Amazon Virtual Private Cloud

User Guide

API Version 2011-07-15

Amazon Web Services

Amazon Virtual Private Cloud: User Guide

Amazon Web Services

Welcome

This is the *Amazon Virtual Private Cloud User Guide*. It explains how to use Amazon VPC through a web-based GUI, the AWS Management Console, which contains consoles for the various Amazon Web Services, including the Amazon VPC console.

Amazon Virtual Private Cloud enables you to create a virtual network topology—including subnets and routing—for your Amazon Elastic Compute Cloud (Amazon EC2) resources.

How Do I...?

How Do I?	Relevant Sections
Get a general product overview and information about pricing	Amazon VPC product page
Get started quickly using Amazon VPC	Amazon Virtual Private Cloud Getting Started Guide
Understand basic scenarios for using Amazon VPC	Scenarios for Using Amazon VPC (p. 8)
Learn about routing in my VPC, including route tables, Elastic IP addresses, and NAT instances	Routing in Your VPC (p. 111)
Learn about security in my VPC	Security in Your VPC (p. 136)
Add an Internet gateway to my VPC	Adding an Internet Gateway to Your VPC (p. 156)
Add a virtual private gateway to my VPC	Adding a Hardware Virtual Private Gateway to Your VPC (p. 164)
Use DHCP options in my VPC	Using DHCP Options with Your VPC (p. 176)
Use Auto Scaling with my VPC	Using Auto Scaling with Your VPC (p. 181)
Launch Dedicated Instances into my VPC	Using EC2 Dedicated Instances Within Your VPC (p. 187)

How Do I?	Relevant Sections
Control who can expose my VPC to the Internet and make changes to my VPC's routing and security	Controlling VPC Management (p. 198)
Learn about Amazon EC2	Amazon Elastic Compute Cloud Getting Started Guide Amazon Elastic Compute Cloud User Guide
Get started using the command line tools (i.e., the EC2 API tools)	Getting Started with the Command Line Tools in the *Amazon Elastic Compute Cloud User Guide*
Find available libraries for programmatically using EC2	Making API Requests in the *Amazon Elastic Compute Cloud User Guide*

Introduction to the Amazon Virtual Private Cloud

Topics

This introduction to Amazon Virtual Private Cloud gives you a high-level overview of this web service.

Overview

Amazon Virtual Private Cloud enables you to create a virtual network topology—including subnets and route tables—for your Amazon Elastic Compute Cloud (Amazon EC2) resources.

If you're familiar with Amazon EC2, you know that each instance you launch is randomly assigned a public IP address in the Amazon EC2 address space. Amazon VPC enables you to create an isolated portion of the Amazon Web Services (AWS) cloud (a VPC) and launch Amazon EC2 instances that have private (RFC 1918) addresses in the range of your choice (e.g., 10.0.0.0/16). A VPC is the first object you create when using Amazon Virtual Private Cloud. You can define *subnets* within your VPC, which enable you to group similar kinds of instances based on IP address range. For more information, see Your VPC and Subnets (p. 105).

By using Amazon VPC with Amazon EC2 (instead of Amazon EC2 alone), you gain the ability to:

- Logically group your Amazon EC2 instances, and assign them private IP addresses
- Control the egress traffic from your Amazon EC2 instances (in addition to controlling the ingress traffic to them)
- Add an additional layer of security to your Amazon EC2 instances in the form of network Access Control Lists (ACLs)
- Connect your VPC to your corporate data center and branch offices with a VPN connection, so that you can use the VPC as an extension of your corporate data center network

Levels of Privacy

When you create a VPC, you can configure it based on the level of privacy you want. In the most private scenario, you can attach only a virtual private gateway, and create an IPsec tunnel between your VPC and home network. In this scenario, your EC2 instances have no direct exposure to the Internet.

In the most public scenario, you can attach only an *Internet gateway* to the VPC and enable traffic to flow between the Internet and all the instances in your VPC.

You can configure your VPC to be somewhere in between, with both a virtual private gateway and an Internet gateway. Here, some instances could receive Internet traffic (e.g., web servers), whereas others could remain unexposed (e.g., database servers). This is a common scenario for running a multi-tier web application in the AWS cloud.

These different scenarios are discussed in more detail in this guide (see Scenarios in This Guide (p. 5)).

Routing and Security

You can configure routing in your VPC to control where traffic flows (e.g., to the Internet gateway, virtual private gateway, etc). With an Internet gateway, your VPC has direct access to other AWS products such as Amazon Simple Storage Service (Amazon S3). If you choose to have only a virtual private gateway with a connection to your home network, you can route your Internet-bound traffic over the VPN and control its egress with your security policies and corporate firewall. In the latter case, you incur additional bandwidth charges when accessing AWS products over the Internet.

You can use *security groups* and *network ACLs* to help secure the instances in your VPC. Security groups might be familiar if you're an Amazon EC2 user, and network ACLs might be familiar if you're a network administrator. Security groups act like a firewall at the instance level, whereas network ACLs are an additional layer of security that act at the subnet level. For more information, see Security in Your VPC (p. 136).

By default, the instances you launch in your VPC have only private IP addresses. If you want an instance to have a public IP address, you can assign it an *Elastic IP address*, which is a static, public address you can assign to any instance in your VPC. For an instance in your VPC to be addressable from the Internet, it must have an Elastic IP address.

You can use Network Address Translation (NAT) to enable instances that don't have Elastic IP addresses to reach the Internet. You can set up the VPC's routing so that traffic from private instances goes through a special NAT instance that has an Elastic IP address. We provide a NAT Amazon Machine Image (AMI) that you can use for this purpose.

For more information about routing, Elastic IP addresses, and NAT in your VPC, see Routing in Your VPC (p. 111).

If You're New to Amazon EC2

Amazon VPC is closely integrated with Amazon EC2. If you're not familiar with EC2, go to the Introduction to Amazon EC2 in the *Amazon Elastic Compute Cloud User Guide* to get a brief overview. We also recommend walking through the Amazon Elastic Compute Cloud Getting Started Guide.

Scenarios in This Guide

This guide presents several simple scenarios for using Amazon VPC:

- Scenario 1: VPC with a Single Public Subnet Only
  We recommend this scenario if you want to run a single-tier, public-facing web application such as a blog or simple website.
- Scenario 2: VPC with Public and Private Subnets
  We recommend this scenario if you want to run a public-facing web application, while still maintaining non-publicly accessible backend servers in a second subnet.
- Scenario 3:VPC with Public and Private Subnets and Hardware VPN Access
  We recommend this scenario if you want to 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
  We recommend this scenario if you want to extend your data center into the cloud and leverage Amazon's elasticity without exposing your network to the Internet.

Note

The preceding scenarios are common ones we chose to present; you can configure your VPC and subnets in other ways to suit your needs.

Each of the preceding scenarios is discussed in detail, with implementation instructions. In these scenarios, you're introduced to the basic concepts you need to understand to use Amazon VPC. For more information, see Scenarios for Using Amazon VPC (p. 8).

Current Limitations

The current implementation of Amazon VPC has the following limitations:

- You can have up to five (5) VPCs per account per Region.

- You can have up to five (5) Amazon VPC Elastic IP Addresses per AWS account per Region.
- You can create up to twenty (20) subnets per Amazon VPC.
- Once you create a VPC or subnet, you can't change its IP address range.
- If you plan to have a VPN connection to your VPC, then you can have up to five virtual private gateways per AWS account per Region (one per VPC), with up to ten VPN connections per virtual private gateway.
- You can't use either broadcast or multicast within your VPC.
- CC1 and t1.micro instances do not work with a VPC.
- Amazon ElastiCache is not available for use in a VPC at this time.
- Amazon RDS for Microsoft SQL Server is not available for use in an Amazon VPC at this time.
- AWS Elastic Beanstalk is not available with your instances in a VPC.
- Amazon DevPay paid AMIs do not work with a VPC.

For more information about VPC limits, see Appendix B: Limits (p. 216).

Amazon VPC Interfaces

You can access Amazon VPC operations through the following interfaces:

- AWS Management Console
- Command line
- API

AWS Management Console

This guide uses the AWS Management Console to perform Amazon VPC tasks, such as creating and deleting virtual private clouds, subnets, and gateways. If you're an Amazon EC2 user, you're probably already familiar with the console. The Amazon EC2 and Amazon VPC consoles are included within the AWS Management Console.

Command Line Interface

The command line interface is a set of simple commands that uses a Java runtime environment. If you're an Amazon EC2 user, you're probably already familiar with this interface (the Amazon EC2 API tools). The commands for Amazon VPC are part of that interface. To get started with the command line interface, go to Getting Started with the Command Line Tools in the *Amazon Elastic Compute Cloud User Guide*. For a complete list of the Amazon EC2 and Amazon VPC commands, go to the Amazon Elastic Compute Cloud Command Line Reference.

API

Because you use Amazon VPC in conjunction with Amazon EC2, the Amazon VPC operations are part of the Amazon EC2 WSDL, and Amazon VPC uses the Amazon EC2 web service entry point (i.e., endpoint). Request authentication for Amazon VPC API calls works the same way it does for Amazon EC2 API calls. For information about how to use the APIs, go to Making API Requests in the *Amazon Elastic Compute Cloud User Guide*. For the API reference for all Amazon EC2 and Amazon VPC API operations, go to the Amazon Elastic Compute Cloud API Reference.

Paying for Amazon Virtual Private Cloud

AWS doesn't charge you to use a VPC, aside from the normal Amazon EC2 instance usage and bandwidth charges for your instances. You're not charged data transfer charges for AWS-bound traffic that goes over the Internet gateway to the same Region as your VPC. Exception: You're charged Regional Data Transfer rates for data transferred between your VPC and Amazon EC2 instances in the same Region, regardless of Availability Zone.

If you choose to create one or more VPN connections to your VPC using a virtual private gateway, you pay for both the connections and the bandwidth of traffic that traverses those connections. For information about the rates for Amazon VPC, go to the Amazon VPC product page.

If you use other Amazon EC2 features (e.g., Amazon EBS, Elastic IP addresses), the normal Amazon EC2 rates for those features also apply. For information about Amazon EC2's rates, go to the Amazon EC2 product page.

Other Documentation

The following table summarizes the other available documentation for Amazon VPC and Amazon EC2.

Description	Documentation
A hands-on introduction to Amazon VPC	Amazon Virtual Private Cloud Getting Started Guide
Information about configuring the customer gateway (if you decide to use multiple VPN connections with your VPC)	Amazon Virtual Private Cloud Network Administrator Guide
A hands-on introduction to Amazon EC2	Amazon Elastic Compute Cloud Getting Started Guide
How to use Amazon EC2	Amazon Elastic Compute Cloud User Guide
Complete descriptions of all the Amazon EC2 and Amazon VPC commands	Amazon Elastic Compute Cloud Command Line Reference
Complete descriptions of the Amazon EC2 and Amazon VPC API operations, data types, and errors	Amazon Elastic Compute Cloud API Reference

Where to Get Additional Help

We recommend that you take advantage of the AWS Discussion Forums. These are community-based forums for users to discuss technical questions related to AWS services. For the Amazon VPC forum, go to https://forums.aws.amazon.com/forum.jspa?forumID=58.

You can also get help if you subscribe to AWS Premium Support, a one-on-one, fast-response support channel (for more information, go to http://aws.amazon.com/premiumsupport).

Scenarios for Using Amazon VPC

Topics

This section presents several basic scenarios for using Amazon VPC:

- Scenario 1: VPC with a Public Subnet Only
  We recommend this scenario if you want to run a single-tier, public-facing web application such as a blog or simple website.
- Scenario 2: VPC with Public and Private Subnets
  We recommend this scenario if you want to run a public-facing web application, while still maintaining non-publicly accessible backend servers in a second subnet.
- Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access
  We recommend this scenario if you want to 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
  We recommend this scenario if you want to extend your data center into the cloud and leverage Amazon's elasticity without exposing your network to the Internet.

Each scenario presents the following information:

- A layout of the basic components used in the scenario
- Routing in the VPC
- Security in the VPC
- How to implement the scenario

Scenario 1: VPC with a Public Subnet Only

Topics

We recommend this scenario if you want to run a single-tier, public-facing web application such as a blog or simple website.

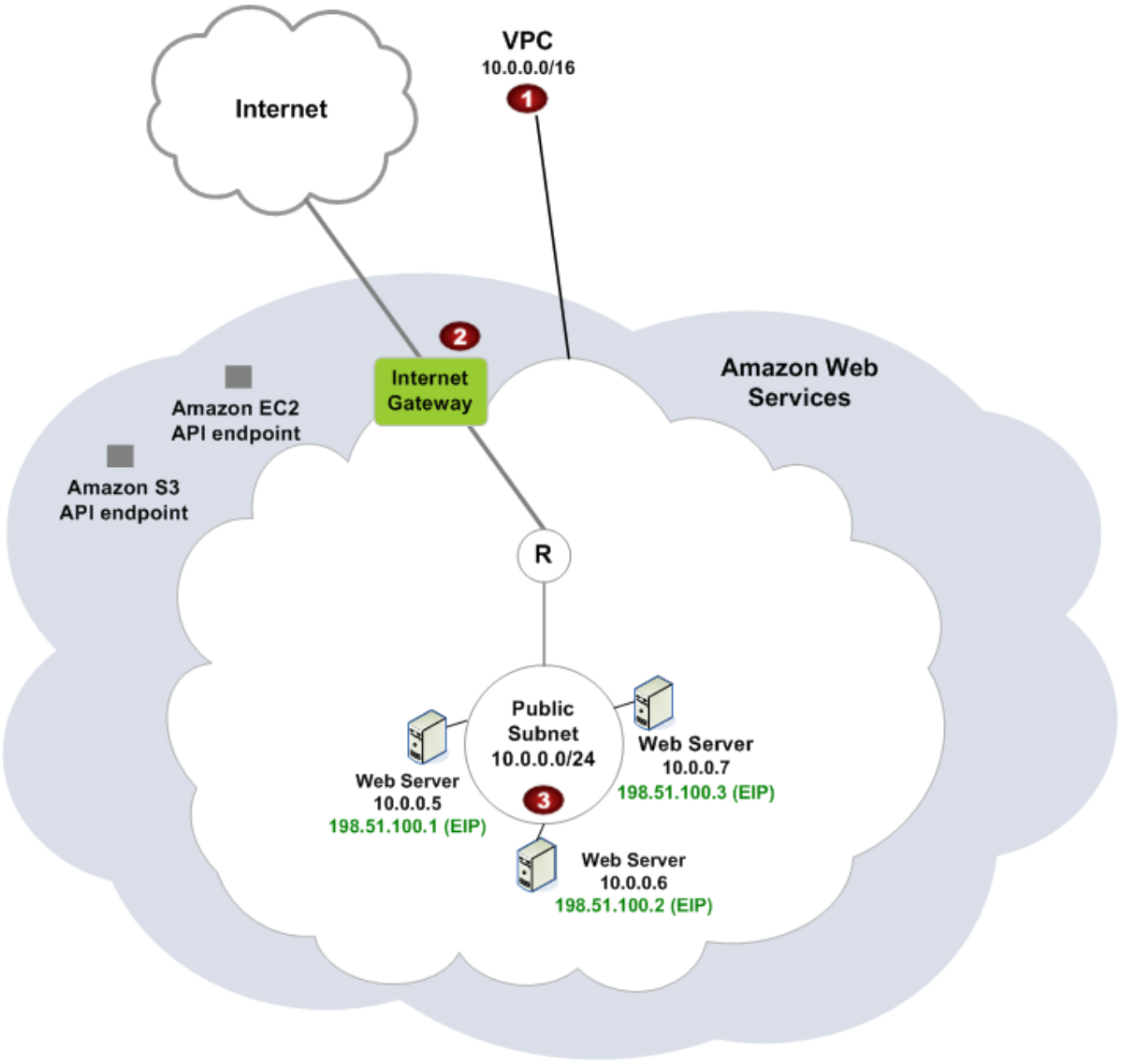
This scenario gives you a VPC with a single public subnet. If you walked through the *Amazon VPC Getting Started Guide*, then you've gone through the steps of implementing this scenario in the AWS Management Console.

Basic Layout

The following diagram shows the basic layout of your VPC in this scenario.

Tip

The AWS Management Console has a wizard in the Amazon VPC console to help you implement this scenario. For more information, go to the Amazon Virtual Private Cloud Getting Started Guide.



1	A size /16 VPC (e.g., 10.0.0.0/16), which means 65,536 private (RFC 1918) IP addresses. For information about CIDR notation and what the "/16" means, go to the Wikipedia article about Classless Inter-Domain Routing.

2	An Internet gateway connecting the VPC to the Internet.
3	A size /24 subnet (e.g., 10.0.0.0/24), which means 256 private IP addresses. For the purposes of this scenario, imagine the subnet contains web servers or other kinds of public instances. Each has a private IP address (e.g., 10.0.0.5) and an Elastic IP address (198.51.100.1), which allows the instance to be reached from the Internet. The addresses shown in the diagram are examples; you'll probably have different values you when implement the scenario. The routing is set up in the VPC so that the subnet can communicate directly with the Internet. Therefore, the subnet is labeled as *public* in the diagram.

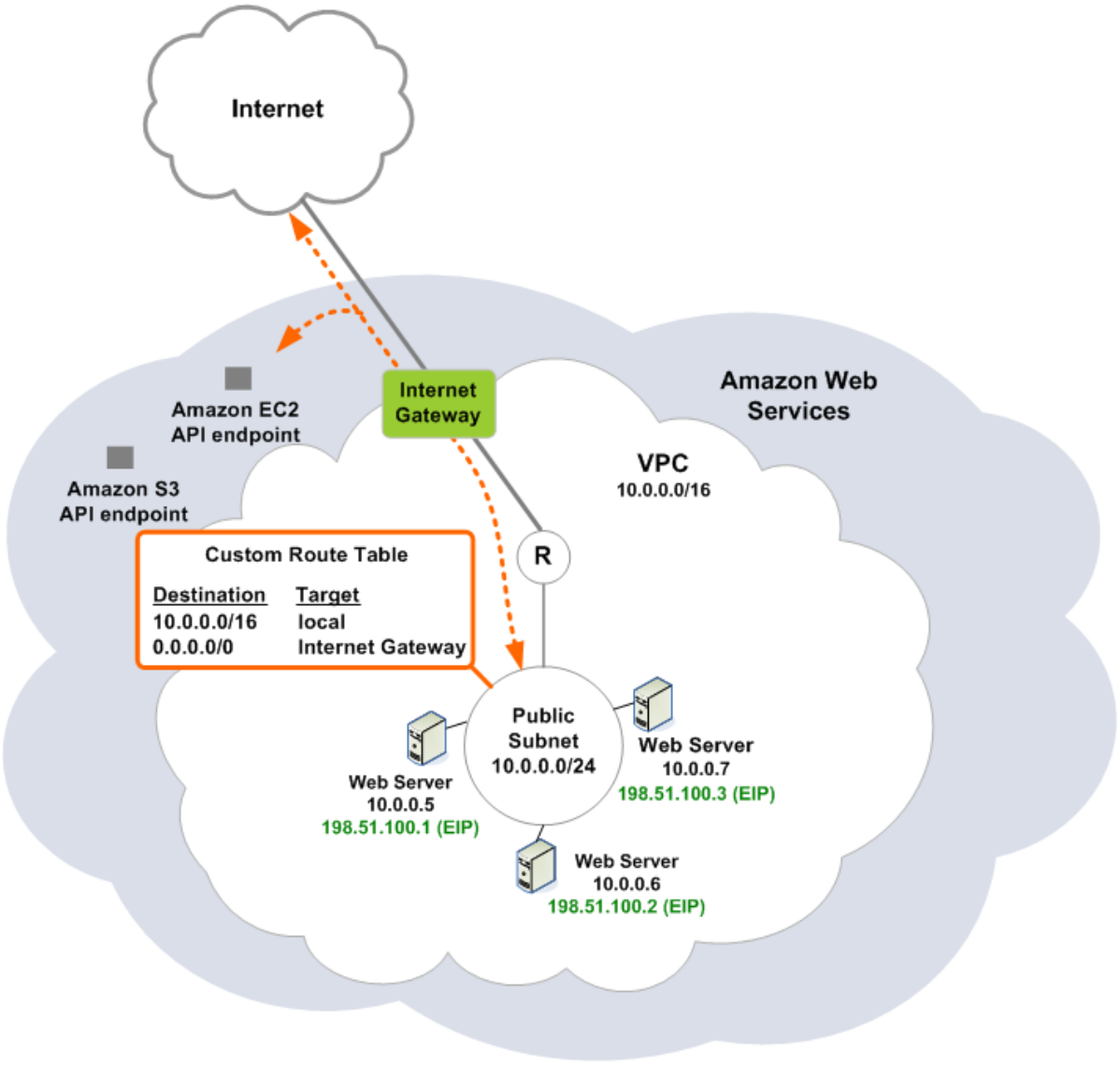
For this scenario, if you want an instance in your VPC to be reachable from the Internet or to reach the Internet, that instance must have an Elastic IP address associated with it. If an instance doesn't have an Elastic IP address, it can still communicate with other instances in the subnet and VPC (assuming the VPC's routing and security settings allow it). For more information about elastic IP addresses, see Elastic IP Addresses (p. 129).

Tip

If you'd like instances in your VPC to be able to reach the Internet without your having to assign each instance an Elastic IP address, see Scenario 2: VPC with Public and Private Subnets (p. 16).

Routing

Your VPC has an implied router (shown in the following diagram as an R in a circle). For this scenario, you create a route table that routes all traffic not destined for other instances in the VPC to the Internet gateway. In the following diagram, this route is indicated by the dotted line.



The following table shows what the route table looks like for this scenario. The first row covers the local routing in the VPC (i.e., allows the instances in the VPC to communicate with each other). The second row routes all other subnet traffic to the Internet gateway, which is specified by its AWS-assigned identifier.

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

Note

If you use the wizard in the console to set up your VPC, the wizard automatically creates this route table and associates it with the subnet. Otherwise, you must create and associate the table yourself.

Any AWS-bound traffic from the subnet (e.g., going to the Amazon EC2 or Amazon S3 API endpoints) goes over the Internet gateway; however, you're not charged for bandwidth if the traffic is bound for the same Region the VPC is in. Exception: You're charged Regional Data Transfer rates for data transferred between your VPC and Amazon EC2 instances in the same Region, regardless of Availability Zone.

Security

AWS provides two ways for you to control security in your VPC: *security groups* and *network ACLs*. They both enable you to control what traffic goes in and out of your instances, but security groups work at the instance level, and network ACLs work at the subnet level. Security groups alone will suffice for many VPC users. However, some users might want 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. 136).

Important

Security groups are a basic Amazon EC2 concept. However, security groups in a VPC have different capabilities than security groups in EC2 (see EC2 vs. VPC Security Groups (p. 139)).

Recommended Security Groups

For scenario 1, you use only security groups and not network ACLs. A security group is just a group of instances that shares a common set of inbound and outbound rules. To use security groups, you create a group, add the rules you want to the group, and then launch instances into the group. You can add and remove rules from the group, and those changes automatically apply to the instances in the group. You can launch an instance into more than one group, and you can change an instance's group membership after launch. For more information about security groups, see Security Groups (p. 137).

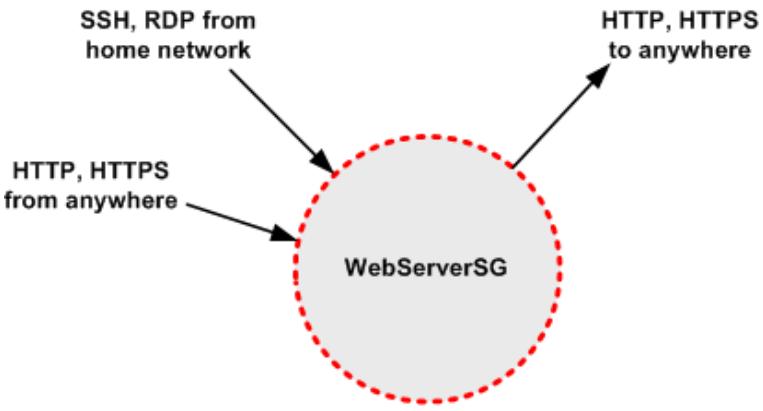
Your VPC comes with a *default security group* whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances in the group. If you don't specify a security group when you launch an instance, the instance automatically goes into this default group. For this scenario, we could just modify the rules for the default group, but the rules you need for your web servers might not be broadly applicable to any instance that might end up in the default group. So for this scenario, we recommend you create a security group (called *WebServerSG*) for the web servers in the public subnet.

The following figure shows the WebServerSG security group as a circle. The circle has arrows indicating the traffic allowed in and out of the security group, based on the rules you add to the group. The rules allow the web servers to receive Internet traffic, as well as SSH and RDP traffic from your home network. The instances can also initiate traffic to the Internet.

Note

Security groups use *stateful filtering*. That is, all response traffic is automatically allowed. For example, if a client on the Internet sends a request to a web server in the WebServerSG, the instance can respond, regardless of any outbound rules on the group. Likewise, if the web server initiates traffic bound to a server on the Internet, the response is allowed back in to the instance, regardless of any inbound rules on the group.

The following table shows the inbound and outbound rules you set up for the WebServerSG group.



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 home network	TCP	22	Allow inbound SSH access to Linux/UNIX instances from your home network
Public IP address range of your home network	TCP	3389	Allow inbound RDP access to Windows instances from your home network
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow outbound HTTP access to servers on the Internet (e.g., for software updates)
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to servers on the Internet (e.g., for software updates)

Even though some instances are in the same security group (e.g, the web servers are together in the WebServerSG), they can't automatically talk to each other. By default, security groups don't contain rules that allow instances in the group to communicate with each other. Exception: the VPC's default security group has such rules. If you want to allow that type of communication, you must add a rule like the one in the following example for the WebServerSG group.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**

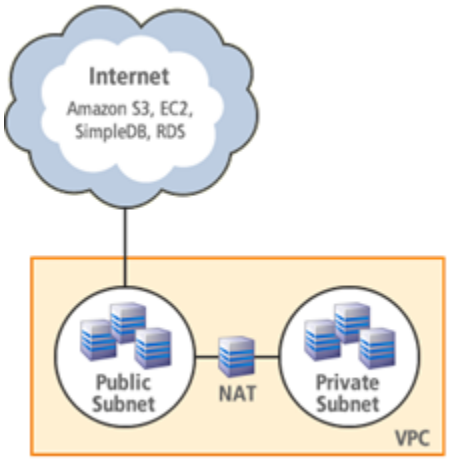
WebServerSG	All	All	Allow inbound traffic from WebServerSG
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
WebServerSG	All	All	Allow outbound traffic from WebServerSG

Implementing the Scenario

For a complete discussion on how to implement this particular scenario, go to the Amazon Virtual Private Cloud Getting Started Guide.

Scenario 2: VPC with Public and Private Subnets

Topics



VPC with Public and Private Subnets

We recommend this scenario if you want to run a public-facing web application, while still maintaining non-publicly accessible backend servers in a second subnet. A common example is a multi-tier website, with web servers in a public subnet, and database servers in a private subnet. You can set up the security in the VPC 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 cannot. The instances in the public subnet can send outbound traffic directly to the Internet, whereas the instances in the private subnet cannot. Instead, they can access the Internet by using a Network Address Translation (NAT) instance that you place in the public subnet.

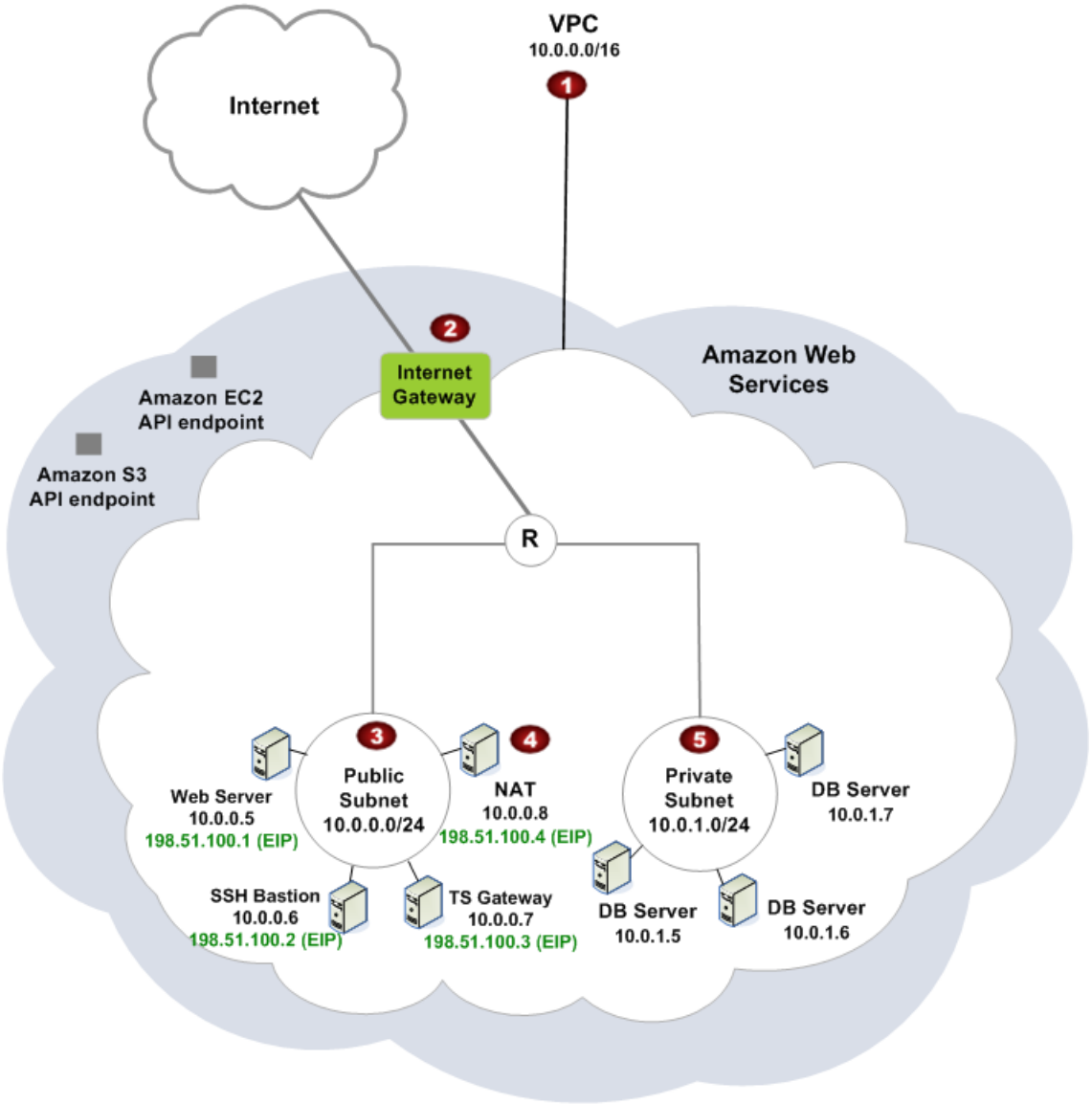
To help manage the instances in the private subnet, you can set up bastion servers in the public subnet to act as proxies. For example, you can set up SSH port forwarders or RDP gateways in the public subnet to proxy the traffic going to your database servers from your home network.

Basic Layout

The following diagram shows the basic layout of your VPC in this scenario. The big white cloud is your VPC (your isolated portion of the AWS cloud). You have an Internet gateway attached to the VPC that enables the VPC to communicate with the Internet. The circle containing an R represents your VPC's implied router. The VPC has two subnets: one public and one private. The table following the diagram gives additional details about the VPC and its layout for this scenario.

Tip

The AWS Management Console has a wizard in the Amazon VPC console to help you implement this scenario. For more information, see Implementing the Scenario (p. 27).



1	A VPC of size /16 (e.g., 10.0.0.0/16), which means 65,536 private (RFC 1918) IP addresses. For information about CIDR notation and what the "/16" means, go to the Wikipedia article about Classless Inter-Domain Routing.
2	An Internet gateway connecting the VPC to the Internet.
3	A subnet of size /24 (e.g., 10.0.0.0/24), which means 256 private IP addresses. For the purposes of this scenario, imagine the subnet contains web servers and bastion hosts (e.g., an SSH bastion for Linux/UNIX instances and a Terminal Services gateway for Windows instances). Each instance has a private IP address (e.g., 10.0.0.5) and an Elastic IP address (198.51.100.1), which allows the instance to be reached from the Internet. The addresses shown in the diagram are examples; you'll probably have different values when you implement the scenario. You're going to set up routing in the VPC so that the subnet can communicate directly with the Internet (see Routing (p. 18)). Therefore, the subnet is labeled as *public* in the diagram.
4	A Network Address Translation (NAT) instance with its own Elastic IP address. This instance enables the instances in the private subnet (see the next item) to send requests out to the Internet (e.g., for software updates). Amazon provides AMIs specifically to act as NAT instances in your VPC. For more information, see NAT Instances (p. 132). You're charged for this instance like any other instance you launch. The NAT instance's primary role is actually Port Address Translation (PAT). However, we use the more widely known term *NAT* when referring to the instance. For information about PAT, go to the Wikipedia article about PAT.
5	Another subnet, also of size /24 (e.g., 10.0.1.0/24). In the diagram, the subnet contains backend services for your website (e.g., database servers). Each server has a private IP address (e.g., 10.0.1.5). Unlike the web servers in the public subnet, these servers don't need to accept incoming traffic from the Internet (and should not). The routing you set up for this subnet prevents traffic going directly from the Internet gateway to the subnet, so we refer to the subnet as *private* in the diagram. However, the routing allows the instances to send requests to the Internet via the NAT instance.

If you want an instance in your VPC to be reachable from the Internet, that instance must be in the public subnet and have an Elastic IP address associated with it.

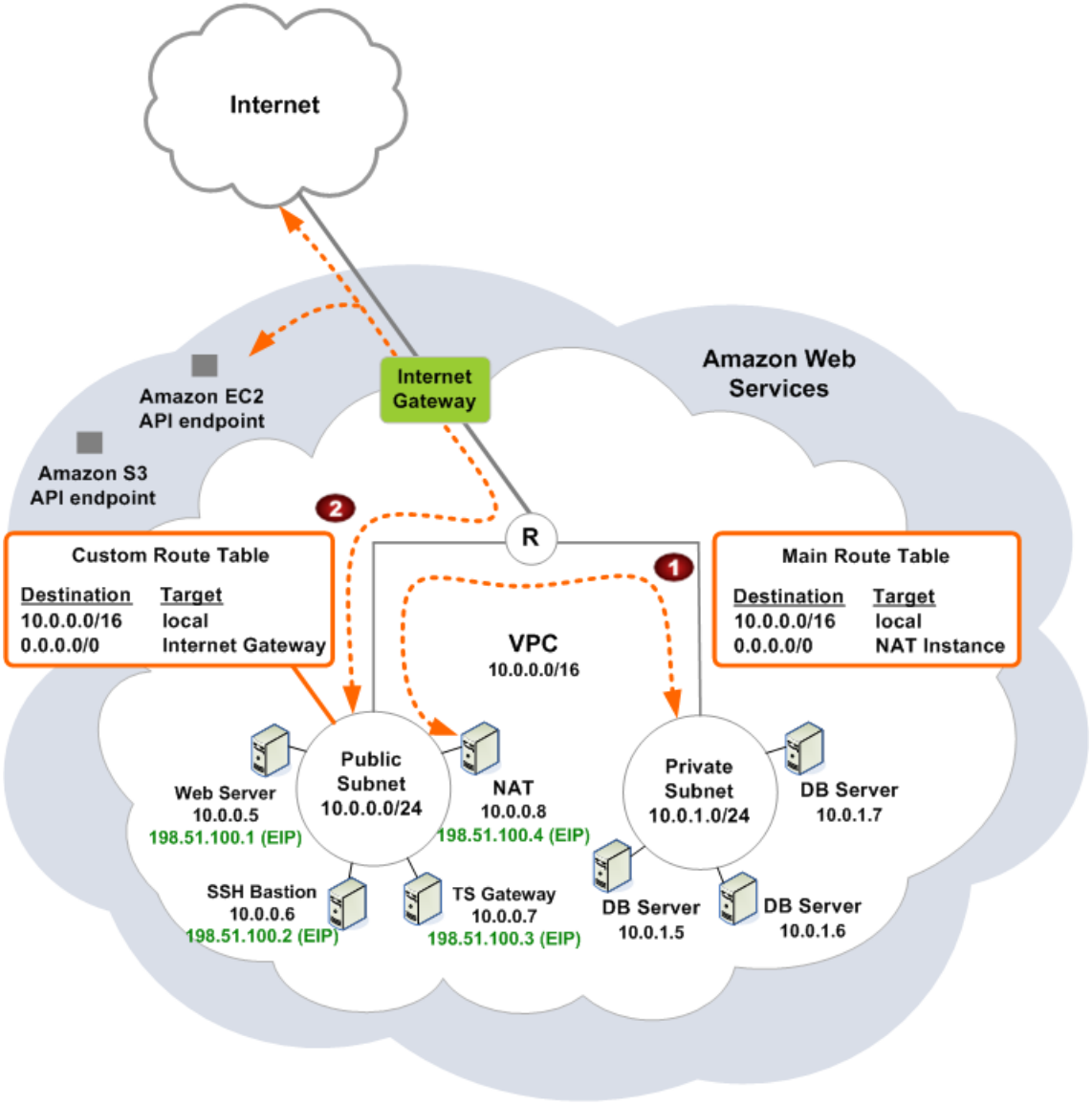
If you want an instance to be able to initiate traffic to the Internet, it must either be in the public subnet and have its own elastic IP address, or it must be in the private subnet and send its Internet-bound traffic to a NAT instance in the public subnet.

If an instance doesn't have an Elastic IP address associated with it, it can still communicate with other instances in the subnet and VPC (assuming the VPC's routing and security settings allow it). For more information about Elastic IP addresses, see Elastic IP Addresses (p. 129).

Routing

Your VPC has an implied router (shown in the following diagram as an R in a circle), as well as a modifiable main route table. You can also create other route tables to use in your VPC. By default, each table has a *local route* that enables instances in your VPC to talk to each other.

The following diagram and table describe the route tables and routes you need to set up in this scenario.



1	The VPC automatically comes with a main route table. Any subnet not explicitly associated with another route table uses the main route table. For this scenario, you update the main route table with a route that sends traffic from the private subnet to the NAT instance in the public subnet (the flow of traffic is indicated by the dotted line adjacent to the table). This route prevents the instances in the subnet from sending traffic directly to the Internet gateway. Also, the database servers in this subnet can't receive traffic directly from the Internet gateway because they don't have Elastic IP addresses. Thus the subnet is labeled *private* in the diagram. However, the servers can send and receive Internet traffic via the NAT instance. They can also receive SSH traffic and Remote Desktop traffic from your home network via an SSH bastion instance and a Terminal Services gateway instance that you launch in the public subnet. You haven't associated the private subnet with a route table, so it uses the routes in the main route table by default. Any new subnets you create use the main route table by default, which means they are *private* by default (not reachable from the Internet). You can always change which route table a subnet is associated with if you want.

2	Your VPC can have other route tables besides the main route table. For this scenario, you must create a route table (it's labeled *Custom Route Table* in the preceding diagram) with a route that sends traffic from the public subnet to the Internet gateway (the flow of traffic is indicated by the dotted line adjacent to the table). After creating the custom route table and the route, you must associate the public subnet with the table. This association is represented by the line connecting the table to the subnet in the diagram. Notice that there's no line connecting the main route table to the private subnet; the absence of a line indicates an implied (default) association with the main route table.

The following two tables show what the route tables look like for this scenario. In each, the first row covers the local routing in the VPC (i.e., allows the instances in the VPC to communicate with each other).

Main Route Table

The first row provides local routing within the VPC. The second row in the main route table sends all the subnet traffic to the NAT instance, which is specified by its AWS-assigned identifier (e.g., i-1a2b3c4d).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	i-xxxxxxxx

Custom Route Table

The first row provides local routing within the VPC. The second row in the custom route table sends all other subnet traffic to the Internet gateway, which is specified by its AWS-assigned identifier (e.g., igw-1a2b3d4d).

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

Note

If you use the wizard in the console to set up your VPC, the wizard automatically updates the main route table and creates a custom route table with the routes shown in the preceding tables. Otherwise, you must make these routing changes yourself.

