **Select** button, then choose an instance type and click **Next: Configure Instance Details**.
4. On the **Configure Instance Details** page, if a VPC is selected in the **Network** list, a **Public IP** check box is displayed. This check box, if selected, will assign a public IP address to your instance. If you selected a default subnet, the **Public IP** check box is selected by default.

Subnet ⓘ	subnet-24271a4a(10.0.0.0/24) us-east-1d ▾	Create new sub
	250 IP Addresses available	
Public IP ⓘ	<input type="checkbox"/> Automatically assign a public IP address to your instances	

The following rules apply:

- A public IP address can only be assigned to a single network interface with the device index of eth0. The **Public IP** check box is not available if you're launching with multiple network interfaces, and is not available for the eth1 network interface.
- You can only assign a public IP address to a new network interface, not an existing one.

This feature is only available during launch. However, whether you assign a public IP address to your instance during launch or not, you can associate an Elastic IP address with your instance after it's launched. For more information, see [Elastic IP Addresses](#) (p. 87).

Modifying Your Subnet's Public IP Addressing Behavior

All subnets have an attribute that determines whether instances launched into that subnet are assigned a public IP address. By default, nondefault subnets have this attribute set to false, and default subnets have this attribute set to true. You can modify the default behavior by changing the subnet's public IP addressing attribute. If you change the public IP addressing attribute for a subnet, you can still override this setting for a specific instance during launch. For more information, see [Assigning a Public IP Address During Launch](#) (p. 86).

Note

A public IP address can only be assigned to a single network interface with the device index of eth0. If you are launching an instance with more than one network interface, or you have specified an existing network interface for eth0, your instance will not receive a public IP address, regardless of how you have defined the default public IP addressing behavior for the subnet.

To modify your subnet's public IP addressing behavior

1. Open the Amazon VPC console.
2. In the navigation pane, click **Subnets**.
3. Select your subnet, and click **Modify Auto-Assign Public IP**.
4. The **Enable Auto-assign Public IP** check box, if selected, requests a public IP address for all instances launched into the selected subnet. Select or clear the check box as required, then click **Save**.

Elastic IP Addresses

An *Elastic IP address* is a static, public IP address designed for dynamic cloud computing. You can associate an Elastic IP address with any instance or network interface for your VPC. With an EIP, you can mask the failure of an instance by rapidly remapping the address to another instance in your VPC. Note that the advantage of associating the Elastic IP address with the network interface instead of directly with the instance is that you can move all the attributes of the network interface from one instance to another in a single step.

Topics

- [Elastic IP Address Basics](#) (p. 88)
- [Working with Elastic IP Addresses](#) (p. 88)
- [API and Command Overview](#) (p. 89)

Elastic IP Address Basics

The following are the basic things that you need to know about Elastic IP addresses:

- You first allocate an EIP for use in a VPC, and then associate it with an instance in your VPC (it can be assigned to only one instance at a time).
- An EIP is a property of network interfaces. You can associate an EIP with an instance by updating the network interface attached to the instance.
- If you associate an EIP with the eth0 network interface of your instance, its current public IP address (if it had one) is released to the EC2-VPC public IP address pool. If you disassociate the EIP, the eth0 network interface is automatically assigned a new public IP address within a few minutes. This doesn't apply if you've attached a second network interface to your instance.
- There are differences between an EIP that you use in a VPC and one that you use in EC2-Classic. For more information, see [Differences Between EC2-Classic and Amazon EC2-VPC](#) in the *Amazon Elastic Compute Cloud User Guide*.
- You can move an EIP from one instance to another. The instance can be in the same VPC or another VPC, but not in EC2-Classic.
- Your EIPs remain associated with your AWS account until you explicitly release them.
- To ensure efficient use of EIPs, we impose a small hourly charge when they aren't associated with a running instance, or when they are associated with a stopped instance or an unattached network interface. While your instance is running, you aren't charged for one EIP associated with the instance, but you are charged for any additional EIPs associated with the instance. For more information, see [Amazon EC2 Pricing](#).
- You're limited to 5 Elastic IP addresses; to help conserve them, you can use a NAT instance (see [NAT Instances](#) (p. 105)).
- An EIP is accessed through the Internet gateway of a VPC. If you have set up a VPN connection between your VPC and your network, the VPN traffic traverses a virtual private gateway, not an Internet gateway, and therefore cannot access the EIP.

Working with Elastic IP Addresses

You can allocate an Elastic IP address and then associate it with an instance in a VPC.

To allocate an Elastic IP address for use in a VPC

1. Open the Amazon VPC console.
2. In the navigation pane, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. In the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.

To view your EIPs

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. To filter the displayed list, start typing part of the EIP or the ID of the instance to which it's assigned in the search box.

To associate an Elastic IP address with a running instance in a VPC

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.

3. Select an Elastic IP address that's allocated for use with a VPC (the **Scope** column has a value of `vpc`), and then click the **Associate Address** button.
4. In the **Associate Address** dialog box, select **Instance** or **Network Interface** from the **Associate with** list, and then either the instance or network interface ID. Select the private IP address to associate the Elastic IP address with from the **Private IP address** list, and then click **Yes, Associate**.

Note

A network interface can have several attributes, including an Elastic IP address. You can create a network interface and attach and detach it from instances in your VPC. The advantage of making the Elastic IP address an attribute of the network interface instead of associating it directly with the instance is that you can move all the attributes of the network interface from one instance to another in a single step. For more information, see [Elastic Network Interfaces](#).

5. (Optional) After you associate the Elastic IP address with your instance, it receives a DNS hostname if DNS hostnames are enabled. For more information, see [Using DNS with Your VPC \(p. 116\)](#).

To change which instance an Elastic IP address is associated with, disassociate it from the currently associated instance, and then associate it with the new instance in the VPC.

To disassociate an Elastic IP address

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. Select the Elastic IP address, and then click the **Disassociate Address** button.
4. When prompted, click **Yes, Disassociate**.

If you no longer need an Elastic IP address, we recommend that you release it (the address must not be associated with an instance). You incur charges for any EIP that's allocated for use with a VPC but not associated with an instance.

To release an Elastic IP address

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. Select the Elastic IP address, and then click the **Release Address** button.
4. When prompted, click **Yes, Release**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Acquire an Elastic IP address

- [allocate-address](#) (AWS CLI)
- [ec2-allocate-address](#) (Amazon EC2 CLI)
- [New-EC2Address](#) (AWS Tools for Windows PowerShell)

Associate an Elastic IP address with an instance or network interface

- [associate-address](#) (AWS CLI)
- [ec2-associate-address](#) (Amazon EC2 CLI)

- [Register-EC2Address](#) (AWS Tools for Windows PowerShell)

Describe one or more Elastic IP addresses

- [describe-addresses](#) (AWS CLI)
- [ec2-describe-addresses](#) (Amazon EC2 CLI)
- [Get-EC2Address](#) (AWS Tools for Windows PowerShell)

Disassociate an Elastic IP address

- [disassociate-address](#) (AWS CLI)
- [ec2-disassociate-address](#) (Amazon EC2 CLI)
- [Unregister-EC2Address](#) (AWS Tools for Windows PowerShell)

Release an Elastic IP address

- [release-address](#) (AWS CLI)
- [ec2-release-address](#) (Amazon EC2 CLI)
- [Remove-EC2Address](#) (AWS Tools for Windows PowerShell)

Assign a public IP address during launch

- Use the `--associate-public-ip-address` or the `--no-associate-public-ip-address` option with the [run-instances](#) command. (AWS CLI)
- Use the `--associate-public-ip-address` option with the [ec2-run-instances](#) command. (Amazon EC2 CLI)
- Use the `-AssociatePublicIp` parameter with the [New-EC2Instance](#) command. (AWS Tools for Windows PowerShell)

Modify a subnet's public IP addressing behavior

- [ec2-modify-subnet-attribute](#) (Amazon EC2 CLI)

Using Elastic Network Interfaces with Your VPC

Each instance in your VPC has a default network interface that is assigned a private IP address from the IP address range of your VPC. You can create and attach an additional network interface, known as an elastic network interface (ENI), to any instance in your VPC. The number of ENIs you can attach varies by instance type. For more information, see [Private IP Addresses Per ENI Per Instance Type](#) in the *Amazon Elastic Compute Cloud User Guide*.

An ENI is a virtual network interface that can include the following attributes:

- a primary private IP address
- one or more secondary private IP addresses
- one Elastic IP address per private IP address
- one public IP address, which can be auto-assigned to the network interface for `eth0` when you launch an instance, but only when you create a network interface for `eth0` instead of using an existing network interface

- one or more security groups
- a MAC address
- a source/destination check flag
- a description

You can create an ENI, attach it to an instance, detach it from an instance, and attach it to another instance. An ENI's attributes follow the ENI as it is attached or detached from an instance and reattached to another instance. When you move an ENI from one instance to another, network traffic is redirected to the new instance.

Attaching multiple ENIs to an instance is useful when you want to:

- Create a management network.
- Use network and security appliances in your VPC.
- Create dual-homed instances with workloads/roles on distinct subnets.
- Create a low-budget, high-availability solution.

For more information about ENIs, and step-by-step instructions for working with them using the Amazon EC2 console, see [Elastic Network Interfaces](#) in the *Amazon Elastic Compute Cloud User Guide*.

Route Tables

A *route table* contains a set of rules, called *routes*, that are used to determine where network traffic is directed.

Each subnet in your VPC must be associated with a route table; the table controls the routing for the subnet. You can associate multiple subnets with the same route table, but you can associate a subnet with only one route table.

Topics

- [Route Table Basics](#) (p. 91)
- [Main Route Tables](#) (p. 92)
- [Custom Route Tables](#) (p. 92)
- [Route Table Association](#) (p. 93)
- [Route Tables for VPC Peering Connections](#) (p. 94)
- [Working with Route Tables](#) (p. 95)
- [API and Command Overview](#) (p. 99)

Route Table Basics

The following are the basic things that you need to know about route tables:

- Your VPC has an implicit router.
- Your VPC automatically comes with a main route table that you can modify.
- You can create additional custom route tables for your VPC.
- Each subnet must be associated with a route table, which controls the routing for the subnet. If you don't explicitly associate a subnet with a particular route table, the subnet uses the main route table.

- You can replace the main route table with a custom table that you've created (so that this table is the default table each new subnet is associated with).
- Each route in a table specifies a destination CIDR and a target (for example, traffic destined for 172.16.0.0/12 is targeted for the virtual private gateway); we use the most specific route that matches the traffic to determine how to route the traffic.

Main Route Tables

When you create a VPC, it automatically has a main route table. On the **Route Tables** page in the VPC console, you can view the main route table for a VPC by looking for **Yes** in the **Main** column.

Initially, the main route table (and every route table in a VPC) contains only a single route: a local route that enables communication within the VPC.

You can't modify the local route in a route table. Whenever you launch an instance in the VPC, the local route automatically covers that instance; you don't need to add the new instance to a route table.

If you don't explicitly associate a subnet with a route table, the subnet is implicitly associated with the main route table. However, you can still explicitly associate a subnet with the main route table. You might do that if you change which table is the main route table (see [Replacing the Main Route Table \(p. 98\)](#)).

The console shows the number of subnets associated with each table. Only explicit associations are included in that number (see [Determining Which Subnets Are Explicitly Associated with a Table \(p. 96\)](#)).

When you add a gateway to a VPC (either an Internet gateway or a virtual private gateway), you must update the route table for any subnet that uses that gateway.

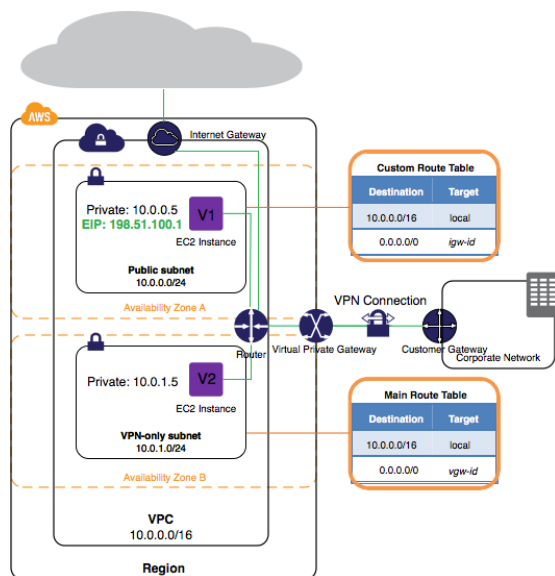
If you've attached a virtual private gateway to your VPC and enabled route propagation on your route table, routes representing your VPN connection automatically appear as propagated routes in your route table's list of routes.

Custom Route Tables

Your VPC can have route tables other than the default table. One way to protect your VPC is to leave the main route table in its original default state (with only the local route), and explicitly associate each new subnet you create with one of the custom route tables you've created. This ensures that you must explicitly control how each subnet's outbound traffic is routed.

For information about the limit on the number of route tables that you can create, see [Amazon VPC Limits \(p. 147\)](#).

The following diagram shows the routing for a VPC with both an Internet gateway and a virtual private gateway, plus a public subnet and a VPN-only subnet. The main route table came with the VPC, and it also has a route for the VPN-only subnet. There's a custom route table that's associated with the public subnet. The custom route table has a route for the public subnet over the Internet gateway (the destination is 0.0.0.0/0, and the target is the Internet gateway).



If you create a new subnet in this VPC, it would be automatically associated with the main route table, which routes its traffic to the virtual private gateway. If you were to set up the reverse configuration (the main route table with the route to the Internet gateway, and the custom route table with the route to the virtual private gateway), then if you create a new subnet, it would automatically have a route to the Internet gateway.

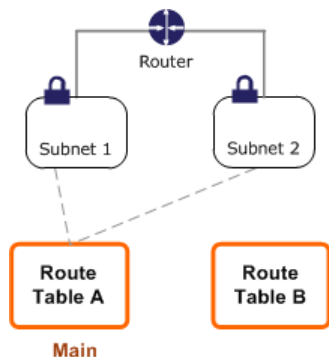
Route Table Association

The main route table is the default table that subnets use if they're not explicitly associated with another table. When you add a new subnet, it automatically uses the routes specified in the main route table. You can change which table is the main route table, and thus change the default for additional new subnets.

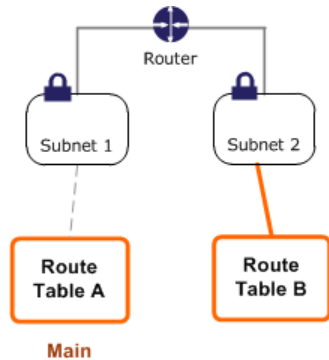
Subnets can be implicitly or explicitly associated with the main route table. Subnets typically won't have an explicit association to the main route table, although it might happen temporarily if you're replacing the main route table.