In this scenario, all AWS-bound traffic from each subnet (e.g., going to the Amazon EC2 or Amazon S3 API endpoints) ultimately goes to the Internet gateway. If the traffic is bound for AWS in the same Region as the VPC, there's no bandwidth charge. Exception: You're charged Regional Data Transfer rates for data transferred between your VPC and Amazon EC2 instances in the same Region, regardless of Availability Zone.

Security

AWS provides two ways for you to control security in your VPC: *security groups* and *network ACLs*. They both enable you to control what traffic goes in and out of your instances, but security groups work at the instance level, and network ACLs work at the subnet level. Security groups alone will suffice for many VPC users. However, some users might want 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. 136).

Important

Security groups are a basic Amazon EC2 concept. However, security groups in a VPC have different capabilities than security groups in EC2 (see EC2 vs. VPC Security Groups (p. 139)).

Recommended Security Groups

For scenario 2, you use only security groups and not network ACLs. A security group is just a group of instances that share a common set of inbound and outbound rules. To use security groups, you create a group, add the rules you want to the group, and then launch instances into the group. You can add and remove rules from the group, and those changes automatically apply to the instances in the group. You can launch an instance into more than one group, and you can change an instance's group membership after launch. For more information about security groups, see Security Groups (p. 137).

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 in the group. If you don't specify a security group when you launch an instance, the instance automatically goes into this default group. You must change the group's rules from the initial default rules if you want the instances to receive traffic from outside the group.

For this scenario, we recommend you not use the default security group and instead create your own groups with the following names (you can use other names if you like):

- **WebServerSG—**For the web servers in the public subnet
- **NATSG—**For the NAT instance in the public subnet
- **BastionSG—**For bastion servers in the public subnet, which act as proxies for SSH and RDP traffic from your home network to the private subnet
- **DBServerSG—**For the database servers in the private subnet

You add rules to each group that let your instances perform only the tasks they need to. For example, you enable the web servers in the public subnet to receive Internet traffic and to post data to the database servers in the private subnet. You enable all instances to receive SSH or RDP traffic from your home network, and so on.

The following figures show each security group as a circle. Some of the figures show a simplified light-gray VPC is in the background to help you understand how the different VPC parts are related. Each circle has arrows indicating the traffic allowed in and out of the security group, based on the rules you add to the group.

Important

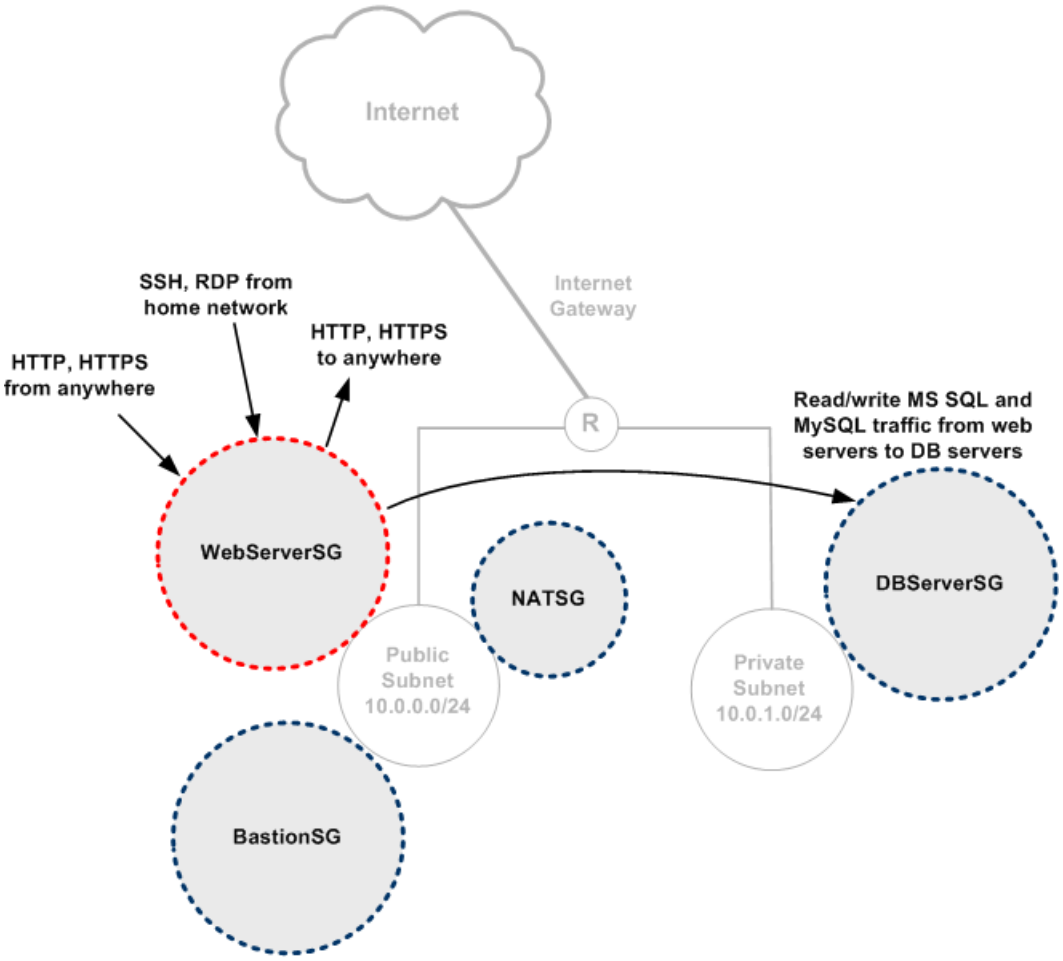
Security groups are independent of network topology. The following diagrams show security groups adjacent to subnets in the VPC. This does not indicate a relationship between the security group and the subnet. Instead, the intention is to show that one or more instances in a given subnet will be launched into each adjacent security group. For example, some instances in the public subnet will be launched into the WebServerSG group, others in that subnet will be launched into the BastionSG group, and one instance in that subnet will be launched into the NATSG group. Therefore, the public subnet is shown adjacent to those three security groups in the diagram.

The instances in a given security group do not have to be in the same subnet. 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, all instances in a given security group in this scenario are in the same subnet.

Let's start with the WebServerSG security group, which you launch your web servers into. Based on the rules in the following table, the web servers can receive Internet traffic, as well as SSH traffic from your home network (for Linux/UNIX instances) and RDP traffic from your home network (for Windows instances). The instances can also initiate traffic to the Internet, and read and write data to the database server instances in the private subnet.

Note

Security groups use *stateful filtering*. That is, all response traffic is automatically allowed. For example, if a client on the Internet sends a request to a web server in the WebServerSG, the instance can respond, regardless of any outbound rules on the group. Likewise, if the web server initiates traffic bound to a server on the Internet, the response is allowed back in to the instance, regardless of any inbound rules on the group.



WebServerSG

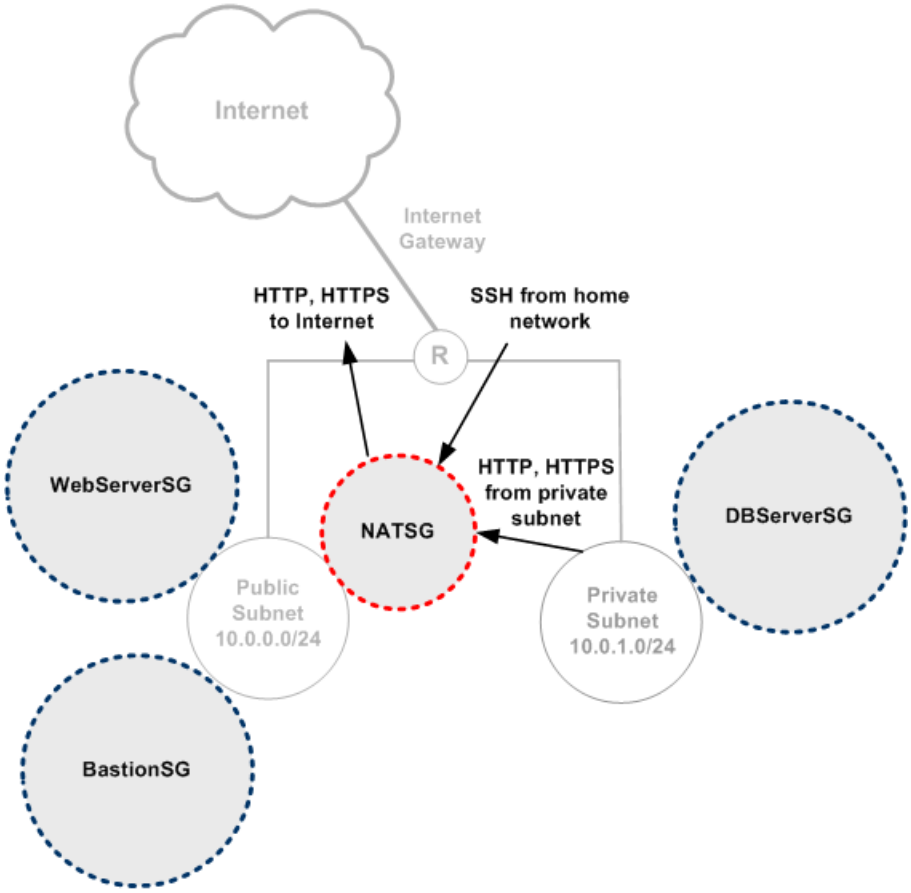
Inbound			
Source	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere

0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
Public IP address range of your home network	TCP	22	Allow inbound SSH access to Linux/UNIX instances from your home network (over the Internet)
Public IP address range of your home network	TCP	3389	Allow inbound RDP access to Windows instances from your home network (over the Internet)
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow web servers to initiate outbound HTTP access to the Internet (e.g., for software updates)
0.0.0.0/0	TCP	443	Allow web servers to initiate outbound HTTPS access to the Internet (e.g., for software updates)
DBServerSG	TCP	1433	Allow outbound SQL access to SQL Server instances in the DBServerSG
DBServerSG	TCP	3306	Allow outbound access to MySQL servers in DBServerSG

Note

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

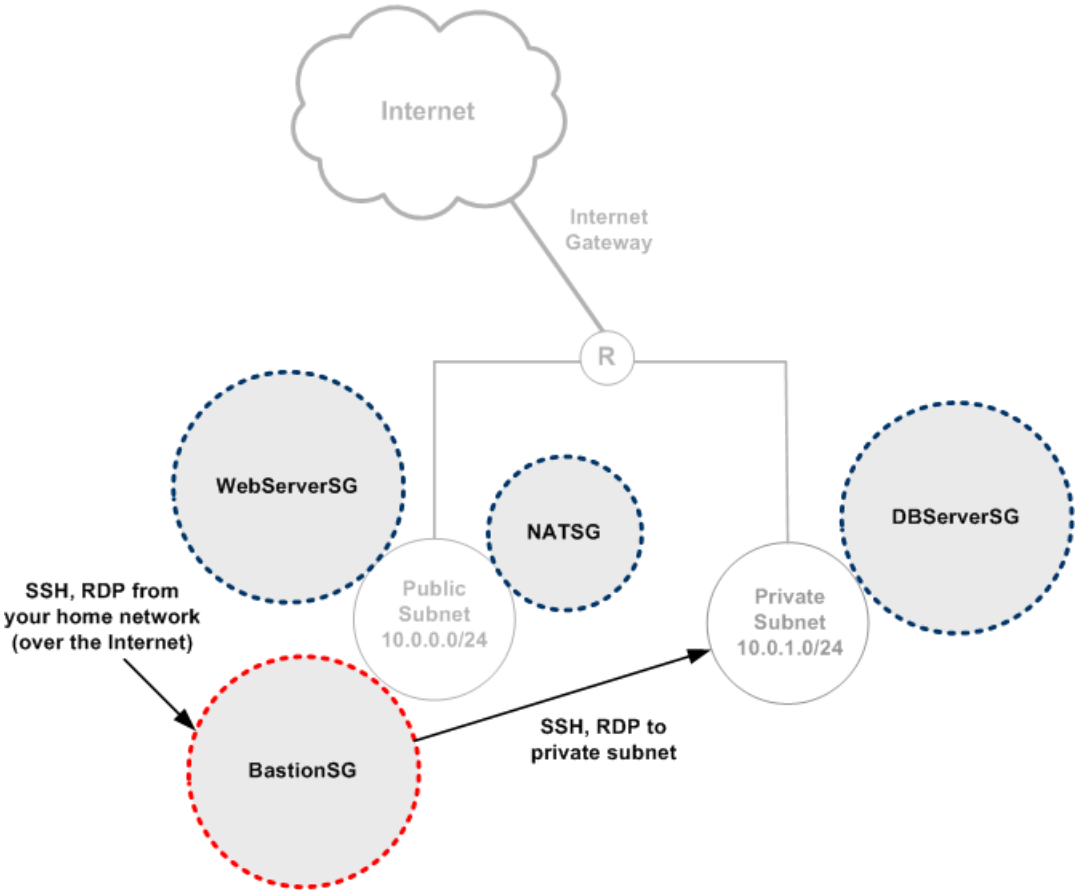
Next is the NATSG security group, which you launch your NAT instance into. Based on the rules in the following table, the NAT instance can receive Internet-bound traffic from the instances in the private subnet, as well as SSH traffic from your home network (the NAT instance is a Linux/UNIX instance). The NAT instance can also send traffic to the Internet. This 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 home network	TCP	22	Allow inbound SSH access to the Linux/UNIX NAT instance from your home network (over the Internet)
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

Next is the BastionSG security group, which you launch proxy instances into. This can include an instance that serves as an SSH proxy, or an instance that serves as a Terminal Services gateway. These instances proxy the SSH and RDP traffic from your home network to the instances in the private subnet.

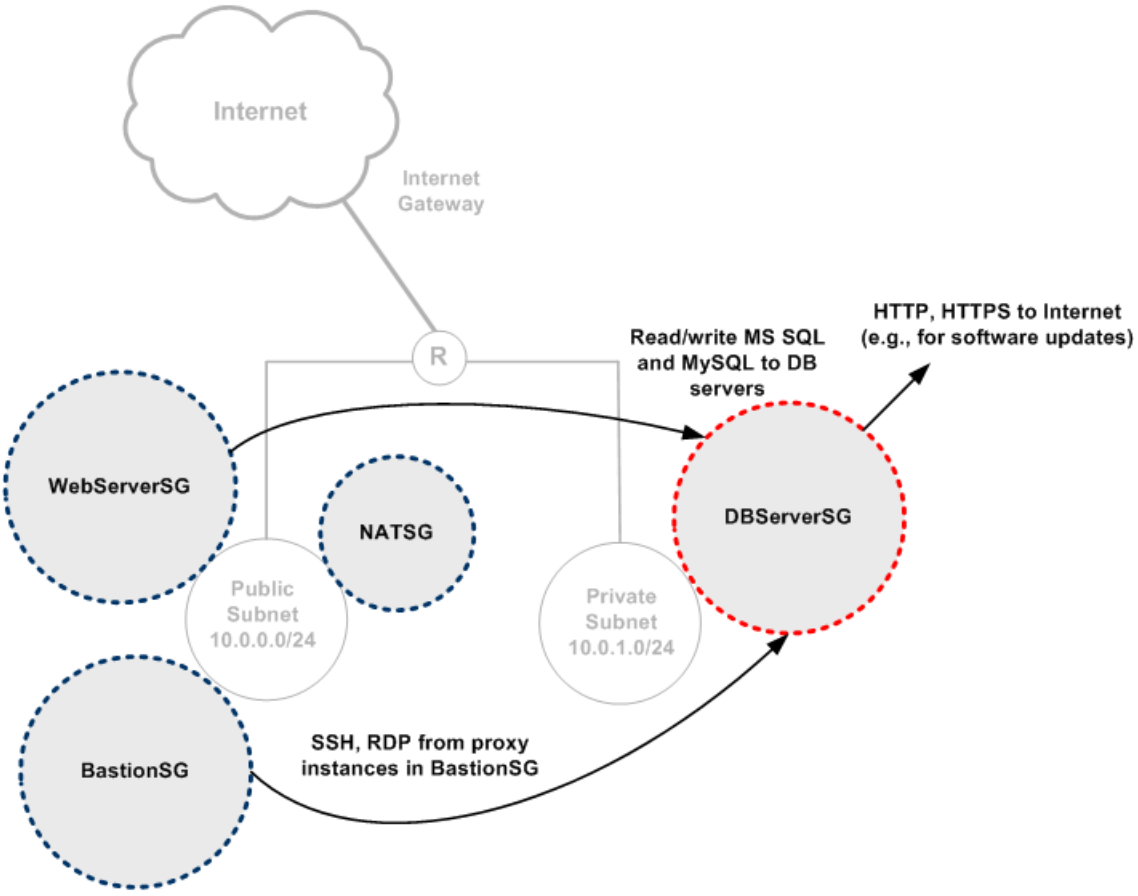


BastionSG

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
Public IP address range of your home network	TCP	22	Allow inbound SSH traffic to Linux/UNIX bastion host from your home network (over the Internet)
Public IP address range of your home network	TCP	3389	Allow inbound RDP traffic to Windows Terminal Services gateway host from your home network (over the Internet)
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
10.0.1.0/24	TCP	22	Allow outbound SSH traffic from Linux/UNIX bastion host to servers in private subnet

10.0.1.0/24	TCP	3389	Allow outbound RDP traffic from Windows Terminal Services gateway host to servers in private subnet

Next is the DBServerSG security group, which you launch your database servers into. Based on the rules in the following table, the database servers allow read or write MS SQL or MySQL requests from the web servers. The database servers also allow SSH and RDP traffic from the proxy servers. The instances can also initiate traffic bound for the Internet (your VPC's routing sends that traffic to the NAT instance, which then forwards it to the Internet).



DBServerSG

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
WebServerSG	TCP	1433	Allow servers in the WebServerSG group to read and write over MS SQL port 1433 to instances in DBServerSG group

WebServerSG	TCP	3306	Allow servers in the WebServerSG group to read and write over MySQL port 3306 to instances in DBServerSG group
BastionSG	TCP	22	Allow inbound SSH traffic from Linux/UNIX bastion host in BastionSG
BastionSG	TCP	3389	Allow inbound RDP traffic from Windows Terminal Services gateway host in BastionSG
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow outbound HTTP access to the Internet (e.g., for software updates)
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (e.g., for software updates)

Even though some instances are in the same security group (e.g, the web servers are together in the WebServerSG), they can't automatically talk to each other. By default, security groups don't contain rules that allow instances in the group to communicate with each other. Exception: the VPC's default security group has such rules. If you want to allow that type of communication, you must add a rule like the one in the following example for the WebServerSG group.

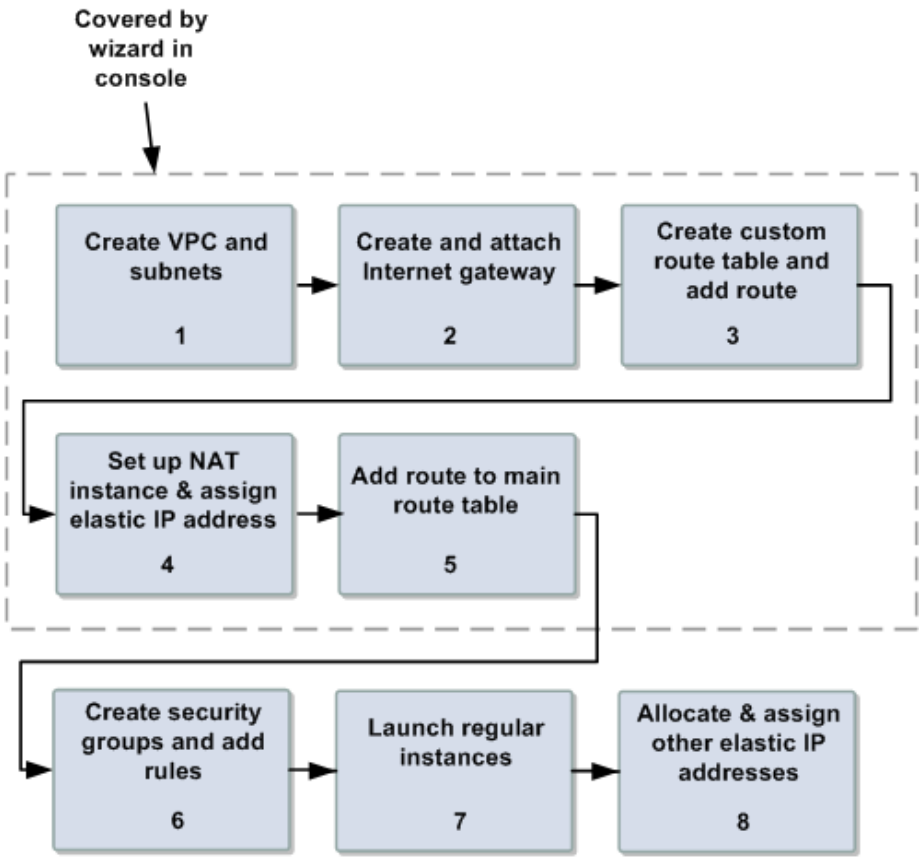
Inbound			
Source	**Protocol**	**Port Range**	**Comments**
WebServerSG	All	All	Allow inbound traffic from WebServerSG
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
WebServerSG	All	All	Allow outbound traffic from WebServerSG

Implementing the Scenario

This section walks you through the process of implementing scenario 2. The following figure and table show the tasks required to implement the scenario.

Tip

Several of the tasks are automatically handled for you if you use the wizard in the console. The following sections describe how to use the wizard, and how to do all the tasks manually.



Process for Implementing Scenario 2

Task 1: Create the VPC and Subnets (p. 31)
Task 2: Create and Attach the Internet Gateway (p. 32)
Task 3: Create a Custom Route Table and Add Routes (p. 32)
Task 4: Set Up the NAT Instance (p. 33)
Task 5: Add a Route to the Main Route Table (p. 37)
Task 6: Create Security Groups and Add Rules (p. 38)
Task 7: Launch Instances into the Subnets (p. 43)
Task 8: Allocate and Assign Elastic IP Addresses (p. 43)

Use the Wizard for Scenario 2

You can have Amazon VPC complete tasks 1-5 for you by using the wizard in the AWS Management Console. This procedure assumes you don't already have a VPC.

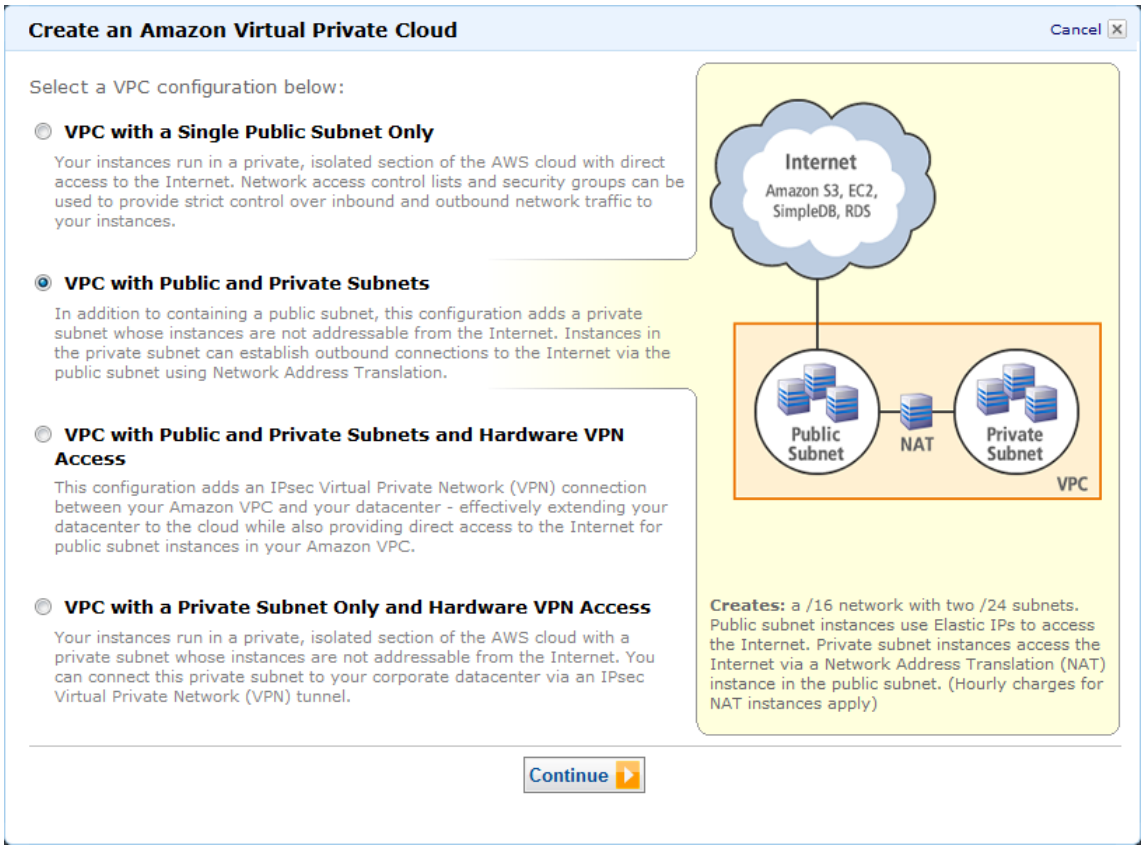
Important

The wizard chooses one of your Amazon EC2 key pairs when launching the NAT instance into the public subnet. The key pair enables you to connect to the instance using SSH or Remote Desktop (RDP). If you don't already have at least one Amazon EC2 key pair in the Region where you're creating the VPC, we recommend you create one before starting the wizard. You can create a new key pair on the **Key Pairs** page on the Amazon EC2 console. For more information about getting key pairs, go to Getting an SSH Key Pair in the *Amazon Elastic Compute Cloud User Guide*.

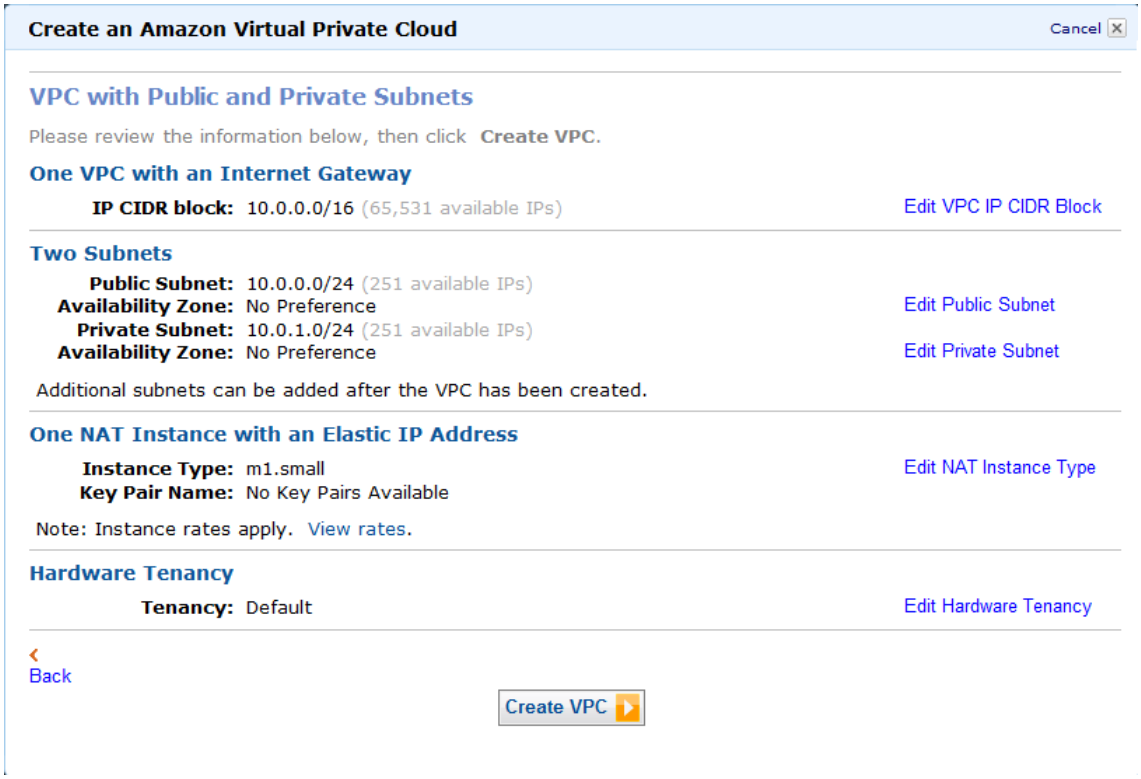
To use the wizard

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **VPC Dashboard**, locate the **Your Virtual Private Cloud** area, and then click **Get started creating a VPC** or **Create another VPC**.

   The wizard opens and displays a page where you can select one of four options.
3. Select the radio button for **VPC with Public and Private Subnets** and click **Continue**.



A confirmation page is displayed showing the CIDR blocks we use for the VPC and subnets. The page also shows the subnets and their associated Availability Zones, the size of the NAT instance we will launch (m1.small), which key pair we'll use to launch the instance, and the instance hardware tenancy of the VPC. You can change any of these values if you want.



4. Click **Continue**.
   The wizard begins to create your VPC, subnets, and Internet gateway. It also updates the main route table and creates a custom route table. Lastly, the wizard launches a NAT instance in the public subnet and prepares it for use. This preparation includes disabling the source/destination check on the instance and assigning the instance an Elastic IP address.

After the wizard completes, you're partway done. The next task is to create the security groups that you need. For more information, see Task 6: Create Security Groups and Add Rules (p. 38)).

Note that the next few sections show you how to manually do tasks that the wizard already completed for you.

Task 1: Create the VPC and Subnets

If you don't use the wizard in the console, you can manually create the VPC and subnets yourself. This section shows you how.

To create your VPC and subnets

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, enter the CIDR range you want for your VPC (e.g., 10.0.0.0/16) and click **Yes, Create**.

   Tip

   For information about choosing the CIDR range for your VPC, see VPC Sizing (p. 105)

   The VPC is created and appears on the **Your VPCs** page. Notice that it has an ID (e.g., vpc-xxxxxxxx).

4. In the **Navigation** pane, click **Subnets**.
5. Click **Create Subnet**.
6. In the **Create Subnet** dialog box, select the VPC and Availability Zone, enter the CIDR range you want for your subnet (e.g., 10.0.0/24), and then click **Yes, Create**.
   The subnet is created and appears on the **Subnets** page. Notice that it has an ID (e.g., subnet-xxxxxxxx). The page also shows the number of available IP addresses in the subnet, the route table associated with the subnet, and the network ACL associated with the subnet. The subnet uses the main route table and default network ACL by default.
7. Create a second subnet (e.g., 10.0.1.0/24) by repeating the preceding steps for creating a subnet.

You've got your VPC and subnets now. Move on to the next section to create and attach an Internet gateway to the VPC.

Task 2: Create and Attach the Internet Gateway

If you don't use the wizard in the console, you can manually create and attach the Internet gateway yourself. This section shows you how.

To create the Internet gateway

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Internet Gateways**.
3. At the top of the **Internet Gateways** page, click **Create Internet Gateway**, and then in the **Create Internet Gateway** dialog box, click **Yes, Create**.
   The Internet gateway is created and appears on the page. Notice that it has an ID (e.g., igw-xxxxxxxx).
4. Select the Internet gateway and click **Attach to VPC**.
5. In the **Attach to VPC** dialog box, in the **VPC** drop-down list box, select a VPC, and then click **Yes, Attach**.

Your VPC has an Internet gateway attached to it now. However, no route table refers to the gateway yet, so no traffic can flow to the gateway. Move on to the next section to set up routing for the public subnet.

Task 3: Create a Custom Route Table and Add Routes

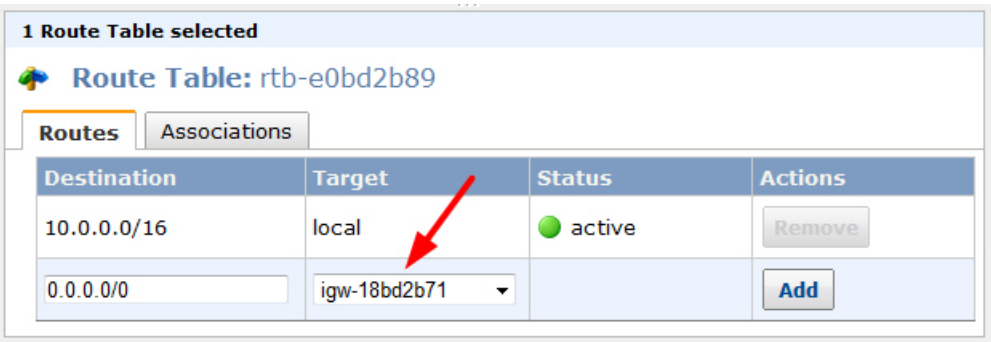
If you don't use the wizard in the console, you can manually create the required custom route table and add routes yourself. This section shows you how.

For this scenario, you create a custom route table with a route to send all the non-local traffic (i.e., 0.0.0.0/0, which means *all* traffic) in the public subnet to the Internet gateway, and you associate the public subnet with the table.

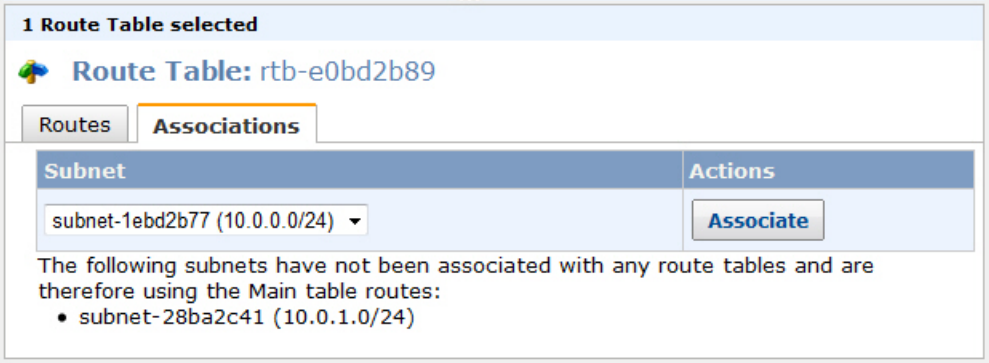
To create a custom route table

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Route Tables**.
   Your VPC's route tables are listed.
3. Click **Create Route Table**.
4. In the **Create Route Table** dialog box, in the **VPC** drop-down list box, select your VPC, and then click **Yes, Create**.
   The new route table is created and appears on the page. Notice that it has an ID (e.g., rtb-xxxxxxxx).
5. Select the check box for the custom route table.

6. On the lower pane, click the **Routes** tab, enter `0.0.0.0/0` in the **Destination** field, select the Internet gateway's ID in the **Target** drop-down list, and then click **Add**.



7. On the **Associations** tab, select the ID of the public subnet and click **Associate**.



   The public subnet is now associated with the custom route table.

The VPC now has a custom route table associated with the public subnet. The table enables traffic to flow between the subnet and the Internet gateway. Move on to the next section to set up the NAT instance in the public subnet.

Task 4: Set Up the NAT Instance

If you don't use the wizard in the console, you can manually launch and set up the NAT instance yourself. This section shows you how.

If you're already familiar with launching Amazon EC2 instances outside a VPC, then you already know most of what you need to know about launching the NAT instance. The additional items to know:

- Amazon provides NAT AMIs you can use (search for AMIs with the string `ami-vpc-nat` in their names).
- You must specify the VPC and subnet when you launch the instance.
- You should put the NAT instance into a security group (you can launch the instance into the default group initially and then later create the NATSG group and move the instance into it).

After the NAT instance is running, you must also do the following tasks to complete the setup:

- Disable the source/destination check on the instance (instructions follow).
- Allocate and assign an Elastic IP address to the instance (instructions follow).

- Create the NATSG security group and move the NAT instance into it (see Task 6: Create Security Groups and Add Rules (p. 38)).

To launch a NAT instance

1. Start the launch wizard:

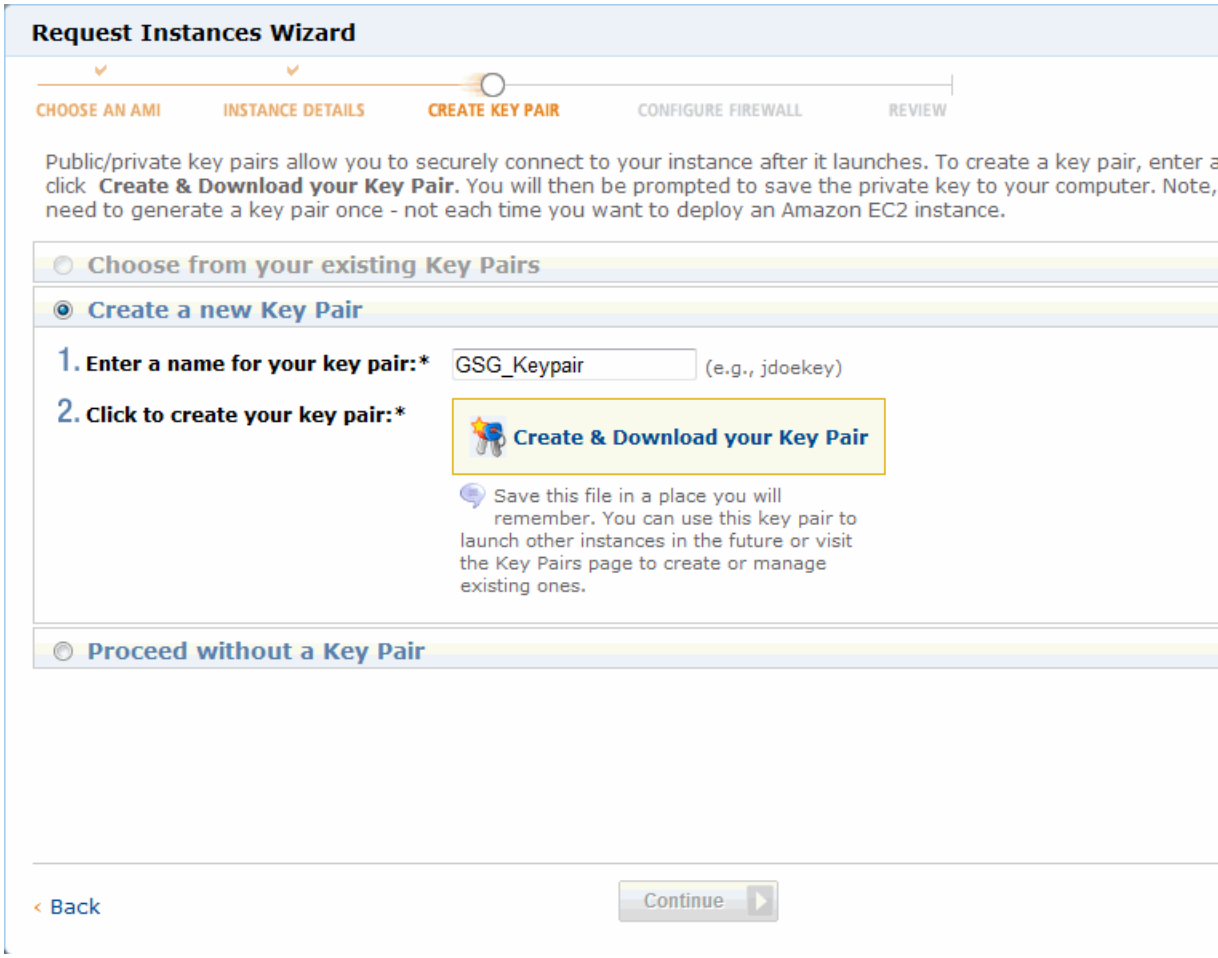
   a. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
   b. In the **Navigation** pane, click **AMIs**.
   c. Change the **Viewing** settings to show Amazon AMIs using the Amazon Linux platform. The NAT AMIs that we provide include the string `ami-vpc-nat` in their names.
   d. Locate the NAT AMI of your choice, right-click it, and select **Launch Instance** to start the launch wizard.