You might want to make changes to the main route table, but to avoid any disruption to your traffic, you decide to first test the route changes using a custom route table. After you're satisfied with the testing, you then replace the main route table with the new custom table.

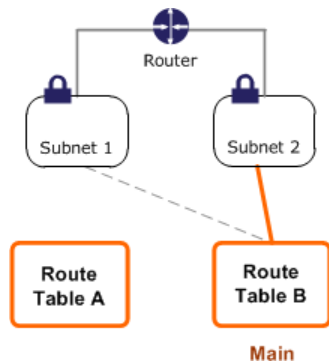
The following diagram shows a VPC with two subnets that are implicitly associated with the main route table (Route Table A), and a custom route table (Route Table B) that isn't associated with any subnets.



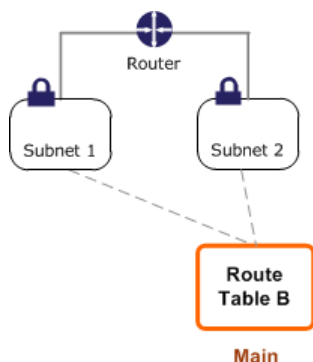
You can create an explicit association between Subnet 2 and Route Table B.



After you've tested Route Table B, you can make it the main route table. Note that Subnet 2 still has an explicit association with Route Table B, and Subnet 1 has an implicit association with Route Table B because it is the new main route table. Route Table A is no longer in use.



If you disassociate Subnet 2 from Route Table B, there's still an implicit association between Subnet 2 and Route Table B. If you no longer need Route Table A, you can delete it.



Route Tables for VPC Peering Connections

A VPC peering connection is a networking connection between two VPCs that allows you to route traffic between them using private IP addresses. Instances in either VPC can communicate with each other as if they are part of the same network.

To enable the routing of traffic between VPCs in a VPC peering connection, you must add a route to one or more of your VPC's route tables that points to the VPC peering connection to access all or part of the CIDR block of the other VPC in the peering connection. Similarly, the owner of the other VPC must add a route to their VPC's route table to route traffic back to your VPC.

For example, you have a VPC peering connection (`pcx-1a2b1a2b`) between two VPCs, with the following information:

- VPC A: `vpc-1111aaaa`, CIDR block is 10.0.0.0/16
- VPC B: `vpc-2222bbbb`, CIDR block is 172.31.0.0/16

To enable traffic between the VPCs and allow access to the entire CIDR block of either VPC, VPC A's route table is configured as follows:

Destination	Target
10.0.0.0/16	Local
172.31.0.0/16	pcx-1a2b1a2b

VPC B's route table is configured as follows:

Destination	Target
172.31.0.0/16	Local
10.0.0.0/16	pcx-1a2b1a2b

For more information about VPC peering connections, see the following topics:

- Working with VPC peering connections in the VPC console: [VPC Peering \(p. 138\)](#)
- Adding routes for VPC peering connections: [Updating Route Tables for Your VPC Peering Connection \(p. 143\)](#)
- Supported VPC peering connection scenarios and routing configurations: [Amazon Virtual Private Cloud Peering Guide](#)

Working with Route Tables

This section shows you how to work with route tables.

Note

When you use the wizard in the console to create a VPC with a gateway, the wizard automatically updates the route tables to use the gateway. If you're using the command line tools or API to set up your VPC, you must update the route tables yourself.

Topics

- [Determining Which Route Table a Subnet Is Associated With \(p. 96\)](#)
- [Determining Which Subnets Are Explicitly Associated with a Table \(p. 96\)](#)
- [Creating a Custom Route Table \(p. 97\)](#)
- [Adding and Removing Routes from a Route Table \(p. 97\)](#)
- [Enabling and Disabling Route Propagation \(p. 97\)](#)
- [Associating a Subnet with a Route Table \(p. 98\)](#)
- [Changing a Subnet's Route Table \(p. 98\)](#)

- [Disassociating a Subnet from a Route Table \(p. 98\)](#)
- [Replacing the Main Route Table \(p. 98\)](#)
- [Deleting a Route Table \(p. 99\)](#)

Determining Which Route Table a Subnet Is Associated With

You can determine which route table a subnet is associated with by looking at the subnet's details in the Amazon VPC Console.

To determine which route table a subnet is associated with

1. Open the Amazon VPC console.
2. Click **Subnets** in the navigation pane, and then select the subnet.

The subnet details are displayed in the **Summary** tab. Click the **Route Table** tab to view the route table ID and its routes. If it's the main route table, the console doesn't indicate whether the association is implicit or explicit. To determine if the association to the main route table is explicit, see [Determining Which Subnets Are Explicitly Associated with a Table \(p. 96\)](#).

Determining Which Subnets Are Explicitly Associated with a Table

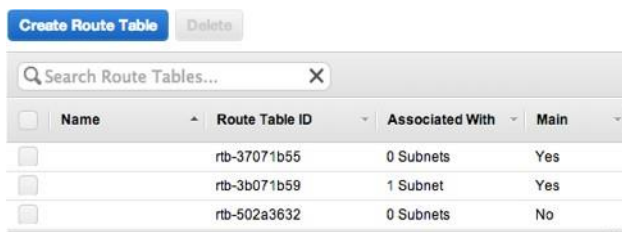
You can determine how many and which subnets are explicitly associated with a route table.

The main route table can have explicit and implicit associations. Custom route tables have only explicit associations.

Subnets that aren't explicitly associated with any route table have an implicit association with the main route table. You can explicitly associate a subnet with the main route table (for an example of why you might do that, see [Replacing the Main Route Table \(p. 98\)](#)).

To determine how many subnets are explicitly associated

1. Open the Amazon VPC console.
2. Click **Route Tables** in the navigation pane.
Check the **Associated With** column to determine the number of explicitly associated subnets.



Create Route Table Delete		<input type="text" value="Search Route Tables..."/>		
<input type="checkbox"/>	Name	Route Table ID	Associated With	Main
<input type="checkbox"/>		rtb-37071b55	0 Subnets	Yes
<input type="checkbox"/>		rtb-3b071b59	1 Subnet	Yes
<input type="checkbox"/>		rtb-502a3632	0 Subnets	No

To determine which subnets are explicitly associated

1. Select the route table of interest.
2. Click the **Subnet Associations** tab in the details pane. The subnets explicitly associated with the table are listed on the tab. Any subnets not associated with any route table (and thus implicitly associated with the main route table) are also listed.

Creating a Custom Route Table

Depending on your situation, you might need to create your own route tables.

To create a custom route table

1. Open the Amazon VPC console.
2. In the navigation pane, click **Route Tables**.
3. Click **Create Route Table**.
4. In the **Create Route Table** dialog box, you can optionally name your route table in the **Name tag** field. Doing so creates a tag with a key of `Name` and a value that you specify. Select your VPC from the **VPC** list, and then click **Yes, Create**.

Adding and Removing Routes from a Route Table

You can't modify routes in a table; you can only add and delete routes.

To add a route to a route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Routes** tab, click **Edit**.
4. Enter a destination CIDR block or a single IP address in the **Destination** field, and then select a target from the **Target** list. Click **Add another route** to add more routes, and then click **Save** when you're done.

To delete a route from a route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Routes** tab, click **Edit**, and then click the **Remove** button for the route you want to delete.
4. Click **Save** when you're done.

Enabling and Disabling Route Propagation

Route propagation allows a virtual private gateway to automatically propagate routes to the route tables so that you don't need to manually enter VPN routes to your route tables. You can enable or disable route propagation.

For more information about VPN routing options, see [VPN Routing Options \(p. 121\)](#).

To enable route propagation

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the details pane, click the **Route Propagation** tab.
4. Click **Edit**, and then select the virtual private gateway. Click **Save**.

To disable route propagation

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

2. In the navigation pane, click **Route Tables**, and then select the route table.
3. On the **Route Propagation** tab, click **Edit**, and then deselect the **Propagate** check box next to the ID of the virtual private gateway. Click **Save**.

Associating a Subnet with a Route Table

To apply a route table's routes to a particular subnet, you must associate the route table with the subnet. A route table can be associated with multiple subnets; however, a subnet can be associated with only one route table. Any subnet not explicitly associated with a table is implicitly associated with the main route table by default.

To associate a table with a subnet

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the details pane, on the **Subnet Associations** tab, click **Edit**.
4. Select the **Associate** check box for the subnet to associate with the route table, and then click **Save**.

Changing a Subnet's Route Table

You can change which route table a subnet is associated with. For example, when you create a subnet, it is implicitly associated with the main route table. You might want to instead associate it with a custom route table you've created.

To change a subnet's route table association

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Subnets**, and then select the subnet.
3. In the **Route Table** tab, click **Edit**.
4. Select the new route table to associate the subnet with from the **Change to** list, and then click **Save**.

Disassociating a Subnet from a Route Table

You might want to disassociate a subnet from a route table. For example, you might have a subnet that is associated with a custom route table, and you instead want it associated with the main route table. By disassociating the subnet from the custom route table, the subnet becomes implicitly associated with the main route table.

To disassociate a subnet from a route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Subnet Associations** tab, click **Edit**.
4. Deselect the **Associate** check box for the subnet, and then click **Save**.

Replacing the Main Route Table

The following procedure describes how to change which route table is the main route table in your VPC.

To replace the main route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. Click **Route Tables** in the navigation pane.
3. Locate the route table that you want to be the new main route table, right-click the table, and then select **Set as Main Table**.
4. In the **Set Main Route Table** dialog box, click **Yes, Set**.

The following procedure describes how to remove an explicit association between a subnet and the main route table. The result is an implicit association between the subnet and the main route table. The process is the same as disassociating any subnet from any route table.

To remove an explicit association with the main route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Subnet Associations** tab, click **Edit**.
4. Deselect the **Associate** check box for the subnet, and then click **Save**.

Deleting a Route Table

You can delete a route table only if there are no subnets associated with it. You can't delete the main route table.

To delete a route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**.
3. Select the route table, and then click the **Delete** button.
4. In the **Delete Route Table** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Create a custom route table

- [create-route-table](#) (AWS CLI)
- [ec2-create-route-table](#) (Amazon EC2 CLI)
- [New-EC2RouteTable](#) (AWS Tools for Windows PowerShell)

Add a route to a route table

- [create-route](#) (AWS CLI)
- [ec2-create-route](#) (Amazon EC2 CLI)
- [New-EC2Route](#) (AWS Tools for Windows PowerShell)

Associate a subnet with a route table

- [associate-route-table](#) (AWS CLI)
- [ec2-associate-route-table](#) (Amazon EC2 CLI)
- [Register-EC2RouteTable](#) (AWS Tools for Windows PowerShell)

Describe one or more route tables

- [describe-route-tables](#) (AWS CLI)
- [ec2-describe-route-tables](#) (Amazon EC2 CLI)
- [Get-EC2RouteTable](#) (AWS Tools for Windows PowerShell)

Delete a route from a route table

- [delete-route](#) (AWS CLI)
- [ec2-delete-route](#) (Amazon EC2 CLI)
- [Remove-EC2Route](#) (AWS Tools for Windows PowerShell)

Replace an existing route in a route table

- [replace-route](#) (AWS CLI)
- [ec2-replace-route](#) (Amazon EC2 CLI)
- [Set-EC2Route](#) (AWS Tools for Windows PowerShell)

Disassociate a subnet from a route table

- [disassociate-route-table](#) (AWS CLI)
- [ec2-disassociate-route-table](#) (Amazon EC2 CLI)
- [Unregister-EC2RouteTable](#) (AWS Tools for Windows PowerShell)

Change the route table associated with a subnet

- [replace-route-table-association](#) (AWS CLI)
- [ec2-replace-route-table-association](#) (Amazon EC2 CLI)
- [Set-EC2RouteTableAssociation](#) (AWS Tools for Windows PowerShell)

Create a static route associated with a VPN connection

- [create-vpn-connection-route](#) (AWS CLI)
- [ec2-create-vpn-connection-route](#) (Amazon EC2 CLI)
- [New-EC2VpnConnectionRoute](#) (AWS Tools for Windows PowerShell)

Delete a static route associated with a VPN connection

- [delete-vpn-connection-route](#) (AWS CLI)
- [ec2-delete-vpn-connection-route](#) (Amazon EC2 CLI)
- [Remove-EC2VpnConnectionRoute](#) (AWS Tools for Windows PowerShell)

Enable a virtual private gateway (VGW) to propagate routes to the routing tables of a VPC

- [enable-vgw-route-propagation](#) (AWS CLI)
- [ec2-enable-vgw-route-propagation](#) (Amazon EC2 CLI)
- [Enable-EC2VgwRoutePropagation](#) (AWS Tools for Windows PowerShell)

Disable a VGW from propagating routes to the routing tables of a VPC

- [disable-vgw-route-propagation](#) (AWS CLI)
- [ec2-disable-vgw-route-propagation](#) (Amazon EC2 CLI)
- [Disable-EC2VgwRoutePropagation](#) (AWS Tools for Windows PowerShell)

Delete a route table

- [delete-route-table](#) (AWS CLI)
- [ec2-delete-route-table](#) (Amazon EC2 CLI)
- [Remove-EC2RouteTable](#) (AWS Tools for Windows PowerShell)

Adding an Internet Gateway to Your VPC

By default, instances that you launch into a virtual private cloud (VPC) can't communicate with the Internet. You can enable access to the Internet from your VPC by attaching an Internet gateway to the VPC, ensuring that your instances have a public IP address, creating a custom route table, and updating your security group rules.

Your default VPC comes with an Internet gateway, and instances launched into a default subnet receive a public IP address by default, unless you specify otherwise during launch, or you modify the subnet's public IP address attribute. Therefore, instances that you launch into a default subnet can automatically communicate with the Internet. For more information, see [Your Default VPC and Subnets](#) (p. 46).

Instances that you launch into a nondefault subnet do not receive a public IP address by default and therefore can't communicate with the Internet, unless you specifically assign one during launch, or you modify the subnet's public IP address attribute. For more information about assigning a public IP address at launch, see [Assigning a Public IP Address During Launch](#) (p. 86). For more information about modifying your subnet's public IP addressing attribute, see [Modifying Your Subnet's Public IP Addressing Behavior](#) (p. 87). You can also enable Internet access for instances that you launch into a nondefault subnet by attaching an Internet gateway to the VPC, creating a custom route table, updating your security group rules, and associating an Elastic IP address with each instance.

When you add a new subnet to your VPC, you must set up the routing and security that you want for the subnet. You can do this manually, as described on this page, or use the VPC wizard to simplify the process. For example, depending on the option that you select, the VPC wizard adds an Internet gateway to your VPC and updates the route table so that your instances can communicate with the Internet. For more information about using the VPC wizard to create a subnet with an Internet gateway, see [Scenario 1: VPC with a Public Subnet Only](#) (p. 7) or [Scenario 2: VPC with Public and Private Subnets](#) (p. 12).