   The wizard opens on the **Instance Details** page. This is where you control settings such as the number and size of instances to launch, and which subnet to launch the instance in.
2. Select the **Launch Instances Into Your Virtual Private Cloud** option, and select the subnet you want to launch the NAT instance in. Keep the other default settings on this page and click **Continue**. The wizard steps to the next page for instance details.
3. The default settings on this page of the wizard and the next page are what you want, so just click **Continue** on each page.
   The **Create Key Pair** page appears.
   A *key pair* is a security credential similar to a password, which you use to securely connect to your instance once it's running. If you're new to Amazon EC2 and haven't created any key pairs yet, when the wizard displays the **Create Key Pair** page, the **Create a new Key Pair** button is selected by default. It's assumed you'll want a new key pair.

   Tip

   If you're already familiar with Amazon EC2 and have an SSH key pair already, you don't need to create a new one now. You can just select one of your existing key pairs instead.
4. Create a key pair:

   a. On the **Create Key Pair** page, enter a name for your key pair (e.g., GSG_Keypair). This is the name of the private key file associated with the pair (with a `.pem` extension).



b. Click **Create & Download your Key Pair**.
You're prompted to save the private key from the key pair to your system.

c. Save the private key in a safe place on your system.

The **Configure Firewall** page is displayed, where you can select a security group for the instance.

5. Select the default security group for now, and click **Continue**.

Note

You'll later create the NATSG security group and move the NAT instance into it.

After you configure the firewall, the wizard steps to the **Review** page where you can review the settings and launch the instance.

6. Review your settings and launch the instance:

a. Click **Launch**.
A confirmation page is displayed to let you know your instance is launching.

b. Click **Close** to close the confirmation page, and then click **Instances** in the navigation pane to view your instance's status. It takes a short time for an instance to launch. The instance's status is *pending* while it's launching. After a short period, your instance's status switches to *running*. You can click **Refresh** to refresh the display.

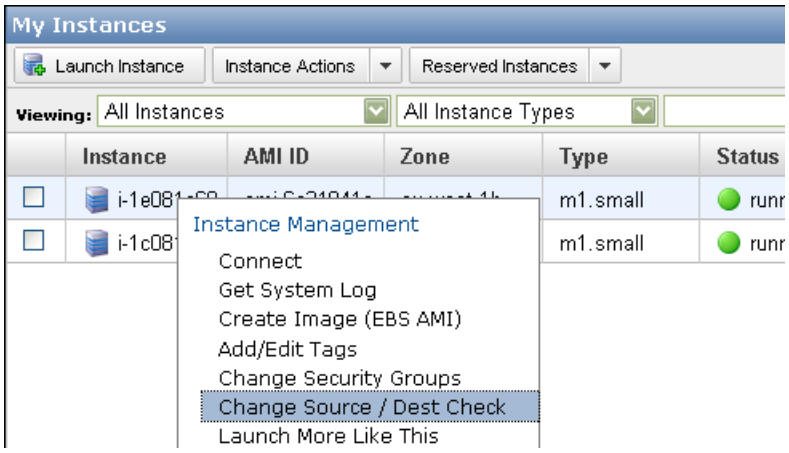
You now have a NAT instance running in your VPC. For the instance to perform network address translation, you must disable source/destination checking on the instance. In other words, each EC2 instance performs source and destination checking by default. This means the instance must be the source or destination of any traffic it sends or receives. However, the NAT instance needs to be able to send and receive traffic where the eventual source or destination is not the NAT instance itself. To enable that behavior, you must disable source/destination checking on the NAT instance.

To disable source/destination checking on the NAT instance

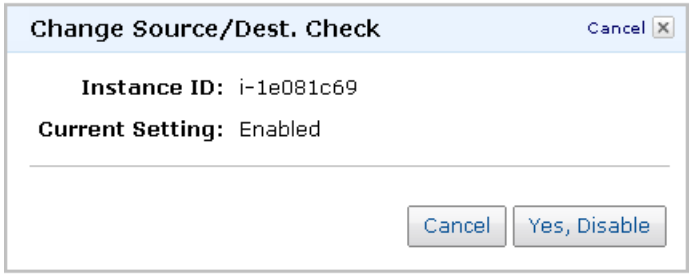
Note

This procedure only works for EC2 instances that are running within a VPC.

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the NAT instance in the list of instances, and select **Change Source / Dest Check**.



The **Change Source/Dest. Check** dialog box opens.



For a regular instance, the value should be *Enabled*, indicating that the instance is performing source/destination checking. For a NAT instance, you want the value to be *Disabled*.

4. Click **Yes, Disable**.

Source/destination checking for the instance is disabled. Your NAT instance also needs an Elastic IP address.

To allocate and assign an elastic IP address to an instance

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

2. In the **Navigation** pane, click **Elastic IPs**, and then click **Allocate New Address**.
3. In the **Allocate New Address** dialog box, in the **EIP used in:** drop-down list, select VPC and click **Yes, Allocate**.
   The new address is allocated and appears on the page.
4. Right-click the IP address in the list and select **Associate**.
5. In the **Associate Address** dialog box, select the instance you want to associate the address with and click **Yes, Associate**.
   The address is associated with the instance. Notice that the instance ID is displayed next to the IP address in the list.

Your NAT instance now has an Elastic IP address associated with it. The instance is currently in the default security group. After you've created your security groups, you need to move the NAT instance into the NATSG group (you'll do that later). Right now, move on to the next section to set up routing for the private subnet.

Task 5: Add a Route to the Main Route Table

If you don't use the wizard in the console, you can manually add the required route to the main route table yourself. This section shows you how.

For this scenario, you add a route that sends all non-local traffic in the private subnet to the NAT instance in the public subnet.

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**.
   Your VPC's route tables are listed.
3. In the list of route tables, select the check box for the main route table.
   The lower pane displays the route table's details.
4. On the **Routes** tab, enter `0.0.0.0/0` in the **Destination** field, select the ID of the NAT instance from the **Target** drop-down list, and click **Add**.

   Tip

   Any instance (running or stopped) in your VPC that has its source/destination checking disabled (e.g., is set up to be a NAT instance) appears in the **Target** drop-down list.

   If you want to select an instance that you haven't yet set up to be a NAT instance, select **Enter Instance ID** in the **Target** drop-down list, and a dialog box opens where you can select the instance. You'll later need to disable source/destination checking on that instance (see Task 4: Set Up the NAT Instance (p. 33)).

   The main route table is updated with a route sending the private subnet's Internet-bound traffic to the NAT instance.

The VPC's main route table now includes the new route. The route enables Internet-bound traffic to flow between the private subnet and the NAT instance. If you click the **Associations** tab (next to the **Routes** tab for the main route table), you can see a bulleted list of the subnets that aren't associated with any other route table and thus are using the main route table. Your VPC's private subnet is listed there.

Move on to the next section to create the security groups you need for this scenario.

Task 6: Create Security Groups and Add Rules

For this scenario, you must create the security groups yourself and add the rules to them. This section shows you how.

You first create all the groups and then add the rules to each. For a list of the groups and their rules for this scenario, see Security (p. 20).

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**.
   Your security groups are listed.

   Note

   This page shows all security groups that belong to your AWS account, including your VPC groups and your EC2 groups. The VPC groups have a value listed in the **VPC ID** column. For information about the different kinds of security groups, see Security Groups (p. 137).
3. Click **Create Security Group**.
   The **Create Security Group** dialog box opens.
4. Enter the name for your security group (e.g., WebServerSG), enter a description of the group, select your VPC's ID from the **VPC** drop-down list, and click **Yes, Create**.
   The security group is created and appears on the **Security Groups** page. Notice that it has an ID (e.g., sg-xxxxxxxx). You might have to turn on the **Group ID** column by clicking **Show/Hide** in the top right corner of the page.
5. Repeat the preceding steps for the remaining security groups you need to create (NATSG, BastionSG, and DBServerSG).

Now that you've created the security groups, you can add rules to them. For a list of the rules to add, see Security (p. 20).

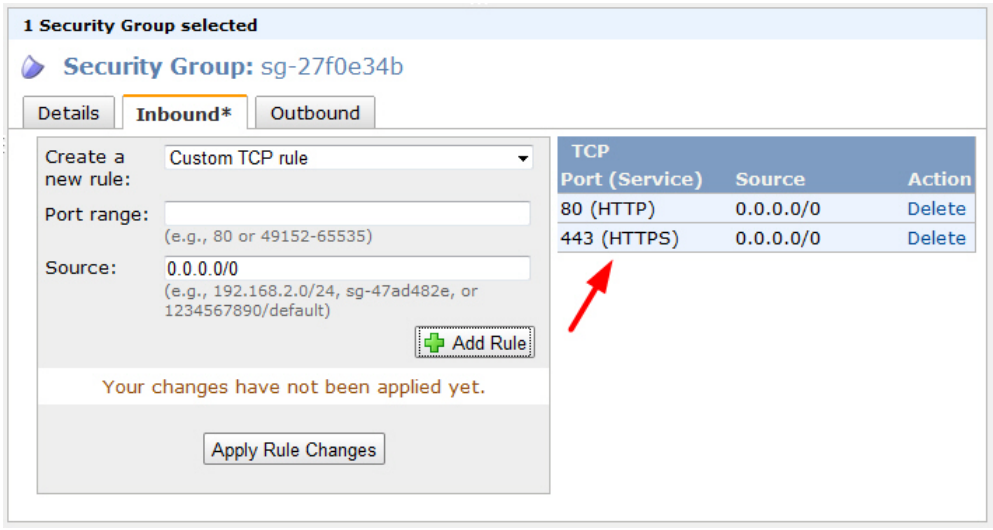
To add rules to the WebServerSG security group

1. In the list of security groups, select the check box for the WebServerSG group you just created.
   The lower pane displays the security group's details.
2. Add rules for inbound HTTP and HTTPS access to the group from anywhere:

   a. On the **Inbound** tab, select `HTTP` from the **Create a new rule** drop-down list.
   b. Make sure the **Source** field's value is `0.0.0.0/0` and click **Add Rule**.

      The rule to allow HTTP access from anywhere (i.e., 0.0.0.0/0) is added to the **Inbound** tab. Notice that the rule on the right is highlighted in blue, and an asterisk appears on the tab. This indicates that you still need to click **Apply Rule Changes** (which you'll do after you've added all the inbound rules).
   c. Select `HTTPS` from the **Create a new rule** drop-down list and click **Add Rule**.

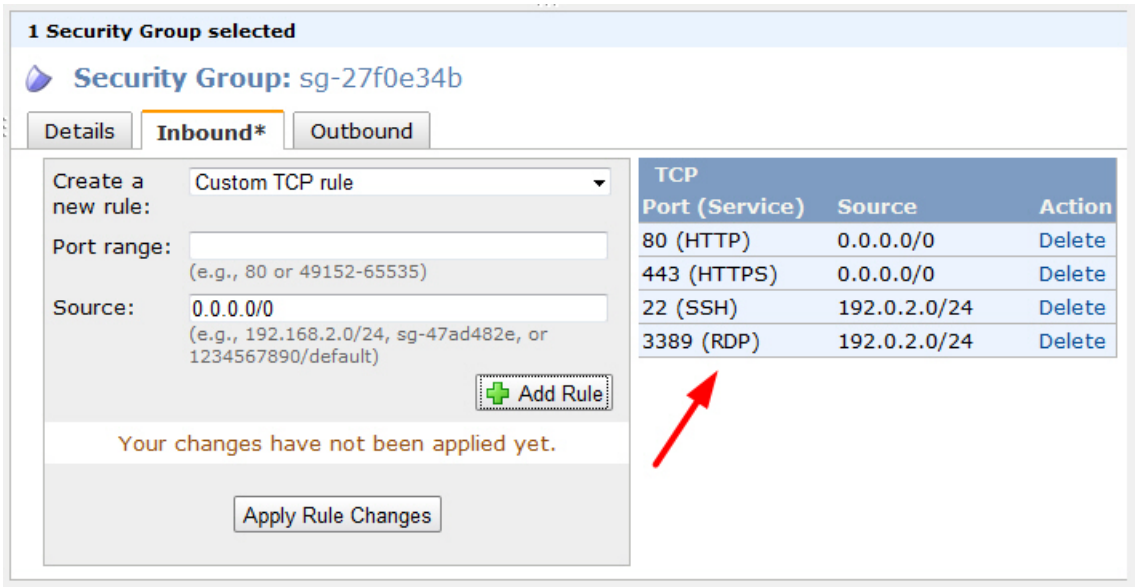
      The rule to allow HTTPS access from anywhere (i.e., 0.0.0.0/0) is added to the **Inbound** tab.



3. Add rules for inbound SSH and Remote Desktop (RDP) access to the group from your home network's public IP address range:

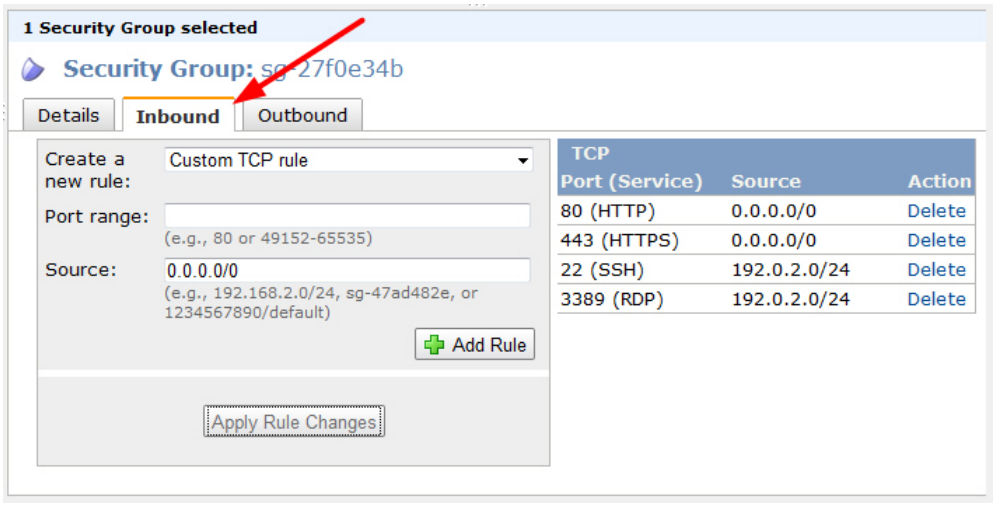
   a. On the **Inbound** tab, select SSH from the **Create a new rule** drop-down list.
   b. In the **Source** field, enter your home network's public IP address range (this example uses 192.0.2.0/24).
   c. Click **Add Rule**.
      The rule is added to the **Inbound** tab.
   d. Select RDP from the **Create a new rule** drop-down list.
   e. In the **Source** field, enter your home network's public IP range.
   f. Click **Add Rule**.

      The rule is added to the **Inbound** tab.



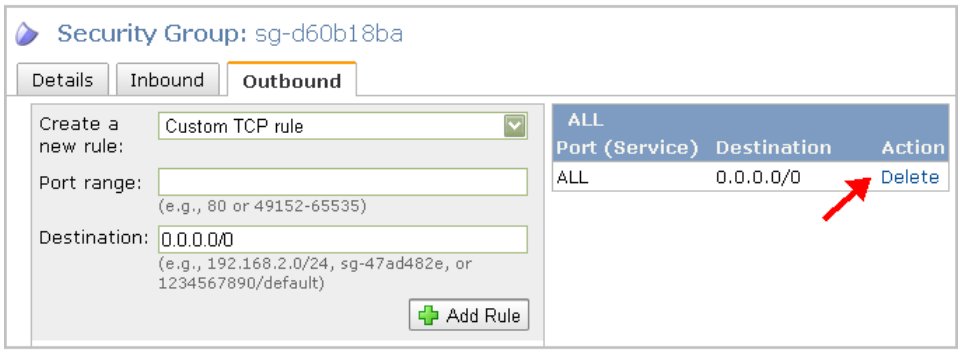
4. Click **Apply Rule Changes**.

   The new inbound rules on the right side of the screen are no longer highlighted in blue, and the asterisk no longer appears on the tab. Those changes indicate that the new inbound rules have been applied.



5. Add the outbound rules to limit egress traffic from the instances:

   a. On the **Outbound** tab, locate the default rule that enables all outbound traffic, and click **Delete**.



   The rule is marked for deletion, and an asterisk appears on the tab. The deletion will not take effect until you click **Apply Rule Changes**, which you'll do after adding new outbound rules to the group.

   b. On the **Outbound** tab, select `HTTP` from the **Create a new rule** drop-down list.

   c. Make sure the **Destination** field's value is `0.0.0.0/0` and click **Add Rule**.
   The rule is added to the **Outbound** tab.

   d. Select `HTTPS` from the **Create a new rule** drop-down list.

   e. Make sure the **Destination** field's value is `0.0.0.0/0` and click **Add Rule**.
   The rule is added to the **Outbound** tab.

   f. Select `MS SQL` (for Microsoft SQL) from the **Create a new rule** drop-down list.

   g. In the **Source** field, start typing `sg-`.
   The drop-down list displays the IDs for your security groups (e.g., sg-xxxxxxxx).

   h. Select the ID for the DBServerSG group and click **Add Rule**.

The rule is added to the **Outbound** tab.

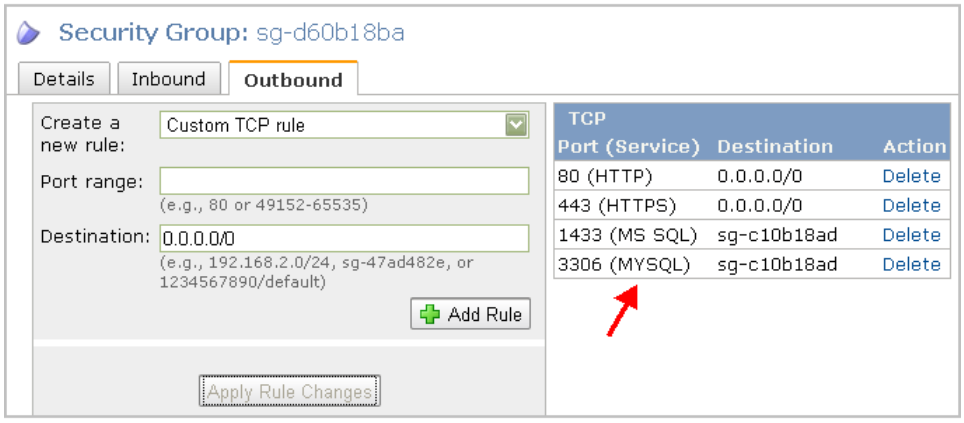
i. Select `MySQL` from the **Create a new rule** drop-down list.

j. In the **Source** field, start typing `DBServerSG`.
The drop-down list displays the ID for the security group (e.g., sg-xxxxxxxx).

k. Select the DBServerSG group from the list and click **Add Rule**.
The rule is added to the **Outbound** tab.

6. Click **Apply Rule Changes**.

The new outbound rules now apply to the security group.



The VPC now includes a security group for the web servers in your subnet. The group allows HTTP/HTTPS access in and out of the group to and from anywhere. The group also allows inbound SSH and RDP access from your home network's IP range. Plus it also allows Microsoft SQL and MySQL access to the DBServerSG group.

Now that you know how to create security groups and add rules to them, you can add rules to the other security groups used in this scenario: NATSG, BastionSG, and DBServerSG. The following images show what the rules look like for each of these groups.

NATSG: Inbound

TCP Port (Service)	Source	Action
80 (HTTP)	10.0.1.0/24	Delete
443 (HTTPS)	10.0.1.0/24	Delete
22 (SSH)	192.0.2.0/24	Delete

NATSG: Outbound

TCP Port (Service)	Destination	Action
80 (HTTP)	0.0.0.0/0	Delete
443 (HTTPS)	0.0.0.0/0	Delete

BastionSG: Inbound

TCP Port (Service)	Source	Action
22 (SSH)	192.0.2.0/24	Delete
3389 (RDP)	192.0.2.0/24	Delete

BastionSG: Outbound

TCP Port (Service)	Destination	Action
22 (SSH)	10.0.1.0/24	Delete
3389 (RDP)	10.0.1.0/24	Delete

DBServerSG: Inbound

TCP Port (Service)	Source	Action
1433 (MS SQL)	sg-d60b18ba	Delete
3306 (MYSQL)	sg-d60b18ba	Delete
22 (SSH)	sg-f20b189e	Delete
3389 (RDP)	sg-f20b189e	Delete

DBServerSG: Outbound

TCP Port (Service)	Destination	Action
80 (HTTP)	0.0.0.0/0	Delete
443 (HTTPS)	0.0.0.0/0	Delete

When you (or the wizard) launched the NAT instance, you put it in the default security group in the VPC. You need to move it into the NATSG group.

To change an instance's group membership

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the NAT instance in the list of instances, and select **Change Security Groups**.

   The **Change Security Groups** dialog box opens, with the default group selected (the instance is in the default group currently).

4. From the drop-down list, select the NATSG group and click **Yes, Change**.

   Tip

   When changing an instance's group membership, you can select multiple groups from the list. The new list of groups you select replaces the instance's current list of groups.

The NAT instance is now in the NATSG security group. Your instances in the private subnet can now reach the Internet via the NAT instance.

Note

The preceding procedure works only for VPC instances. You can't change security group membership for standard (EC2) instances.

Move on to the next section to launch instances into your subnets.

Task 7: Launch Instances into the Subnets

After you have your VPC, subnets, Internet gateway, routing, NAT instance, and security groups, you can launch instances using AMIs of your choice into your VPC. For example, you launch instances of a web server AMI into the public subnet, and instances of a database server AMI into the private subnet. If you're not familiar with the general procedure, see Task 4: Set Up the NAT Instance (p. 33).

After you've launched instances, move on to the next section to associate Elastic IP addresses with web servers in the public subnet.

Task 8: Allocate and Assign Elastic IP Addresses

You should have at least one instance running in each of your subnets. Now you can allocate and assign Elastic IP addresses to any instances that need them (i.e., the web servers in the public subnet).

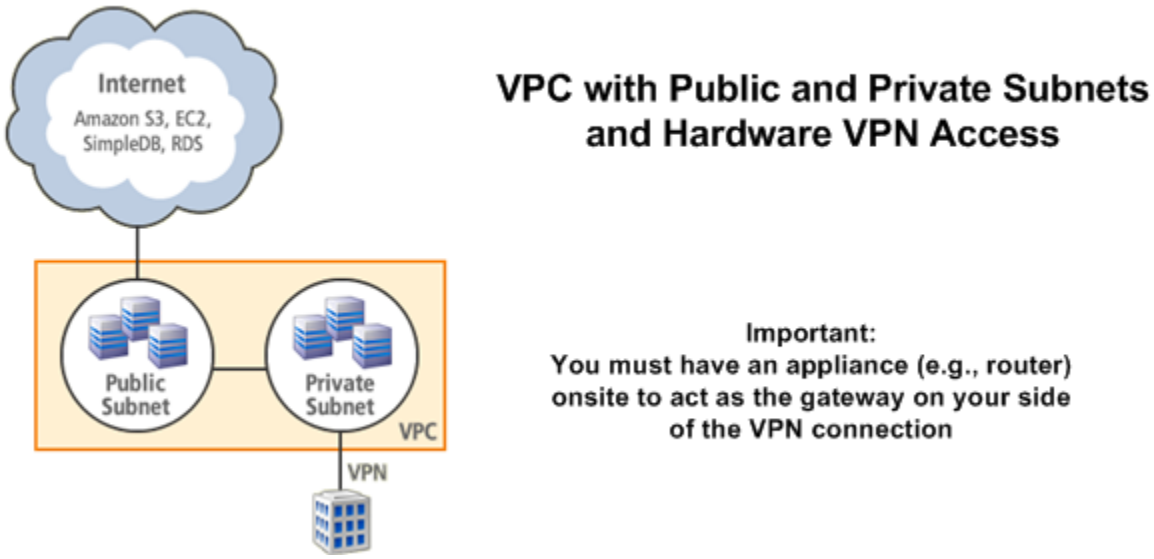
If you don't know how to allocate and associate an Elastic IP address to an instance in your VPC, see Task 4: Set Up the NAT Instance (p. 33). You need to repeat the allocation and association procedures only for the instances in the public subnet.

Congratulations! You've implemented scenario 2. You've got a VPC with two subnets containing instances that can initiate traffic to the Internet, but only one subnet's instances are reachable from the Internet.

You can now connect to your instances in the VPC. For instructions on how to connect to a Linux/UNIX instance, go to Connect to Your Linux/UNIX Instance in the *Amazon Elastic Compute Cloud Getting Started Guide*. For instructions on how to connect to a Windows instance, go to Connect to Your Windows Instance.

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

Topics



We recommend this scenario if you want to extend your data center 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 frontend in a public subnet, and to house your data in a private subnet that is connected to your data center by an IPsec VPN connection.

Important

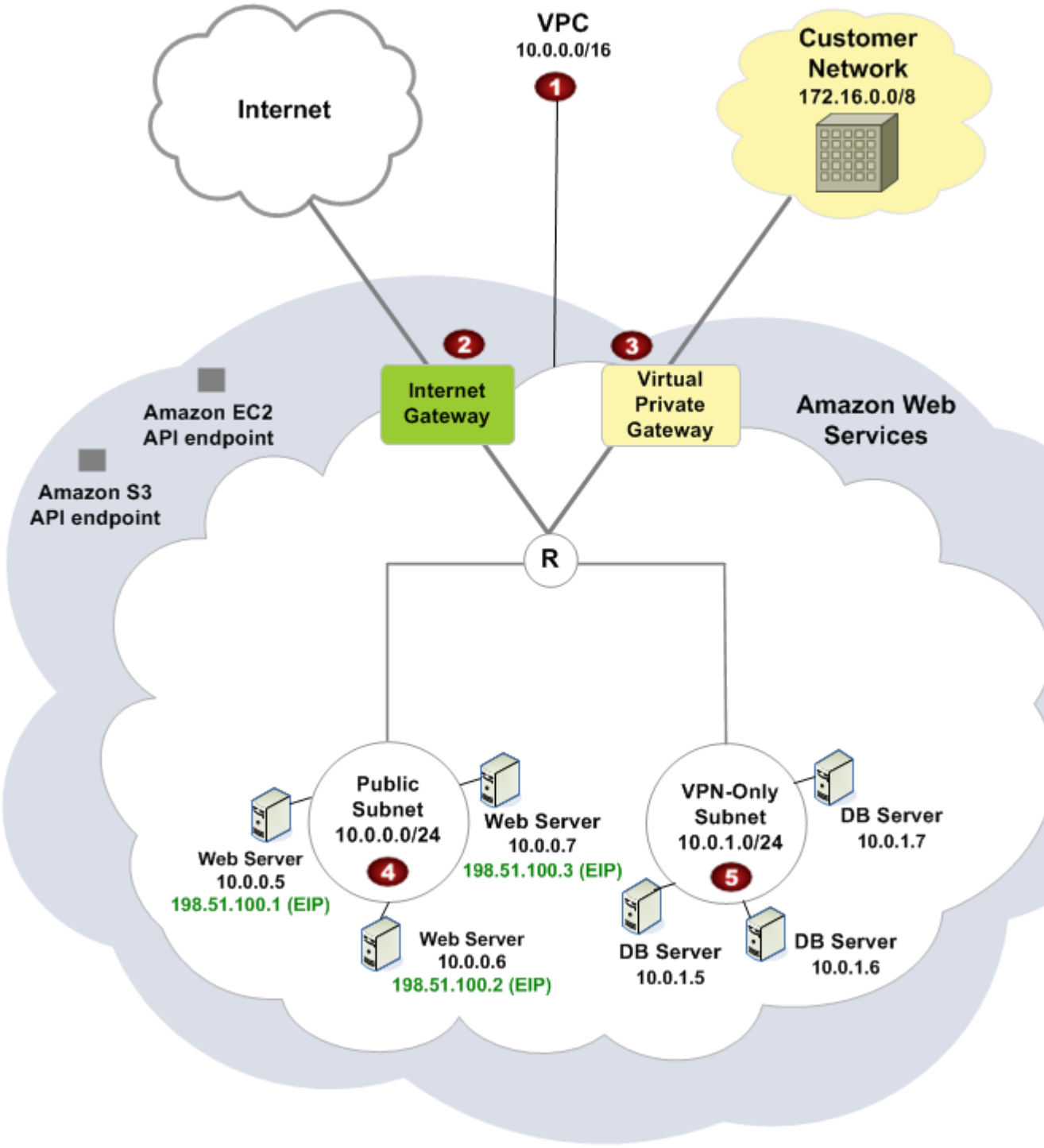
For this scenario, your network administrator needs the Amazon Virtual Private Cloud Network Administrator Guide in order to configure the customer gateway on your side of the VPN connection.

Basic Layout

The following diagram shows the basic layout of your VPC in this scenario. The big white cloud is your VPC (your isolated portion of the AWS cloud). You have an Internet gateway attached to the VPC that enables the VPC to communicate with the Internet. You also have a virtual private gateway that enables the VPC to communicate with your home network over an IPsec VPN tunnel. The circle containing an R represents your VPC's built-in routing function. The VPC has two subnets. The table following the diagram gives additional details about the VPC and its layout for this scenario.

Tip

The AWS Management Console has a wizard in the Amazon VPC console to help you implement this scenario. For more information, see Implementing the Scenario (p. 55).



1	A size /16 VPC (e.g., 10.0.0.0/16), which means 65,536 private (RFC 1918) IP addresses. For information about CIDR notation and what the "/16" means, go to the Wikipedia article about Classless Inter-Domain Routing.

2	An Internet gateway connecting the VPC to the Internet.
3	A VPN between your VPC and home network. The entire VPN setup consists of a customer gateway, virtual private gateway, VPN attachment (connecting the virtual private gateway to the VPC), and a VPN connection. For this scenario, we refer to the VPN setup generally as your virtual private gateway or VPN connection. For more information about your VPN connection, see Adding a Hardware Virtual Private Gateway to Your VPC (p. 164). To enable the VPN connection, you must have an appliance (e.g., router) in your home network that acts as the anchor on your side of the connection (for more information, go to the Amazon Virtual Private Cloud Network Administrator Guide).
4	A size /24 subnet (e.g., 10.0.1.0/24), which means 256 private IP addresses. The diagram shows the subnet containing several web servers; however, they could be any kind of instance you want. Each has a private IP address (e.g., 10.0.0.5) and an Elastic IP address (e.g., 192.0.2.1), which allows the instance to be reached from the Internet. The addresses shown in the diagram are examples; you'll probably have different values when you implement the scenario. You're going to set up routing in the VPC so that the subnet can send traffic directly to the Internet (see Routing (p. 18)). Therefore, the subnet is labeled as *public* in the diagram.
5	Another subnet, also size /24. In the diagram, the subnet contains backend services for your website (e.g., database servers). Each server has a private IP address (e.g., 10.0.1.5). Unlike the web servers in the public subnet, these servers don't need to accept incoming traffic from the Internet (and should not). You're going to set up the VPC so that the subnet can receive and send traffic only from your home network (in addition to talking to other subnets). Therefore, we refer to the subnet as *VPN-only* in the diagram.

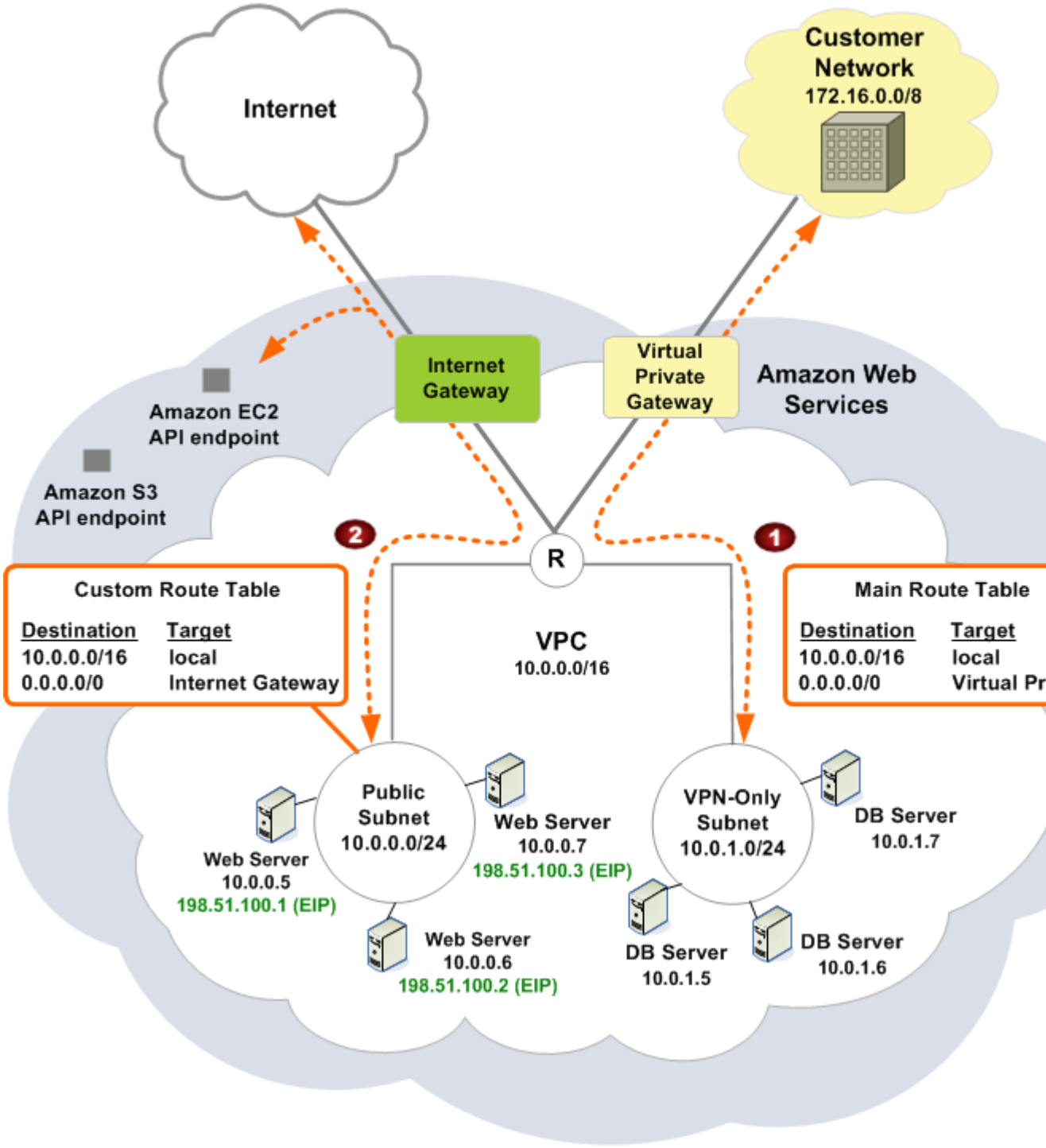
If you want an instance in the public subnet to be reachable from the Internet, that instance must have an Elastic IP address associated with it. For more information about Elastic IP addresses, see Elastic IP Addresses (p. 129).

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

Routing

Your VPC has an implied router (shown in the following diagram as an R in a circle), as well as a modifiable main route table. You can also create other route tables to use in your VPC. By default, each table has a *local route* that enables instances in your VPC to talk to each other.

The following diagram and table describe the route tables and routes you need to set up in this scenario.


Internet
Customer Network
172.16.0.0/8
Internet Gateway
Virtual Private Gateway
Amazon Web Services
Amazon EC2 API endpoint
Amazon S3 API endpoint
2
1
R
Custom Route Table
Destination
Target
10.0.0.0/16
local
0.0.0.0/0
Internet Gateway
VPC
10.0.0.0/16
Main Route Table
Destination
Target
10.0.0.0/16
local
0.0.0.0/
Virtual Pr
Public Subnet
10.0.0.0/24
VPN-Only Subnet
10.0.1.0/24
Web Server
10.0.0.5
198.51.100.1 (EIP)
Web Server
10.0.0.7
198.51.100.3 (EIP)
Web Server
10.0.0.6
198.51.100.2 (EIP)
DB Server
10.0.1.5
DB Server
10.0.1.6
DB Server
10.0.1.7

1	The VPC automatically comes with a main route table. Any subnet not explicitly associated with another route table uses the main route table. For this scenario, you update the main route table with a route that sends traffic from the VPN-only subnet to the virtual private gateway (the flow of traffic is indicated by the dotted line adjacent to the table). You don't explicitly associate the VPN-only subnet with any route table, so it implicitly uses the routes in the main route table.
2	Your VPC can have other route tables besides the main route table. For this scenario, you must create a route table (it's labeled *Custom Route Table* in the preceding diagram) with a route that sends traffic from the public subnet to the Internet gateway (the flow of traffic is indicated by the dotted line adjacent to the table). After creating the custom route table and the route, you must associate the public subnet with the table. This association is represented by the line connecting the table to the subnet in the diagram. Notice that there's no line connecting the main route table to the VPN-only subnet; the lack of line indicates an implied association with the main route table.

Note

For this scenario, any traffic from your home network going to 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 home network over the virtual private gateway to the public subnet.

The following two tables show what the route tables look like for this scenario. In each, the first row covers the local routing in the VPC (i.e., allows the instances in the VPC to communicate with each other).

Main Route Table

The first row provides local routing within the VPC. The second row sends all the subnet traffic over the virtual private gateway, which is specified by its AWS-assigned identifier (e.g., vgw-1a2b3c4d).

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

Note

If you use the wizard in the console to set up your VPC, the wizard automatically updates the main route table with the route between the VPN-only subnet and the virtual private gateway. Otherwise, you must update the main route table yourself.

Custom Route Table

The first row provides local routing within the VPC. The second row sends all traffic from the public subnet to the Internet gateway, which is specified by its AWS-assigned identifier (e.g., igw-1a2b3c4d).

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

Note

If you use the wizard in the console to set up your VPC, the wizard automatically creates the custom route table and associates the public subnet with it. Otherwise, you must do that yourself.

Any AWS-bound traffic from the public subnet (e.g., going to the Amazon EC2 or Amazon S3 API endpoints) is routed to the Internet gateway. If the traffic is bound for AWS in the same Region as the VPC, there's no bandwidth charge. Exception: You're charged Regional Data Transfer rates for data transferred between the Internet gateway attached to your VPC and Amazon EC2 instances in the same Region, regardless of Availability Zone.

Any AWS-bound traffic from the VPN-only subnet is routed to the virtual private gateway. The traffic must egress your home network to the Internet, so you're charged for both the bandwidth across the virtual private gateway, and the Internet bandwidth costs.

Security

AWS provides two ways for you to control security in your VPC: *security groups* and *network ACLs*. They both enable you to control what traffic goes in and out of your instances, but security groups work at the instance level, and network ACLs work at the subnet level. Security groups alone will suffice for many VPC users. However, some users might want 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. 136).

Important

Security groups are a basic Amazon EC2 concept. However, security groups in a VPC have different capabilities than security groups in EC2 (see EC2 vs. VPC Security Groups (p. 139)).

Recommended Security Groups

For scenario 3, you use only security groups and not network ACLs. A security group is just a group of instances that share a common set of inbound and outbound rules. To use security groups, you create a group, add the rules you want to the group, and then launch instances into the group. You can add and remove rules from the group, and those changes automatically apply to the instances in the group. You can launch an instance into more than one group, and you can change an instance's group membership after launch. For more information about security groups, see Security Groups (p. 137).

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 in the group. If you don't specify a security group when you launch an instance, the instance automatically goes into this default group. You must change the group's rules from the initial default rules if you want the instances to receive traffic from outside the group.

For this scenario, we recommend you not use the default security group and instead create the following security groups:

- **WebServerSG—**For the web servers in the public subnet
- **DBServerSG—**For the database servers in the VPN-only subnet

The following figures show each security group as a circle. A simplified light-gray VPC is in the background to help you understand how the different VPC parts are related. Each figure has a corresponding table that lists the inbound and outbound rules for the group and what they do.