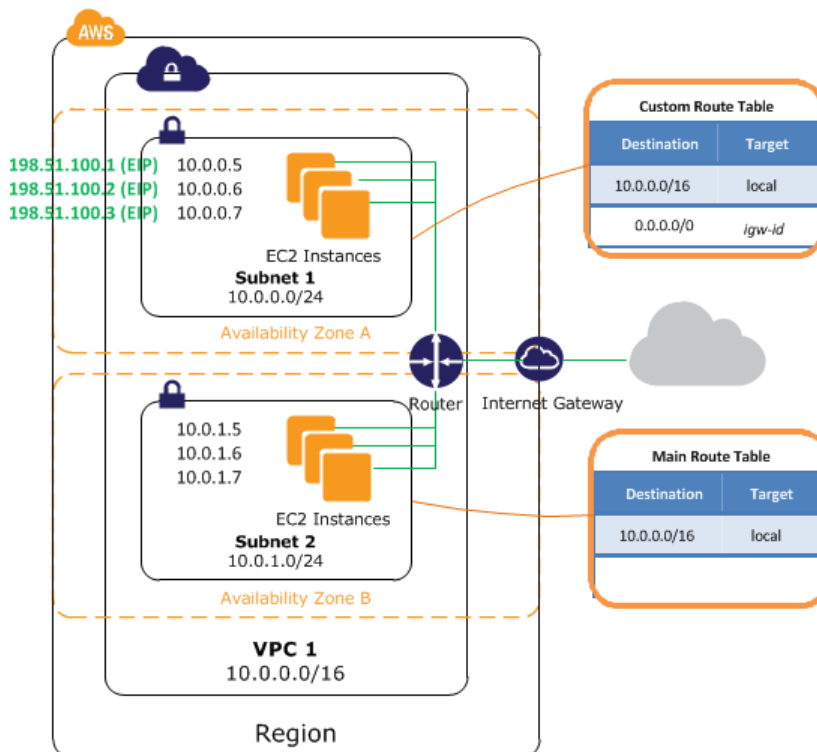
The following sections describe how to set up a subnet manually to support Internet access.

Topics

- [Creating a Subnet](#) (p. 102)
- [Attaching an Internet Gateway](#) (p. 102)
- [Creating a Custom Route Table](#) (p. 103)
- [Updating the Security Group Rules](#) (p. 103)

- [Adding Elastic IP Addresses \(p. 104\)](#)
- [Detaching an Internet Gateway from Your VPC \(p. 104\)](#)
- [Deleting an Internet Gateway \(p. 104\)](#)
- [API and Command Overview \(p. 105\)](#)

When you are finished setting up the subnet, your VPC is configured as shown in the following diagram.



Creating a Subnet

To add a subnet to your VPC

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Subnets**, and then click **Create Subnet**.
3. In the **Create Subnet** dialog box, select the VPC, select the Availability Zone, specify the CIDR range for the subnet, and then click **Yes, Create**.

For more information about subnets, see [Your VPC and Subnets \(p. 37\)](#).

Attaching an Internet Gateway

To create an Internet gateway and attach it to your VPC

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Internet Gateways**, and then click **Create Internet Gateway**.
3. In the **Create Internet Gateway** dialog box, you can optionally name your Internet gateway, and then click **Yes, Create**.

4. Select the Internet gateway that you just created, and then click **Attach to VPC**.
5. In the **Attach to VPC** dialog box, select your VPC from the list, and then click **Yes, Attach**.

Creating a Custom Route Table

When you create a subnet, we automatically associate it with the main route table for the VPC. By default, the main route table doesn't contain a route to an Internet gateway. The following procedure creates a custom route table with a route that sends traffic destined outside the VPC to the Internet gateway, and then associates it with your subnet.

To create a custom route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**, and then click **Create Route Table**.
3. In the **Create Route Table** dialog box, optionally name your route table, then select your VPC, and then click **Yes, Create**.
4. Select the custom route table that you just created. The details pane displays tabs for working with its routes, associations, and route propagation.
5. On the **Routes** tab, click **Edit**, specify `0.0.0.0/0` in the **Destination** box, select the Internet gateway ID in the **Target** list, and then click **Save**.
6. On the **Subnet Associations** tab, click **Edit**, select the **Associate** check box for the subnet, and then click **Save**.

For more information about route tables, see [Route Tables \(p. 91\)](#).

Updating the Security Group Rules

Your VPC comes with a default security group. Each instance that you launch into a VPC is automatically associated with its default security group. The default settings for a default security group allow no inbound traffic from the Internet and allow all outbound traffic to the Internet. Therefore, to enable your instances to communicate with the Internet, create a new security group that allows public instances to access the Internet.

To create a new security group and associate it with your instances

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Security Groups**, and then click **Create Security Group**.
3. In the **Create Security Group** dialog box, specify a name for the security group and a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
4. Select the security group. The details pane displays the details for the security group, plus tabs for working with its inbound rules and outbound rules.
5. On the **Inbound Rules** tab, click **Edit**. Click **Add Rule**, and complete the required information. For example, select **HTTP** or **HTTPS** from the **Type** list, and enter the **Source** as `0.0.0.0/0`. Click **Save** when you're done.
6. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
7. In the navigation pane, click **Instances**.
8. Right-click the instance, and then select **Change Security Groups**.
9. In the **Change Security Groups** dialog box, clear the check box for the currently selected security group, and select the new one. Click **Assign Security Groups**.

For more information about security groups, see [Security Groups for Your VPC \(p. 53\)](#).

Adding Elastic IP Addresses

After you've launched an instance into the subnet, you must assign it an Elastic IP address if you want it to be reachable from the Internet.

Note

If you assigned a public IP address to your instance during launch, then your instance is reachable from the Internet, and you do not need to assign it an Elastic IP address. For more information about IP addressing for your instance, see [IP Addressing in Your VPC \(p. 85\)](#).

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, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. In the **Allocate New Address** dialog box, in the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.
5. Select the Elastic IP address from the list, and then click the **Associate Address** button.
6. In the **Associate Address** dialog box, select **Instance** or **Network Interface** from the **Associate with** list, and then either the instance or network interface ID. Select the private IP address to associate the Elastic IP address with from the **Private IP address** list, and then click **Yes, Associate**.

For more information about Elastic IP addresses, see [Elastic IP Addresses \(p. 87\)](#).

Detaching an Internet Gateway from Your VPC

If you no longer need Internet access for instances that you launch into a nondefault VPC, you can detach an Internet gateway from a VPC. You can't detach an Internet gateway if the VPC has instances with associated Elastic IP addresses.

To detach an Internet gateway

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Elastic IPs**.
3. Select the IP address, click the **Disassociate Address** button, and then click **Yes, Disassociate**.
4. In the navigation pane, click **Internet Gateways**.
5. Select the Internet gateway and click **Detach from VPC**.
6. In the **Detach from VPC** dialog box, click **Yes, Detach**.

Deleting an Internet Gateway

If you no longer need an Internet gateway, you can delete it. You can't delete an Internet gateway if it's still attached to a VPC.

To delete an Internet gateway

1. Select the Internet gateway and click **Delete**.
2. In the **Delete Internet Gateway** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Create an Internet gateway

- [create-internet-gateway](#) (AWS CLI)
- [ec2-create-internet-gateway](#) (Amazon EC2 CLI)
- [New-EC2InternetGateway](#) (AWS Tools for Windows PowerShell)

Attach an Internet gateway to a VPC

- [attach-internet-gateway](#) (AWS CLI)
- [ec2-attach-internet-gateway](#) (Amazon EC2 CLI)
- [Add-EC2InternetGateway](#) (AWS Tools for Windows PowerShell)

Describe an Internet gateway

- [describe-internet-gateways](#) (AWS CLI)
- [ec2-describe-internet-gateways](#) (Amazon EC2 CLI)
- [Get-EC2InternetGateway](#) (AWS Tools for Windows PowerShell)

Detach an Internet gateway from a VPC

- [detach-internet-gateway](#) (AWS CLI)
- [ec2-detach-internet-gateway](#) (Amazon EC2 CLI)
- [Dismount-EC2InternetGateway](#) (AWS Tools for Windows PowerShell)

Delete an Internet gateway

- [delete-internet-gateway](#) (AWS CLI)
- [ec2-delete-internet-gateway](#) (Amazon EC2 CLI)
- [Remove-EC2InternetGateway](#) (AWS Tools for Windows PowerShell)

NAT Instances

Instances that you launch into a private subnet in a virtual private cloud (VPC) can't communicate with the Internet. You can optionally use a network address translation (NAT) instance in a public subnet in your VPC to enable instances in the private subnet to initiate outbound traffic to the Internet, but prevent the instances from receiving inbound traffic initiated by someone on the Internet.

For a general overview of VPCs and subnets, see [What is Amazon VPC? \(p. 1\)](#). For more information about public and private subnets, see [Subnet Routing \(p. 42\)](#).

Note

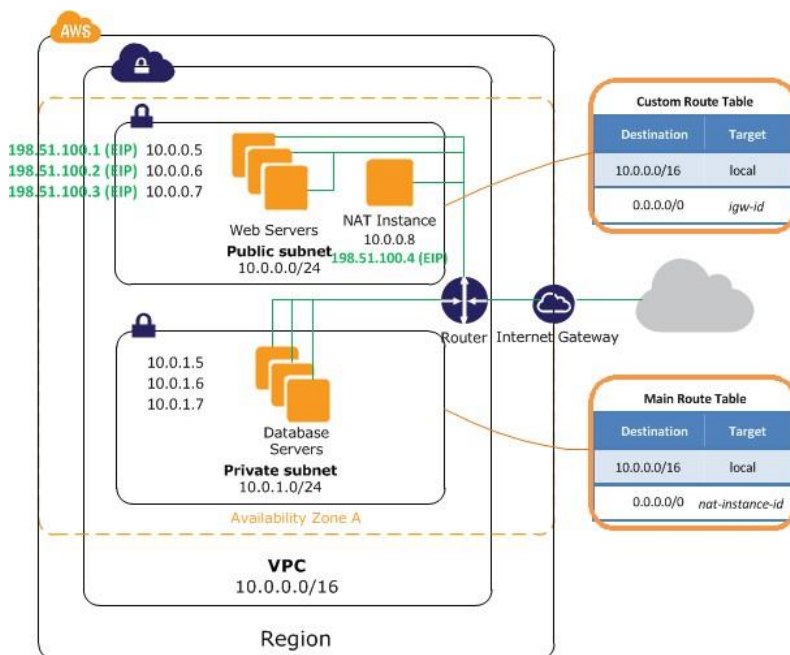
We use the term *NAT instance*; however, the primary role of a NAT instance is actually port address translation (PAT). We chose to use the more widely known term, NAT. For more information about NAT and PAT, see the [Wikipedia article about network address translation](#).

Topics

- [NAT Instance Basics](#) (p. 106)
- [Setting up the NAT Instance](#) (p. 106)
- [Creating the NATSG Security Group](#) (p. 108)
- [Disabling Source/Destination Checks](#) (p. 109)
- [Updating the Main Route Table](#) (p. 110)
- [Testing Your NAT Instance Configuration](#) (p. 110)

NAT Instance Basics

The following figure illustrates the NAT instance basics. The main route table sends the traffic from the instances in the private subnet to the NAT instance in the public subnet. The NAT instance sends the traffic to the Internet gateway for the VPC. The traffic is attributed to the Elastic IP address of the NAT instance. The NAT instance specifies a high port number for the response; if a response comes back, the NAT instance sends it to an instance in the private subnet based on the port number for the response.



Setting up the NAT Instance

You can use the VPC wizard to set up a VPC with a NAT instance; for more information, see [Scenario 2: VPC with Public and Private Subnets](#) (p. 12). Otherwise, you can set up the NAT instance manually using the steps below.

1. Create a VPC with two subnets.
 - a. Create a VPC (see [Creating a VPC](#) (p. 39))
 - b. Create two subnets (see [Creating a Subnet](#) (p. 102))
 - c. Attach an Internet gateway to the VPC (see [Attaching an Internet Gateway](#) (p. 102))
 - d. Create a custom route table that sends traffic destined outside the VPC to the Internet gateway, and then associate it with one subnet, making it a public subnet (see [Creating a Custom Route Table](#) (p. 103))

2. Create the NATSG security group (see [Creating the NATSG Security Group \(p. 108\)](#)). You'll specify this security group when you launch the NAT instance.
3. Launch an instance into your public subnet from an AMI that's been configured to run as a NAT instance. Amazon provides Amazon Linux AMIs that are configured to run as NAT instances. These AMIs include the string `amzn-ami-vpc-nat` in their names, so you can search for them in the Amazon EC2 console.
 - a. Open the Amazon EC2 console.
 - b. On the dashboard, click the **Launch Instance** button, and complete the wizard as follows:
 - i. On the **Choose an Amazon Machine Image (AMI)** page, select the **Community AMIs** category, and search for `amzn-ami-vpc-nat`. In the results list, each AMI's name includes the version to enable you to select the most recent AMI, for example, `2013.09`. Click **Select**.
 - ii. On the **Choose an Instance Type** page, select the instance type, then click **Next: Configure Instance Details**.
 - iii. On the **Configure Instance Details** page, select the VPC you created from the **Network** list, and select your public subnet from the **Subnet** list.
 - iv. (Optional) Select the **Public IP** check box to request that your NAT instance receives a public IP address. If you choose not to assign a public IP address now, you can allocate an Elastic IP address and assign it to your instance after it's launched. For more information about assigning a public IP at launch, see [Assigning a Public IP Address During Launch \(p. 86\)](#). Click **Next: Add Storage**.
 - v. You can choose to add storage to your instance, and on the next page, you can add tags. Click **Next: Configure Security Group** when you are done.
 - vi. On the **Configure Security Group** page, select the **Select an existing security group** option, and select the NATSG security group that you created. Click **Review and Launch**.
- vii. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.
4. (Optional) Log on to the NAT instance, make any modifications that you need, and then create your own AMI that's configured to run as a NAT instance. You can use this AMI the next time that you need to launch a NAT instance. For more information about creating an AMI, see [Creating Amazon EBS-Backed AMIs](#) in the *Amazon Elastic Compute Cloud User Guide*.
5. Disable the `SrcDestCheck` attribute for the NAT instance (see [Disabling Source/Destination Checks \(p. 109\)](#))
6. If you did not assign a public IP address to your NAT instance during launch (step 3), you need to associate an Elastic IP address with it.
 - a. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 - b. Click **Elastic IPs** in the navigation pane.
 - c. Click the **Allocate New Address** button.
 - d. In the **Allocate New Address** dialog box, in the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.
 - e. Select the Elastic IP address from the list, and then click the **Associate Address** button.
 - f. In the **Associate Address** dialog box, select the network interface for the NAT instance. Select the address to associate the EIP with from the **Private IP address** list, and then click **Yes, Associate**.
7. Update the main route table to send traffic to the NAT instance. For more information, see [Updating the Main Route Table \(p. 110\)](#).

Launching a NAT Instance Using the Command Line

To launch a NAT instance into your subnet, use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon VPC \(p. 5\)](#).

- [run-instances](#) (AWS CLI)
- [ec2-run-instances](#) (Amazon EC2 CLI)
- [New-EC2Instance](#) (AWS Tools for Windows PowerShell)

To get the ID of an AMI that's configured to run as a NAT instance, use a command to describe images, and use filters to return results only for AMIs that are owned by Amazon, and that have the `amzn-ami-vpc-nat` string in their names. The following example uses the AWS CLI:

```
PROMPT> aws ec2 describe-images --filter Name="owner-alias",Values="amazon" --filter Name="name",Values="amzn-ami-vpc-nat*"
```

Creating the NATSG Security Group

Define the NATSG security group as described in the following table to enable your NAT instance to receive Internet-bound traffic from instances in a private subnet, as well as SSH traffic from your network. The NAT instance can also send traffic to the Internet, which enables the instances in the private subnet to get software updates.

NATSG: Recommended Rules

Inbound			
Source	Protocol	Port Range	Comments
10.0.1.0/24	TCP	80	Allow inbound HTTP traffic from servers in the private subnet
10.0.1.0/24	TCP	443	Allow inbound HTTPS traffic from servers in the private subnet
Public IP address range of your network	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
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet

To create the NATSG security group

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. Click **Security Groups** in the navigation pane.
3. Click the **Create Security Group** button.

4. In the **Create Security Group** dialog box, specify `NATSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
5. Select the NATSG security group that you just created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
6. Add rules for inbound traffic using the **Inbound Rules** tab as follows:
 - a. Click **Edit**.
 - b. Click **Add another rule**, and select **HTTP** from the **Type** list. In the **Source** field, specify the IP address range of your private subnet.
 - c. Click **Add another rule**, and select **HTTPS** from the **Type** list. In the **Source** field, specify the IP address range of your private subnet.
 - d. Click **Add another rule**, and select **SSH** from the **Type** list. In the **Source** field, specify the public IP address range of your network.
 - e. Click **Save**.
7. Add rules for outbound traffic using the **Outbound Rules** tab as follows:
 - a. Click **Edit**.
 - b. Click **Add another rule**, and select **HTTP** from the **Type** list. In the **Source** field, specify `0.0.0.0/0`
 - c. Click **Add another rule**, and select **HTTPS** from the **Type** list. In the **Source** field, specify `0.0.0.0/0`
 - d. Click **Save**.

For more information about security groups, see [Security Groups for Your VPC \(p. 53\)](#).

Disabling Source/Destination Checks

Each EC2 instance performs source/destination checks by default. This means that the instance must be the source or destination of any traffic it sends or receives. However, a NAT instance must be able to send and receive traffic when the source or destination is not itself. Therefore, you must disable source/destination checks on the NAT instance.

You can disable the `SrcDestCheck` attribute for a NAT instance that's either running or stopped using the console or the command line.

To disable source/destination checking using the console

1. Open the Amazon EC2 console.
2. Click **Instances** in the navigation pane.
3. Select the NAT instance, click **Actions**, and then click **Change Source/Dest. Check**.
4. For a NAT instance, verify that this attribute is disabled. Otherwise, click **Yes, Disable**.

To disable source/destination checking using the command line

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon VPC \(p. 5\)](#).

- `modify-instance-attribute` (AWS CLI)
- `ec2-modify-instance-attribute` (Amazon EC2 CLI)

- [Edit-EC2InstanceAttribute](#) (AWS Tools for Windows PowerShell)

Updating the Main Route Table

Update the main route table as described in the following procedure. By default, the main route table enables the instances in your VPC to communicate with each other. We'll add a route that sends all other subnet traffic to the NAT instance.

To update the main route table

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Route Tables**.
3. Select the main route table for your VPC. The details pane displays tabs for working with its routes, associations, and route propagation.
4. On the **Routes** tab, click **Edit**, specify `0.0.0.0/0` in the **Destination** box, select the instance ID of the NAT instance from the **Target** list, and then click **Save**.
5. On the **Subnet Associations** tab, click **Edit**, and then select the **Associate** check box for the subnet. Click **Save**.

For more information about route tables, see [Route Tables](#) (p. 91).

Testing Your NAT Instance Configuration

After you have launched a NAT instance and completed the configuration steps above, you can perform a test to check if an instance in your private subnet can access the Internet through the NAT instance. To do this, update your NAT instance's security group rules to accept inbound ICMP traffic, launch an instance into your private subnet, connect to your instance, and then test the Internet connectivity.

To update your NAT instance's security group

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Security Groups**.
3. Find the security group associated with your NAT instance, and click **Edit** in the **Inbound** tab.
4. Click **Add Rule**, select **All ICMP** from the **Type** list, and select **Custom IP** from the **Source** list. Enter the IP address range of your private subnet, for example, `10.0.1.0/24`. Click **Save**.

To launch an instance into your private subnet

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Instances**.
3. Launch an instance into your private subnet. For more information, see [Launching an Instance into Your Subnet](#) (p. 43). Ensure that you configure the following options in the launch wizard, and then click **Launch**:
 - On the **Choose an Amazon Machine Image (AMI)** page, select an Amazon Linux AMI from the **Quick Start** category.
 - On the **Configure Instance Details** page, select your private subnet from the **Subnet** list, and do not assign a public IP address to your instance.
 - On the **Configure Security Group** page, ensure that your security group includes a rule that allows SSH access from your NAT instance's private IP address, or from the IP address range of your public subnet.

4. Locate the private key pair file for your instance, and copy it to your NAT instance. For more information about transferring files to your instance from a Windows computer, see [Using PuTTY Secure Copy Client](#) in the *Amazon Elastic Compute Cloud User Guide*. For more information about transferring files to your instance from a Linux/Unix computer, see [Transferring Files Using SCP](#).

To connect to your instance and test the Internet connection

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Instances**.
3. Connect to your NAT instance. For more information about connecting to an instance, see [Connect to Your Instance](#) in the *Amazon Elastic Compute Cloud User Guide*.
4. Test that your NAT instance can communicate with the Internet by running the `ping` command for a website that has ICMP enabled; for example:

```
PROMPT> ping ietf.org

PING ietf.org (4.31.198.44) 56(84) bytes of data.
64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=1 ttl=48 time=74.9 ms
64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=2 ttl=48 time=75.1 ms
...
```

Press **Ctrl+C** on your keyboard to cancel the `ping` command.

5. From your NAT instance, use the `chmod` command to make sure that your public key file isn't publicly viewable:

```
PROMPT> chmod 400 my-key-pair.pem
```

6. From your NAT instance, connect to your instance in your private subnet by using its private IP address, for example:

```
PROMPT> ssh -i my-key-pair.pem ec2-user@10.0.1.123
```

7. From your private instance, test that you can connect to the Internet by running the `ping` command:

```
PROMPT> ping ietf.org

PING ietf.org (4.31.198.44) 56(84) bytes of data.
64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=1 ttl=47 time=86.0 ms
64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=2 ttl=47 time=75.6 ms
...
```

Press **Ctrl+C** on your keyboard to cancel the `ping` command.

If the `ping` command fails, check the following information:

- Check that your NAT instance's security group rules allow inbound ICMP traffic from your private subnet. If not, your NAT instance cannot receive the `ping` command from your private instance.
- Check that you've configured your route tables correctly. For more information, see [Updating the Main Route Table \(p. 110\)](#).
- Ensure that you've disabled source/destination checking for your NAT instance. For more information, see [Disabling Source/Destination Checks \(p. 109\)](#).

- Ensure that you are pinging a website that has ICMP enabled. If not, you will not receive reply packets. To test this, perform the same `ping` command from the command line terminal on your own computer.
8. (Optional) Terminate your private instance if you no longer require it. For more information, see [Terminate Your Instance](#) in the *Amazon Elastic Compute Cloud User Guide*.

DHCP Options Sets

This topic describes DHCP options sets and how to specify the DHCP options for your VPC.

Topics

- [Overview of DHCP Options Sets](#) (p. 112)
- [Amazon DNS Server](#) (p. 113)
- [Changing DHCP Options](#) (p. 113)
- [Working with DHCP Options Sets](#) (p. 113)
- [API and Command Overview](#) (p. 115)

Overview of DHCP Options Sets

The Dynamic Host Configuration Protocol (DHCP) provides a standard for passing configuration information to hosts on a TCP/IP network. The *options* field of a DHCP message contains the configuration parameters. Some of those parameters are the domain name, domain name server, and the netbios-node-type.

DHCP options sets are associated with your AWS account so that you can use them across all of your virtual private clouds (VPC).

The Amazon EC2 instances you launch into a nondefault VPC are private by default; they're not assigned a public IP address unless you specifically assign one during launch, or you modify the subnet's public IP address attribute. By default, all instances in a nondefault VPC receive an unresolvable host name that AWS assigns (for example, ip-10-0-0-202). You can assign your own domain name to your instances, and use up to four of your own DNS servers. To do that, you must specify a special set of DHCP options to use with the VPC. This set can contain other commonly used DHCP options (see the following table for the full list of supported options). For more information about the options, see [RFC 2132](#).

DHCP Option Name	Description
domain-name-servers	The IP addresses of up to four domain name servers, or AmazonProvidedDNS. The default DHCP option set specifies AmazonProvidedDNS.
domain-name	If you're using AmazonProvidedDNS in <code>us-east-1</code> , specify <code>ec2.internal</code> . If you're using AmazonProvidedDNS in another region, specify <code>region.compute.internal</code> (for example, <code>ap-northeast-1.compute.internal</code>). Otherwise, specify a domain name (for example, <code>MyCompany.com</code>).
ntp-servers	The IP addresses of up to four Network Time Protocol (NTP) servers.
netbios-name-servers	The IP addresses of up to four NetBIOS name servers.

DHCP Option Name	Description
netbios-node-type	The NetBIOS node type (1, 2, 4, or 8). We recommend that you specify 2 (broadcast and multicast are not currently supported). For more information about these node types, see RFC 2132 .

Amazon DNS Server

When you create a VPC, we automatically create a set of DHCP options and associate them with the VPC. This set includes two options: `domain-name-servers=AmazonProvidedDNS`, and `domain-name=domain-name-for-your-region`. `AmazonProvidedDNS` is an Amazon DNS server, and this option enables DNS for instances that need to communicate over the VPC's Internet gateway. The string `AmazonProvidedDNS` maps to a DNS server running on a reserved IP address at the base of the VPC network range "plus two". For example, the DNS Server on a 10.0.0.0/16 network is located at 10.0.0.2.

Services that use the Hadoop framework, such as Amazon EMR, require instances to resolve their own fully qualified domain names (FQDN). In such cases, DNS resolution can fail if the `domain-name-servers` option is set to a custom value. To ensure proper DNS resolution, consider adding a conditional forwarder on your DNS server to forward queries for the domain `region-name.compute.internal` to the Amazon DNS server. For more information about launching an Amazon EMR cluster into a VPC, see [Setting Up a VPC to Host Clusters](#) in the *Amazon Elastic MapReduce Developer Guide*.

Note

You can use the Amazon DNS server IP address 169.254.169.253, though some servers don't allow its use. Windows Server 2008, for example, disallows the use of a DNS server located in the 169.254.x.x network range.

Changing DHCP Options

After you create a set of DHCP options, you can't modify them. If you want your VPC to use a different set of DHCP options, you must create a new set and associate them with your VPC. You can also set up your VPC to use no DHCP options at all.

You can have multiple sets of DHCP options, but you can associate only one set of DHCP options with a VPC at a time. If you delete a VPC, the DHCP options set associated with the VPC are also deleted.

After you associate a new set of DHCP options with a VPC, any existing instances and all new instances that you launch in the VPC use these options. You don't need to restart or relaunch the instances. They automatically pick up the changes within a few hours, depending on how frequently the instance renews its DHCP lease. If you want, you can explicitly renew the lease using the operating system on the instance.

Working with DHCP Options Sets

This section shows you how to work with DHCP options sets.

Topics

- [Creating a DHCP Options Set \(p. 114\)](#)
- [Changing the Set of DHCP Options a VPC Uses \(p. 114\)](#)
- [Changing a VPC to use No DHCP Options \(p. 115\)](#)
- [Deleting a DHCP Options Set \(p. 115\)](#)

Creating a DHCP Options Set

You can create as many additional DHCP options sets as you want. However, you can only associate a VPC with one set of DHCP options at a time. After you create a set of DHCP options, you must configure your VPC to use it. For more information, see [Changing the Set of DHCP Options a VPC Uses](#) (p. 114).

To create a DHCP options set

1. Open the Amazon VPC console.
2. Click **DHCP Options Sets** in the navigation pane, and then click the **Create DHCP Options Set** button.
3. In the **Create DHCP Options Set** dialog box, enter values for the options that you want to use, and then click **Yes, Create**.

Important

If your VPC has an Internet gateway, make sure to specify your own DNS server or Amazon's DNS server (AmazonProvidedDNS) for the **Domain name servers** value. Otherwise, the instances that need to communicate with the Internet won't have access to DNS.

The new set of DHCP options appears in your list of DHCP options. The following image shows an example of the list, with both the set of DHCP options you just created and the set that automatically came with your VPC (where the only option is domain-name-servers=AmazonProvidedDNS).



	Name	DHCP Options Set ID	Options
<input checked="" type="checkbox"/>	My DHCP Options	dopt-89dcd6eb	domain-name = myCompany.com;domain-name-servers = 172.16.16.16;
<input type="checkbox"/>		dopt-e0fe0e88	domain-name = ap-southeast-1.compute.internal;domain-name-servers = AmazonProvidedDNS;

4. Make a note of the ID of the new set of DHCP options (dopt-xxxxxxx). You will need it to associate the new set of options with your VPC.

Although you've created a set of DHCP options, you must associate it with your VPC for the options to take effect. You can create multiple sets of DHCP options, but you can associate only one set of DHCP options with your VPC at a time.

Changing the Set of DHCP Options a VPC Uses

You can change which set of DHCP options your VPC uses. If you want the VPC to use no DHCP options, see [Changing a VPC to use No DHCP Options](#) (p. 115).

Note

The following procedure assumes that you've already created the DHCP options set you want to change to. If you haven't, create the options set now. For more information, see [Creating a DHCP Options Set](#) (p. 114).

To change the DHCP options set associated with a VPC

1. Open the Amazon VPC console.
2. Click **Your VPCs** in the navigation pane.
3. Select the VPC, and click **Edit** in the **Summary** tab.
4. In the **DHCP options set** field, select a set of options from the list, and then click **Save**.