Important

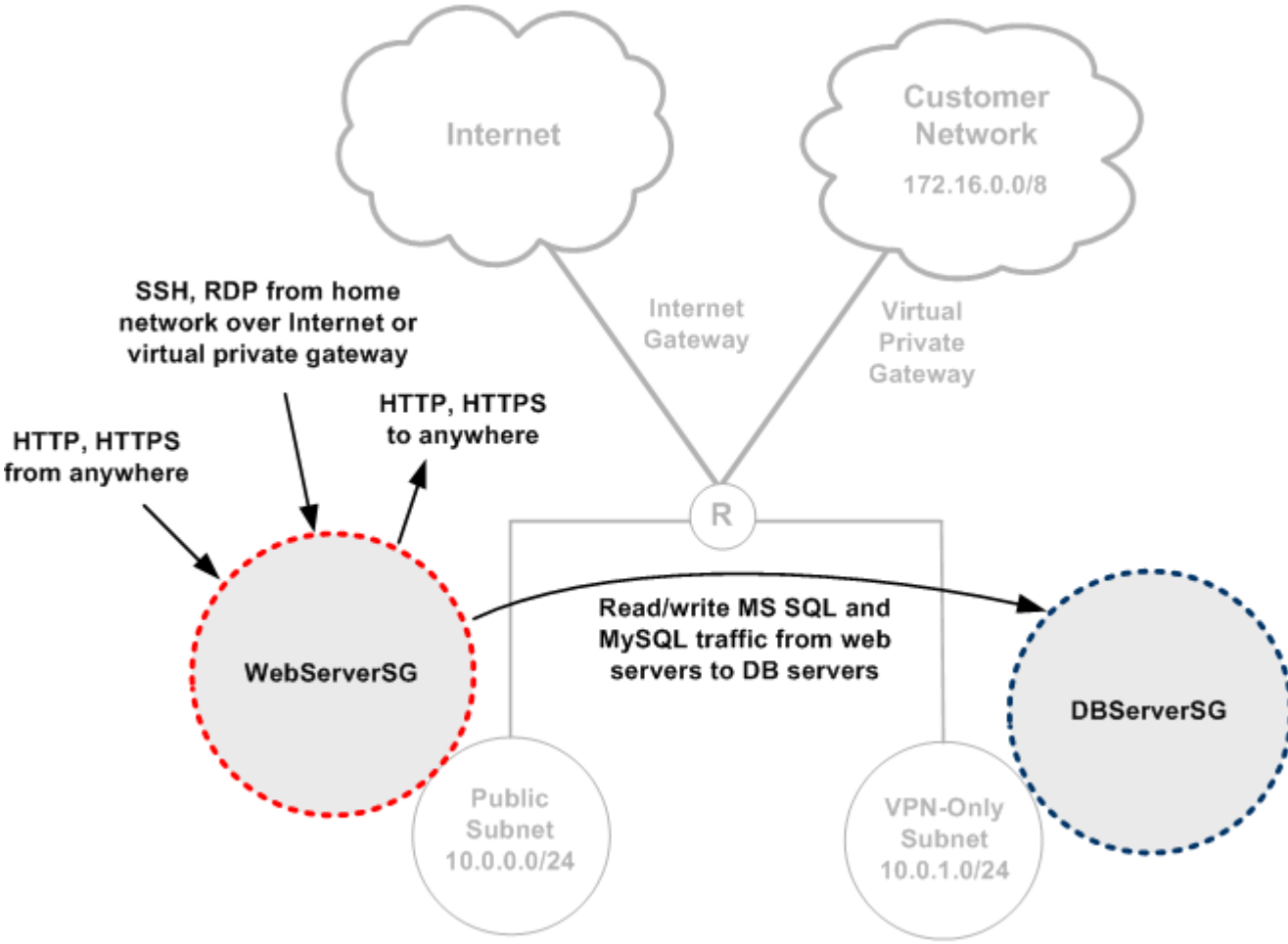
Security groups are independent of network topology. The following diagrams show security groups adjacent to subnets in the VPC. This does not indicate a relationship between the security group and the subnet. Instead, the intention is to show that one or more instances in a given subnet will be launched into the adjacent security group. For example, instances in the public subnet will be launched into the WebServerSG group, so the public subnet is shown adjacent to that group.

The instances in a given security group do not have to be in the same subnet. 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, all instances in a given security group in this scenario are in the same subnet.

Let's start with the WebServerSG security group, which you launch your web servers into. Based on the rules in the following table, the web servers can receive Internet traffic, as well as SSH and RDP traffic from your home network. The instances can also initiate traffic to the Internet and read and write data to the database server instances in the private subnet.

Note

Security groups use *stateful filtering*. That is, all response traffic is automatically allowed. For example, if a client on the Internet sends a request to a web server in the WebServerSG, the instance can respond, regardless of any outbound rules on the group. Likewise, if the web server initiates traffic bound to a server on the Internet, the response is allowed back in to the instance, regardless of any inbound rules on the group.



WebServerSG

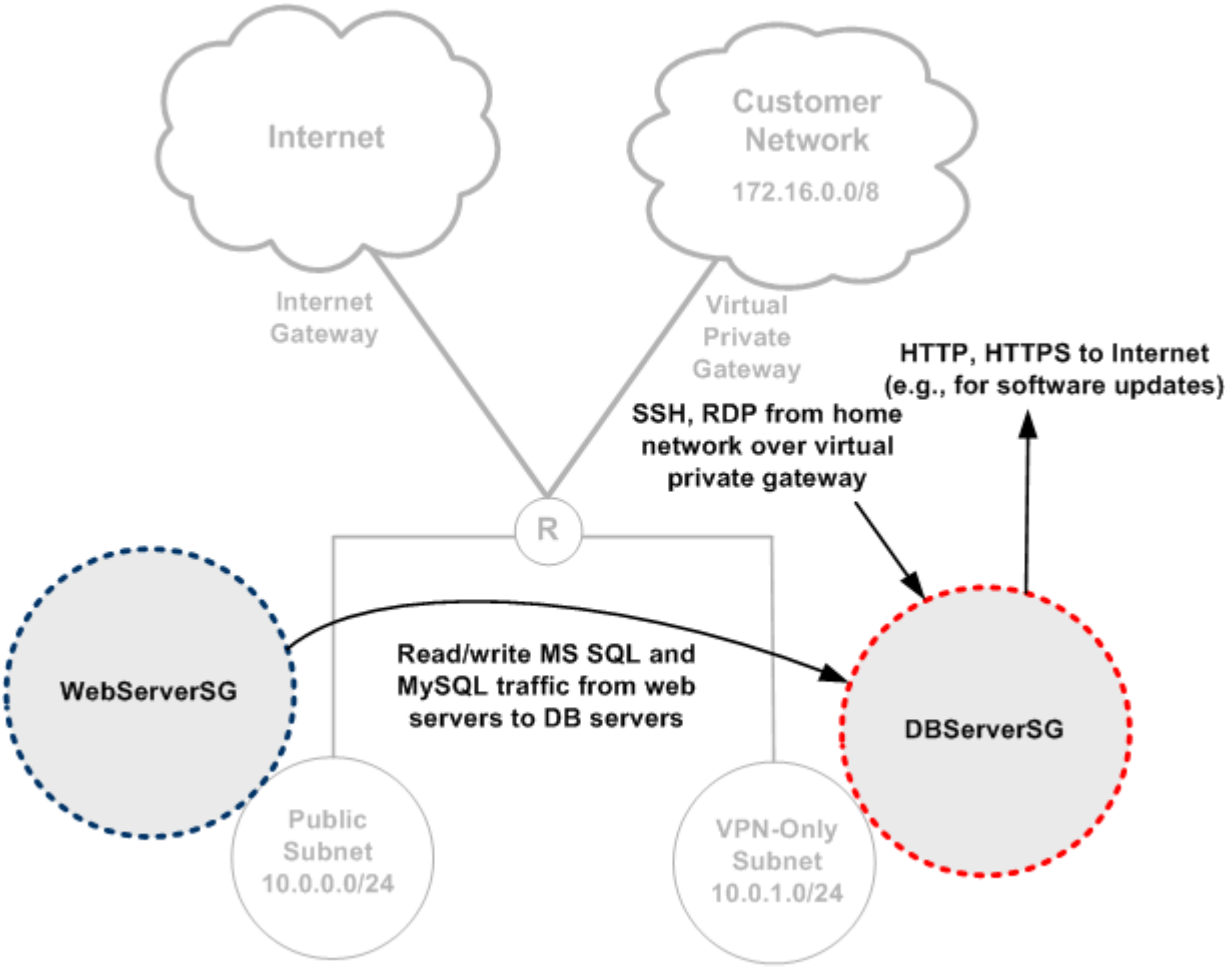
Inbound			
Source	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anyone
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anyone
Your home network's public IP address range	TCP	22	Allow inbound SSH access to Linux/UNIX instances from your home network (over the Internet gateway)
Your home network's public IP address range	TCP	3389	Allow inbound RDP access to Windows instances from your home network (over the Internet gateway)
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**

0.0.0.0/0	TCP	80	Allow web servers to initiate outbound HTTP access to the Internet (e.g., for software updates)
0.0.0.0/0	TCP	443	Allow web servers to initiate outbound HTTPS access to the Internet (e.g., for software updates)
DBServerSG	TCP	1433	Allow outbound Microsoft SQL Server access to the database servers in DBServerSG
DBServerSG	TCP	3306	Allow outbound MySQL access to the database servers in DBServerSG

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/UNIX (SSH and MySQL) or Windows (RDP and MS SQL).

Next is the DBServerSG security group, which you launch your database servers into. Based on the rules in the following table, the database servers allow Microsoft SQL Server and MySQL read or write requests from the web servers. They allow in SSH and RDP traffic from your home network. They can also initiate traffic bound for the Internet over the virtual private gateway.



DBServerSG

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
WebServerSG	TCP	1433	Allow servers in the WebServerSG to read and write over MS SQL port 1433 to instances in DBServerSG group
WebServerSG	TCP	3306	Allow servers in the WebServerSG to read and write over MySQL port 3306 to instances in DBServerSG group
10.0.0.0/8	TCP	22	Allow inbound SSH traffic to Linux/UNIX instances from home network (over the virtual private gateway)
10.0.0.0/8	TCP	3389	Allow inbound RDP traffic to Windows instances from home network (over the VPN gateway)

Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow outbound HTTP access to the Internet (e.g., for software updates) over the virtual private gateway
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (e.g., for software updates) over the virtual private gateway

Even though some instances are in the same security group (e.g, the web servers are together in the WebServerSG), they can't automatically talk to each other. By default, security groups don't contain rules that allow instances in the group to communicate with each other. Exception: the VPC's default security group has such rules. If you want to allow that type of communication, you must add a rule like the one in the following example for the WebServerSG group.

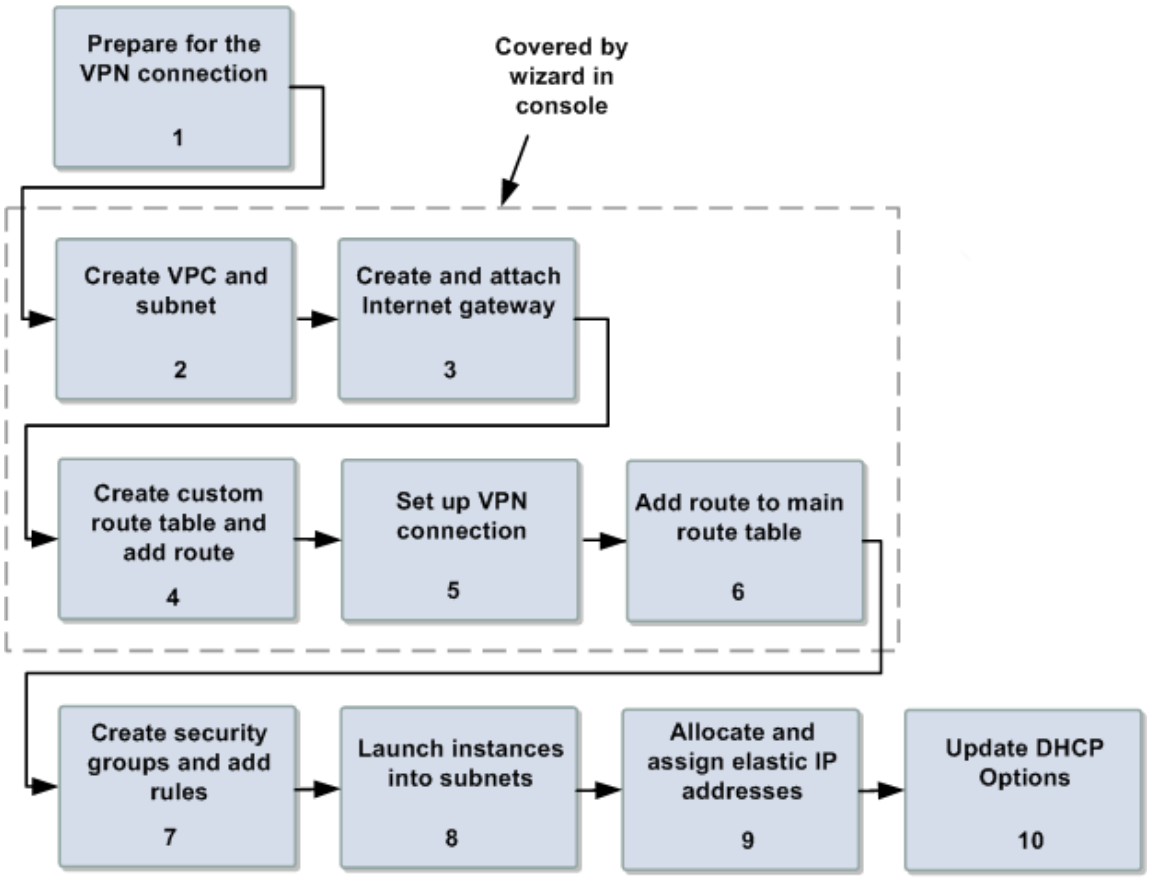
Inbound			
Source	**Protocol**	**Port Range**	**Comments**
WebServerSG	All	All	Allow inbound traffic from WebServerSG
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
WebServerSG	All	All	Allow outbound traffic from WebServerSG

Implementing the Scenario

This section walks you through the process of implementing scenario 3. The following figure and table show the tasks required to implement the scenario.

Tip

Several of the tasks are automatically handled for you if you use the wizard in the AWS Management Console. The following sections describe how to use the wizard and how to do all the tasks manually.



Process for Implementing Scenario 3

Task 1: Prepare for the VPN Connection (p. 57)
Task 2: Create the VPC and Subnets (p. 60)
Task 3: Create and Attach the Internet Gateway (p. 61)
Task 4: Create a Custom Route Table and Add Rules (p. 61)
Task 5: Set Up the VPN Connection (p. 62)
Task 6: Add a Route to the Main Route Table (p. 63)
Task 7: Create Security Groups and Add Rules (p. 64)
Task 8: Launch Instances into the Subnets (p. 68)
Task 9: Allocate and Assign Elastic IP Addresses (p. 72)
Task 10: Update DHCP Options (p. 72)

Task 1: Prepare for the VPN Connection

In scenario 3, you set up a VPN connection between your home network and your VPC. The connection requires an appliance onsite (e.g., router) to act as your customer gateway. You need help from a network administrator in your organization to:

- Determine the appliance that will be your customer gateway
- Provide you 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))

For more information about the requirements for your customer gateway, go to the Amazon Virtual Private Cloud Network Administrator Guide.

If you want to use the wizard to set up your VPC, see Use the Wizard for Scenario 3 (p. 57). Otherwise, see Task 2: Create the VPC and Subnets (p. 60) to perform the process manually.

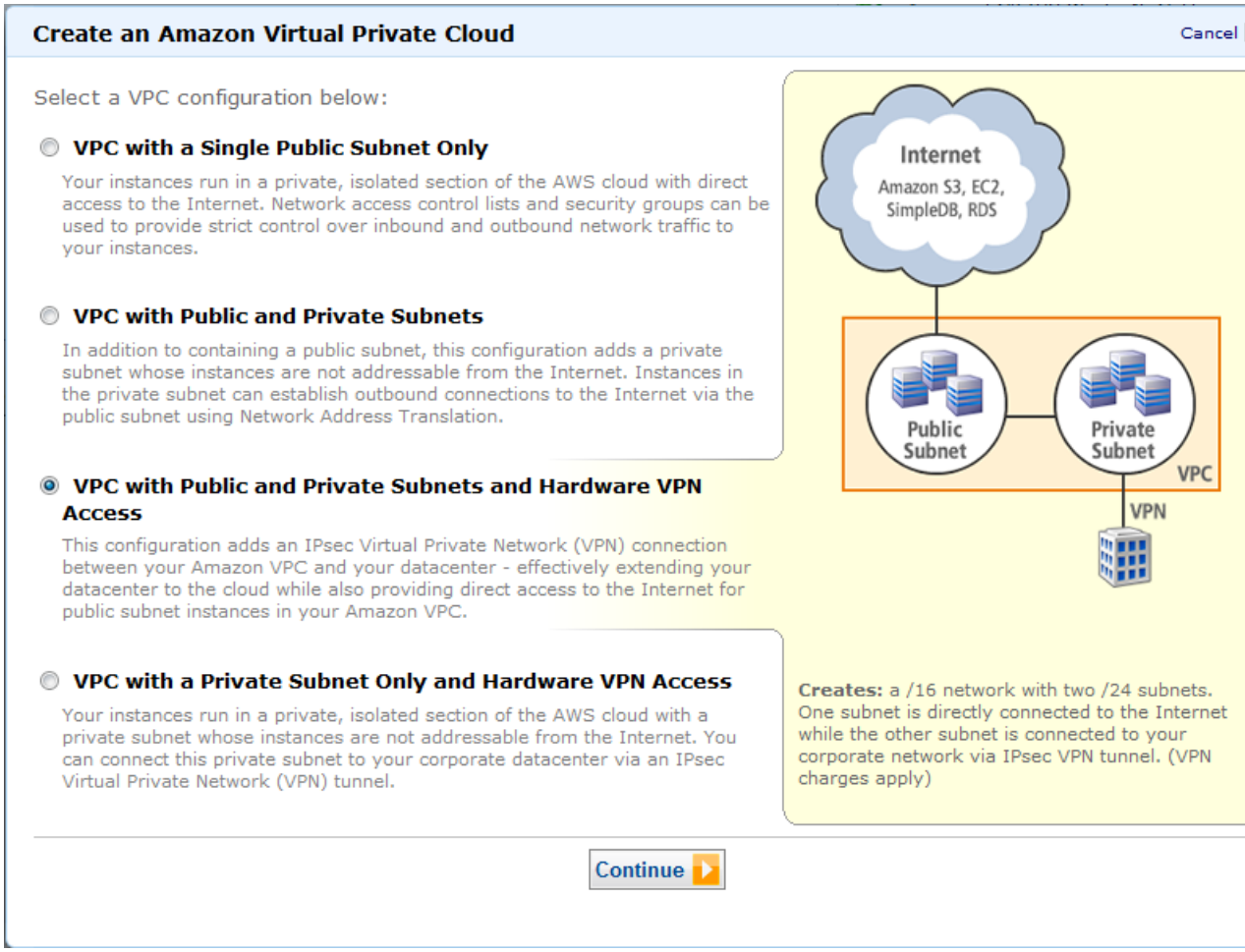
Use the Wizard for Scenario 3

You can have Amazon VPC complete tasks 2-6 for you by using the wizard in the AWS Management Console. This procedure assumes you don't already have a VPC, and that you have the IP address for your customer gateway (see the preceding section).

To use the wizard

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **VPC Dashboard**.
3. On the **VPC Dashboard**, locate the **Your Virtual Private Cloud** area and click **Get started creating a VPC**.

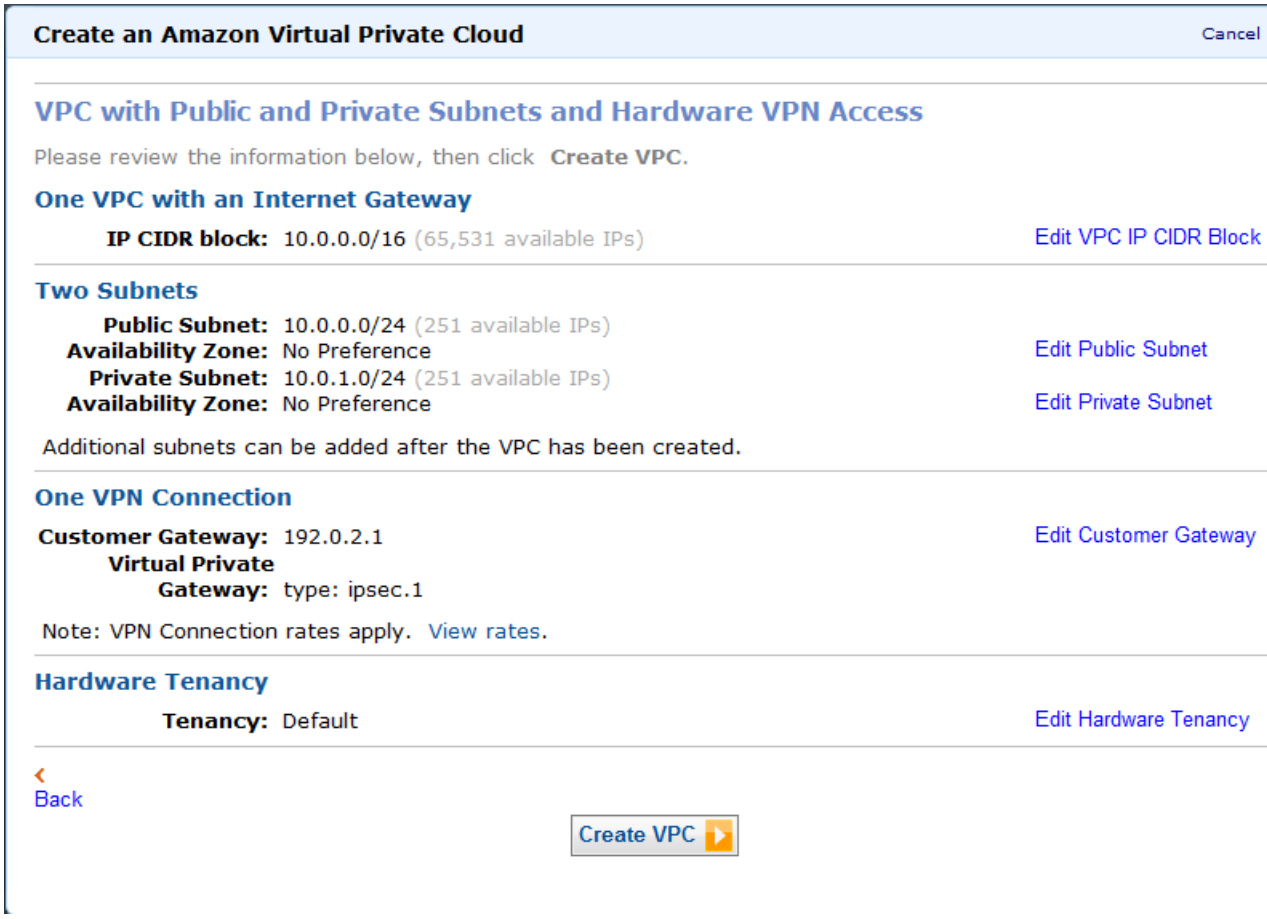
   The wizard opens and displays a page where you can select one of four options.
4. Select the radio button for **VPC with Public and Private Subnets and Hardware VPN Access** and click **Continue**.



A dialog box opens with a field for your customer gateway's IP address.

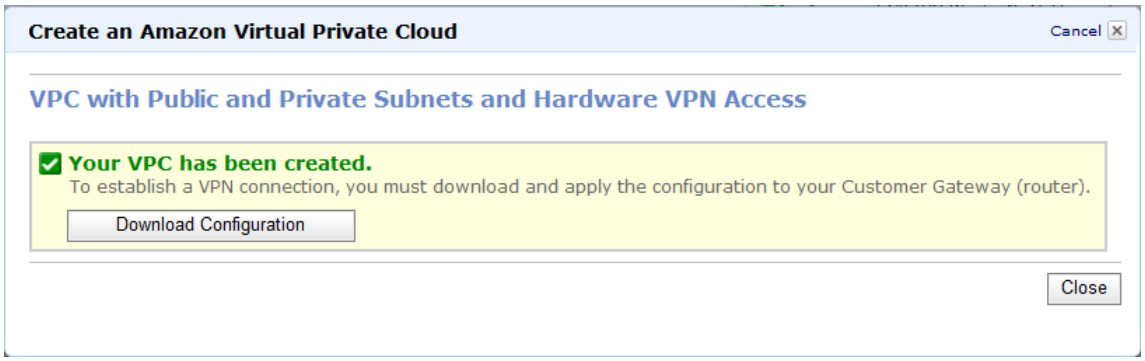
5. Enter your customer gateway's IP address and click **Continue**.

A confirmation page is displayed showing the CIDR blocks we use for the VPC and subnets. It also shows the IP address that you just provided for the customer gateway, as well as the instance hardware tenancy of the VPC. You can change any of these values if you want.



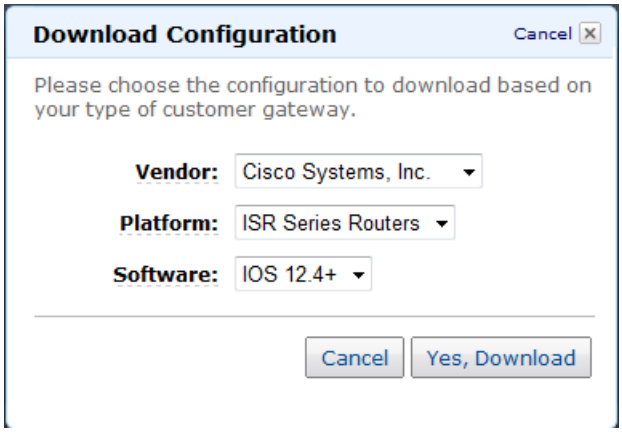
6. Make any changes you want and click **Continue**.
   The wizard begins to create your VPC, subnets, Internet Gateway, and VPN connection. It also updates the main route table, creates a custom route table, and adds routes.

   When the wizard is done, a confirmation dialog box is displayed with a button for downloading the configuration for your customer gateway.



7. Click **Download Configuration**.

8. In the **Download Configuration** dialog box, select the customer gateway's vendor, platform, and software version, and then click **Yes, Download**.



9. Save the text file containing the 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.

After the wizard completes, you're partway done. The next task is to create the recommended security groups. For more information, see Task 7: Create Security Groups and Add Rules (p. 64)).

Note that the next few sections show how to manually do tasks that the wizard already completed for you.

Task 2: Create the VPC and Subnets

If you don't use the wizard in the console, you can manually create the VPC and subnets yourself. This section shows you how.

To create your VPC and subnets

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, enter the CIDR range you want for your VPC (e.g., 10.0.0.0/16), and then click **Yes, Create**.

   Tip

   For information about choosing the CIDR range for your VPC, see VPC Sizing (p. 105).

   The VPC is created and appears on the **Your VPCs** page. Notice that it has an ID (e.g., vpc-xxxxxxxx).
4. In the **Navigation** pane, click **Subnets**.
5. At the top of the **Subnets** page, click **Create Subnet**.
6. In the **Create Subnet** dialog box, select the VPC and Availability Zone, enter the CIDR range you want for your subnet (e.g., 10.0.0.0/24), and then click **Yes, Create**.
   The subnet is created and appears on the **Subnets** page. Notice that it has an ID (e.g., subnet-xxxxxxxx). The page also shows the number of available IP addresses in the subnet, the route table associated with the subnet, and the network ACL associated with the subnet. The subnet uses the main route table and default network ACL by default.
7. Create a second subnet (e.g., 10.0.1.0/24) by repeating the preceding steps for creating a subnet.

You've got your VPC and subnets now. Move on to the next section to create and attach an Internet gateway to the VPC.

Task 3: Create and Attach the Internet Gateway

If you don't use the wizard in the console, you can manually create and attach the Internet gateway yourself. This section shows you how.

To create the Internet gateway

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In in the **Navigation** pane, click **Internet Gateways**, and then click **Create Internet Gateway**.
3. In the **Create Internet Gateway** dialog box, click **Yes, Create**.
   The Internet gateway is created and appears on the page. Notice that it has an ID (e.g., igw-xxxxxxxx).
4. Select the Internet gateway and click **Attach to VPC**.
5. In the **Attach to VPC** dialog box, click **Yes, Attach**.

Your VPC has an Internet gateway attached to it now. However, no route table refers to the gateway yet, so no traffic can flow to the gateway. Move on to the next section to set up routing for the public subnet.

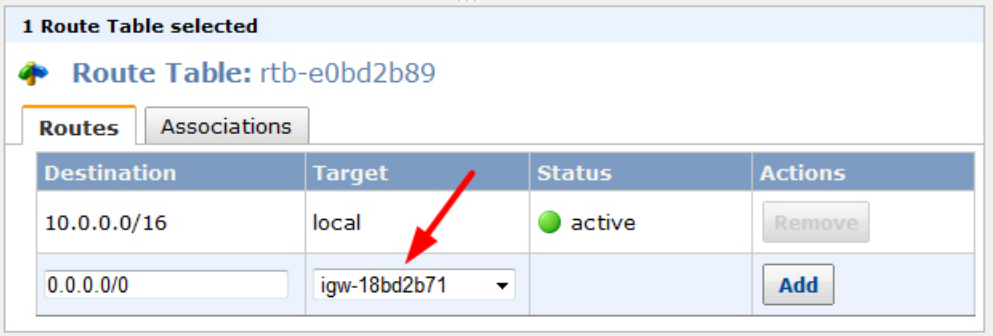
Task 4: Create a Custom Route Table and Add Rules

If you don't use the wizard in the console, you can manually create the required custom route table and add routes yourself. This section shows you how.

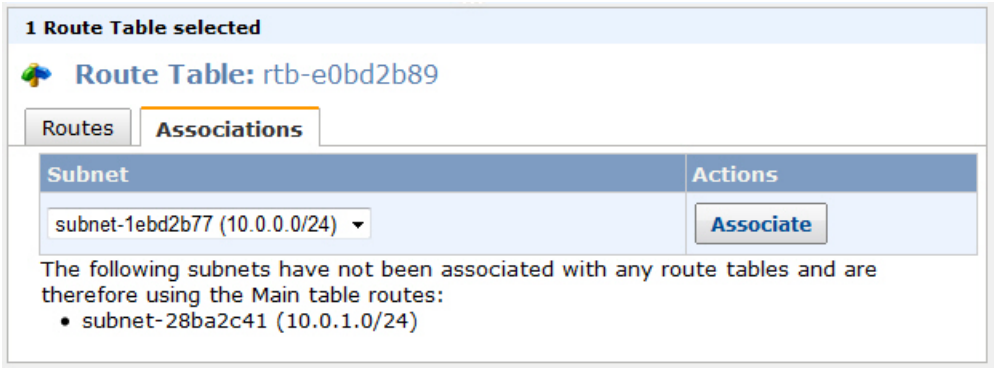
For this scenario, you create a custom route table with a route to send all the non-local traffic (i.e., 0.0.0.0/0) in the public subnet to the Internet gateway, and you associate the public subnet with the table.

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**.
   Your VPC's route tables are listed.
3. Click **Create Route Table**.
4. In the **Create Route Table** dialog box, make sure your VPC is selected and click **Yes, Create**.
   The new route table is created and appears on the page. Notice that it has an ID (e.g., rtb-xxxxxxxx).
5. Select the check box for the custom route table.
   The lower pane displays the route table's details.
6. On the **Routes** tab, enter `0.0.0.0/0` in the **Destination** field, select the ID for the Internet gateway in the **Target** drop-down list, and click **Add**.



7. On the **Associations** tab, select the ID of the public subnet and click **Associate**.



   The public subnet is now associated with the custom route table.

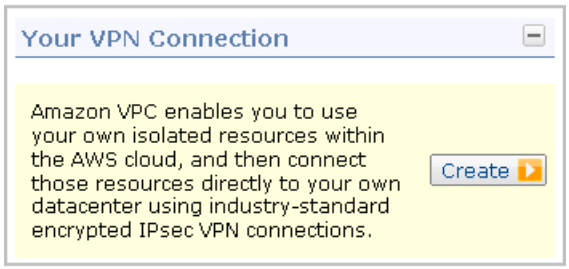
The VPC now has a custom route table associated with the public subnet. The table enables traffic to flow between the subnet and the Internet gateway. Move on to the next section to set up the VPN connection for your VPC.

Task 5: Set Up the VPN Connection

If you don't use the wizard in the console, you can manually set up the VPN connection yourself. This section shows you how. You must have already prepared for the VPN connection (see Task 1: Prepare for the VPN Connection (p. 57)).

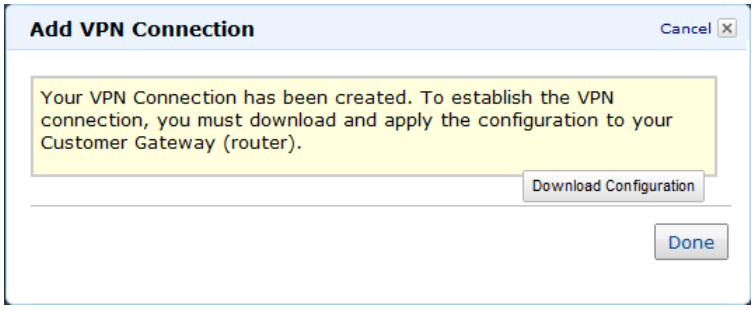
To set up the VPN connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **VPC Dashboard**.
3. In the **Your VPN Connections** area of the page, click **Create** (if this is your first VPN) or click **Add VPN Connection**.

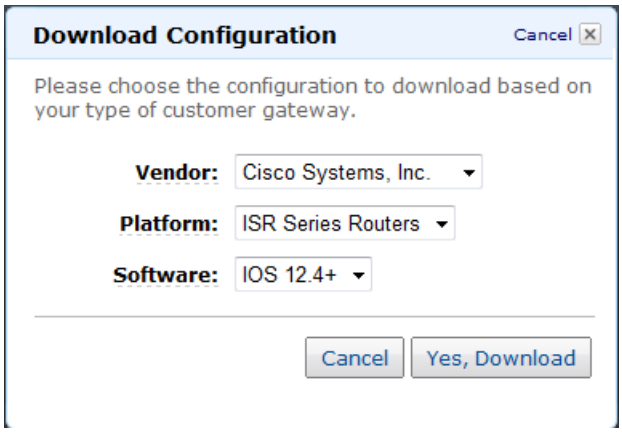


4. In the **Add VPN Connection** dialog box, enter the IP address for your customer gateway (e.g., 198.0.2.1), and then click **Yes, Create**.

   We create your customer gateway and your virtual private gateway, attach the virtual private gateway to the VPC, and create a VPN connection. When the wizard is done, a confirmation dialog box is displayed with a button for downloading the configuration for your customer gateway.



5. Click **Download Configuration**.
6. In the **Download Configuration** dialog box, select the customer gateway's vendor, platform, and software version, and then click **Yes, Download**.



7. Save the text file contain and give it to the network administrator along with this guide: Amazon Virtual Private Cloud Network Administrator Guide.

You now have a customer gateway, a virtual private gateway attached to your VPC, and a VPN connection. However, the VPN won't work until your network administrator configures your customer gateway. Also, no route table refers to the gateway yet, so no traffic can flow to the gateway. Move on to the next section to set up routing for the VPN-only subnet.

Task 6: Add a Route to the Main Route Table

If you don't use the wizard in the console, you can manually add the required route to the main route table yourself. This section shows you how.

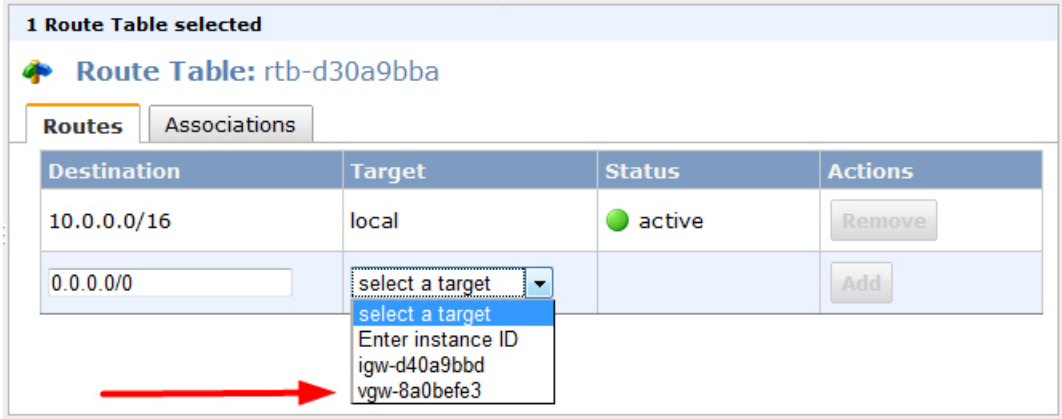
For this scenario, you add a route that sends the traffic from the VPN-only subnet to the virtual private gateway. You don't need to associate the route table with the subnet, because it's the main route table (which is automatically associated with any subnet that isn't explicitly associated with a subnet).

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**.

   Your VPC's route tables are listed.
3. In the list of route tables, select the check box for the main route table.
   The lower pane displays the route table's details.

4. On the **Routes** tab, enter `0.0.0.0/0` in the **Destination** field, and select the virtual private gateway's ID in the **Target** drop-down list, and click **Add**.



The VPC's main route table now includes the new route. The route enables traffic to flow between the VPN-only subnet and the virtual private gateway. If you click the **Associations** tab (next to the **Routes** tab for the main route table), you can see which subnets use the main route table. Your VPN-only subnet is listed there because you haven't explicitly associated it to any route table.

Move on to the next section to create the security groups you need for this scenario.

Task 7: Create Security Groups and Add Rules

If you don't use the wizard in the console, you can manually create the security groups yourself and add the rules to them. This section shows you how.

You first create both groups and then add the rules to each. For details about the groups and their rules for this scenario, see Security (p. 50).

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**.
   Your VPC's security groups are listed.
3. Click **Create Security Group**.
4. In the **Create Security Group** dialog box, enter the name for your security group (e.g., WebServerSG), enter a description of the group, select your VPC's ID from the **VPC** menu, and click **Yes, Create**. The security group is created and appears on the **Security Groups** page. Notice that it has an ID (e.g., sg-xxxxxxxx). You might have to turn on the **Group ID** column by clicking **Show/Hide** in the top right corner of the page.

   Note

   This page shows all security groups that belong to your AWS account, including your VPC groups and your EC2 groups. The VPC groups have a value listed in the **VPC ID** column. For information about the different kinds of security groups, see Security Groups (p. 137).
5. Repeat the preceding steps for the other group you need (DBServerSG).

Now that you've created the security groups, you can add rules to them. For a list of the rules to add, see Security (p. 50).

To add rules to the WebServerSG security group

1. In the list of security groups, select the check box for the WebServerSG group you just created. The lower pane displays the security group's details.
2. Add rules for inbound HTTP and HTTPS access to the group from anywhere:

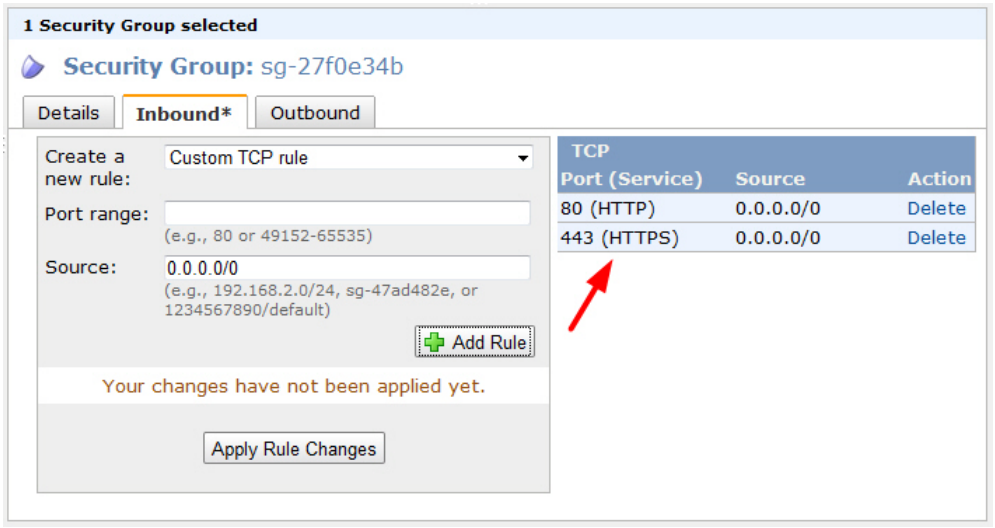
   a. On the **Inbound** tab, select `HTTP` from the **Create a new rule** drop-down list.

   b. Make sure the **Source** field's value is `0.0.0.0/0` and click **Add Rule**.