After you associate a new set of DHCP options with the VPC, any existing instances and all new instances that you launch in that VPC use the options. You don't need to restart or relaunch the instances. They automatically pick up the changes within a few hours, depending on how frequently the instance renews its DHCP lease. If you want, you can explicitly renew the lease using the operating system on the instance.

Changing a VPC to use No DHCP Options

You can set up your VPC to use no set of DHCP options.

1. Open the Amazon VPC console.
2. Click **Your VPCs** in the navigation pane.
3. Select the VPC, and click **Edit** in the **Summary** tab.
4. In the **DHCP options set** field, select **Default** from the list, and then click **Save**.

You don't need to restart or relaunch the instances. They automatically pick up the changes within a few hours, depending on how frequently the instance renews its DHCP lease. If you want, you can explicitly renew the lease using the operating system on the instance.

Deleting a DHCP Options Set

When you no longer need a DHCP options set, use the following procedure to delete it. The VPC must not be using the set of options.

To delete a DHCP options set

1. Open the Amazon VPC console.
2. Click **DHCP Options Sets** in the navigation pane.
3. Select the set of DHCP options to delete, and then click **Delete**.
4. In the **Delete DHCP Options Set** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Create a set of DHCP options for your VPC

- [create-dhcp-options](#) (AWS CLI)
- [ec2-create-dhcp-options](#) (Amazon EC2 CLI)
- [New-EC2DhcpOption](#) (AWS Tools for Windows PowerShell)

Associate a set of DHCP options with the specified VPC, or no DHCP options

- [associate-dhcp-options](#) (AWS CLI)
- [ec2-associate-dhcp-options](#) (Amazon EC2 CLI)
- [Register-EC2DhcpOption](#) (AWS Tools for Windows PowerShell)

Describes one or more sets of DHCP options

- [describe-dhcp-options](#) (AWS CLI)
- [ec2-describe-dhcp-options](#) (Amazon EC2 CLI)

- [Get-EC2DhcpOption](#) (AWS Tools for Windows PowerShell)

Deletes a set of DHCP options

- [delete-dhcp-options](#) (AWS CLI)
- [ec2-delete-dhcp-options](#) (Amazon EC2 CLI)
- [Remove-EC2DhcpOption](#) (AWS Tools for Windows PowerShell)

Using DNS with Your VPC

Amazon EC2 instances need IP addresses to communicate. Public IP addresses enable communication over the Internet, while private IP addresses enable communication within the network of the instance (either EC2-Classic or a VPC).

Domain Name System (DNS) is a standard by which names used on the Internet are resolved to their corresponding IP addresses. A DNS hostname is a name that uniquely and absolutely names a computer; it's composed of a host name and a domain name. DNS servers resolve DNS hostnames to their corresponding IP addresses.

We provide an Amazon DNS server. To use your own DNS server, update the DHCP options set for your VPC. For more information, see [DHCP Options Sets \(p. 112\)](#).

To enable an EC2 instance to be publicly accessible, it must have a public IP address, a DNS hostname, and DNS resolution.

Viewing DNS Hostnames for Your EC2 Instance

When you launch an instance into the EC2-Classic platform or into a default VPC, we provide the instance with public and private DNS hostnames. Instances that you launch into a nondefault VPC might have public and private DNS hostnames, depending on the settings you specify for the VPC and for the instance.

You can view the DNS hostnames for a running instance or a network interface using the Amazon EC2 console or the command line.

Instance

To view DNS hostnames for an instance using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Instances**.
3. Select the instance from the list.
4. In the **Description** tab in the details pane, review the values of the **Public DNS** and **Private DNS** fields.

To view DNS hostnames for an instance using the command line

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon VPC \(p. 5\)](#).

- [describe-instances](#) (AWS CLI)
- [ec2-describe-instances](#) (Amazon EC2 CLI)
- [Get-EC2Instance](#) (AWS Tools for Windows PowerShell)

Network Interface

To view DNS hostnames for a network interface using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Network Interfaces**.
3. Select the network interface from the list.
4. In the **Details** tab for the network interface, review the values of the **Public DNS** and **Private DNS** fields.

To view DNS hostnames for a network interface using the command line

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon VPC \(p. 5\)](#).

- [describe-network-interfaces](#) (AWS CLI)
- [ec2-describe-network-interfaces](#) (Amazon EC2 CLI)
- [Get-EC2NetworkInterface](#) (AWS Tools for Windows PowerShell)

Updating DNS Support for Your VPC

When you launch an instance into a VPC, we provide the instance with public and private DNS hostnames only if DNS hostnames are enabled for the VPC. By default, DNS hostnames are enabled only for default VPCs and VPCs that you create using the VPC console.

We support the following VPC attributes to control DNS support. Be sure to set both attributes to `true` if you want your instances that have public DNS hostnames that are accessible from the Internet.

Attribute	Description
<code>enableDnsHostnames</code>	Indicates whether the instances launched in the VPC get DNS hostnames. If this attribute is <code>true</code> , instances in the VPC get DNS hostnames; otherwise, they do not.
<code>enableDnsSupport</code>	Indicates whether the DNS resolution is supported for the VPC. If this attribute is <code>false</code> , the Amazon provided DNS service in the VPC that resolves public DNS hostnames to IP addresses is not enabled. If this attribute is <code>true</code> , queries to the Amazon provided DNS server at the 169.254.169.253 IP address, or the reserved IP address at the base of the VPC network range "plus two" will succeed. For more information, see Amazon DNS Server (p. 113) .

If you enable DNS hostnames in a VPC that didn't previously support them, an instance that you already launched into that VPC gets a public DNS hostname if it has a public IP address or an Elastic IP address.

To describe and update DNS support for a VPC using the console

1. Open the Amazon VPC console.
2. In the navigation pane, click **Your VPCs**.

3. Select the VPC from the list.
4. Review the information in the **Summary** tab. In this example, both settings are enabled.

DNS resolution: yes

DNS hostnames: yes

5. To update these setting, click **Edit**, select either the **Yes** or **No** option for the **DNS resolution** and **DNS hostname** fields, and then click **Save**.

To describe DNS support for a VPC using the command line

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon VPC \(p. 5\)](#).

- [describe-vpc-attribute](#) (AWS CLI)
- [ec2-describe-vpc-attribute](#) (Amazon EC2 CLI)
- [Get-EC2VpcAttribute](#) (AWS Tools for Windows PowerShell)

To update DNS support for a VPC using the command line

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon VPC \(p. 5\)](#).

- [modify-vpc-attribute](#) (AWS CLI)
- [ec2-modify-vpc-attribute](#) (Amazon EC2 CLI)
- [Edit-EC2VpcAttribute](#) (AWS Tools for Windows PowerShell)

Adding a Hardware Virtual Private Gateway to Your VPC

By default, instances that you launch into a virtual private cloud (VPC) can't communicate with your own network. You can enable access to your network from your VPC by attaching a virtual private gateway to the VPC, creating a custom route table, and updating your security group rules.

You can complete this process manually, as described on this page, or let the VPC creation wizard take care of many of these steps for you. For more information about using the VPC creation wizard to set up the virtual private gateway, see [Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access](#) (p. 22) or [Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access](#) (p. 31).

Although the term *VPN connection* is a general term, in the Amazon VPC documentation, a VPN connection refers to the connection between your VPC and your own network.

Topics

- [Components of Your VPN](#) (p. 119)
- [VPN Configuration Examples](#) (p. 120)
- [VPN Routing Options](#) (p. 121)
- [What You Need for a VPN Connection](#) (p. 121)
- [Configuring Two VPN Tunnels for Your VPN Connection](#) (p. 122)
- [Using Redundant VPN Connections to Provide Failover](#) (p. 123)
- [Setting Up the VPN Connection](#) (p. 124)
- [Testing the End-to-End Connectivity of Your Instance](#) (p. 127)
- [Replacing Compromised Credentials](#) (p. 128)
- [Deleting a VPN Connection](#) (p. 128)

For information about how you're charged for using a VPN connection with your VPC, see the [Amazon VPC product page](#).

Components of Your VPN

A VPN connection consists of the following components.

Virtual Private Gateway

A *virtual private gateway* is the VPN concentrator on the Amazon side of the VPN connection.

For information about how many virtual private gateways you can have per region, as well as the limits for other components within your VPC, see [Amazon VPC Limits \(p. 147\)](#).

Customer Gateway

A *customer gateway* is a physical device or software application on your side of the VPN connection.

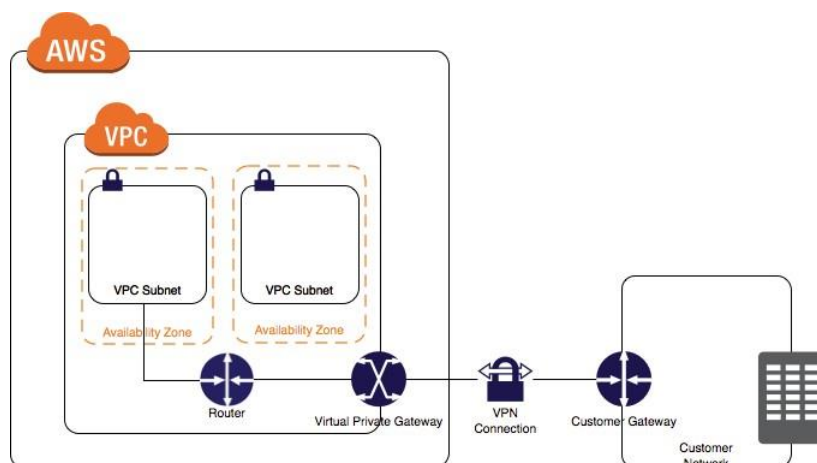
For a list of customer gateways that we have tested with Amazon VPC, see [Amazon Virtual Private Cloud FAQs](#).

VPN Configuration Examples

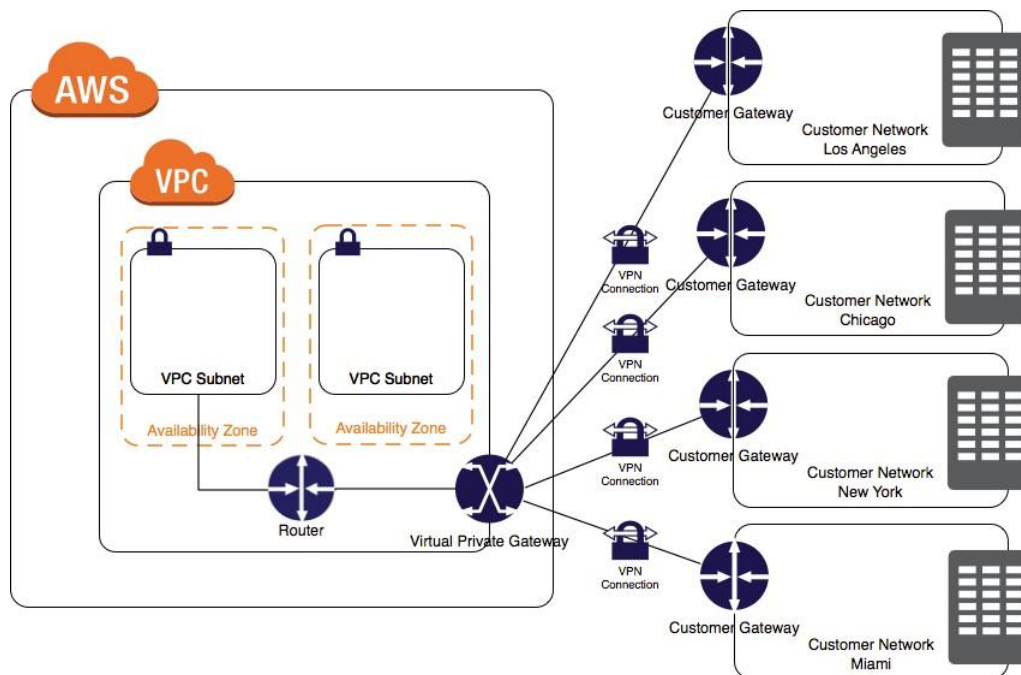
The following diagrams illustrate single and multiple VPN connections. The VPC has an attached virtual private gateway, and your network includes a customer gateway, which you must configure to enable the VPN connection. You set up the routing so that any traffic from the VPC bound for your network is routed to the virtual private gateway.

When you create multiple VPN connections to a single VPC, you can configure a second customer gateway to create a redundant connection to the same external location. You can also use it to create VPN connections to multiple geographic locations.

Single VPN Connection



Multiple VPN connections



VPN Routing Options

When you create a VPN connection, you must specify the type of routing that you plan to use. The type of routing that you select can depend on the make and model of your VPN devices. If your VPN device supports Border Gateway Protocol (BGP), specify dynamic routing when you configure your VPN connection. If your device does not support BGP, specify static routing. For a list of static and dynamic routing devices that have been tested with Amazon VPC, see the [Amazon Virtual Private Cloud FAQs](#).

When you use a BGP device, you don't need to specify static routes to the VPN connection because the device uses BGP to advertise its routes to the virtual private gateway. If you use a device that doesn't support BGP, you must select static routing and enter the routes (IP prefixes) for your network that should be communicated to the virtual private gateway. Only IP prefixes that are known to the virtual private gateway, whether through BGP advertisement or static route entry, can receive traffic from your VPC.

We recommend that you use BGP-capable devices, when available, because the BGP protocol offers robust liveness detection checks that can assist failover to the second VPN tunnel if the first tunnel goes down. Devices that don't support BGP may also perform health checks to assist failover to the second tunnel when needed.

What You Need for a VPN Connection

To use Amazon VPC with a VPN connection, you or your network administrator must designate a physical appliance as your customer gateway and configure it. We provide you with the required configuration information, including the VPN preshared key and other parameters related to setting up the VPN connection. Your network administrator typically performs this configuration. For information about the

customer gateway requirements and configuration, see the [Amazon Virtual Private Cloud Network Administrator Guide](#).

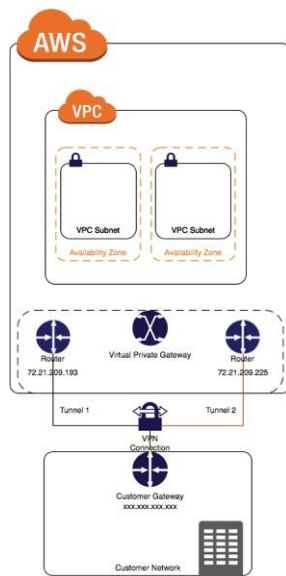
The following table lists the information that you need to have so that we can establish your VPN connection.