      The rule to allow HTTP access from anywhere (i.e., 0.0.0.0/0) is added to the **Inbound** tab. Notice that the rule on the right is highlighted in blue, and an asterisk appears on the tab. This indicates that you still need to click **Apply Rule Changes** (which you'll do after you've added all the inbound rules).

   c. Select `HTTPS` from the **Create a new rule** drop-down list and click **Add Rule**.

      The rule to allow HTTPS access from anywhere (i.e., 0.0.0.0/0) is added to the **Inbound** tab.



3. Add rules for inbound SSH and Remote Desktop (RDP) access to the group from your home network's public IP address range:

   a. On the **Inbound** tab, select `SSH` from the **Create a new rule** drop-down list.

   b. In the **Source** field, enter your home network's public IP address range (this example uses 192.0.2.0/24).

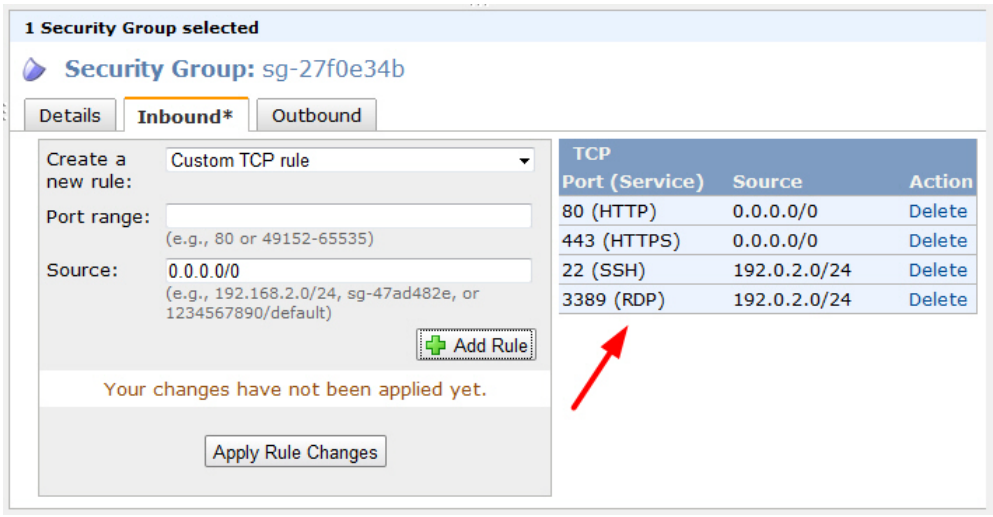
   c. Click **Add Rule**.
      The rule is added to the **Inbound** tab.

   d. Select `RDP` from the **Create a new rule** drop-down list.

   e. In the **Source** field, enter your home network's public IP range.

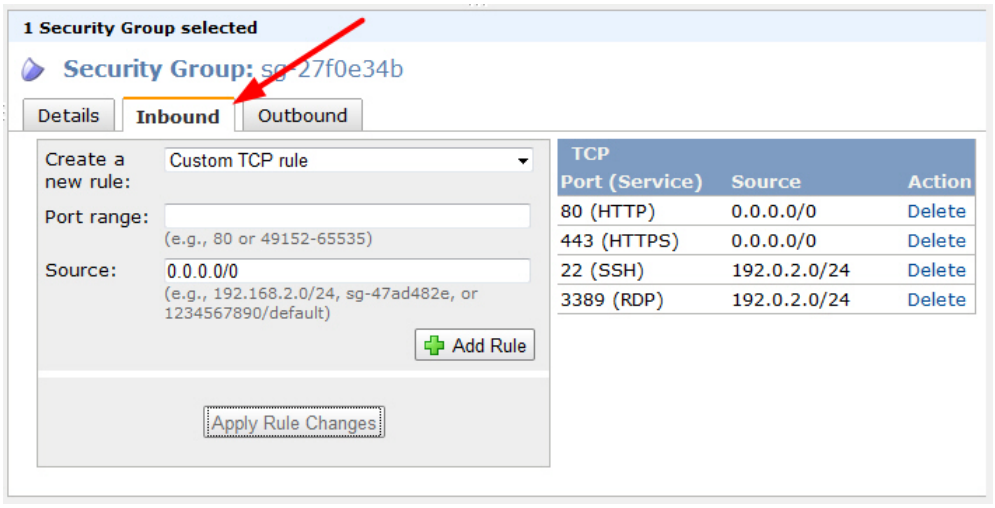
   f. Click **Add Rule**.

      The rule is added to the **Inbound** tab.



4. Click **Apply Rule Changes**.

   The new inbound rules on the right side of the screen are no longer highlighted in blue, and the asterisk no longer appears on the tab. Those changes indicate that the new inbound rules have been applied.



5. Add the outbound rules to limit egress traffic from the instances:

   a. On the **Outbound** tab, locate the default rule that enables all outbound traffic, and click **Delete**.

The rule is marked for deletion, and an asterisk appears on the tab. The deletion will not take effect until you click **Apply Rule Changes**, which you'll do after adding new outbound rules to the group.

b. On the **Outbound** tab, select `HTTP` from the **Create a new rule** drop-down list.

c. Make sure the **Destination** field's value is `0.0.0.0/0` and click **Add Rule**.
The rule is added to the **Outbound** tab.

d. Select `HTTPS` from the **Create a new rule** drop-down list.

e. Make sure the **Destination** field's value is `0.0.0.0/0` and click **Add Rule**.
The rule is added to the **Outbound** tab.

f. Select `MS SQL` (for Microsoft SQL) from the **Create a new rule** drop-down list.

g. In the **Source** field, start typing `sg-`.
The drop-down list displays the IDs for your security groups (e.g., sg-xxxxxxxx).

h. Select the ID for the DBServerSG group and click **Add Rule**.
The rule is added to the **Outbound** tab.

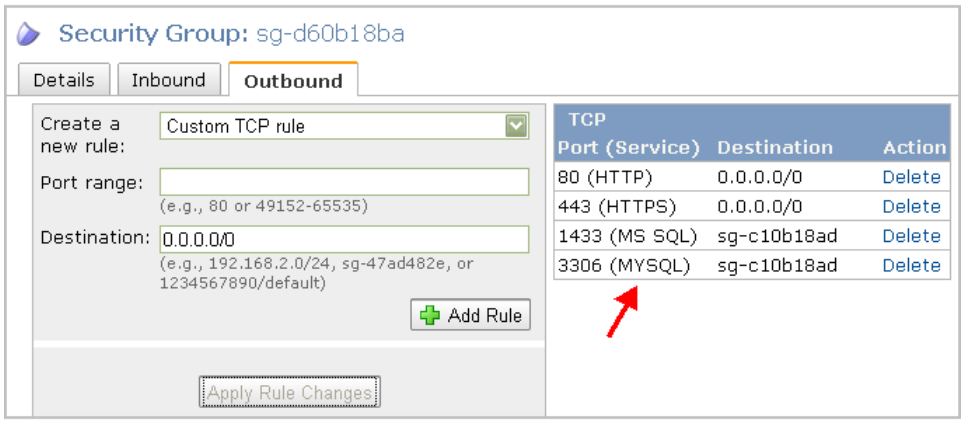
i. Select `MySQL` from the **Create a new rule** drop-down list.

j. In the **Source** field, start typing `sg-`.
The drop-down list displays the IDs for your security groups (e.g., sg-xxxxxxxx).

k. Select the ID for the DBServerSG group and click **Add Rule**.
The rule is added to the **Outbound** tab.

6. Click **Apply Rule Changes**.

The new outbound rules now apply to the security group.



The VPC now includes a security group for the web servers in your subnet. The group allows HTTP/HTTPS access in and out of the group to and from anywhere. The group also allows inbound SSH and RDP access from your home network's IP range. It also allows Microsoft SQL and MySQL access to the DBServerSG group.

Now that you know how to create security groups and add rules to them, you can add rules to the DBServerSG. The following image shows what the rules look like for the DBServerSG.

DBServerSG: Inbound

TCP		
Port (Service)	Source	Action
1433 (MS SQL)	sg-d60b18ba	Delete
3306 (MYSQL)	sg-d60b18ba	Delete
22 (SSH)	10.0.0.0/8	Delete
3389 (RDP)	10.0.0.0/8	Delete

DBServerSG: Outbound

TCP		
Port (Service)	Destination	Action
80 (HTTP)	0.0.0.0/0	Delete
443 (HTTPS)	0.0.0.0/0	Delete

Move on to the next section to launch instances in your subnets.

Task 8: Launch Instances into the Subnets

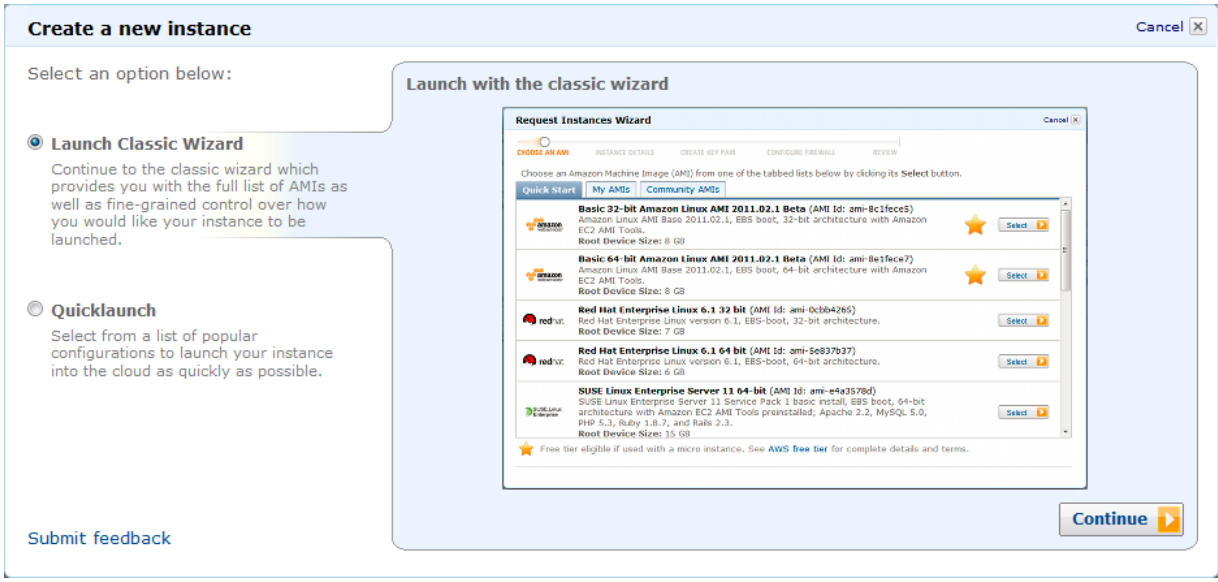
After your network administrator configures your customer gateway, you can launch instances into your VPC. If you haven't launched instances before, use the following procedure. If you're already familiar with launching Amazon EC2 instances outside a VPC, then you already know most of what you need to know. The additional items to know:

- You must specify the VPC and subnet you want to launch the instances in.
- You must specify the VPC security group you want the instance to be in (e.g., WebServerSG, etc.).

To launch an instance

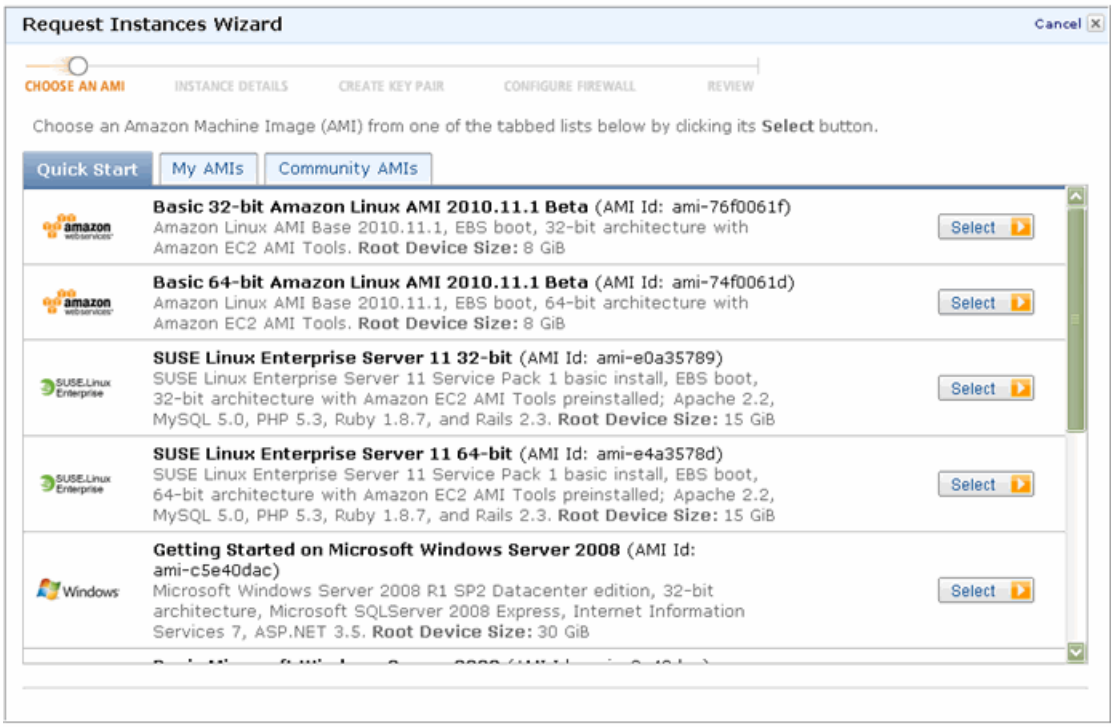
1. Start the launch wizard:
   a. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
   b. Click **Launch Instance** to start the Request Instances Wizard.

c. On the **Create a New Instance** screen, select **Classic Wizard**, and then click **Continue**.



The first page of the wizard displays tabs that list different types of AMIs.

2. Select an AMI from one of the tabs. If you don't have a particular AMI you want to launch, select either the *Basic 32-bit Amazon Linux AMI*, or the *Getting Started on Microsoft Windows Server 2008* AMI on the **Quick Start** tab.



After you select an AMI, the wizard steps to the **Instance Details** page. This is where you control settings such as the number and size of instances to launch, and which subnet to launch the instance in.

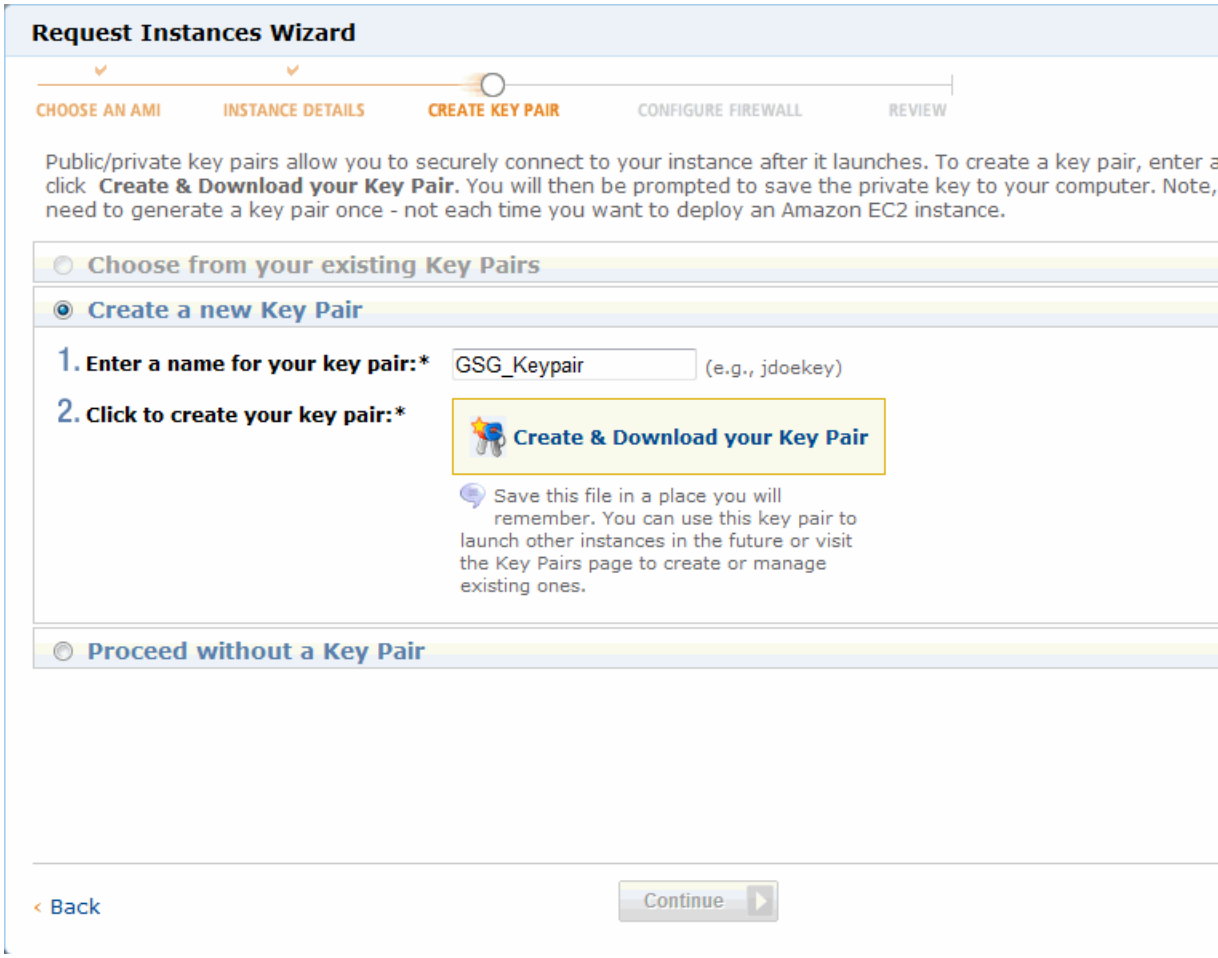
3. Select the **Launch Instances Into Your Virtual Private Cloud** option, and select the subnet you want to launch the instance in. Keep the other default settings on this page and click **Continue**. The wizard steps to the next page for instance details.
4. The default settings on this page of the wizard and the next page are what we want, so just click **Continue** on each page.
   The **Create Key Pair** page appears.

   A *key pair* is a security credential similar to a password, which you use to securely connect to your instance once it's running. If you're new to Amazon EC2 and haven't created any key pairs yet, when the wizard displays the **Create Key Pair** page, the **Create a new Key Pair** button is selected by default. We assume you'll want a new key pair.
5. Create a key pair:

   Tip

   If you're already familiar with Amazon EC2 and have an SSH key pair already, you don't need to create a new one now. You can just select one of your existing key pairs instead.

   a. On the **Create Key Pair** page, enter a name for your key pair (e.g., GSG_Keypair). This is the name of the private key file associated with the pair (with a `.pem` extension).



b. Click **Create & Download your Key Pair**.
You're prompted to save the private key from the key pair to your system.

c. Save the private key in a safe place on your system. Note the location because you'll need to use the key soon to connect to the instance.

6. On the **Configure Firewall** page of the wizard, select the security group you want to use for the instance (e.g., WebServerSG or DBServerSG), and click **Continue**.

   After you configure the firewall, the wizard steps to the **Review** page where you can review the settings and launch the instance.

7. Review your settings and launch the instance:

   a. Click **Launch**.
   A confirmation page is displayed to let you know your instance is launching.

   b. Click **Close** to close the confirmation page, and then click **Instances** in the navigation pane to view your instance's status. It takes a short time for an instance to launch. The instance's status is *pending* while it's launching. After a short period, your instance's status switches to *running*. You can click **Refresh** to refresh the display.

For the instances running in the VPN-only subnet, you can test their connectivity by pinging them from your home network. For more information, see How to Test the End-to-End Connectivity of Your Instance (p. 171).

You now have instances running in your VPC. Move on to the next section to associate Elastic IP addresses with web servers in the public subnet.

Task 9: Allocate and Assign Elastic IP Addresses

You should have at least one instance running in each of your subnets. Now you can allocate and assign Elastic IP addresses to instances in the public subnet.

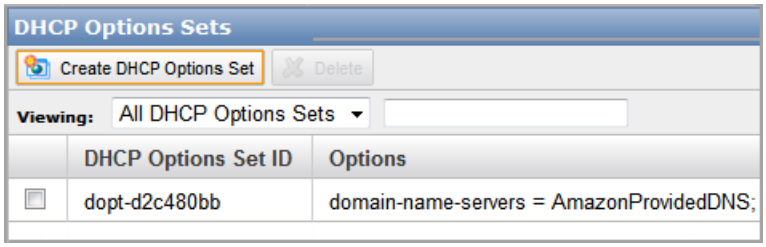
To allocate and assign an elastic IP address to an instance

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Elastic IPs**, and then click **Allocate New Address**.
3. In the **Allocate New Address** dialog box, in the **EIP used in:** drop-down list, select VPC and click **Yes, Allocate**.
   The new address is allocated and appears on the page.
4. Right-click the IP address in the list and select **Associate**.
5. In the **Associate Address** dialog box, select the instance you want to associate the address with and click **Yes, Associate**.
   The address is associated with the instance. Notice that the instance ID is displayed next to the IP address in the list.

Your instance now has an Elastic IP address associated with it. The instance is now accessible from the Internet. You can also access it using SSH or Remote Desktop from your home network over the Internet. Make sure to use the instance's elastic IP address and not its private IP address when you connect with SSH or RDP.

Task 10: Update DHCP Options

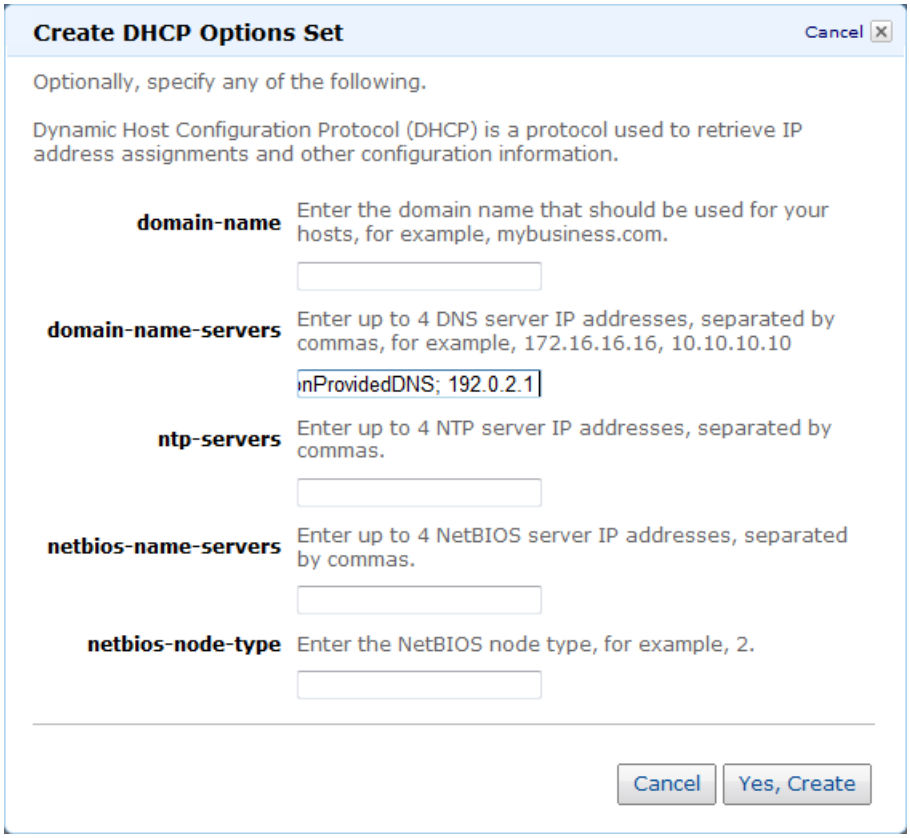
In scenario 3, you need a DNS server that enables your public subnet to communicate with servers on the Internet, and you need another DNS server that enables your VPN-only subnet to communicate with servers in your home network. Amazon provides the first DNS server (AmazonProvidedDNS). In order for your VPC to use that DNS server, your VPC must use a set of DHCP options that includes the option `domain-name-servers=AmazonProvidedDNS`. Your VPC automatically has a set of DHCP options with only that option (see the following image). For more information about DHCP options, see Using DHCP Options with Your VPC (p. 176).



If you want DNS to work with your home network, you must provide your own DNS server, and add it to the list of DNS servers your VPC uses. To do this for scenario 3, you must create a new set of DHCP options that includes both your DNS server and the one from Amazon, and then configure the VPC to use that set of options. Sets of DHCP options aren't modifiable once they exist, so you can't just add your DNS server to the existing set of options.

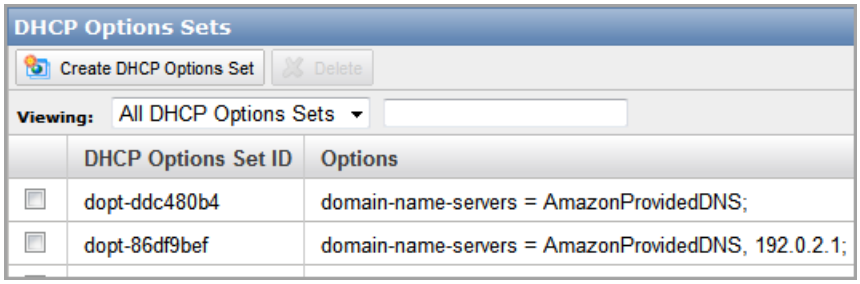
To update the DHCP options

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **DHCP Options Sets**, and then click **Create DHCP Options Set**.
3. In the **Create DHCP Options Set** dialog box, in the **domain-name-servers** field, enter the Amazon DNS server IP address (AmazonProvidedDNS) and your corporate DNS server, separated by a comma. In this example, your DNS server is 192.0.2.1.



4. Click **Yes, Create**.

   The new set of DHCP options is created. You now have the original set that your VPC comes with and the new set you just created.



5. Write down the ID of the new set of options you just created.
6. In the **Navigation** pane, click **Your VPCs**.
7. Select the VPC and click **Change DHCP Options Set**.

8. In the **Change DHCP Options Set** dialog box, select the ID of the new set of options and click **Yes, Change**.

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.

Congratulations! You've implemented scenario 3. You've got a VPC with a public subnet containing instances that are reachable from the Internet (and your home network), and that can initiate traffic to the Internet. You've also got a VPN-only subnet that can communicate with your home network.

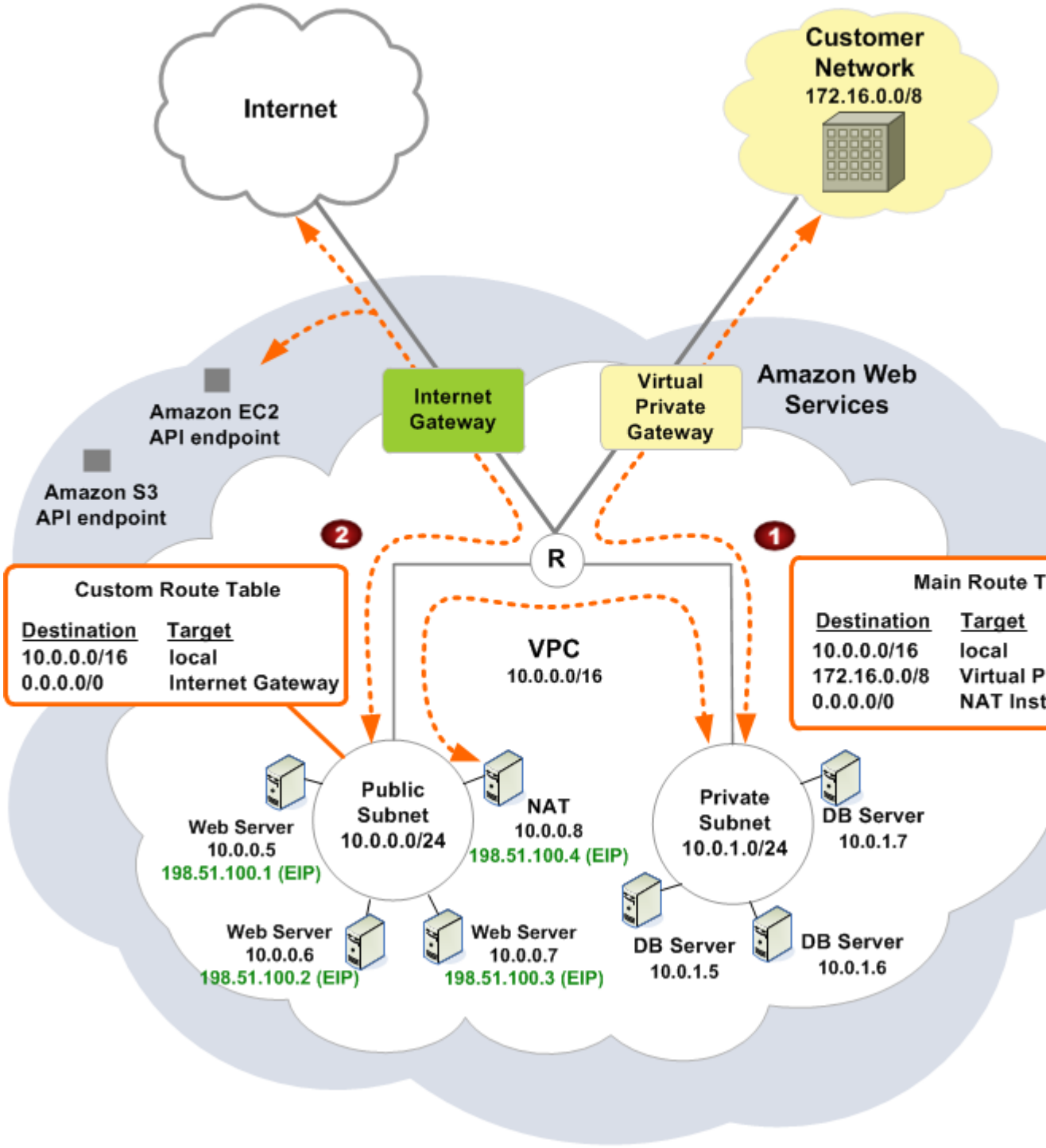
Alternate Routing

The preceding sections assume you want the Internet-bound traffic from the VPN-only subnet to go over the virtual private gateway to your home network. You could instead set up the routing so the subnet's Internet-bound traffic goes to a Network Address Translation (NAT) instance in the public subnet (scenario 2 uses a similar setup). The NAT instance enables the instances in the VPN-only subnet to send requests out to the Internet over the Internet gateway (e.g., for software updates). Amazon provides AMIs specifically to act as NAT instances in your VPC. For more information, see NAT Instances (p. 132).

Note

The NAT instance's primary role is actually Port Address Translation (PAT). However, we use the more widely known term *NAT* when referring to the instance. For information about PAT, go to the Wikipedia article about PAT.

The following diagram shows the routing for the alternative version of scenario 3. Notice that the VPN-only subnet is now labeled as *private* instead of VPN-only. The subnet no longer routes all its traffic to the virtual private gateway. However, it still can't be reached directly from the Internet, so we label it *private*.



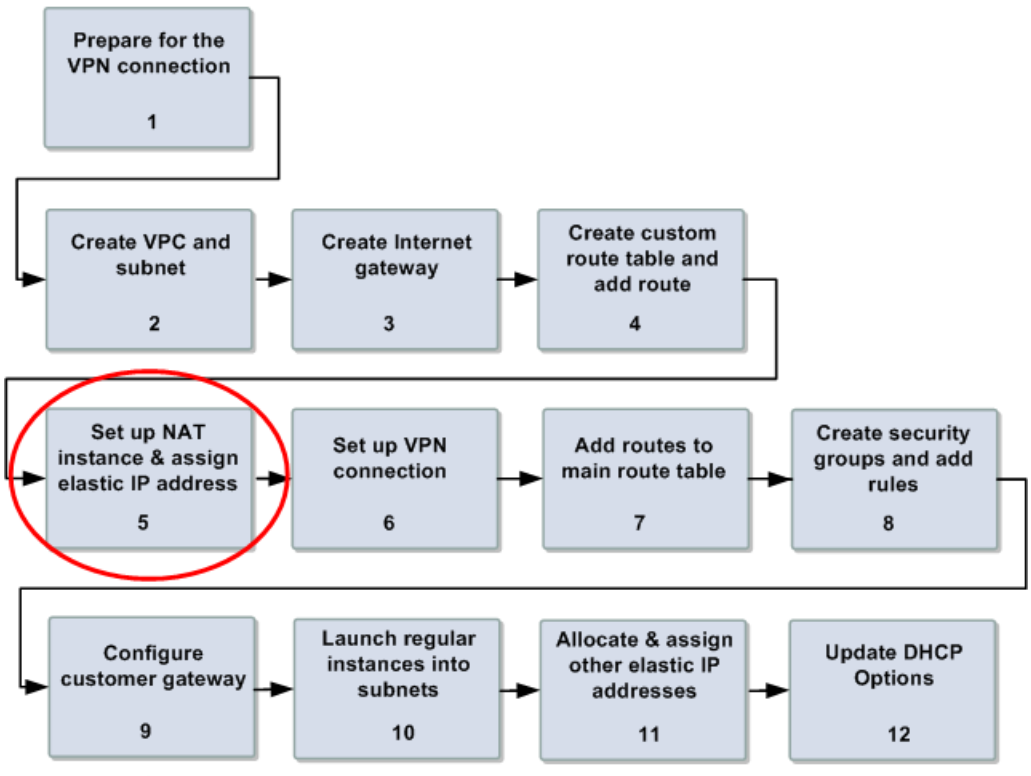
To enable the private subnet's Internet-bound traffic to go to the NAT instance, you must add a route to the main route table.

Main Route Table

The first row provides local routing within the VPC. The second row sends all traffic bound for the home network's IP address range to the virtual private gateway, which is specified by its AWS-assigned identifier. The third row sends all remaining traffic (i.e., Internet-bound traffic) to the NAT instance, which is specified by its AWS-assigned identifier.

Destination	Target
10.0.0.0/16	local
172.16.0.0/8	vgw-xxxxxxxx
0.0.0.0/0	i-xxxxxxxx

To implement the alternative version of scenario 3, use the process in the following diagram. It's very similar to the process shown for the original version of scenario 3, but with one additional task (circled in the diagram).



The console doesn't have a wizard to handle this scenario, so you must do each task yourself. Most of the basic tasks are covered earlier in this section (see Implementing the Scenario (p. 55)). However, the following sections describe how to:

- Set up the NAT instance
- Change the routing to accommodate the NAT instance
- Modify the security groups and create a special security group for the NAT instance

Set Up the NAT Instance

If you're already familiar with launching Amazon EC2 instances outside a VPC, then you already know most of what you need to know about launching the NAT instance. The additional items to know:

- Amazon provides NAT AMIs you can use
- You must specify the VPC and subnet when you launch the instance

After the NAT instance is running, you must also do the following tasks to complete the setup:

- Disable the source/destination check on the instance (instructions follow)
- Allocate and assign an Elastic IP address to the instance (instructions follow)
- Update the main route table with a route going to the instance
- Create a NATSG security group and move the instance into the group

To launch a NAT instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **AMIs**.
3. Change the **Viewing** settings to show Amazon AMIs using the Amazon Linux platform.
   The NAT AMIs that we provide include the string `ami-vpc-nat` in their names
4. Locate the NAT AMI of your choice, right-click it, and select **Launch Instance** to start the launch wizard.

   The wizard opens on the **Instance Details** page. This is where you control settings such as the number and size of instances to launch, and which subnet to launch the instance in.
5. Select the **Launch Instances Into Your Virtual Private Cloud** option, and select the public subnet in your VPC. Keep the other default settings on this page and click **Continue**.
   The wizard steps to the next page for instance details.
6. The default settings on this page of the wizard and the next page are what we want, so just click **Continue** on each page.

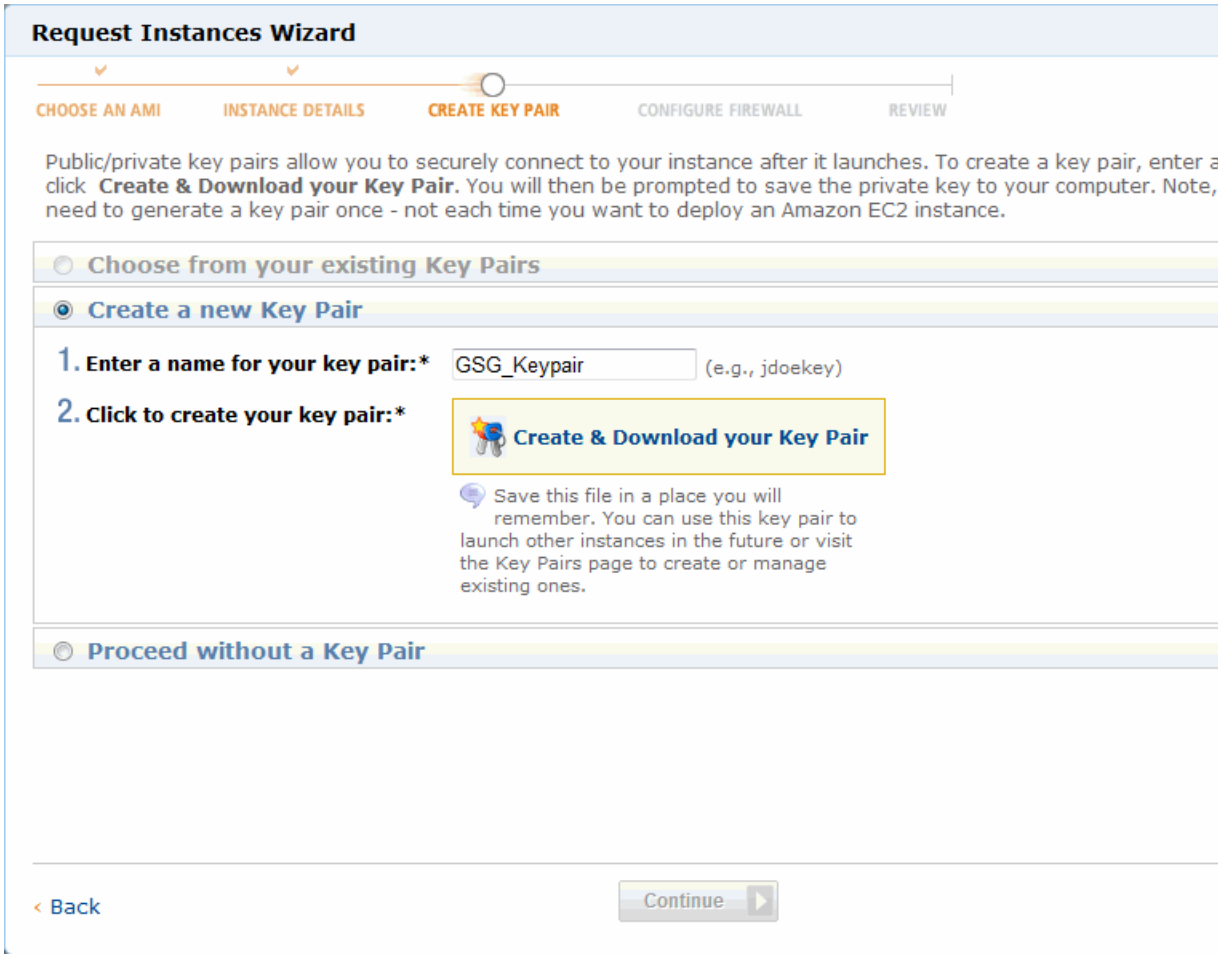
   The **Create Key Pair** page appears.

   Tip

   If you're already familiar with Amazon EC2 and have an SSH key pair already, you don't need to create a new one now.

   A *key pair* is a security credential similar to a password, which you use to securely connect to your instance once it's running. If you're new to Amazon EC2 and haven't created any key pairs yet, when the wizard displays the **Create Key Pair** page, the **Create a new Key Pair** button is selected by default. It's assumed you'll want a new key pair.
7. Create a key pair:

   a. On the **Create Key Pair** page, enter a name for your key pair (e.g., GSG_Keypair). This is the name of the private key file associated with the pair (with a `.pem` extension).



b. Click **Create & Download your Key Pair**.
You're prompted to save the private key from the key pair to your system.

c. Save the private key in a safe place on your system. Note the location because you'll need to use the key soon to connect to the instance.

8. On the **Configure Firewall** page of the wizard, select the VPC's default security group for now and click **Continue**. Later you'll create a NATSG group and move the instance into that group.
After you configure the firewall, the wizard steps to the **Review** page where you can review the settings and launch the instance.

9. Review your settings and launch the instance:

a. Click **Launch**.
A confirmation page is displayed to let you know your instance is launching.

b. Click **Close** to close the confirmation page, and then click **Instances** in the navigation pane to view your instance's status. It takes a short time for an instance to launch. The instance's status is *pending* while it's launching. After a short period, your instance's status switches to *running*. You can click **Refresh** to refresh the display.

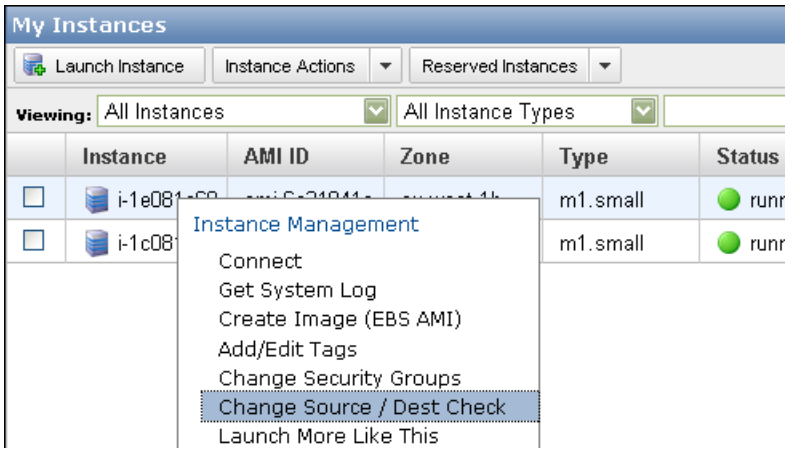
You now have a NAT instance running in your VPC. For the instance to perform network address translation, you must disable source/destination checking on the instance. In other words, each EC2 instance performs source and destination checking by default. This means the instance must be the source or destination of any traffic it sends or receives. However, the NAT instance needs to be able to send and receive traffic where the eventual source or destination is not the NAT instance itself. To enable that behavior, you must disable source/destination checking on the NAT instance.

To disable source/destination checking on the NAT instance

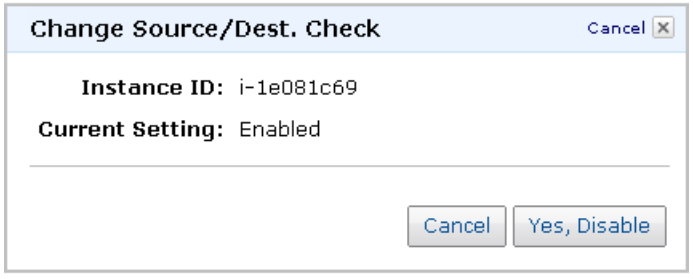
Note

This procedure only works for EC2 instances that are running within a VPC.

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the NAT instance in the list of instances, and select **Change Source / Dest Check**.



The **Change Source/Dest. Check** dialog box opens.



For a regular instance, the value should be *Enabled*, indicating that the instance is performing source/destination checking. For a NAT instance, you want the value to be *Disabled*.

4. Click **Yes, Disable**.

Source/destination checking for the instance is disabled. Your NAT is one step closer to being able to do its job.

Your NAT instance also needs an Elastic IP address.

To allocate and assign an Elastic IP address to an instance

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Elastic IPs**, and then click **Allocate New Address**.
3. In the **Allocate New Address** dialog box, in the **EIP used in:** drop-down list, select VPC, and then click **Yes, Allocate**.
   The new address is allocated and appears on the page.
4. Right-click the IP address in the list and select **Associate**.
5. In the **Associate Address** dialog box, select the instance you want to associate the address with and click **Yes, Associate**.
   The address is associated with the instance. Notice that the instance ID is displayed next to the IP address in the list.

Your NAT instance now has an Elastic IP address associated with it. The instance is currently in the default security group. After you've created your security groups, you need to move the NAT instance into the NATSG group (see Task 6: Create Security Groups and Add Rules (p. 38)).

Add Routes to Main Route Table

For this alternate scenario, you must set up the main route table so that traffic bound for the home network goes to the virtual private gateway, and all remaining traffic goes to the NAT instance. The following table shows what the main route table would look like.

Main Route Table

The first row provides local routing in the VPC. The second row in the table sends the subnet traffic bound for the home network to the virtual private gateway, which is specified by its AWS-assigned identifier (e.g., vgw-xxxxxxxx). The third row sends all remaining subnet traffic to the NAT instance, which is specified by its AWS-assigned identifier (e.g., i-xxxxxxxx).

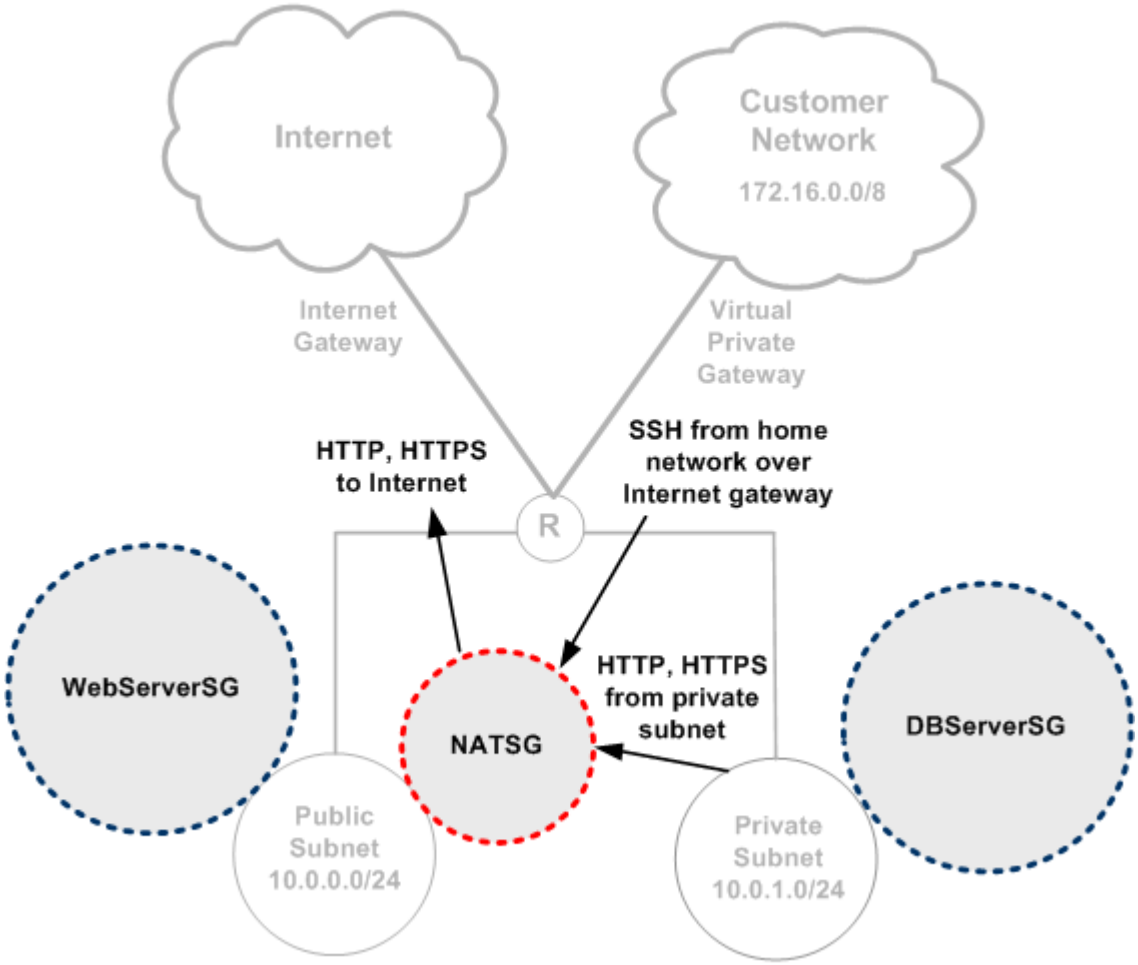
Destination	Target
10.0.0.0/16	local
172.16.0.0/8	vgw-xxxxxxxx
0.0.0.0/0	i-xxxxxxxx

For general instructions on creating route tables and adding routes, see Task 4: Create a Custom Route Table and Add Rules (p. 61).

Create NATSG Security Group

For this alternate scenario, you must create a security group for the NAT instance (we'll call it NATSG) in addition to the other security groups described earlier (see Security (p. 50)). You also must move the NAT instance into the NATSG group.

The following figure and table show the NATSG group and its rules.



NATSG

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
10.0.1.0/24	TCP	80	Allow inbound HTTP traffic from servers in private subnet
10.0.1.0/24	TCP	443	Allow inbound HTTPS traffic from servers in private subnet
Public IP address range of your home network	TCP	22	Allow inbound SSH access to the Linux/UNIX NAT instance from your home 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

The following image shows what the rules look like for the NATSG security group.

NATSG: Inbound

TCP Port (Service)	Source	Action
80 (HTTP)	10.0.1.0/24	Delete
443 (HTTPS)	10.0.1.0/24	Delete
22 (SSH)	192.0.2.0/24	Delete

NATSG: Outbound

TCP Port (Service)	Destination	Action
80 (HTTP)	0.0.0.0/0	Delete
443 (HTTPS)	0.0.0.0/0	Delete

For general instructions on how to create a security group, see Task 7: Create Security Groups and Add Rules (p. 64).

When you launched the NAT instance, you put it in the default security group in the VPC. You now need to move it into the NATSG group.

To change an instance's group membership

Note

The following procedure works only for VPC instances. You can't change security group membership for standard (EC2) instances.

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the NAT instance in the list of instances, and select **Change Security Groups**.

   The **Change Security Groups** dialog box opens, with the default group selected (the instance is in the default group currently).

4. From the drop-down list, select the NATSG group, and then click **Yes, Change**.

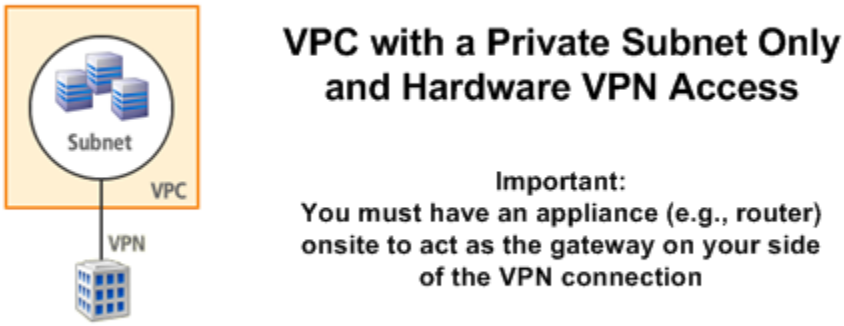
   Tip

   When changing an instance's group membership, you can select multiple groups from the list. The new list of groups you select replaces the instance's current list of groups. An instance can be in a maximum of five VPC security groups.

The NAT instance is now in the NATSG security group. Your instances in the private subnet can now reach the Internet via the NAT instance.

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

Topics



We recommend this scenario if you want extend your data center into the cloud and leverage Amazon's elasticity without exposing your network to the Internet. This scenario includes a VPN connection from your home network to your VPC, and no Internet gateway. This is the basic layout that Amazon VPC has supported since the initial release of the service.

Important

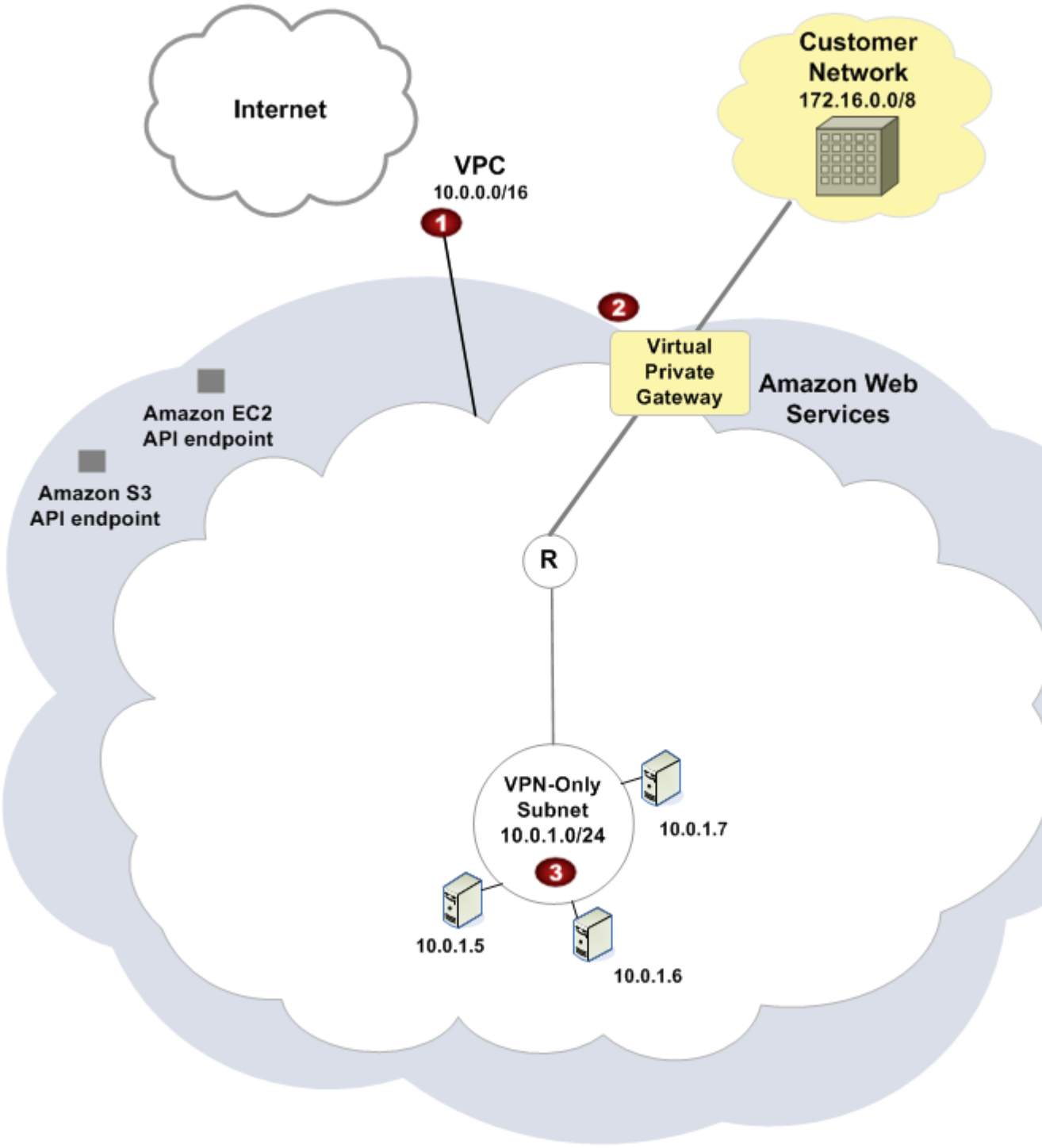
For this scenario, your network administrator needs the Amazon Virtual Private Cloud Network Administrator Guide in order to configure the Amazon VPC customer gateway on your side of the VPN connection.

Basic Layout

The following diagram shows the basic layout of your VPC in this scenario. The big white cloud is your VPC (your isolated portion of the AWS cloud). You have a virtual private gateway that enables the VPC to communicate with your home network over an IPsec VPN tunnel. The circle containing an R represents your VPC's built-in routing function. The VPC has one subnet. The table following the diagram gives additional details about the VPC and its layout for this scenario.

Tip

The AWS Management Console has a wizard In the Amazon VPC console to help you implement this scenario. For more information, see Implementing the Scenario (p. 90).



1	A size /16 VPC (e.g., 10.0.0.0/16), which means 65,536 private (RFC 1918) IP addresses. For information about CIDR notation and what the "/16" means, go to the Wikipedia article about Classless Inter-Domain Routing.

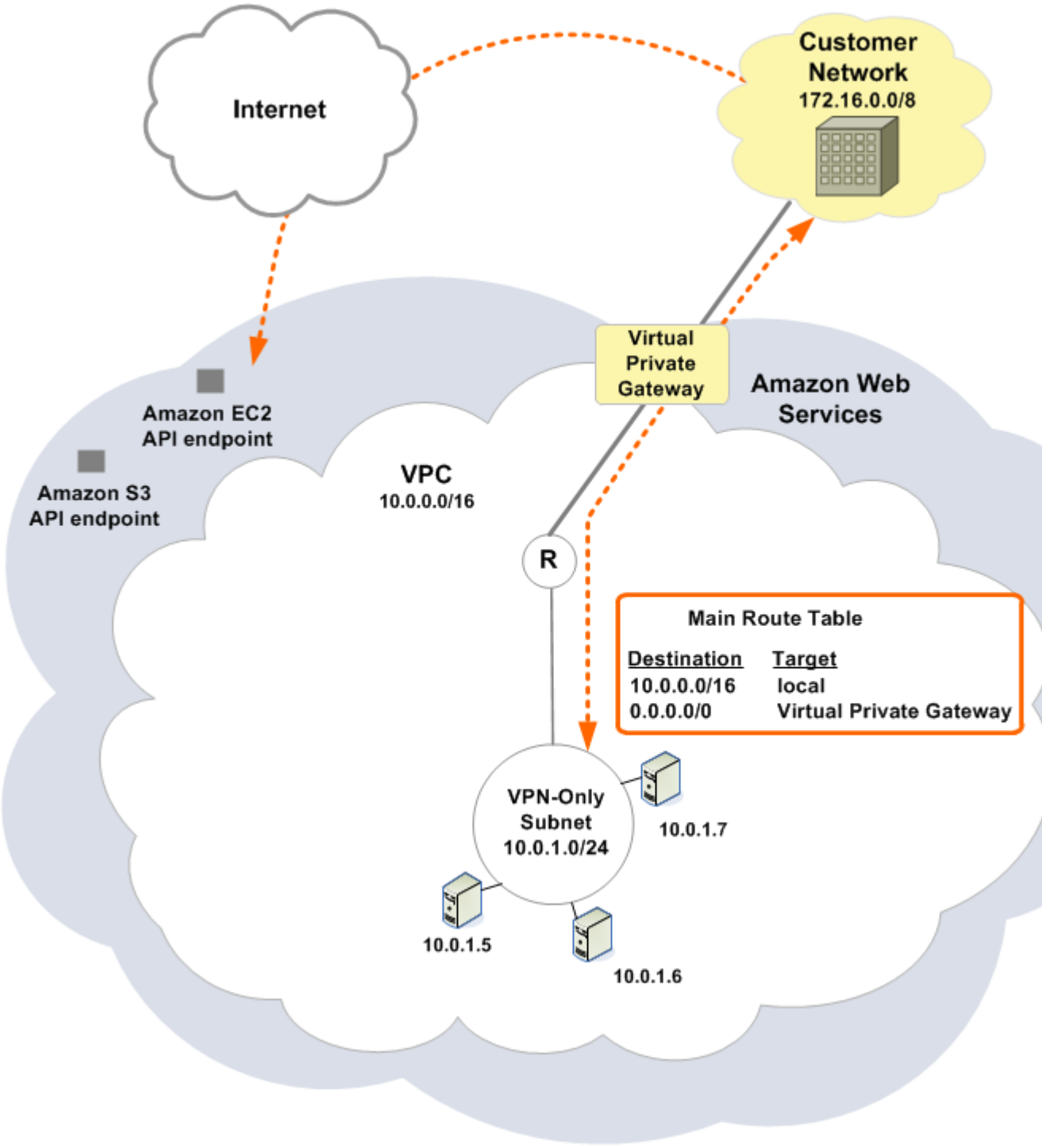
2	A VPN between your VPC and home network. The entire VPN setup consists of a customer gateway, virtual private gateway, VPN attachment (connecting the virtual private gateway to the VPC), and a VPN connection. For this scenario, we refer to the VPN setup generally as your virtual private gateway or VPN connection. For more information about your VPN connection, see Adding a Hardware Virtual Private Gateway to Your VPC (p. 164). To enable the VPN connection, you must have an appliance (e.g., router) in your home network that acts as the anchor on your side of the connection (for more information, go to Amazon Virtual Private Cloud Network Administrator Guide.
3	A size /24 subnet (e.g., 10.0.1.0/24), which means 256 private IP addresses. In the diagram, the subnet contains generic servers. Each has a private IP address (e.g., 10.0.1.5). You're going to set up routing in the VPC so that the subnet can send traffic only to your home network over the virtual private gateway (see Routing (p. 18)). Therefore, the subnet is labeled as *VPN-only* in the diagram.

The instances in your VPC can't reach the Internet directly; any Internet-bound traffic must traverse the virtual private gateway to your home network first, where the traffic is then subject to your firewall and corporate security policies. If the instances send any AWS-bound traffic (e.g., requests to the Amazon S3 or Amazon EC2 APIs), the requests must go over the virtual private gateway to your home network and then egress to the Internet before reaching AWS.

Routing

Your VPC has an implied router (shown in the following diagram as an R in a circle), as well as a modifiable main route table. You can also create other route tables to use in your VPC. By default, each table has a *local route* that enables instances in your VPC to talk to each other.

The following diagram and table describe the main route table and routes you need to set up in this scenario.



The VPC automatically comes with a main route table. Any subnet not explicitly associated with another route table uses the main route table. For this scenario, you update the main route table with a route that sends traffic from the VPN-only subnet to the virtual private gateway (the flow of traffic is indicated by the dotted line adjacent to the table). You don't explicitly associate the subnet with any route table, so it implicitly uses the routes in the main route table.

The following table shows what the main route table looks like for this scenario. The first row covers the local routing in the VPC (i.e., allows the instances in the VPC to communicate with each other).

Main Route Table

The first row provides local routing within the VPC. The second row sends all the subnet traffic over the virtual private gateway, which is specified by its AWS-assigned identifier (e.g., vgw-xxxxxxxx).

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

> **Note**
>
> If you use the wizard in the AWS Management Console to set up your VPC, the wizard automatically updates the main route table with the route between the subnet and the virtual private gateway. Otherwise, you must update the main route table yourself.

Any AWS-bound traffic from the subnet (e.g., going to the Amazon EC2 or Amazon S3 API endpoints) is routed to the virtual private gateway. The traffic must egress your home network to the Internet, so you're charged for both the data transfer across the virtual private gateway, and the Internet data transfer costs to access your AWS resources.

Security

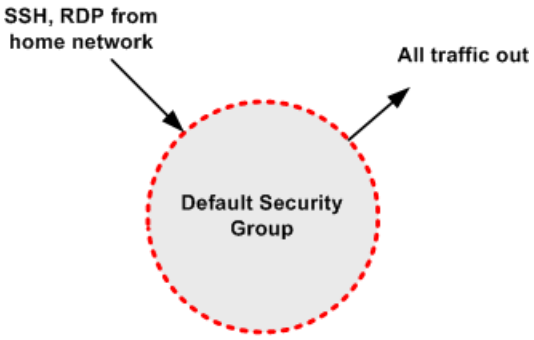
AWS provides two ways for you to control security in your VPC: *security groups* and *network ACLs*. They both enable you to control what traffic goes in and out of your instances, but security groups work at the instance level, and network ACLs work at the subnet level. Security groups alone will suffice for many VPC users. However, some users might want 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. 136).

> **Important**
>
> Security groups are a basic Amazon EC2 concept. However, security groups in a VPC have different capabilities than security groups in EC2 (see EC2 vs. VPC Security Groups (p. 139)).

Security Groups

For scenario 4, you use only the default security group that comes with your VPC. Its initial settings are to deny all inbound traffic, allow all outbound traffic, and allow all traffic between the instances in the group. We recommend you change the default security group's rules to allow only inbound SSH traffic (for Linux/UNIX instances) and Remote Desktop traffic (for Windows instances) from your home network.

The following figure shows the default security group as a circle. The figure has a corresponding table that lists the recommended inbound and outbound rules to use with the default security group and what they do.



Default Security Group

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
Private IP address range of your home network (e.g., 172.16.0.0/8)	TCP	22	Allow inbound SSH traffic to Linux/UNIX instances from your home network
Private IP address range of your home network (e.g., 172.16.0.0/8)	TCP	3389	Allow inbound RDP traffic to Windows instances from your home network
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	All	All	Allow all outbound traffic from the instances

Note

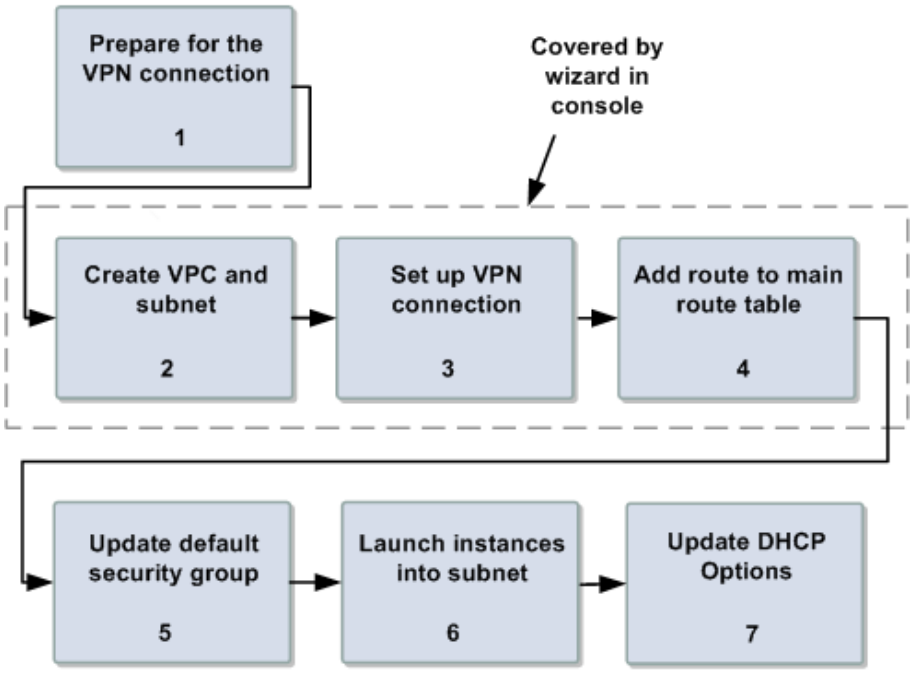
The default security group automatically allows its instances to talk to each other, so you don't have to add a rule to the group to specifically allow that. For groups other than the default security group, you must add that type of rule if you want the instances in the group to communicate with each other.

Implementing the Scenario

This section walks you through the process of implementing Scenario 4. The following figure and table show the tasks required to implement the scenario.

Tip

Three of the tasks are automatically handled for you if you use the wizard in the AWS Management Console. The following sections describe how to use the wizard, and how to do all the tasks manually.



Process for Implementing Scenario 4

Task 1: Prepare for the VPN Connection (p. 91)
Task 2: Create the VPC and Subnet (p. 95)
Task 3: Set Up the VPN Connection (p. 96)
Task 4: Add a Route to the Main Route Table (p. 97)
Task 5: Update the Default Security Group (p. 98)
Task 6: Launch Instances into the Subnet (p. 99)
Task 7: Update DHCP Options (p. 103)

Task 1: Prepare for the VPN Connection

In scenario 4, you set up a VPN connection between your home network and your VPC. The connection requires an appliance onsite (e.g., router) to act as your customer gateway. You need help from a network administrator in your organization to:

- Determine the appliance that will be your customer gateway
- Provide you 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))

For more information about the requirements for your customer gateway, go to the Amazon Virtual Private Cloud Network Administrator Guide.

If you want to use the wizard to set up your VPC, see Use the Wizard for Scenario 4 (p. 92). Otherwise, see Task 2: Create the VPC and Subnet (p. 95) to perform the process manually.

Use the Wizard for Scenario 4

You can have Amazon VPC complete tasks 2-4 for you by using the wizard in the AWS Management Console. This procedure assumes you don't already have a VPC, and that you have the IP address for your customer gateway (see the preceding section).

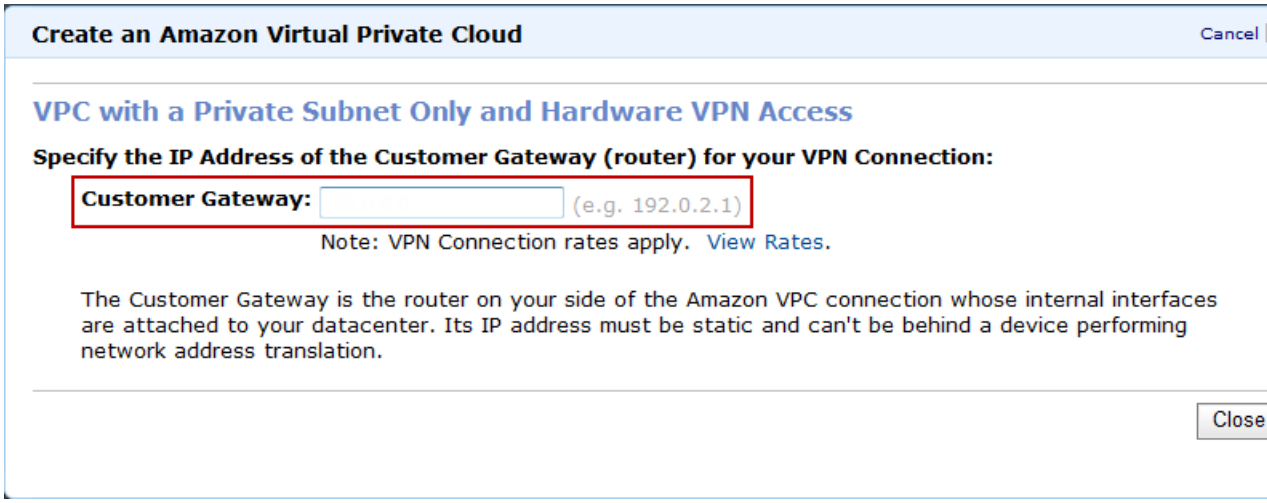
To use the wizard

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **VPC Dashboard**.
3. On the **VPC Dashboard**, locate the **Your Virtual Private Cloud** area and click **Get started creating a VPC**.

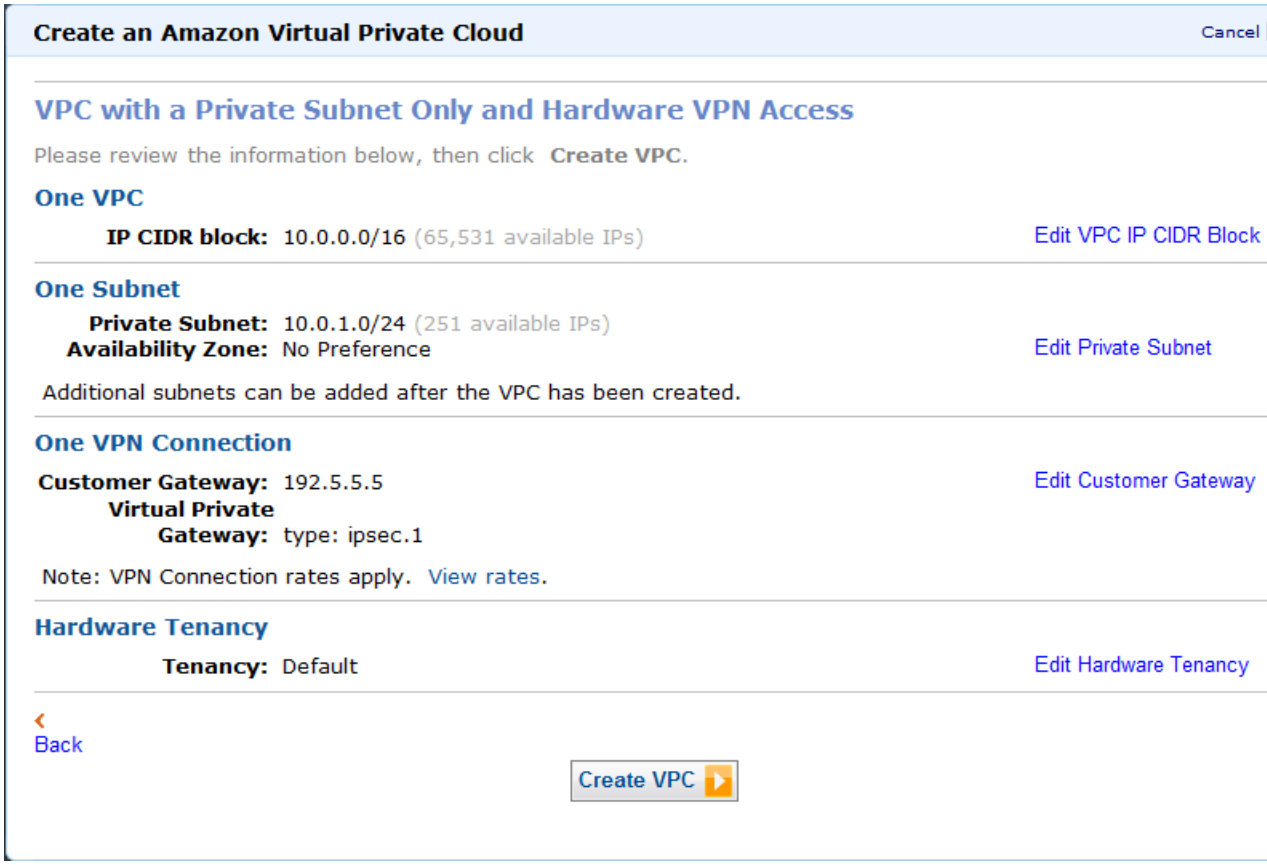
The wizard opens and displays a page where you can select one of four options.

4. Select **VPC with a Private Subnet Only and Hardware VPN Access** and click **Continue**.

A dialog box opens with a field for your customer gateway's IP address.

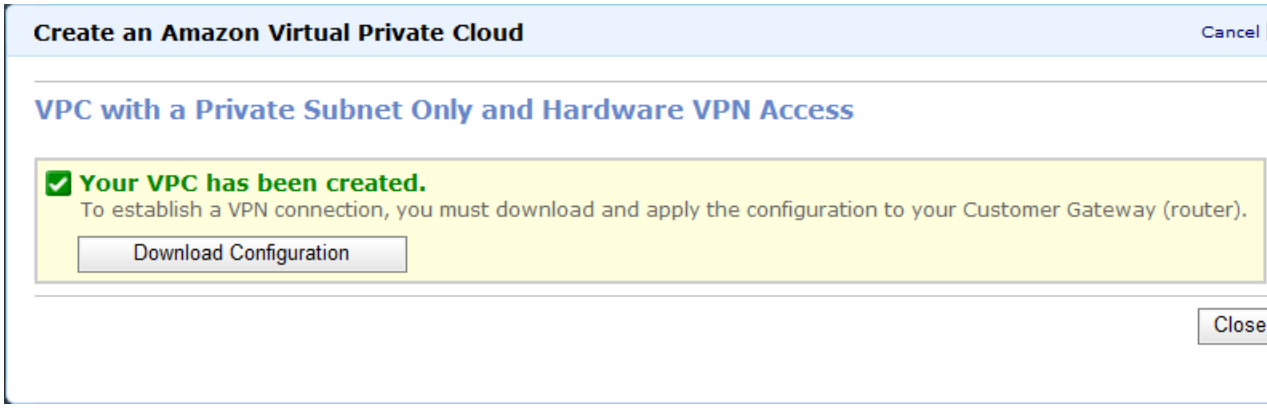


5. Enter your customer gateway's IP address and click **Continue**.
   A confirmation page is displayed showing the CIDR blocks we use for the VPC and subnet. It also shows the IP address that you just provided for the customer gateway, as well as the instance hardware tenancy of the VPC. You can change any of these values if you want.

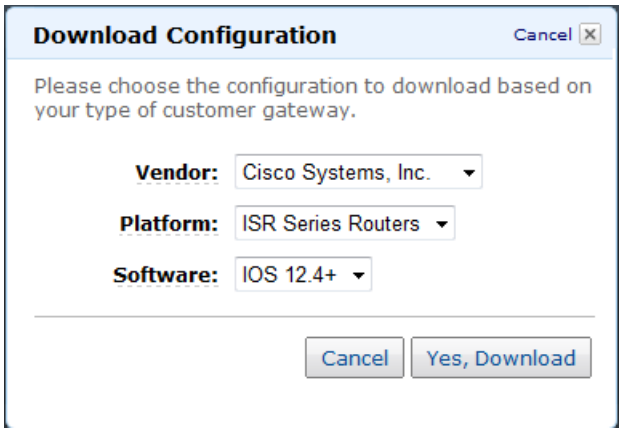


6. Make any changes you want and click **Create VPC**.

   The wizard begins to create your VPC, subnet, and VPN connection. It also updates the main route table and adds routes. When the wizard is done, a confirmation dialog box is displayed with a button for downloading the configuration for your customer gateway.



7. Click **Download Configuration**.
8. In the **Download Configuration** dialog box, select the customer gateway's vendor, platform, and software version, and then click **Yes, Download**.



9. 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.

After the wizard completes, you're partway done. The next task is to update the default security group. For more information, see Task 5: Update the Default Security Group (p. 98)).

Note that the next few sections show how to manually do tasks that the wizard already completed for you.

Task 2: Create the VPC and Subnet

If you don't use the wizard in the console, you can manually create the VPC and subnet yourself. This section shows you how.

To create your VPC and subnet

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, enter the CIDR range you want for your VPC (e.g., 10.0.0.0/16), and then click **Yes, Create**.

   Tip

   For information about choosing the CIDR range for your VPC, see VPC Sizing (p. 105).

   The VPC is created and appears on the **Your VPCs** page. Notice that it has an ID (e.g., vpc-xxxxxxxx).
4. In the **Navigation** pane, click **Subnets**.
5. Click **Create Subnet**.
6. In the **Create Subnet** dialog box, select the VPC and Availability Zone, enter the CIDR range you want for your subnet (e.g., 10.0.1.0/24), and then click **Yes, Create**.
   The subnet is created and appears on the **Subnets** page. Notice that it has an ID (e.g., subnet-xxxxxxxx). The page also shows the number of available IP addresses in the subnet, the route table associated with the subnet, and the network ACL associated with the subnet. The subnet uses the main route table and default network ACL by default.

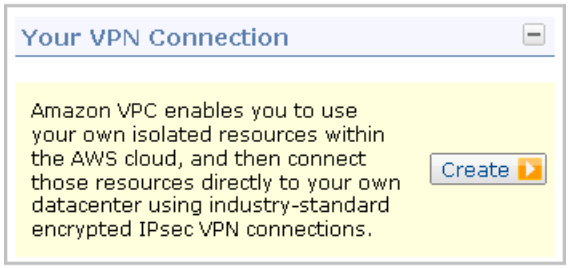
You've got your VPC and subnet now.

Task 3: Set Up the VPN Connection

If you don't use the wizard in the console, you can manually set up the VPN connection yourself. This section shows you how.

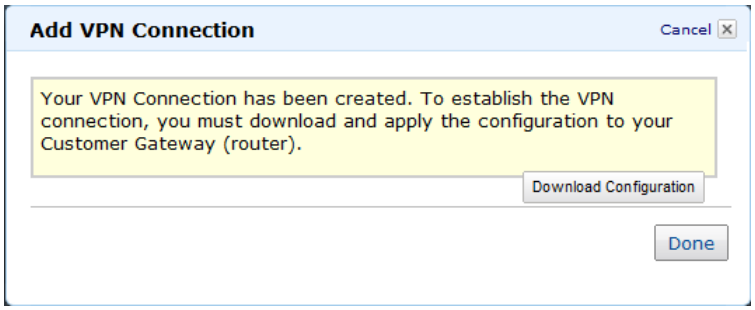
To set up the VPN connection

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **VPC Dashboard**.
3. In the **Your VPN Connections** area of the page, click **Create** (if this is your first VPN) or click **Add VPN Connection**.