Item	How Used	Comments
The type of customer gateway (for example, Cisco ASA, Juniper J-Series, Juniper SSG, Yamaha)	Specifies how to format the returned information that you use to configure the customer gateway.	For information about the specific devices that we've tested, see What customer gateway devices are known to work with Amazon VPC? in the Amazon VPC FAQ.
Internet-routable IP address (static) of the customer gateway's external interface.	Used to create and configure your customer gateway (it's referred to as YOUR_UPLINK_ADDRESS)	The value must be static and can't be behind a device performing network address translation (NAT).
(Optional) Border Gateway Protocol (BGP) Autonomous System Number (ASN) of the customer gateway, if you are creating a dynamically routed VPN connection.	Used to create and configure your customer gateway (referred to as YOUR_BGP_ASN). If you use the wizard in the console to set up your VPC, we automatically use 65000 as the ASN.	You can use an existing ASN assigned to your network. If you don't have one, you can use a private ASN (in the 64512–65534 range). For more information about ASNs, see the Wikipedia article . Amazon VPC supports 2-byte ASN numbers.
Internal network IP ranges that you want advertised over the VPN connection to the VPC.	Used to specify static routes.	

Configuring Two VPN Tunnels for Your VPN Connection

You use a VPN connection to connect your network to a VPC. Each VPN connection has two tunnels, with each tunnel using a unique virtual private gateway public IP address. It is important to configure both tunnels for redundancy. When one tunnel becomes unavailable (for example, down for maintenance), network traffic is automatically routed to the available tunnel for that specific VPN connection.

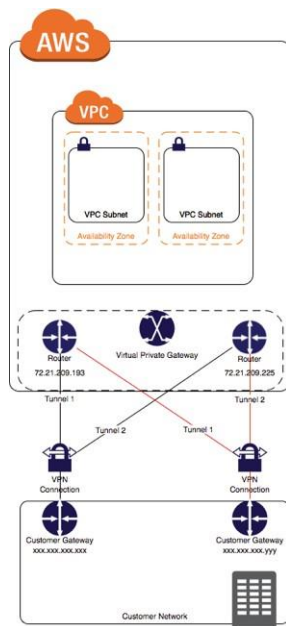
The following diagram shows the two tunnels of the VPN connection.



Using Redundant VPN Connections to Provide Failover

As described earlier, a VPN connection has two tunnels to help ensure connectivity in case one of the VPN connections becomes unavailable. To protect against a loss of connectivity in case your customer gateway becomes unavailable, you can set up a second VPN connection to your VPC by using a second customer gateway. By using redundant VPN connections and customer gateways, you can perform maintenance on one of your customer gateways while traffic continues to flow over the second customer gateway's VPN connection. To establish redundant VPN connections and customer gateways on your network, you need to set up a second VPN connection. The customer gateway IP address for the second VPN connection must be publicly accessible and can't be the same public IP address that you are using for the first VPN connection.

The following diagram shows the two tunnels of the VPN connection and two customer gateways.



Dynamically routed VPN connections use the Border Gateway Protocol (BGP) to exchange routing information between your customer gateways and the virtual private gateways. Statically routed VPN connections require you to enter static routes for the network on your side of the customer gateway. BGP-advertised and statically entered route information allow gateways on both sides to determine which tunnels are available and reroute traffic if a failure occurs. We recommend that you configure your network to use the routing information provided by BGP (if available) to select an available path. The exact configuration depends on the architecture of your network.

Setting Up the VPN Connection

Use the following procedures to manually set up the VPN connection. Alternatively, you can create the VPC and subnets and complete the first five steps in this procedure using the VPC wizard. For more information, see [Implementing Scenario 3](#) (p. 27) or [Implementing Scenario 4](#) (p. 34).

To set up a VPN connection, you need to complete the following steps:

- [Step 1: Create a Customer Gateway](#) (p. 124)
- [Step 2: Create a Virtual Private Gateway](#) (p. 125)
- [Step 3: Update Your Route Tables and Enable Route Propagation](#) (p. 125)
- [Step 4: Update Your Security Group to Enable Inbound SSH, RDP and ICMP Access](#) (p. 126)
- [Step 5: Create a VPN Connection and Configure the Customer Gateway](#) (p. 126)
- [Step 6: Launch an Instance Into Your Subnet](#) (p. 126)

These procedures assume that you have a VPC with one or more subnets, and that you have the required network information (see [What You Need for a VPN Connection](#) (p. 121)).

Step 1: Create a Customer Gateway

To create a customer gateway

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

2. In the navigation pane, click **Customer Gateways**, and then click **Create Customer Gateway**.
3. In the **Create Customer Gateway** dialog box, complete the following and then click **Yes, Create**:
 - In the **Name tag** field, optionally enter a name for your customer gateway. Doing so creates a tag with a key of `Name` and the value that you specify.
 - Select the routing type from the **Routing** list.
 - If you selected dynamic routing, enter the Border Gateway Protocol (BGP) Autonomous System Number (ASN) in the **BGP ASN** field.
- Enter the static, Internet-routable IP address for your customer gateway device in the **IP Address** field. The address cannot be behind a device that performs network address translation (NAT).

Step 2: Create a Virtual Private Gateway

To create a virtual private gateway

1. In the navigation pane, click **Virtual Private Gateways**, and then click **Create Virtual Private Gateway**.
2. You can optionally enter a name for your virtual private gateway, and then click **Yes, Create**.
3. Select the virtual private gateway that you created, and then click **Attach to VPC**.
4. In the **Attach to VPC** dialog box, select your VPC from the list, and then click **Yes, Attach**.

Step 3: Update Your Route Tables and Enable Route Propagation

For this step, the action you take differs depending on whether you are using static or dynamic routing for your VPN connection. If you are using static routing, you must update your route table to include the static route used by the VPN connection, and point it to your virtual private gateway. If you are using dynamic routing, you don't need to specify static routes to the VPN connection because the device uses BGP to advertise its routes to the virtual private gateway. You only need to enable route propagation to ensure that the virtual private gateway automatically propagates the routes to the route table.

To enable route propagation for dynamic routing

1. In the navigation pane, click **Route Tables**, and then select the route table that's associated with the subnet; by default, this is the main route table for the VPC.
2. On the **Route Propagation** tab in the details pane, click **Edit**, select the virtual private gateway that you created in the previous procedure, and then click **Save**.

Note

If you configure your VPN connection to use dynamic routing and you enable route propagation, the BGP-advertised routes from your customer gateway won't appear in the route table unless the status of the VPN connection is `UP`.

To update your route table for static routing

1. In the navigation pane, click **Route Tables**, and then select the route table that's associated with the subnet; by default, this is the main route table for the VPC.
2. On the **Routes** tab in the details pane, click **Edit**. Add the static route used by your VPN connection in the **Destination** field, select the virtual private gateway ID from the **Target** list, and then click **Save**.

Step 4: Update Your Security Group to Enable Inbound SSH, RDP and ICMP Access

To add rules to your security group to enable inbound SSH, RDP and ICMP access

1. In the navigation pane, click **Security Groups**, and then select the default security group for the VPC.
2. On the **Inbound** tab in the details pane, add rules that allow inbound SSH, RDP and ICMP access from your network, and then click **Save**. For more information about adding inbound rules, see [Adding and Removing Rules](#) (p. 56).

Step 5: Create a VPN Connection and Configure the Customer Gateway

To create a VPN connection and configure the customer gateway

1. In the navigation pane, click **VPN Connections**.
2. Click **Create VPN Connection**.
3. In the **Create VPN Connection** dialog box, do the following, and then click **Yes, Create**:
 - In the **Name tag** field, optionally enter a name for your VPN connection. Doing so creates a tag with a key of `Name` and the value that you specify.
 - Select the virtual private gateway that you created earlier.
 - Select the customer gateway that you created earlier.
 - Select one of the routing options based on whether your VPN router supports Border Gateway Protocol (BGP):
 - If your VPN router supports BGP, select **Dynamic (requires BGP)**.
 - If your VPN router does not support BGP, select **Static**. In the **Static IP Prefixes** field, specify each IP prefix for the private network of your VPN connection, separated by commas.
4. It may take a few minutes to create the VPN connection. When it's ready, select the connection, and then click **Download Configuration**.
5. In the **Download Configuration** dialog box, select the vendor, platform, and software that corresponds to your customer gateway device or software, and then click **Yes, Download**.
6. Give the configuration file to your network administrator, along with this guide: [Amazon Virtual Private Cloud Network Administrator Guide](#). After the network administrator configures the customer gateway, the VPN connection is operational.

Step 6: Launch an Instance Into Your Subnet

To launch an instance into your subnet

1. Open the Amazon EC2 console.
2. On the dashboard, click **Launch Instance**.
3. On the **Choose an Amazon Machine Image (AMI)** page, choose an AMI, and then click **Select**.
4. Choose an instance type, and then click **Next: Configure Instance Details**.
5. On the **Configure Instance Details** page, select your VPC from the **Network** list, and your subnet from the **Subnet** list. Click **Next** until you reach the **Configure Security Group** page.

6. Select the **Select an existing security group** option, and then select the default group that you modified earlier. Click **Review and Launch**.
7. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to select a key pair and launch the instance.

Testing the End-to-End Connectivity of Your Instance

After you set up your VPN connection and launch an instance, you can test the connection by pinging the instance. You need to use an AMI that responds to ping requests, and you need to ensure that your instance's security group is configured to enable inbound ICMP. We recommend you use one of the Amazon Linux AMIs. If you are using instances running Windows Server, you'll need to log in to the instance and enable inbound ICMPv4 on the Windows firewall in order to ping the instance.

Important

You must configure any security group or network ACL in your VPC that filters traffic to the instance to allow inbound and outbound ICMP traffic.

You can monitor the status of your VPN connections using the Amazon VPC console or by using the Amazon EC2 API/CLI. You can view information about your VPN connections, including its state, the time since last state change, and descriptive error text.