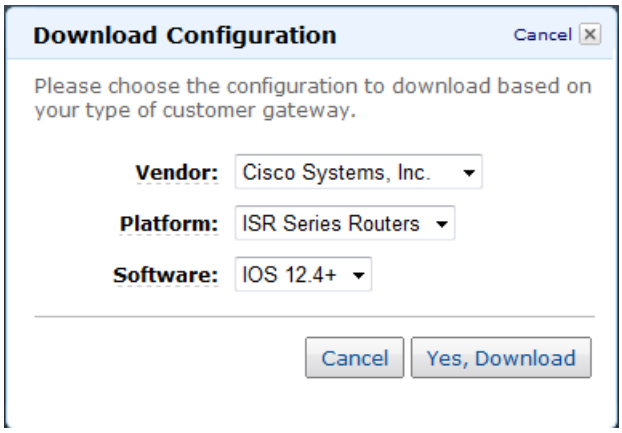


4. In the **Add VPN Connection** dialog box, enter the IP address for your customer gateway (e.g., 198.0.2.1), and then click **Yes, Create**.

   We create your customer gateway and your virtual private gateway, attach the virtual private gateway to the VPC, and create a VPN connection. When the wizard is done, a confirmation dialog box is displayed with a button for downloading the configuration for your customer gateway.



5. Click **Download Configuration**.
6. In the **Download Configuration** dialog box, select the customer gateway's vendor, platform, and software version, and then click **Yes, Download**.



7. Save the text file contain and give it to the network administrator along with this guide: Amazon Virtual Private Cloud Network Administrator Guide.

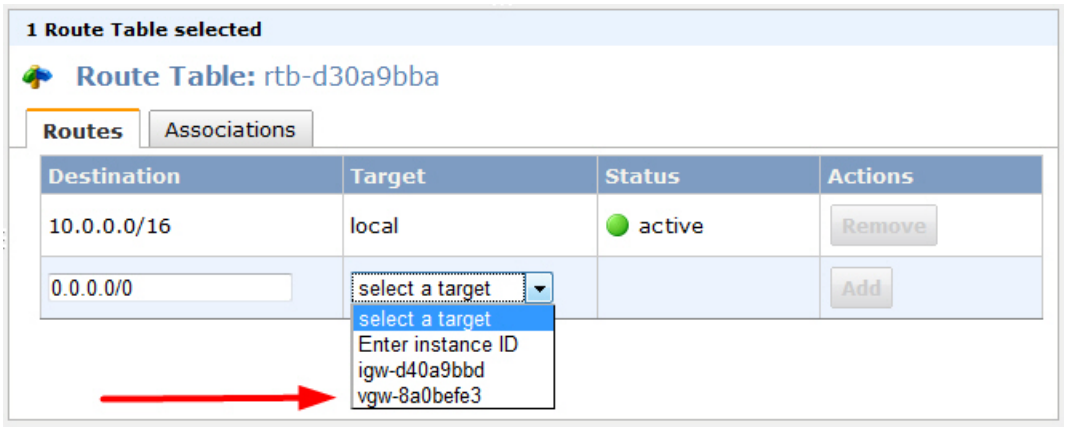
You now have a customer gateway, a virtual private gateway attached to your VPC, and a VPN connection. However, the VPN won't work until your network administrator configures the customer gateway. Also, no route table refers to the gateway yet, so no traffic can flow to the gateway. Move on to the next section to set up routing for the VPN-only subnet.

Task 4: Add a Route to the Main Route Table

If you don't use the wizard in the console, you can manually add the required route to the main route table yourself. This section shows you how.

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**.
   Your VPC's route tables are listed.
3. In the list of route tables, select the check box for the main route table.
4. In the lower pane, in the **Routes** tab, enter `0.0.0.0/0` in the **Destination** field, and select the virtual private gateway's ID in the **Target** drop-down list, and click **Add**.



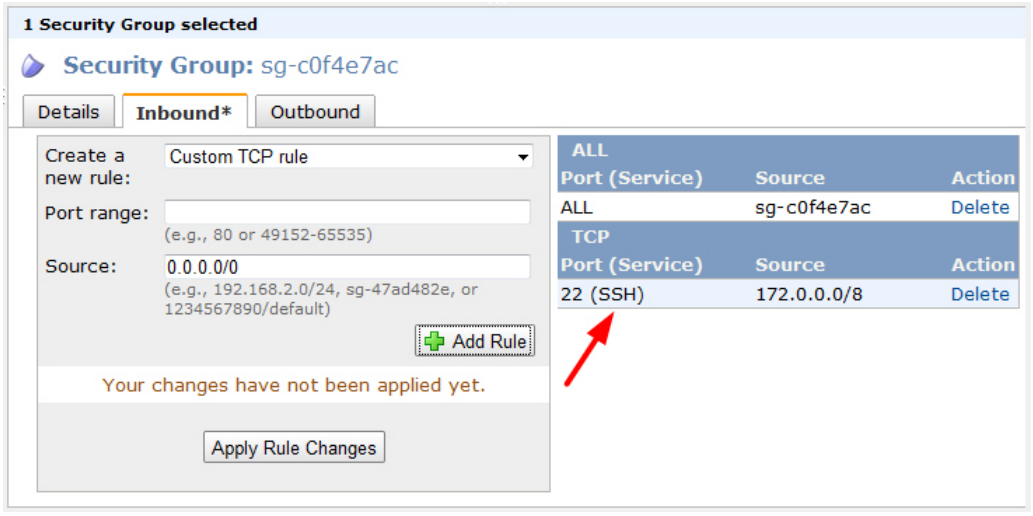
The VPC's main route table now includes the new route. The route enables traffic to flow between the subnet and the virtual private gateway. If you click the **Associations** tab (next to the **Routes** tab for the main route table), you can see which subnets are using the main route table. Your VPC's subnet is listed there because you haven't explicitly associated your subnet to any route table.

Task 5: Update the Default Security Group

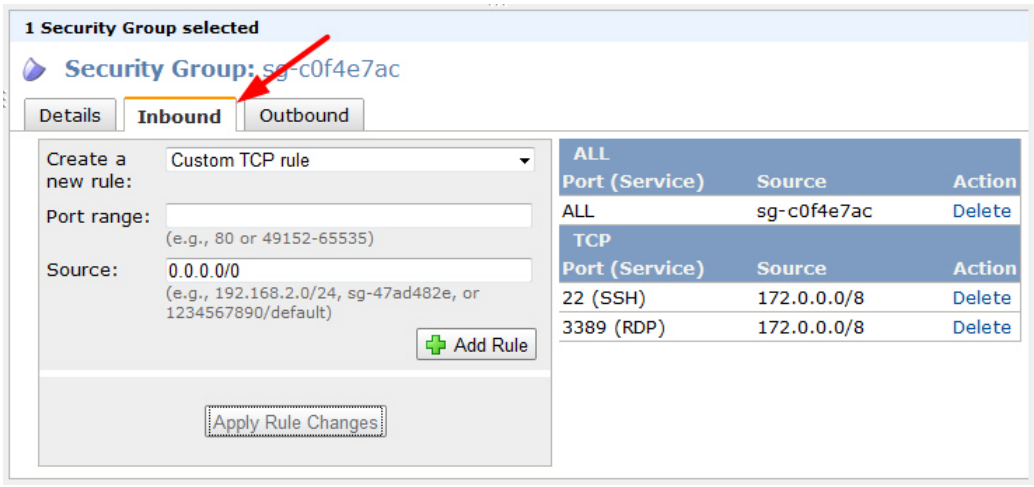
For this scenario, you update the default security group with new inbound rules that allow SSH and Remote Desktop (RDP) access from your home network. Reminder: the initial settings of the default security group block all inbound traffic, allow all outbound traffic, and allow all instances in the group to talk to each other.

To update the rules for the default security group

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Security Groups**, and then select the check box for the VPC's default security group.
3. In the lower pane, add a rule for inbound SSH access to the group from your home network:
   a. On the **Inbound** tab, select SSH from the **Create a new rule** drop-down list.
   b. In the **Source** field, enter your home network's private IP address range (e.g., 172.16.0.0/8).
   c. Click **Add Rule**.
   The rule is added to the **Inbound** tab. However, the rule isn't applied to the group until you click **Apply Rule Changes** (which you'll do after you've added all the inbound rules).



4. Add a rule for inbound RDP access to the group from your home network:
   a. On the **Inbound** tab, select RDP from the **Create a new rule** drop-down list.
   b. In the **Source** field, enter your home network's private IP address range (e.g., 172.16.0.0/8).
   c. Click **Add Rule**.
   The rule is added to the **Inbound** tab.
5. Click **Apply Rule Changes**.



The new inbound rules now apply to the default security group.

The default security group now allows SSH and RDP access from your home network to the instances. Move on to the next section to launch instances into the subnet.

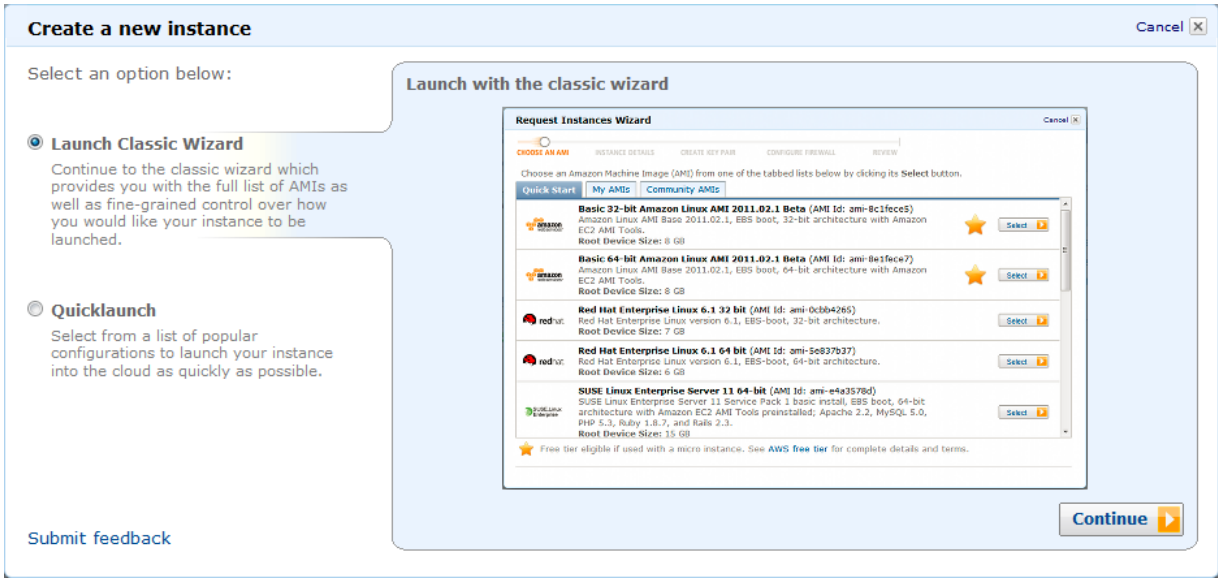
Task 6: Launch Instances into the Subnet

After your network administrator has configured your customer gateway, you can launch instances into your VPC. If you haven't launched instances before, use the following procedure. If you're already familiar with launching Amazon EC2 instances outside a VPC, then you already know most of what you need to know. You just need to specify the VPC and subnet when launching the instance.

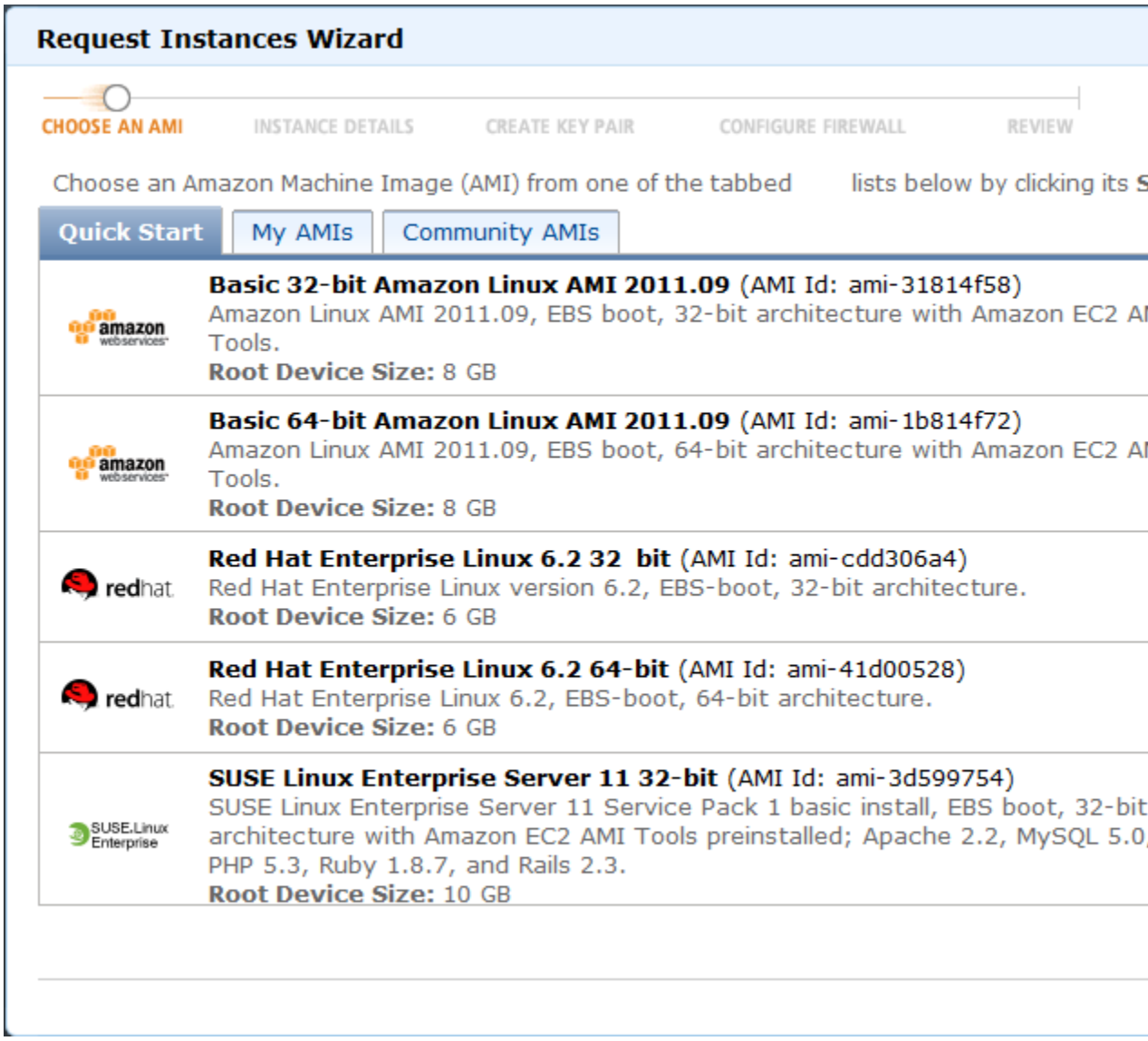
To launch an instance

1. Start the launch wizard:
   a. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
   b. Click **Launch Instance** to start the Request Instances Wizard.

   c. On the **Create a New Instance** screen, select **Classic Wizard**, and then click **Continue**.



2. Select an AMI from one of the tabs. If you don't have a particular AMI you want to launch, select either the *Basic 32-bit Amazon Linux AMI*, or the *Getting Started on Microsoft Windows Server 2008* AMI on the **Quick Start** tab.



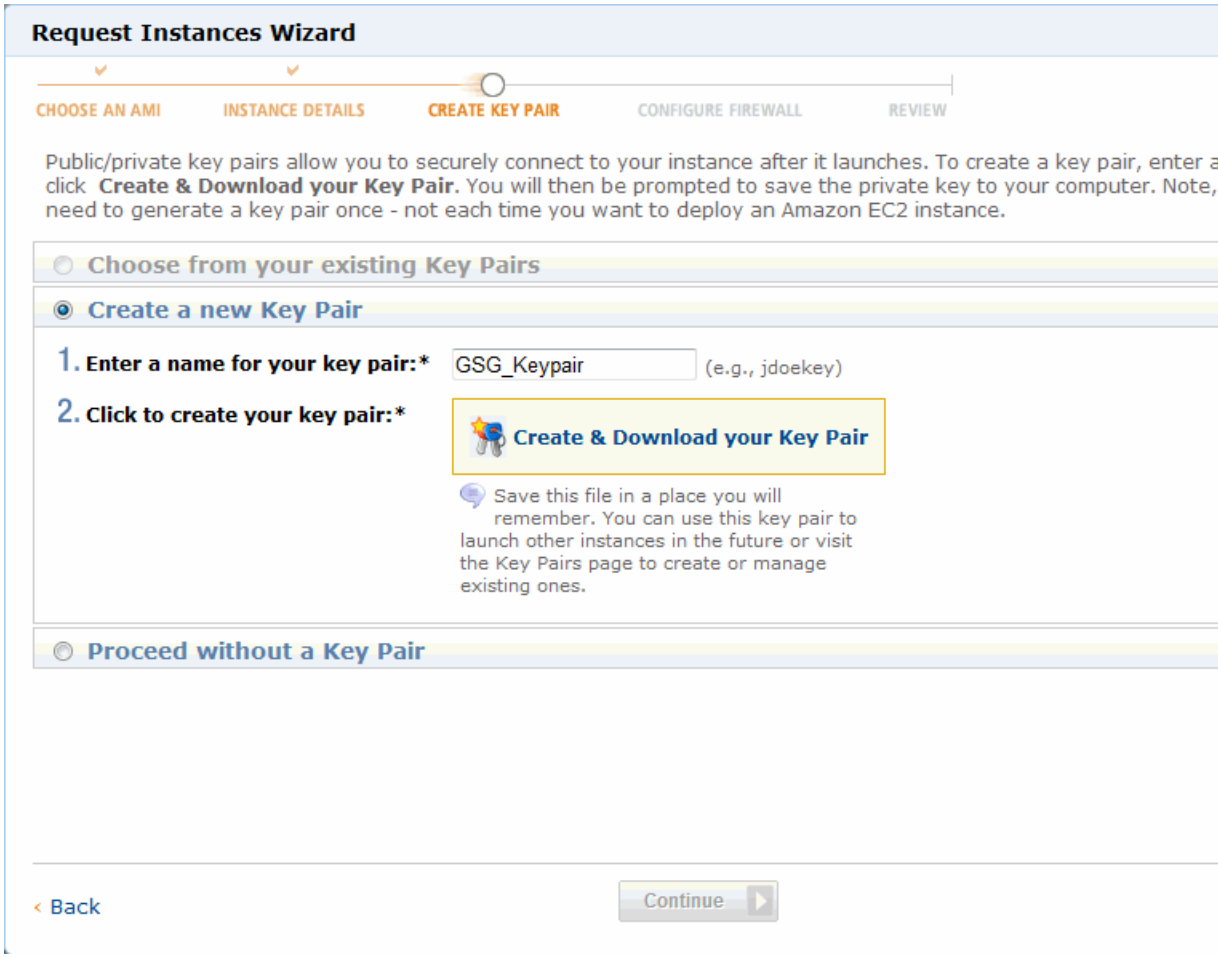
After you select an AMI, the wizard steps to the **Instance Details** page. This is where you control settings such as the number and size of instances to launch, and which subnet to launch the instance in.

3. Select the **Launch Instances Into Your Virtual Private Cloud** option, and select the subnet you want to launch the instance in. Keep the other default settings on this page and click **Continue**. The wizard steps to the next page for instance details.
4. The default settings on this page of the wizard and the next page are what we want, so just click **Continue** on each page.
5. Create a key pair:
   A *key pair* is a security credential similar to a password, which you use to securely connect to your instance once it's running. If you're new to Amazon EC2 and haven't created any key pairs yet, when the wizard displays the **Create Key Pair** page, the **Create a new Key Pair** button is selected by default. It's assumed you'll want a new key pair.

Tip

If you're already familiar with Amazon EC2 and have an SSH key pair already, you don't need to create a new one now. You can just select one of your existing key pairs instead.

a. On the **Create Key Pair** page, enter a name for your key pair (e.g., GSG_Keypair). This is the name of the private key file associated with the pair (with a `.pem` extension).



b. Click **Create & Download your Key Pair**.
You're prompted to save the private key from the key pair to your system.

c. Save the private key in a safe place on your system. Note the location because you'll need to use the key soon to connect to the instance.

6. On the **Configure Firewall** page of the wizard, select the default security group and click **Continue**. After you configure the firewall, the wizard steps to the **Review** page where you can review the settings and launch the instance.
7. Review your settings and launch the instance:

   a. Click **Launch**.
   A confirmation page is displayed to let you know your instance is launching.

   b. Click **Close** to close the confirmation page, and then click **Instances** in the navigation pane to view your instance's status. It takes a short time for an instance to launch. The instance's status

is *pending* while it's launching. After a short period, your instance's status switches to *running*. You can click **Refresh** to refresh the display.

You now have an instance running in your VPC. You can test the connectivity to the instance by pinging it from your home network. For more information, see How to Test the End-to-End Connectivity of Your Instance (p. 171).

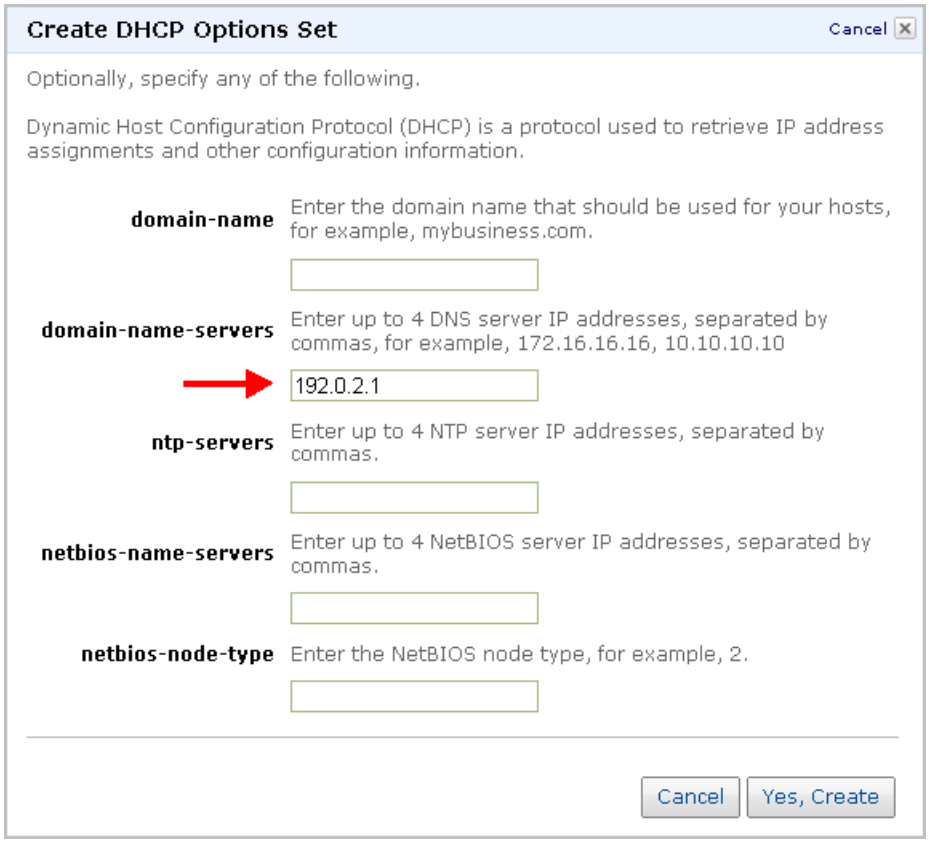
You can now use SSH or Remote Desktop to connect to your instance in the VPC. For instructions on how to connect to a Linux/UNIX instance, go to Connect to Your Linux/UNIX Instance in the *Amazon Elastic Compute Cloud Getting Started Guide*. For instructions on how to connect to a Windows instance, go to Connect to Your Windows Instance.

Task 7: Update DHCP Options

In scenario 4, you need a DNS server that enables your VPN-only subnet to communicate with servers in your home 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.

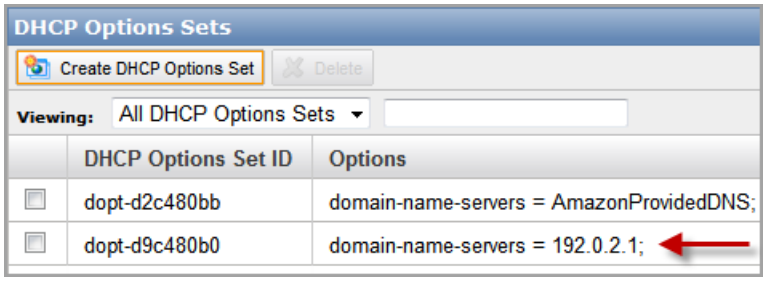
To update the DHCP options

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **DHCP Options Sets**.
3. Click **Create DHCP Options Set**.
4. In the **Create DHCP Options Set** dialog box, in the **domain-name-servers** box, enter the address of your DNS server. In this example, your DNS server is 192.0.2.1.



5. Click **Yes, Create**.

   The new set of DHCP options is created.



   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.

6. Write down the ID of the new set of options you just created.
7. In the **Navigation** pane, click **Your VPCs**.
8. Select the VPC and click **Change DHCP Options Set**.
9. In the **Change DHCP Options Set** dialog box, select the ID of the new set of options and click **Yes, Change**.

The VPC now uses this new set of DHCP options and therefore has access to your corporate DNS server.

Note

After you associate a new set of 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.

Congratulations! You've implemented scenario 4. You've got a VPC with a VPN-only subnet that can communicate only with your home network.

If in the future you want to add an Internet gateway to your VPC and a public subnet, you can. Scenario 3 covers that setup. See Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 44).

Your VPC and Subnets

Topics

This section describes basic things you should know about your VPC and subnets.

Your VPC

A VPC is the first object you create when using Amazon Virtual Private Cloud. When creating the VPC, you simply provide the set of IP addresses you want the VPC to cover. You specify this set of addresses in the form of a Classless Inter-Domain Routing (CIDR) block. For example, 10.0.0.0/16. For information about CIDR notation and what the "/16" means, go to the Wikipedia article about Classless Inter-Domain Routing.

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 (for an explanation of the math, go to the Wikipedia article about Classless Inter-Domain Routing). Currently, you can't change the size of a VPC or its subnets once you create them. If your VPC ends up too small, you must terminate all the instances in the VPC, delete the VPC and its components, and then create a new, larger VPC. For information about deleting a VPC, see Deleting Your VPC (p. 109).

If You Have a Virtual Private Gateway

If you have an IP address prefix in your VPC that overlaps with one of your home networks' prefixes, any traffic to the home network's prefix is dropped. For example, let's say 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

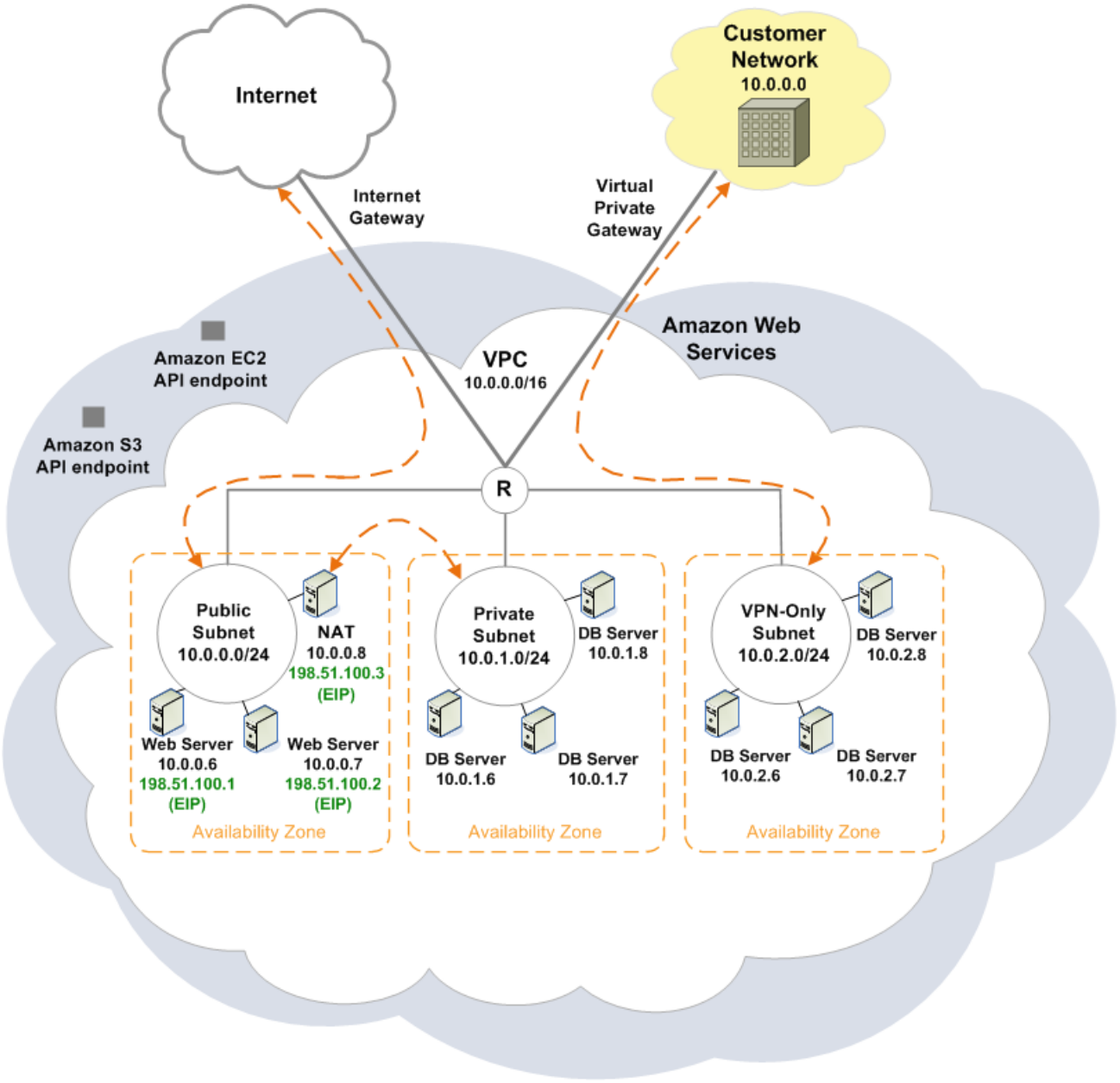
When those instances in the VPC try to talk to hosts in the 10.0.37.0/24 address space, the traffic is dropped because 10.0.37.0/24 is part of the larger prefix assigned to the VPC (10.0.0.0/16). The instances can talk to hosts in the 10.1.38.0/24 space because that block isn't part of 10.0.0.0/16.

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 network.

Subnets in the VPC

You can create a VPC that spans multiple Availability Zones. 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 Availability Zones. Availability Zones are distinct locations that are engineered to be insulated from failures in other Availability Zones. By launching instances in separate Availability Zones, you can protect your applications from failure of a single location. AWS assigns a unique ID to each. Some services (e.g., Elastic Load Balancing, Amazon Relational Database Service, and Auto Scaling) leverage VPC subnets in different ways. For more information about subnet requirements, please see the AWS documentation for the specific service.

The following diagram shows a VPC that has been configured with subnets in multiple Availability Zones.



After creating a VPC, you add one or more subnets. AWS assigns a unique ID to each.

Subnet Sizing

When creating each subnet, you provide the VPC ID, Availability Zone, and the CIDR block for the subnet. The subnet's CIDR block can be the same as the VPC's CIDR (assuming you want only a single subnet in the VPC), or a subset of the VPC's CIDR. If you create more than one subnet in a VPC, the CIDR blocks of the subnets must not overlap.

For example, let's say you create a VPC with CIDR block 10.0.0.0/24, which provides 256 addresses. You break the VPC's CIDR block into two subnets, which means each has 128 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).

Tip

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

Important

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

Important

If you launch an instance in a VPC using an Amazon EBS-backed AMI, the IP address doesn't change if you stop and restart the instance (unlike a similar instance launched outside a VPC, which gets a new IP address when restarted). It's therefore possible to have a subnet with no running instances (they're all stopped), but with no remaining IP addresses available. For more information about Amazon EBS-backed AMIs, go to AMIs in the *Amazon Elastic Compute Cloud User Guide*.

Subnet Routing: Public, Private, VPN-Only

By design, each subnet must be associated with a route table, which specifies the allowed routes for the traffic leaving the subnet. Every new subnet you create is automatically associated with the VPC's main route table. You can change the association, and you can change the contents of the main route table. For more information, see Route Tables (p. 111).

This guide and the wizard in the console typically label subnets as *public*, *private*, or *VPN-only*. Public means that the subnet's traffic is routed to the Internet gateway. You can determine whether a subnet is public by looking at the route table associated with the subnet. If the subnet is public, its route table includes the route shown in the following table. The destination 0.0.0.0/0 means *all traffic*, and the target is the ID for the Internet gateway (e.g., igw-1a2b3c4d).

Destination	Target
0.0.0.0/0	igw-xxxxxxxx

If a subnet is *private*, it doesn't have a route to the Internet gateway. Instead, its Internet-bound traffic is routed to a NAT instance in a public subnet. Its route table includes the route shown in the following table. The target is the ID for the NAT instance (e.g., i-1a2b3c4d).

Destination	Target
0.0.0.0/0	i-xxxxxxxx

If a subnet is labeled in this guide as *VPN-only*, it doesn't have a route to the Internet gateway. Instead, all its traffic is routed to the virtual private gateway. Its route table includes the route shown in the following table. The target is the ID for the virtual private gateway (e.g., vgw-1a2b3c4d).

Destination	Target
0.0.0.0/0	vgw-xxxxxxxx

Subnet Security

By design, each subnet must be associated with a network ACL, which provides subnet-level security for the instances in the subnet. Every new subnet 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. 144).

Adding a Subnet to Your VPC

When you add a new subnet to your VPC, you must set up routing and any security you want for the subnet.

Process for Adding a Subnet

1	Create the subnet. For an example, see Task 1: Create the VPC and Subnets (p. 31) in the discussion of scenario 2.
2	Set up routing for the subnet. By default, the subnet is associated with the VPC's main route table. If that table isn't sufficient, associate the subnet with another route table that contains the routes you want for the subnet. You might have to create a new custom route table with those routes. For more information about route tables, see Route Tables (p. 111).
3	Set up any new security groups you might need for instances in this new subnet. Also update any existing security groups if they need to refer to these new security groups or the new subnet. For more information about security groups, see Security Groups (p. 137).
4	If you use network ACLs in your VPC, set up the network ACL for the subnet. By default, the subnet is associated with the VPC's default network ACL. If that ACL isn't sufficient, associate the subnet with another ACL that contains the rules you want for the subnet. You might have to create a new custom network ACL with those rules. For more information about network ACLs, see Network ACLs (p. 144).

Deleting Your VPC

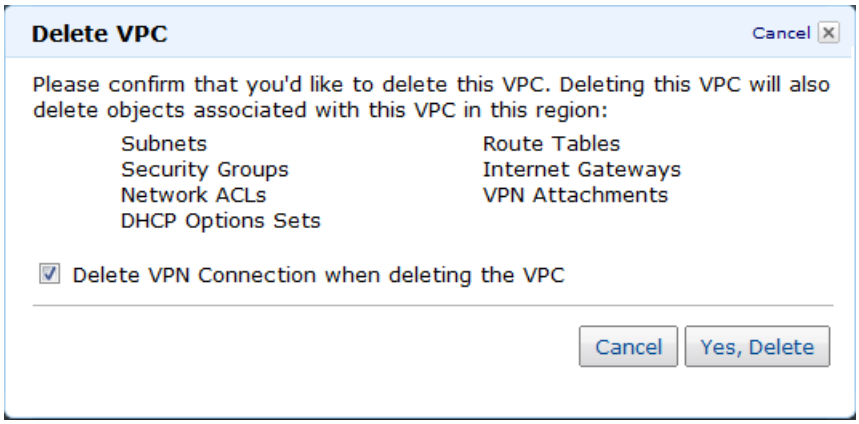
You can delete your VPC at any time (for example, if you decide it's too small). However, be aware that we will also delete all other components related to the VPC. These components include instances, subnets, security groups, network ACLs, route tables, the Internet gateway, and DHCP options. Any VPC Elastic IP addresses you've allocated are not released.

> **Important**
>
> If you have a VPN connection, you don't have to delete it or the other components related to the VPN (customer gateway, virtual private gateway, VPN attachment). If you plan to reuse the customer gateway with another VPC, we recommend you keep the VPN connection and the gateways. Otherwise, your network administrator will need to configure the customer gateway again after you create a new VPN connection.

The Amazon VPC console in the AWS Management Console can do the work to disassemble and delete the VPC for you, assuming you've terminated all the instances first.

To delete your VPC

1. Terminate all instances in the VPC, including any NAT instances.
2. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
3. In the **Navigation** pane, click **Your VPCs**.
4. Right-click the VPC that you want to delete, and then click **Delete**.



5. If you have a VPN connection and want to delete it, select the check box.
6. Click **Yes, Delete**.

We begin deleting your VPC and its components. The dialog box displays the progress.

Routing in Your VPC

Topics

You use the following components to control routing in your VPC:

- Route tables
- Elastic IP addresses
- NAT instances

You only need to use Elastic IP addresses and possibly a NAT instance if you attach an Internet gateway to your VPC and want instances in your VPC to communicate with the Internet.

Route Tables

Topics

This section describes route tables in your VPC and how they work.

Basic Things to Know about Route Tables

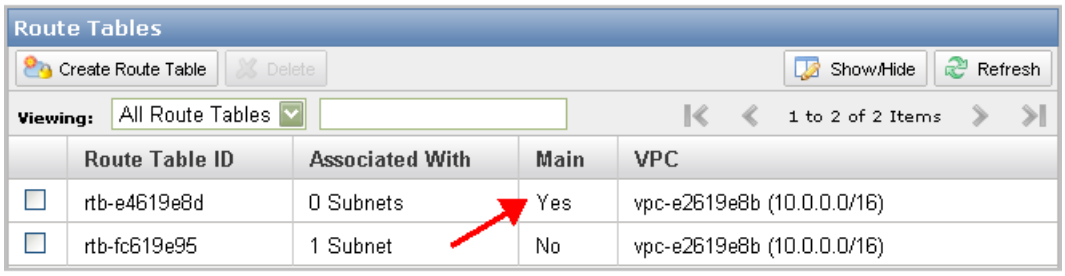
Here are the basic things you need to know about VPC route tables:

- Your VPC has an implicit router (represented by the R enclosed in a circle in the diagrams in this guide).

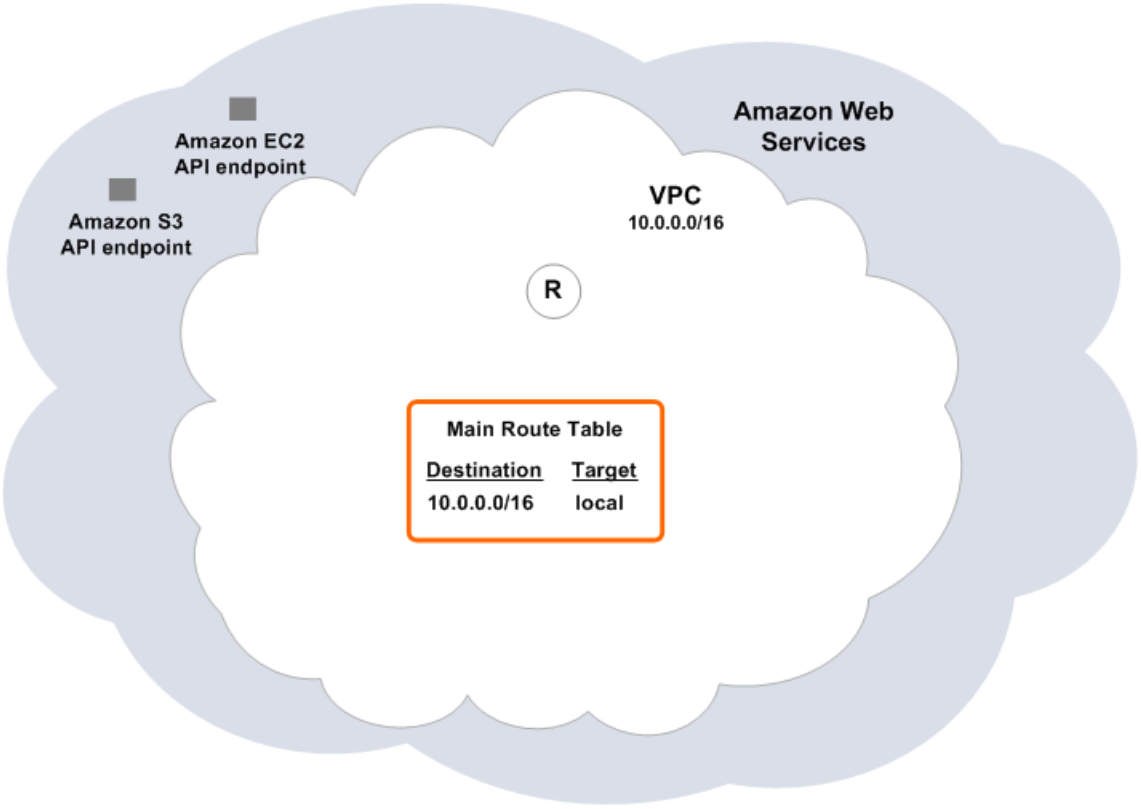
- Your VPC automatically comes with a modifiable main route table.
- You can create other route tables in your VPC (for the limit on the number you can create, see Appendix B: Limits (p. 216)).
- Each subnet must be associated with a route table, which controls the routing for the subnet. If you don't explicitly associate a subnet with a particular table, the subnet uses the main route table.
- You can replace the main route table with a custom table you've created (if you want a different table to be the default table each new subnet is associated with).
- Each route in a table specifies a destination CIDR and a target (e.g., traffic destined for 172.16.0.0/8 is targeted for the virtual private gateway); we use the most specific route that matches the traffic to determine how to route the traffic.