To test end-to-end connectivity

1. After the instance is running, get its private IP address (for example, 10.0.0.4). The Amazon EC2 console displays the address as part of the instance's details.
2. From a computer in your network that is behind the customer gateway, use the **ping** command with the instance's private IP address. A successful response is similar to the following:

```
PROMPT> ping 10.0.0.4
Pinging 10.0.0.4 with 32 bytes of data:

Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.4:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),

Approximate round trip times in milliseconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

You can now use SSH or RDP to connect to your instance in the VPC. For more information about how to connect to a Linux instance, see [Connect to Your Linux Instance](#) in the *Amazon Elastic Compute Cloud User Guide*. For more information about how to connect to a Windows instance, see [Connect to Your Windows Instance](#) in the *Amazon Elastic Compute Cloud Microsoft Windows Guide*.

Replacing Compromised Credentials

If you believe that the tunnel credentials for your VPN connection have been compromised, you can change the IKE preshared key. To do so, delete the VPN connection, create a new one using the same virtual private gateway, and configure the new keys on your customer gateway. You also need to confirm that the tunnel's inside and outside addresses match, because these might change when you recreate the VPN connection. While you perform the procedure, communication with your instances in the VPC stops, but the instances continue to run uninterrupted. After the network administrator implements the new configuration information, your VPN connection uses the new credentials, and the network connection to your instances in the VPC resumes.

Important

This procedure requires assistance from your network administrator group.

To change the IKE pre-shared key

1. Delete the VPN connection. For more information, see [Deleting a VPN Connection \(p. 128\)](#). You don't need to delete the VPC or the virtual private gateway.
2. Create a new VPN connection and download the new configuration file. For more information, see [Step 5: Create a VPN Connection and Configure the Customer Gateway \(p. 126\)](#).

Deleting a VPN Connection

If you no longer need a VPN connection, you can delete it.

Important

If you delete your VPN connection and then create a new one, you have to download new configuration information and have your network administrator reconfigure the customer gateway.

To delete a VPN connection

1. Open the Amazon VPC console.
2. In the navigation pane, click **VPN Connections**.
3. Select the VPN connection and click **Delete**.
4. In the **Delete VPN Connection** dialog box, click **Yes, Delete**.

If you no longer require a customer gateway, you can delete it. You can't delete a customer gateway that's being used in a VPN connection.

To delete a customer gateway

1. In the navigation pane, click **Customer Gateways**.
2. Select the customer gateway to delete and click **Delete**.
3. In the **Delete Customer Gateway** dialog box, click **Yes, Delete**.

If you no longer require a virtual private gateway for your VPC, you can detach it.

To detach a virtual private gateway

1. In the navigation pane, click **Virtual Private Gateways**.
2. Select the virtual private gateway and click **Detach from VPC**.
3. In the **Detach from VPC** dialog box, click **Yes, Detach**.

If you no longer require a detached virtual private gateway, you can delete it. You can't delete a virtual private gateway that's still attached to a VPC.

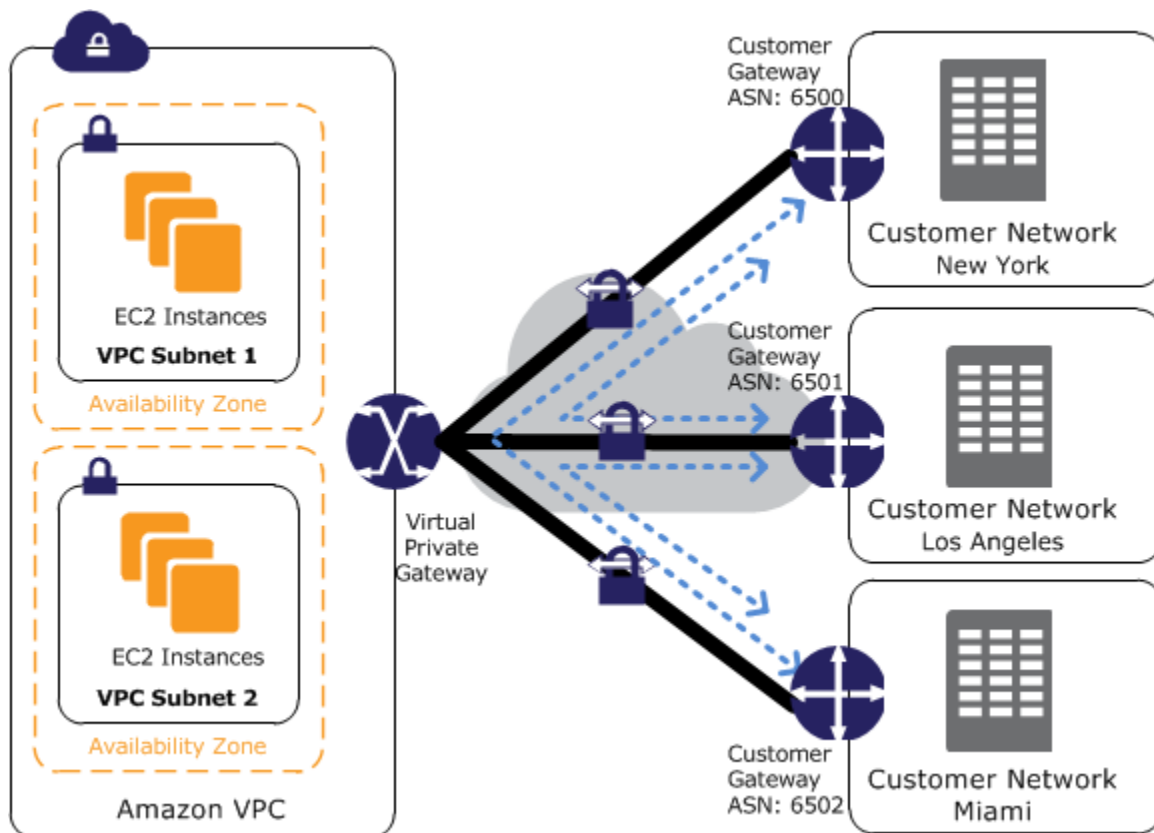
To delete a virtual private gateway

1. Select the virtual private gateway to delete and click **Delete**.
2. In the **Delete Virtual Private Gateway** dialog box, click **Yes, Delete**.

Providing Secure Communication Between Sites Using VPN CloudHub

If you have multiple VPN connections, you can provide secure communication between sites using the AWS VPN CloudHub. The VPN CloudHub operates on a simple hub-and-spoke model that you can use with or without a VPC. This design is suitable for customers with multiple branch offices and existing Internet connections who'd like to implement a convenient, potentially low-cost hub-and-spoke model for primary or backup connectivity between these remote offices.

The following diagram shows the VPN CloudHub architecture, with blue dashed lines indicating network traffic between remote sites being routed over their VPN connections.



To use the AWS VPN CloudHub, you must create a virtual private gateway with multiple customer gateways, each with unique Border Gateway Protocol (BGP) Autonomous System Numbers (ASNs). Customer gateways advertise the appropriate routes (BGP prefixes) over their VPN connections. These routing advertisements are received and re-advertised to each BGP peer, enabling each site to send data to and receive data from the other sites. The routes for each spoke must have unique ASNs and the sites must not have overlapping IP ranges. Each site can also send and receive data from the VPC as if they were using a standard VPN connection.

Sites that use AWS Direct Connect connections to the virtual private gateway can also be part of the AWS VPN CloudHub. For example, your corporate headquarters in New York can have an AWS Direct Connect connection to the VPC and your branch offices can use VPN connections to the VPC. The branch offices in Los Angeles and Miami can send and receive data with each other and with your corporate headquarters, all using the AWS VPN CloudHub.

To configure the AWS VPN CloudHub, you use the AWS Management Console to create multiple customer gateways, each with the unique public IP address of the gateway and a unique ASN. Next, you create a VPN connection from each customer gateway to a common virtual private gateway. Each VPN connection must advertise its specific BGP routes. This is done using the network statements in the VPN configuration files for the VPN connection. The network statements differ slightly depending on the type of router you use.

When using an AWS VPN CloudHub, you pay typical Amazon VPC VPN connection rates. You are billed the connection rate for each hour that each VPN is connected to the virtual private gateway. When you send data from one site to another using the AWS VPN CloudHub, there is no cost to send data from your site to the virtual private gateway. You only pay standard AWS data transfer rates for data that is relayed from the virtual private gateway to your endpoint. For example, if you have a site in Los Angeles and a second site in New York and both sites have a VPN connection to the virtual private gateway, you pay \$.05 per hour for each VPN connection (for a total of \$.10 per hour). You also pay the standard AWS data transfer rates for all data that you send from Los Angeles to New York (and vice versa) that traverses

each VPN connection; network traffic sent over the VPN connection to the virtual private gateway is free but network traffic sent over the VPN connection from the virtual private gateway to the endpoint is billed at the standard AWS data transfer rate. For more information, see [VPN Connection Pricing](#).

Dedicated Instances

Dedicated Instances are Amazon EC2 instances that run in a virtual private cloud (VPC) on hardware that's dedicated to a single customer. Your Dedicated Instances are physically isolated at the host hardware level from your instances that aren't Dedicated Instances and from instances that belong to other AWS accounts.

This topic discusses the basics of Dedicated Instances and shows you how to implement them.

Topics

- [Dedicated Instance Basics \(p. 133\)](#)
- [Working with Dedicated Instances \(p. 134\)](#)
- [API and Command Overview \(p. 136\)](#)

Dedicated Instance Basics

Each instance that you launch into a VPC has a tenancy attribute. You can't change the tenancy of an instance after you launch it. This attribute has the following values.

Value	Description
default	Your instance runs on shared hardware.
dedicated	Your instance runs on single-tenant hardware.

Each VPC has a related instance tenancy attribute. You can't change the instance tenancy of a VPC after you create it. This attribute has the following values.

Value	Description
default	An instance launched into the VPC is a Dedicated Instance if the tenancy attribute for the instance is <code>dedicated</code> .
dedicated	All instances launched into the VPC are Dedicated Instances, regardless of the value of the tenancy attribute for the instance.

If you are planning to use Dedicated Instances, you can implement them using either method:

- Create the VPC with the instance tenancy set to `dedicated` (all instances launched into this VPC are Dedicated Instances).
- Create the VPC with the instance tenancy set to `default`, and specify dedicated tenancy for any instances that should be Dedicated Instances when you launch them.

Dedicated Instances Limitations

There are services in AWS that won't work with a VPC with the instance tenancy set to `dedicated`. For example, with Amazon RDS, you can't launch a DB instance into such a VPC.

Amazon EBS with Dedicated Instances

When you launch an Amazon EBS-backed Dedicated Instance, the EBS volume doesn't run on single-tenant hardware.

Reserved Instances with Dedicated Tenancy

To guarantee that sufficient capacity will be available to launch Dedicated Instances, you can purchase Dedicated Reserved Instances. For more information about Reserved Instances, see [On-Demand and Reserved Instances](#).

When you purchase a Dedicated Reserved Instance, you are purchasing the capacity to launch a Dedicated Instance into a VPC at a much reduced usage fee; the price break in the hourly charge applies only if you launch an instance with dedicated tenancy. However, if you purchase a Reserved Instance with a default tenancy value, you won't get a Dedicated Reserved Instance if you launch an instance with `dedicated` instance tenancy.

In addition, you can't change the tenancy of a Reserved Instance after you've purchased it.

Auto Scaling of Dedicated Instances

For information about using Auto Scaling to launch Dedicated Instances, see [Auto Scaling in Amazon Virtual Private Cloud](#) in the *Auto Scaling Developer Guide*.

Pricing for Dedicated Instances

We have a separate pricing model for Dedicated Instances. For more information, see the [Amazon EC2 Dedicated Instances product page](#).

Working with Dedicated Instances

This section shows you how to complete the following tasks.

Topics

- [Creating a VPC with an Instance Tenancy of Dedicated](#) (p. 135)
- [Launching Dedicated Instances into a VPC](#) (p. 135)
- [Displaying Tenancy Information](#) (p. 135)

Creating a VPC with an Instance Tenancy of Dedicated

When you create a VPC, you have the option of specifying its instance tenancy. You can accept the default, or you can specify an instance tenancy of `dedicated` for your VPC. In this section, we show you how to create a VPC with an instance tenancy of `dedicated`.

To create a VPC with an instance tenancy of dedicated (VPC Wizard)

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. From the dashboard, click the **Start VPC Wizard** button.
3. Select a VPC configuration, and then click **Select**.
4. On the next page of the wizard, select **Dedicated** from the **Hardware tenancy** list.
5. Click the **Create VPC** button to create the VPC.

To create a VPC with an instance tenancy of dedicated (Create VPC dialog box)

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Your VPCs**, and then click **Create VPC**.
3. In the **Create VPC** dialog box, select **Dedicated** from the **Tenancy** drop-down list. Specify the **CIDR Block**, and then click **Yes, Create**.

Launching Dedicated Instances into a VPC

If you launch an instance into a VPC that has an instance tenancy of `dedicated`, your instance is automatically a Dedicated Instance, regardless of the tenancy of the instance. The following procedure shows you how to launch a Dedicated Instance into a VPC that has default instance tenancy.

To launch a Dedicated Instance into a VPC with default instance tenancy

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. Create a VPC, or decide to use an existing VPC with default instance tenancy.
3. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
4. Click the **Launch Instance** button.
5. On the **Choose an Amazon Machine Image (AMI)** page, choose an AMI and click **Select**.
6. On the **Choose an Instance Type** page, select the type of instance you want to launch, then click **Next: Configure Instance Details**.
7. On the **Configure Instance Details** page, select a VPC and subnet. Select **Dedicated tenancy** from the **Tenancy** list, and then click **Next: Add Storage**.
8. Continue as prompted by the wizard. When you've finished reviewing your options on the **Review Instance Launch** page, click **Launch** to choose a key pair and launch the Dedicated Instance.

Displaying Tenancy Information

To display tenancy information for your VPC

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Your VPCs**.
3. Check the instance tenancy of your VPC in the **Tenancy** column.

4. If the **Tenancy** column is not displayed, click the **Show/Hide** button, select **Tenancy** from the **Show/Hide Columns** dialog box, and then click **Close**.

To display tenancy information for your instance

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, click **Instances**.
3. Check the tenancy of your instance in the **Tenancy** column.
4. If the **Tenancy** column is not displayed, do one of the following:
 - Click the **Show/Hide** button, select **Tenancy** from the **Show/Hide Columns** dialog box, and then click **Close**.
 - Select the instance. The **Description** tab in the details pane displays information about the instance, including its tenancy.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Set the supported tenancy options for instances that you launch into a VPC

- [create-vpc](#) (AWS CLI)
- [ec2-create-vpc](#) (Amazon EC2 CLI)
- [New-EC2Vpc](#) (AWS Tools for Windows PowerShell)

Describe the supported tenancy options for instances launched into the VPC

- [describe-vpcs](#) (AWS CLI)
- [ec2-describe-vpcs](#) (Amazon EC2 CLI)
- [Get-EC2Vpc](#) (AWS Tools for Windows PowerShell)

Set the tenancy option for an instance

- [run-instances](#) (AWS CLI)
- [ec2-run-instances](#) (Amazon EC2 CLI)
- [New-EC2Instance](#) (AWS Tools for Windows PowerShell)

Describe the tenancy value of an instance

- [describe-instances](#) (AWS CLI)
- [ec2-describe-instances](#) (Amazon EC2 CLI)
- [Get-EC2Instance](#) (AWS Tools for Windows PowerShell)

Describes the tenancy value of a Reserved Instance

- [describe-reserved-instances](#) (AWS CLI)
- [ec2-describe-reserved-instances](#) (Amazon EC2 CLI)

- [Get-EC2ReservedInstance](#) (AWS Tools for Windows PowerShell)

Describes the tenancy value of a Reserved Instance offering

- [describe-reserved-instances-offerings](#) (AWS CLI)
- [ec2-describe-reserved-instances-offerings](#) (Amazon EC2 CLI)
- [Get-EC2ReservedInstancesOffering](#) (AWS Tools for Windows PowerShell)

VPC Peering

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IP 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 within a single region.

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 can help 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. For more examples of scenarios in which you can use a VPC peering connection, see the [Amazon Virtual Private Cloud Peering Guide](#).

Topics

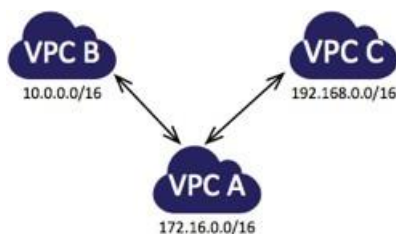
- [VPC Peering Basics](#) (p. 138)
- [Working with VPC Peering Connections](#) (p. 140)
- [API and CLI Overview](#) (p. 145)
- [Controlling Access to VPC Peering Connections](#) (p. 146)

VPC Peering Basics

To establish a VPC peering connection, the owner of the *requester VPC* (or *local VPC*) sends a request to the owner of the *peer VPC* to create the VPC peering connection. The peer 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. The owner of the peer VPC has to accept the VPC peering connection request to activate the VPC peering connection. To enable the flow of traffic between the peer VPCs using private IP addresses, add a route to one or more of your VPC's route tables that points to the IP address range of the peer VPC. The owner of the peer VPC adds a route to one of their VPC's route tables that points to the IP address range of your VPC. You may also need to update the security group rules that are associated with your instance to ensure that traffic to and from the peer VPC is not restricted. For more information about security groups, see [Security Groups for Your VPC](#) (p. 53).

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 will 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.

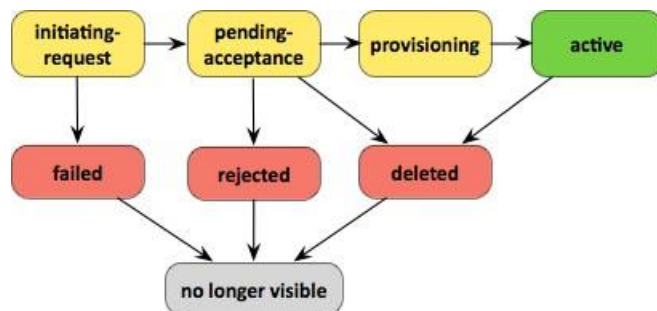


For more information about VPC peering limitations, see [VPC Peering Limitations \(p. 140\)](#). For more information and examples of peering relationships that are supported, see the [Amazon Virtual Private Cloud Peering Guide](#).

You are charged for data transfer within a VPC peering connection at the same rate as you are charged for data transfer across Availability Zones. For more information, see [Amazon EC2 Pricing](#).

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 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 or rejected. 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 peer VPC. During this state, the owner of the requester VPC can delete the request, and the owner of the peer VPC can accept or reject the request. If no action is taken on the request, it will expire 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 peer 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 peer 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. During this state, either of the VPC owners can delete the VPC peering connection, but cannot reject it.
- **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.

VPC Peering Limitations

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

- You cannot create a VPC peering connection between VPCs that have matching or overlapping CIDR blocks.
- You cannot create a VPC peering connection between VPCs in different regions.
- You have a limit on the number active and pending VPC peering connections that you can have per VPC. For more information about VPC limits, see [Amazon VPC Limits \(p. 147\)](#).
- VPC peering does not support transitive peering relationships; in a VPC peering connection, your VPC will not have access to any other VPCs that the peer VPC may be peered with. This includes VPC peering connections that are established entirely within your own AWS account. For more information and examples of peering relationships that are supported, see the [Amazon Virtual Private Cloud Peering Guide](#).
- You cannot have more than one VPC peering connection between the same two VPCs at the same time.
- The Maximum Transmission Unit (MTU) across a VPC peering connection is 1500 bytes.
- A placement group can span peered VPCs; however, you will not get full-bisection bandwidth between instances in peered VPCs. For more information about placement groups, see [Placement Groups](#) in the *Amazon Elastic Compute Cloud User Guide*.
- Unicast reverse path forwarding in VPC peering connections is not supported. For more information, see [Routing for Response Traffic](#) in the *Amazon Virtual Private Cloud Peering Guide*.
- You cannot reference a security group from the peer VPC as a source or destination for ingress or egress rules in your security group. Instead, reference CIDR blocks of the peer VPC as the source or destination of your security group's ingress or egress rules.

Working with VPC Peering Connections

Topics

- [Creating a VPC Peering Connection \(p. 141\)](#)
- [Accepting a VPC Peering Connection \(p. 142\)](#)
- [Rejecting a VPC Peering Connection \(p. 143\)](#)
- [Updating Route Tables for Your VPC Peering Connection \(p. 143\)](#)
- [Describing Your VPC Peering Connections \(p. 144\)](#)
- [Deleting a VPC Peering Connection \(p. 144\)](#)

Creating 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. To activate the request, the owner of the peer VPC must accept the request.

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.

To create a VPC peering connection in your account

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
 2. In the navigation pane, click **Peering Connections**.
 3. Click **Create VPC Peering Connection**.
 4. In the dialog, configure the following information, and click **Create** when you are done:
 - **Name:** You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify.
 - **Local VPC to peer:** Select the VPC in your account with which you want to create the VPC peering connection.
 - **Select a VPC to peer with:** Ensure **My account** is selected, and select another of your VPCs from the **VPC ID** list. Only VPCs in the current region are displayed.
- Important**
Ensure that your VPCs do not have overlapping CIDR blocks. If they do, the status of the VPC peering connection immediately goes to `failed`.
5. A confirmation dialog box provides the ID of the VPC peering connection, as well as information about the VPCs in the peering connection. Click **OK**.
 6. To view all VPC peering connections that are pending your acceptance, select **Requests pending my approval** from the filter list.
 7. Select the VPC peering connection that you've created, and click **Accept request**.
 8. In the confirmation dialog, click **Yes, Accept**. A second confirmation dialog displays; click **Modify my route tables now** to go directly to the route tables page, or click **Close** to do this later.

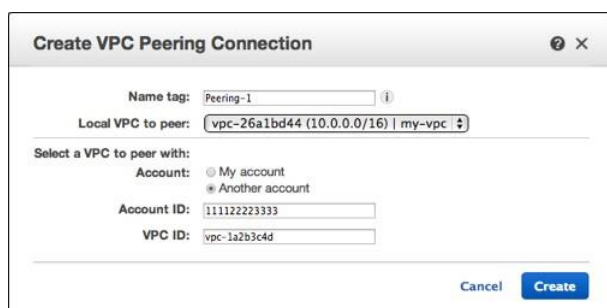
Now that your VPC peering connection is active, you must add an entry to your VPCs' route tables to enable traffic to be directed between the peered VPCs. For more information, see [Updating Route Tables for Your VPC Peering Connection](#) (p. 143).

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 peer VPC must accept the VPC peering connection to activate it.

To create a VPC peering connection with a remote VPC

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Peering Connections**.
3. Click **Create VPC Peering Connection**.
4. In the dialog, configure the information as follows, and click **Create** when you are done:
 - **Name:** 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.
 - **Local VPC to peer:** Select the VPC in your account with which to create the VPC peering connection.
 - **Select a VPC to peer with:** Select **Another account**, and enter the AWS account ID and the ID of the VPC with which to create the VPC peering connection.



Important

If your VPC and the peer VPC have overlapping 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`.

5. A confirmation dialog box provides the ID of the VPC peering connection, as well as information about the VPCs in the peering connection. Click **OK**.

The VPC peering connection that you've created is not active. To activate it, the owner of the peer VPC must accept the VPC peering connection request. To enable traffic to be directed to the peer VPC, update your VPC's route table. For more information, see [Updating Route Tables for Your VPC Peering Connection](#) (p. 143).

Accepting a VPC Peering Connection

A VPC peering connection that's in the `pending-acceptance` state must be accepted by the owner of the peer 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.

Important

Do not accept VPC peering connections from AWS accounts that you do not know. 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 about rejecting VPC peering requests, see [Rejecting a VPC Peering Connection](#) (p. 143). You can also ignore the request and let it expire; by default, the request expires after 7 days.

To accept a VPC peering connection

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Peering Connections**.
3. To view all VPC peering connections that are pending your acceptance, select **Requests pending my approval** from the filter list.
4. Select the VPC peering connection, and click **Accept request**.
5. In the confirmation dialog box, click **Yes, Accept**. A second confirmation dialog displays; click **Modify my route tables now** to go directly to the route tables page, or click **Close** to do this later.

Now that your VPC peering connection is active, you must add an entry to your VPC's route table to enable traffic to be directed to the peer VPC. For more information, see [Updating Route Tables for Your VPC Peering Connection](#) (p. 143).

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, click **Peering Connections**.
3. To view all VPC peering connections that are pending your acceptance, select **Requests pending my approval** from the filter list.
4. Select the VPC peering connection, and click **Reject request**.
5. In the confirmation dialog box, click **Yes, Reject**.

Updating Route Tables for Your VPC Peering Connection

To send traffic between instances in peered VPCs using private IP addresses, you must add a route to a route table that's associated with your VPC. The route points to the CIDR block (or portion of the CIDR block) of the other VPC in the VPC peering connection.

Similarly, the owner of the other VPC in the peering connection must add a route to one of their VPC's route tables to direct traffic back to your VPC. For more examples of supported route table configurations for VPC peering connections, see the [Amazon Virtual Private Cloud Peering Guide](#).

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.

For more information about route tables, see [Route Tables](#) (p. 91).

Warning

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 more information, see [Routing for Response Traffic](#) in the *Amazon Virtual Private Cloud Peering Guide*.

To add a route for a VPC peering connection using the console

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **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. Click the **Routes** tab, and then click **Edit**.
5. Click **Add Route**.
6. In the **Destination** field, enter the IP address range to which the network traffic in the VPC peering connection must be directed. You can specify the entire CIDR block of the peer VPC, a specific range, or an individual IP 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.
7. Select the VPC peering connection from the **Target** list, and then click **Save**.

Destination	Target	Status	Propagated	Remove
192.168.0.0/28	local	Active	No	
10.0.0.0/28	pcx-c37b9faa	Active	No	✖

Describing Your VPC Peering Connections

You can view all your VPC peering connections in the 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. 139).

To view your VPC peering connections

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, click **Peering Connections**.
3. All your VPC peering connections are listed. Use the filter lists and search field to narrow your results. For example, to view VPC peering connection requests that you've sent that are waiting for approval, select **Outstanding requests I've sent** from the filter list.

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.

Note

Deleting a VPC in the VPC console that's part of an active VPC peering connection 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 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.

To delete a VPC peering connection

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

API and CLI Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see [Accessing Amazon VPC \(p. 5\)](#).

Create a VPC peering connection

- [create-vpc-peering-connection](#) (AWS CLI)
- [ec2-create-vpc-peering-connection](#) (Amazon EC2 CLI)
- [New-EC2VpcPeeringConnection](#) (AWS Tools for Windows PowerShell)
- [CreateVpcPeeringConnection](#)

Accept a VPC peering connection

- [accept-vpc-peering-connection](#) (AWS CLI)
- [ec2-accept-vpc-peering-connection](#) (Amazon EC2 CLI)
- [Approve-EC2VpcPeeringConnection](#) (AWS Tools for Windows PowerShell)
- [AcceptVpcPeeringConnection](#)

Describe VPC peering connections

- [describe-vpc-peering-connections](#) (AWS CLI)
- [ec2-describe-vpc-peering-connections](#) (Amazon EC2 CLI)
- [Get-EC2VpcPeeringConnections](#) (AWS Tools for Windows PowerShell)
- [DescribeVpcPeeringConnections](#)

Reject VPC peering connections

- [reject-vpc-peering-connection](#) (AWS CLI)
- [ec2-reject-vpc-peering-connection](#) (Amazon EC2 CLI)
- [Deny-EC2VpcPeeringConnection](#) (AWS Tools for Windows PowerShell)
- [RejectVpcPeeringConnection](#)

Delete a VPC peering connection

- [delete-vpc-peering-connection](#) (AWS CLI)
- [ec2-delete-vpc-peering-connection](#) (Amazon EC2 CLI)
- [Remove-EC2VpcPeeringConnection](#) (AWS Tools for Windows PowerShell)
- [DeleteVpcPeeringConnection](#)

Add a route to a route table

- [create-route](#) (AWS CLI)
- [ec2-create-route](#) (Amazon EC2 CLI)
- [New-EC2Route](#) (AWS Tools for Windows PowerShell)
- [CreateRoute](#)

Replace a route in a route table

- [replace-route](#) (AWS CLI)
- [ec2-replace-route](#) (Amazon EC2 CLI)
- [Set-EC2Route](#) (AWS Tools for Windows PowerShell)
- [ReplaceRoute](#)

Controlling Access to VPC Peering Connections

By default, IAM users cannot create or modify VPC peering connections. You can create an IAM policy that grants users permission to create or modify VPC peering connections, and you can control which resources users have access to during those requests. For more information about IAM policies for Amazon EC2, see [IAM Policies for Amazon EC2](#) in the *Amazon Elastic Compute Cloud User Guide*.

For example policies for working with VPC peering connections, see [Controlling Access to Amazon VPC Resources](#) (p. 75).

Amazon VPC Limits

The following table lists the limits related to Amazon VPC. To request to increase in any of these limits, see the [Amazon VPC Limits form](#).

Component	Limit	Comments
VPCs per region	5	This limit can be increased upon request.
Subnets per VPC	200	This limit can be increased upon request.
Internet gateways per region	5	You can create as many Internet gateways as your 'VPCs per region' limit. Only one Internet gateway can be attached to a VPC at a time.
Virtual private gateways per region	5	Only one virtual private gateway can be attached to a VPC at a time.
Customer gateways per region	50	This limit can be increased upon request.
VPN connections per region	50	Ten per virtual private gateway.
Route tables per VPC	200	Including the main route table. You can associate one route table to one or more subnets in a VPC.
Entries per route table	50	This is the limit for the number of non-propagated entries per route table. This limit can be increased upon request; however, network performance may be impacted as the number of non-propagated route entries increases.
Elastic IP addresses per region for each AWS account	5	This is the limit for the number of VPC Elastic IPs you can allocate within a region. This is a separate limit from the EC2 Elastic IP address limit.

Component	Limit	Comments
Security groups per VPC	100	This limit can be increased upon request; however, network performance may be impacted as the number of security groups is increased, depending on the way the security groups are configured.
Rules per security group	50	This limit can be increased or decreased upon request, however, the multiple of 'rules per security group' and 'security groups per network interface' cannot exceed 250. For example, if you want 100 rules per security group, we'd need to decrease your number of security groups per network interface to 2.
Security groups per network interface	5	This limit can be increased or decreased upon request; however, the multiple of 'security groups per network interface' and 'rules per security group' cannot exceed 250. For example, if you want 10 security groups per network interface, we'd need to decrease your number of rules per security group to 25.
Network ACLs per VPC	200	You can associate one network ACL to one or more subnets in a VPC. This limit is not the same as the number of rules per network ACL.
Rules per network ACL	20	This is the sum of the number of rules for both ingress and egress rules in a single network ACL. The maximum limit is 40 rules per network ACL.
BGP Advertised Routes per VPN Connection	100	This limit can be increased upon request; however, network performance may be impacted as the number of advertised routes is increased.
Active VPC peering connections per VPC	50	This limit can be increased via special request to AWS Developer Support. The maximum limit is 125 peering connections per VPC. The number of entries per route table should be increased accordingly; however, network performance may be impacted as the number of entries in a route table is increased.
Outstanding VPC peering connection requests	25	This is the limit for the number of outstanding VPC peering connection requests that you've requested from your account. This limit can be increased via special request to AWS Developer Support.
Expiry time for an unaccepted VPC peering connection request	1 week (168 hours)	This limit can be increased via special request to AWS Developer Support.

Document History

The following table describes the important changes in each release of this Amazon VPC guide.

Feature	API Version	Description	Release Date
Modify a subnet's public IP addressing attribute	2014-06-15	You can modify the public IP addressing attribute of your subnet to indicate whether instances launched into that subnet should receive a public IP address. For more information, see Modifying Your Subnet's Public IP Addressing Behavior (p. 87).	21 June 2014
VPC peering	2014-02-01	You can create a VPC peering connection between two VPCs, which allows instances in either VPC to communicate with each other using private IP addresses - as if they are within the same VPC. For more information, see VPC Peering (p. 138).	24 March 2014
New launch wizard	2013-10-01	Added information about the redesigned EC2 launch wizard. For more information, see Launch an Instance Into Your VPC in the <i>Amazon Virtual Private Cloud Getting Started Guide</i> .	10 October 2013
Assigning a public IP address	2013-07-15	Added information about a new public IP addressing feature for instances launched in a VPC. For more information, see Assigning a Public IP Address During Launch (p. 86).	20 August 2013

Feature	API Version	Description	Release Date
Enabling DNS hostnames and disabling DNS resolution	2013-02-01	<p>By default, DNS resolution is enabled. You can now disable DNS resolution using the Amazon VPC console, the Amazon EC2 command line interface, or the Amazon EC2 API actions.</p> <p>By default, DNS hostnames are disabled for nondefault VPCs. You can now enable DNS hostnames using the Amazon VPC console, the Amazon EC2 command line interface, or the Amazon EC2 API actions.</p> <p>For more information, see Using DNS with Your VPC (p. 116).</p>	11 March 2013
VPN connections using static routing configuration	2012-08-15	You can create IPsec VPN connections to Amazon VPC using static routing configurations. Previously, VPN connections required the use of the Border Gateway Protocol (BGP). We now support both types of connections and are excited to announce that you can now establish connectivity from devices that do not support BGP, including Cisco ASA and Microsoft Windows Server 2008 R2.	13 September 2012
Automatic route propagation	2012-08-15	You can now configure automatic propagation of routes from your VPN and Direct Connect links to your VPC routing tables. This feature simplifies the effort to create and maintain connectivity to Amazon VPC.	13 September 2012
AWS VPN CloudHub and redundant VPN connections		You can securely communicate from one site to another with or without a VPC. You can use redundant VPN connections to provide a fault-tolerant connection to your VPC.	29 September 2011
VPC Everywhere	2011-07-15	Support in five AWS regions, VPCs in multiple Availability Zones, multiple VPCs per AWS account, multiple VPN connections per VPC, Microsoft Windows Server 2008 R2 and Microsoft SQL Server Reserved Instances.	03 August 2011
Dedicated Instances	2011-02-28	Dedicated Instances are Amazon EC2 instances launched within your VPC that run hardware dedicated to a single customer. Dedicated Instances let you take full advantage of the benefits of Amazon VPC and AWS elastic provisioning, pay only for what you use, and a private, isolated virtual network—all while isolating your instances at the hardware level.	27 March 2011

AWS Glossary

Blank

placeholder

This page redirects to the AWS Glossary in the *AWS General Reference*.