Route Table Details

When you create a VPC, it automatically has a main route table. The following image from the VPC console shows the main route table in the list of route tables for a VPC.



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. The following diagram shows an empty VPC with a main route table.



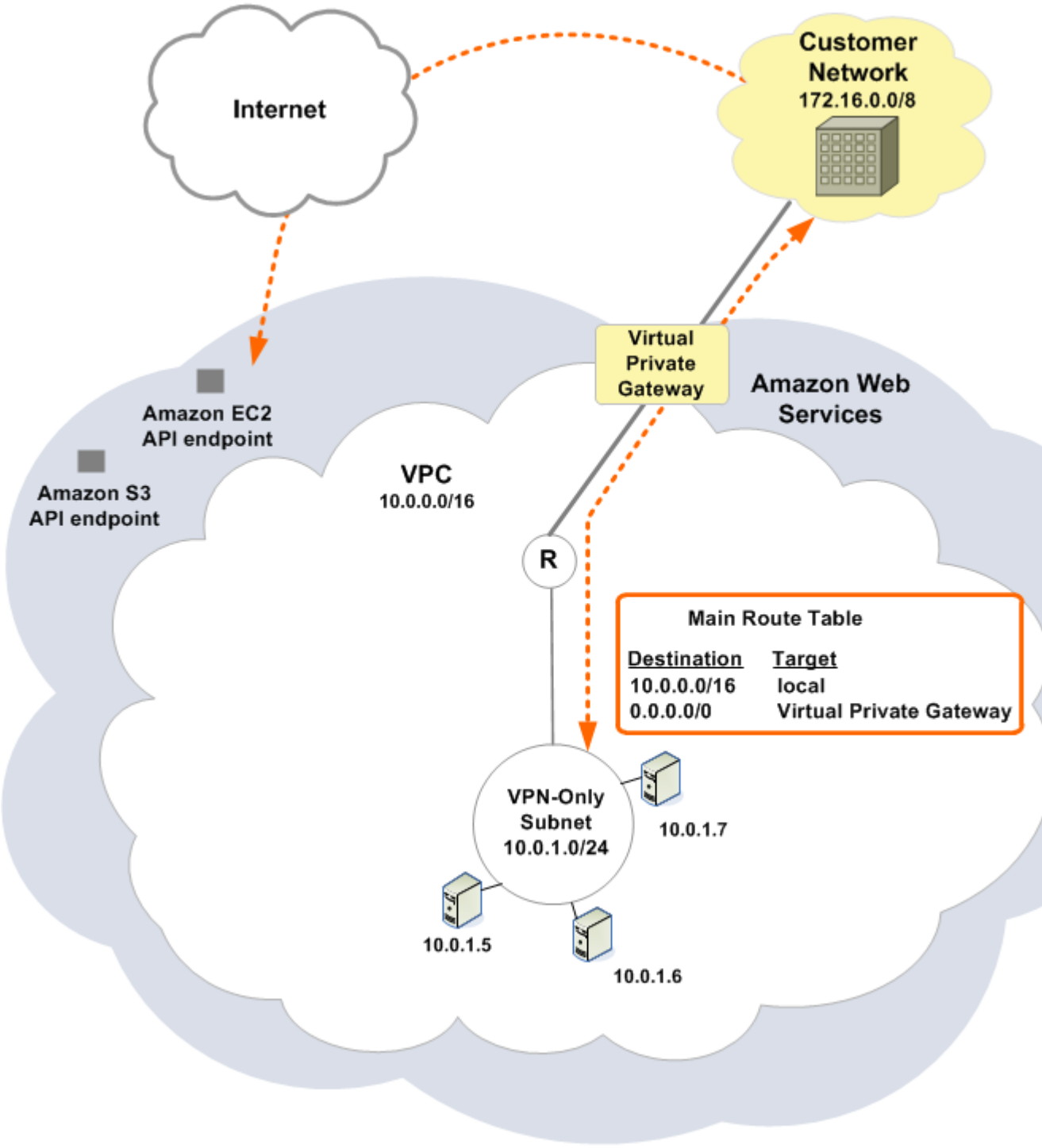
You can't modify the local route in a route table. Whenever you launch an instance in the VPC, the local route automatically covers that instance; you don't need to add the new instance to a route table.

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

If you don't explicitly associate a subnet with a 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. 122)).

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. 119)).

When you add a gateway to the VPC (either an Internet gateway or a virtual private gateway), you must update the route table for any subnet that needs to use that gateway. For example, in the following diagram, you've added a virtual private gateway and a subnet that needs to use that gateway. The subnet uses the main route table by default, so you add a route to the main table that routes all the subnet's traffic to the VPN gateway.



Note

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

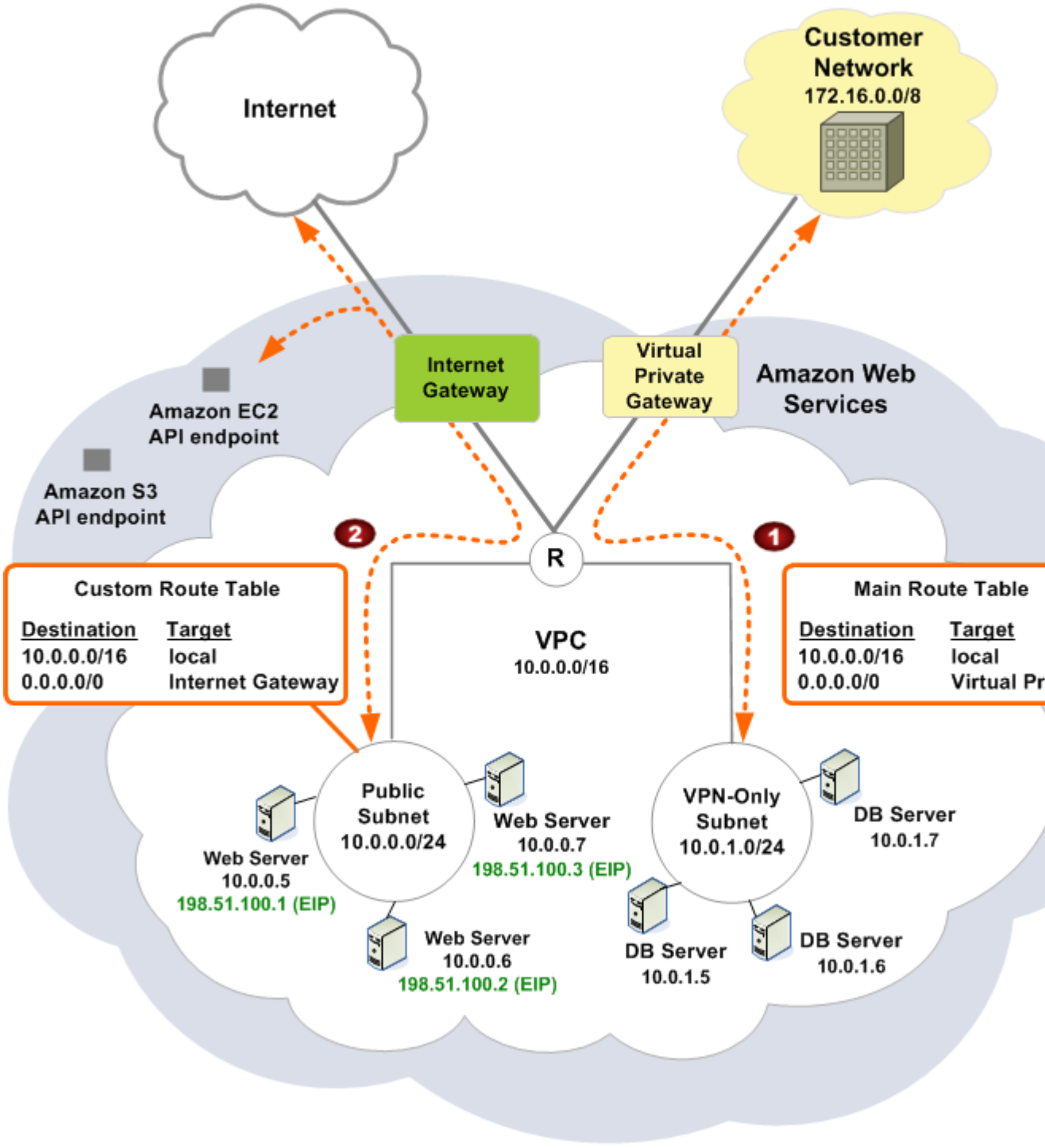
Custom Route Tables

Your VPC can have other route tables 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.

The following diagram shows the routing for the VPC from scenario 3 earlier in this guide (see Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 44)). The VPC has both an Internet gateway and a virtual private gateway, plus a public subnet and a VPN-only subnet. The VPC has a main route table which came with the VPC (labeled 1 in the diagram), and a custom route table that is associated with the public subnet (labeled 2).

The custom route table has a route to cover the public subnet's communication over the Internet gateway (Destination=0.0.0.0/0, and Target=Internet gateway).

The main route table also has a route to cover the VPN-only subnet's communication over the VPN gateway.



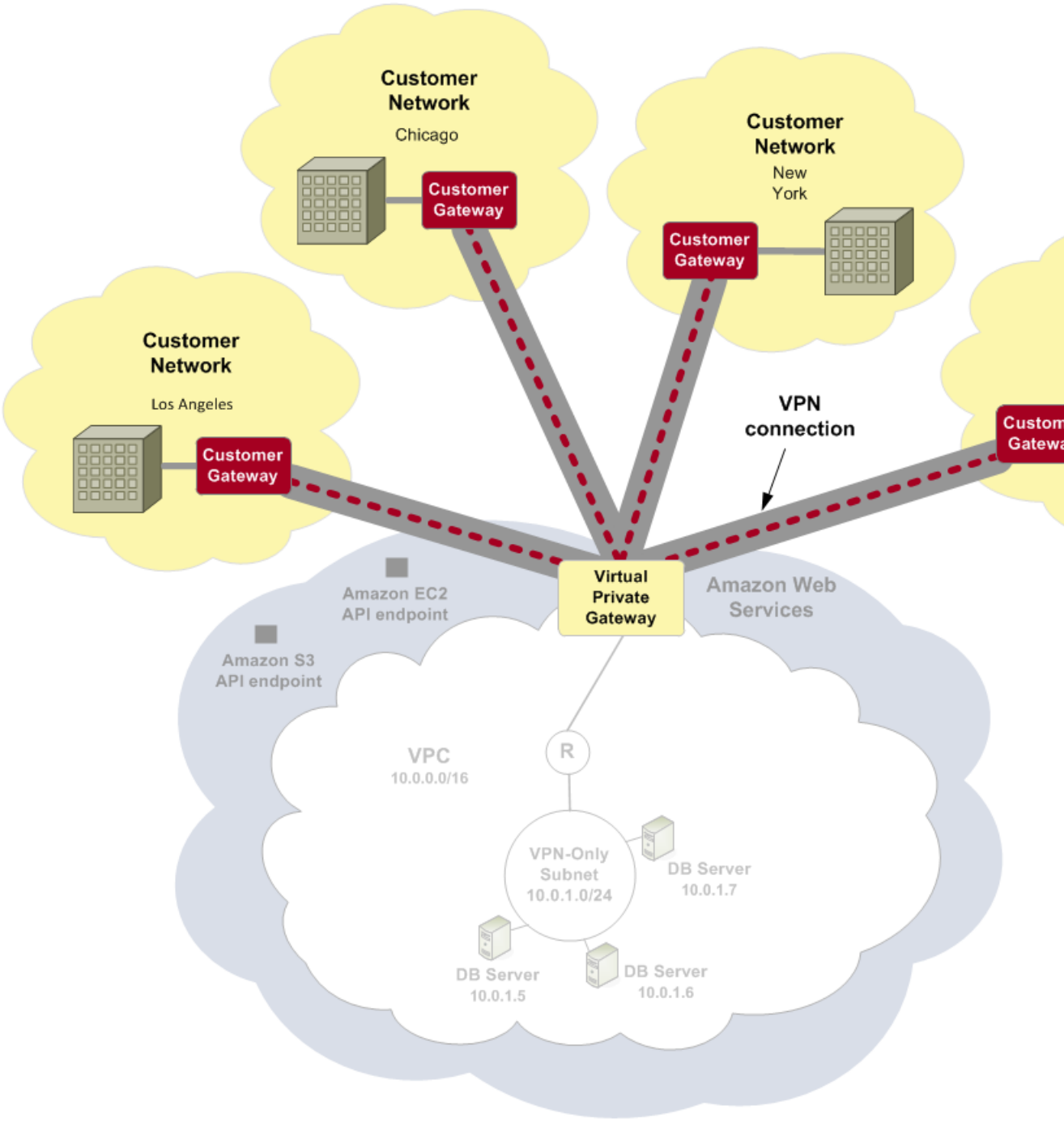
The VPN-only subnet isn't explicitly associated with any route table, so it uses the main route table. This implicit association is indicated in the diagram by an absence of a line between the VPN-only subnet and the table.

The custom route table is explicitly associated with the private subnet, so the table is connected with a line to the subnet in the diagram.

If you were to create a new subnet in this VPC, it would automatically be associated with the main route table, which routes its traffic to the virtual private gateway in this scenario. For the purposes of controlling your subnet's exposure to the Internet, this is the preferred configuration. 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 when you created a new subnet, it would automatically have a route to the Internet gateway.

Adding Multiple VPN Connections

You can add up to ten VPN connections to a single VPC. Multiple VPN connections enable you to establish VPN connections from each of your branch offices to your VPC. For example, if you have offices in Los Angeles, Chicago, New York, and Miami, you can link each of these offices to your VPC. In addition, multiple VPN connections provide for hardware redundancy. You can configure a second Customer Gateway on the same physical network as your first Customer Gateway. If one Customer Gateway needs to be taken down for maintenance, traffic continues to flow with the VPC over the second Customer Gateway.



Working with Route Tables

Topics

This section gives procedures for working with route tables.

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 AWS Management Console.

To determine which route table 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 check box for the subnet.

   Its details are displayed in the lower pane. The ID of the route table the subnet is associated with is included in the details (see the following image). 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. 119).

Determining Which Subnets Are Explicitly Associated with a Table

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

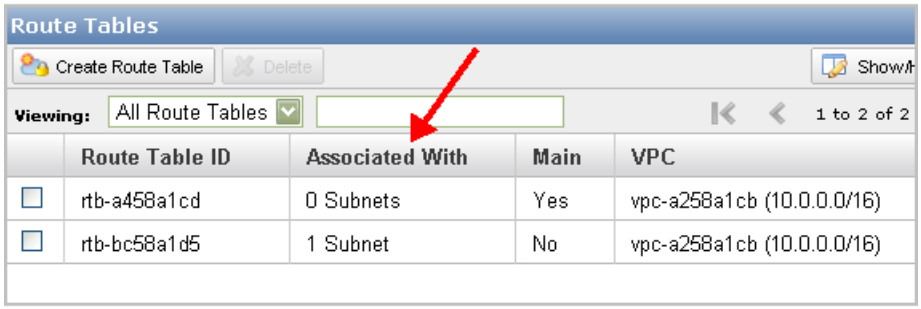
The main route table can have explicit and implicit associations. Custom route tables have only explicit associations.

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. 122)).

To determine how many subnets are explicitly associated

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

2. In the **Navigation** pane, click **Route Tables**.
   Your VPC's route tables are listed. The list includes an **Associated With** column that indicates the number of explicitly associated subnets.



To determine which subnets are explicitly associated

1. Select the check box for the route table of interest.
   Its details are displayed in the lower pane.
2. Click the **Associations** tab.
   The subnets explicitly associated with the table are listed on the tab. The 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. Some of the scenarios presented in this guide include instructions for creating your own route table. For more information, see Task 3: Create a Custom Route Table and Add Routes (p. 32) in scenario 2.

Adding and Removing Routes from a Table

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

Some of the scenarios presented in this guide include instructions for adding routes to route tables. For more information, see Task 3: Create a Custom Route Table and Add Routes (p. 32) in scenario 2.

To delete a route from a 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 check box for the route table.
3. Right-click the route you want to delete, and then click **Delete**.
4. In the **Delete Route Table** dialog box, click **Yes, Delete**.
   The route is deleted from the route table.

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**.
   Your VPC's route tables are listed.
3. Select the check box for the route table.
4. In the lower pane, on the **Associations** tab, select the subnet to associate with the table and click **Associate**.
5. In the **Associate Route Table** dialog box, click **Yes, Associate**.
   The route table is associated with the subnet. The instances in the subnet are now subject to the routes in the table.

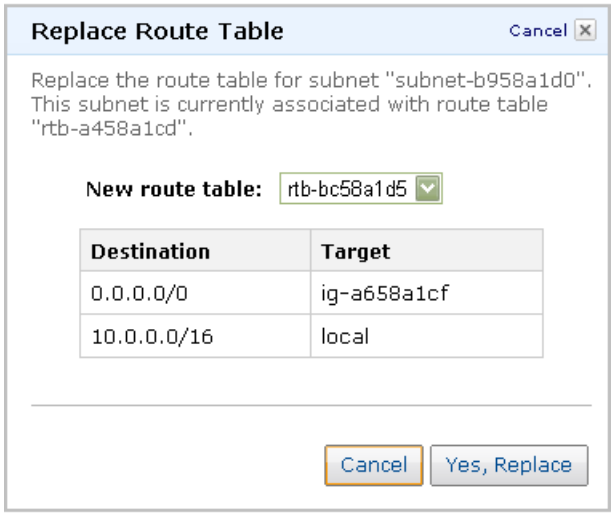
Changing a Subnet's Route Table

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 check box for the subnet.
3. In the lower pane, next to the ID of the route table associated with the subnet, click **Replace**.

The **Replace Route Table** dialog box is displayed.



4. From the drop-down list, select the route table to associate the subnet with and click **Yes, Replace**. The subnet is associated with the route table. The instances in the subnet are now subject to the routes in the new table.

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 implicitly becomes 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**.
3. Select the route table you want to disassociate, and then in the lower pane, click its **Associations** tab.
   On the tab, you can verify that the subnet is currently associated with the table.
4. Click **Disassociate** for the subnet you want to disassociate.
5. In the **Disassociate Route Table** dialog box, click **Yes, Disassociate**.
   The subnet is no longer associated with the route table; it's now implicitly associated with the main route table. You can confirm this association by looking at the subnet's details on the **Subnets** page.

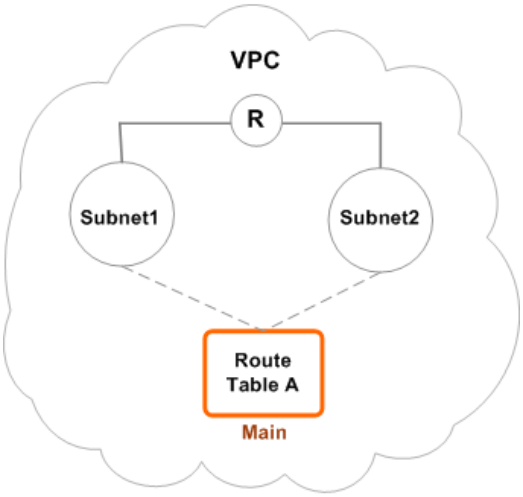
Replacing the Main Route Table

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

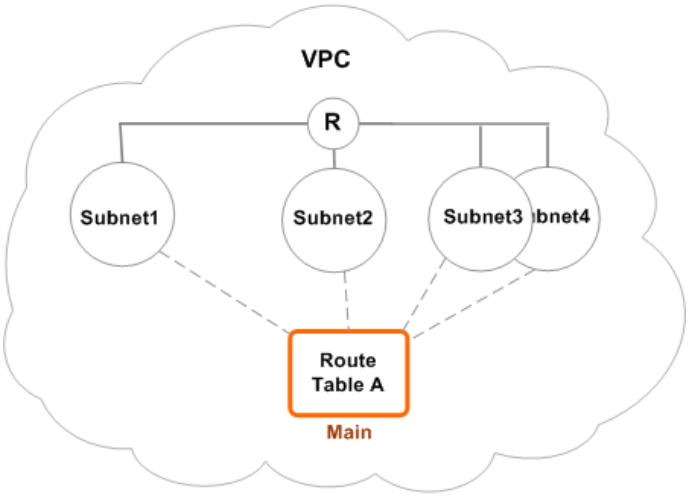
Subnets can be implicitly or explicitly associated with the main route table. Subnets typically won't have an explicit association to the main route table, although it might happen temporarily if you're replacing the main route table.

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 series of diagrams illustrates the process in more detail.

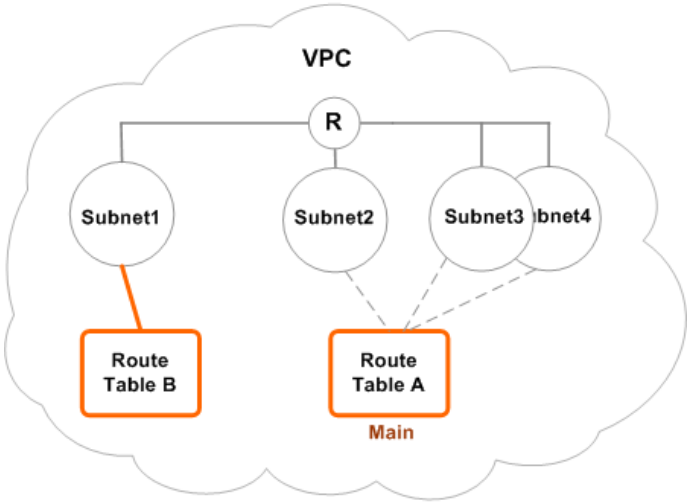
In the first diagram, you have a VPC with two subnets that are implicitly associated with the main route table (Route Table A).



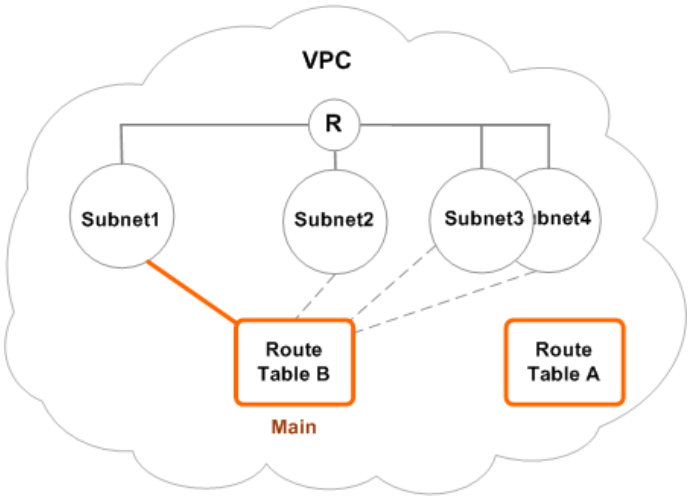
Note that if you add any additional subnets, they are implicitly associated with the main route table by default. The following diagram illustrates that concept with the addition of Subnet3 and Subnet4.



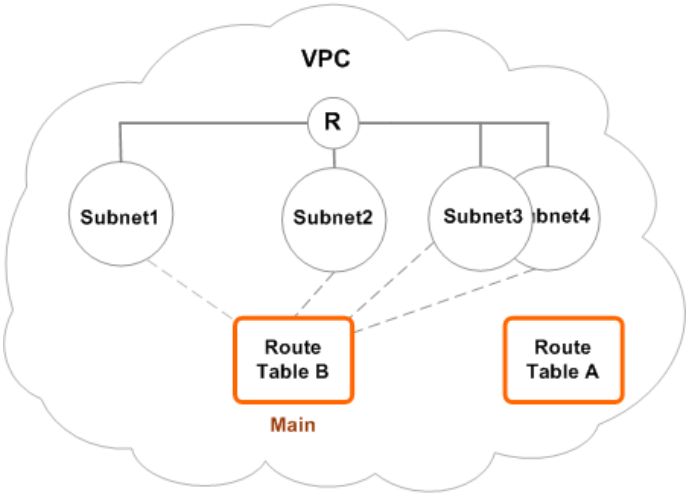
You want to make a change to the main route table, but you want to test it first with Subnet1. So you create a custom route table (Route Table B) with the same routes as Route Table A and explicitly associate Subnet1 with the new table. The following diagram shows Subnet1 explicitly associated with the new route table.



After you've tested the changes, you replace the main route table association with the new custom route table you just tested. After you do this, Route Table B is now the main route table. As shown in the following diagram, Subnet1 still has an explicit association with the new main route table, and the other subnets have implicit associations with it. Route Table A is no longer in use.



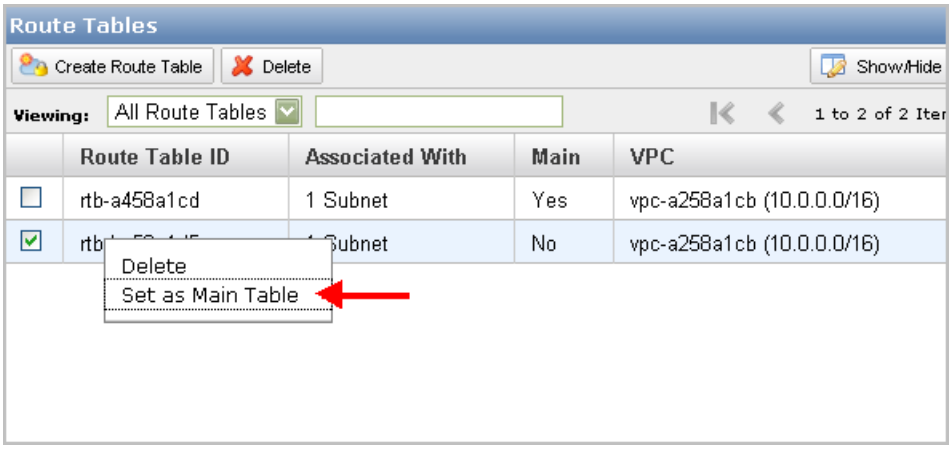
You then disassociate Subnet1 from the new main route table, leaving an implicit association as shown in the following diagram. If you no longer need Route Table A, you can delete it.



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

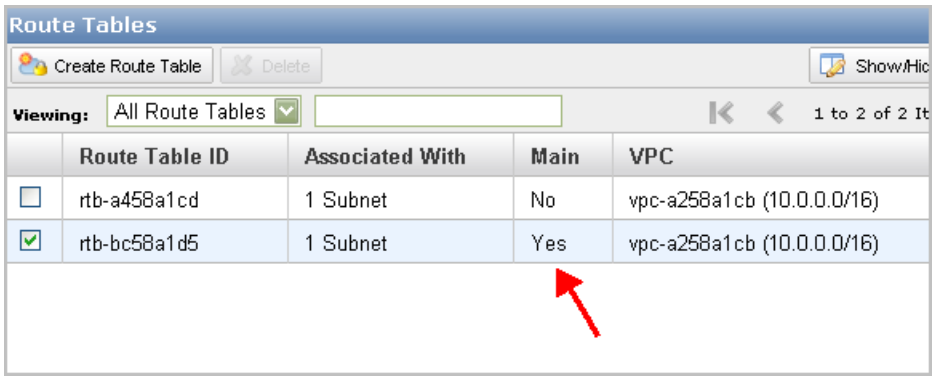
To replace the main route table

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



4. In the **Set Main Route Table** dialog box, click **Yes, Set**.

   The table is now the new main route table. You can confirm this by looking at the table in the list of tables.



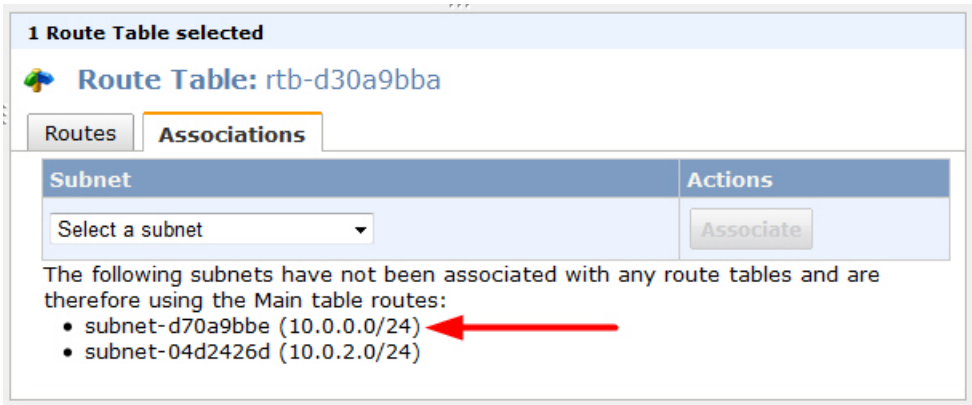
The following procedure describes how to remove an explicit association between a subnet and the main route table. The result is an implicit association between the subnet and the main route table. The process is the same as disassociating any subnet from any route table.

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**.
3. On the **Route Tables** page, select the main route table and click its **Associations** tab.
4. Click **Disassociate**.

5. In the **Disassociate Route Table** dialog box, click **Yes, Disassociate**.

   The subnet is now implicitly associated with the main route table. You can confirm this by refreshing the page and looking at the associations for the table.



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.

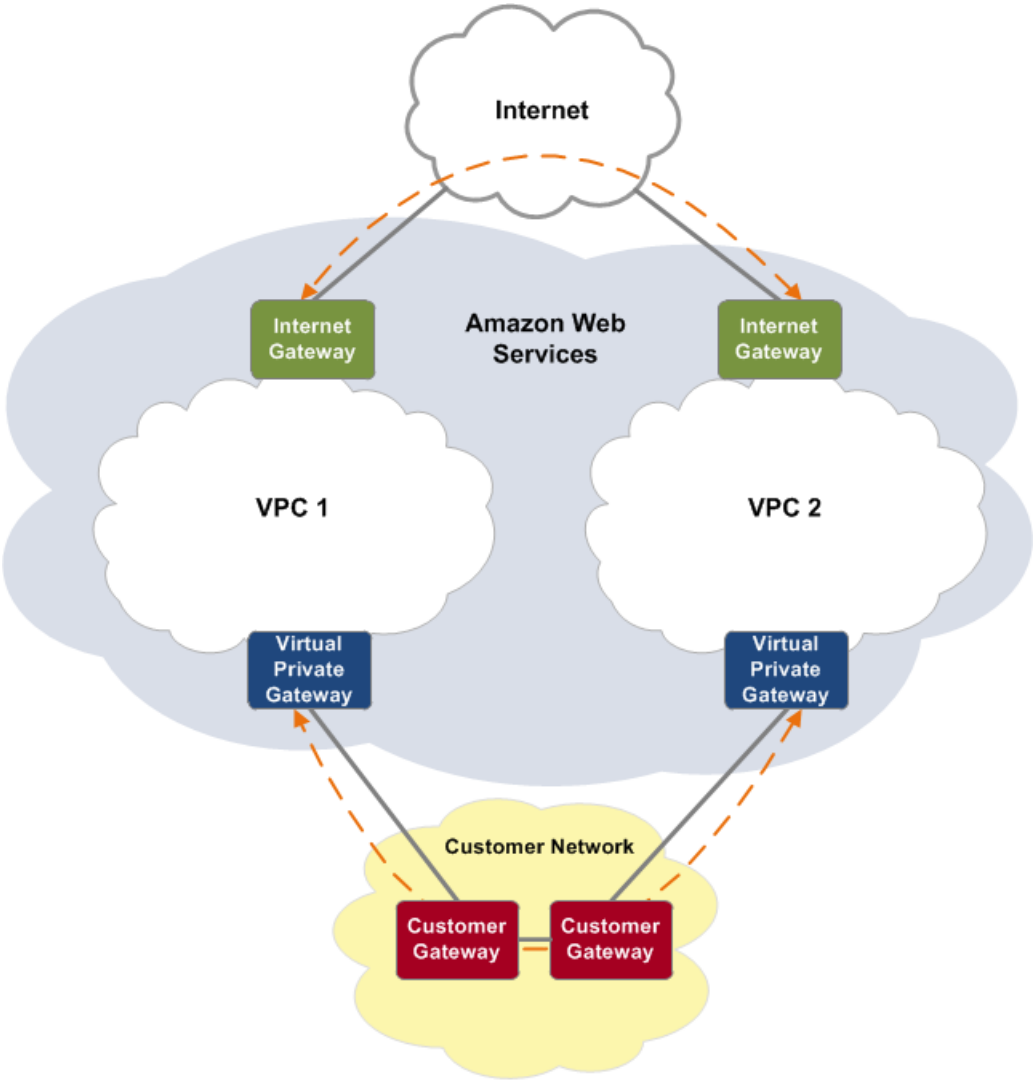
To delete a route table

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Route Tables** page.
3. Select the check box for the route table and click **Delete**.
4. In the **Delete Route Table** dialog box, click **Yes, Delete**.

VPC to VPC Communication

If you're using multiple VPCs, you can enable communication between them through the Internet or through your own virtual private gateways. If you use the virtual private gateway to communicate between the VPCs, you will experience slower connections and will pay standard data transfer rates. It is not possible for VPCs to communicate with each other without using an Internet Gateway or a Virtual Private Gateway.

The following diagram shows how you can send data from VPC to VPC in the same region when those VPCs are not linked internally within AWS.



API and Command Overview

The following table summarizes the available route table commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-create-route-table` `CreateRouteTable`	Creates a custom route table for your VPC.
`ec2-describe-route-tables` `DescribeRouteTables`	Lists the route tables in your VPC.

Command and API Action	Description
`ec2-delete-route-table` `DeleteRouteTable`	Deletes a route table from your VPC.
`ec2-create-route` `CreateRoute`	Adds a new route to a route table.
`ec2-delete-route` `DeleteRoute`	Removes a route from a route table.
`ec2-replace-route` `ReplaceRoute`	Replaces an existing route in a route table (i.e., changes the target for a destination CIDR range specified in the route table).
`ec2-associate-route-table` `AssociateRouteTable`	Associates a subnet with a route table.
`ec2-disassociate-route-table` `DisassociateRouteTable`	Disassociates a subnet from a route table.
`ec2-replace-route-table-association` `ReplaceRouteTableAssociation`	Changes the route table that a subnet is associated with. Also changes which route table is the main route table.

Elastic IP Addresses

Topics

This section applies to you only if you add an Internet gateway to your VPC and want one or more of your Amazon VPC instances to directly communicate with the Internet (including a NAT instance).

VPC instances only have private IP addresses, so if you want an instance to communicate with the Internet, you must allocate an Elastic IP address for use with Amazon VPC and then assign that address to the instance. The address is a static, public IP address that you can assign to any instance or elastic network interface in your VPC. With an Elastic IP address, you can mask an instance failure by rapidly reassigning the address to another instance in your VPC.

Important

If you're already an Amazon EC2 user, you might be familiar with Elastic IP addresses. The Elastic IP addresses you use with instances outside your VPC (i.e., *EC2 addresses*) are not available to use in your VPC. You must allocate a separate set of addresses to use in your VPC (i.e., *VPC addresses*). The two types of addresses differ in their characteristics (see Differences Between EC2 Addresses and VPC Addresses (p. 130)).

You have a separate limit on the number of EC2 addresses and VPC addresses you can have (5 of each type). To request to increase your VPC Elastic IP address limit, submit the Amazon VPC Limits form.

Basic Things to Know about Elastic IP Address

Following are the basic things you need to know about Amazon VPC Elastic IP addresses:

- Any instance that needs to communicate with the Internet (i.e., over the Internet gateway) must have an Elastic IP address associated with it.
- You first allocate an Elastic IP address for your VPCs, and then assign it to an instance in your VPC (it can be assigned to only one instance at a time).
- Elastic IP addresses you use in a VPC are different from ones you use outside a VPC (for a list of the differences, see the next section).
- You can move an Elastic IP address from one instance to another in the same VPC, or in any other VPCs that you are running, but not to instances outside the VPC.
- Any addresses you've allocated to your VPC remain with your VPC until you explicitly release them.
- To ensure efficient use of Elastic IP addresses, we impose a small hourly charge when these IP addresses are not associated with a running instance, or when they are associated with a stopped instance or an unattached network interface. You can associate an Elastic IP address to an elastic network interface (ENI), however, if that ENI is not attached to a running instance, you'll be charged for the Elastic IP address.
- You're limited to 5 VPC Elastic IP addresses; to help conserve them, you can use a NAT instance (see NAT Instances (p. 132)).

Differences Between EC2 Addresses and VPC Addresses

The following table lists the differences between EC2 Elastic IP addresses and those you can use in a VPC.

EC2	VPC
When you allocate an address, it's associated with your AWS account, but for use only outside a VPC.	When you allocate an address, it's associated with your AWS account, but for use only in a VPC.
If you try to associate an address that's already associated with another instance, the address is automatically associated with the new instance.	If you try to associate an address that's already associated with another instance, you can allow reassociation to reassociate the address.
If you stop an instance, its Elastic IP address is unmapped, and you must remap it when you restart the instance.	If you stop an instance, its Elastic IP address stays mapped.
Instances support only a single private IP adddress and a corresponding Elastic IP address.	Instances support multiple IP addresses and each one can have a corresponding Elastic IP address. For more information, see Using Instance IP Addresses in the *Amazon Elastic Compute Cloud User Guide*.

Working with Elastic IP Addresses

Allocating and Associating an Elastic IP Address

Some of the scenarios presented earlier in this guide include instructions for allocating and associating an Elastic IP address. For more information, see Task 8: Allocate and Assign Elastic IP Addresses (p. 43) in scenario 2.

Disassociating an Elastic IP Address

You might want to disassociate an Elastic IP address from the instance it's associated with.

To disassociate an Elastic IP address

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Elastic IPs**.
3. Select the address and click **Disassociate Address**.
   The address is disassociated. You can now either release it or associate it with a different instance in your VPC.

Associating an Address with a Different Instance

To change which instance an Elastic IP address is associated with, you just disassociate the address from the original instance, and then associate the address with the new instance. The instance must be in your VPC.

Releasing an Elastic IP Address

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 address that is allocated for use with Amazon VPC but not associated with an instance.

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Elastic IPs**.
3. Select the address and click **Release Address**.

API and Command Overview

The following table summarizes the available Elastic IP address commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

> **Important**
>
> You use the same set of commands and actions for both EC2 Elastic IP addresses and VPC addresses.

Command and API Action	Description
`ec2-allocate-address` `AllocateAddress`	Acquires an Elastic IP address for use with Amazon VPC.

Command and API Action	Description
`ec2-associate-address` `AssociateAddress`	Associates an Elastic IP address with an instance in your VPC.
`ec2-describe-addresses` `DescribeAddresses`	Lists your Elastic IP addresses (both EC2 addresses and VPC addresses).
`ec2-disassociate-address` `DisassociateAddress`	Disassociates an Elastic IP address from the instance it's associated with.
`ec2-release-address` `ReleaseAddress`	Releases an Elastic IP address from your AWS account. After releasing an Elastic IP address, it is released to the IP address pool and might no longer be available to you.

NAT Instances

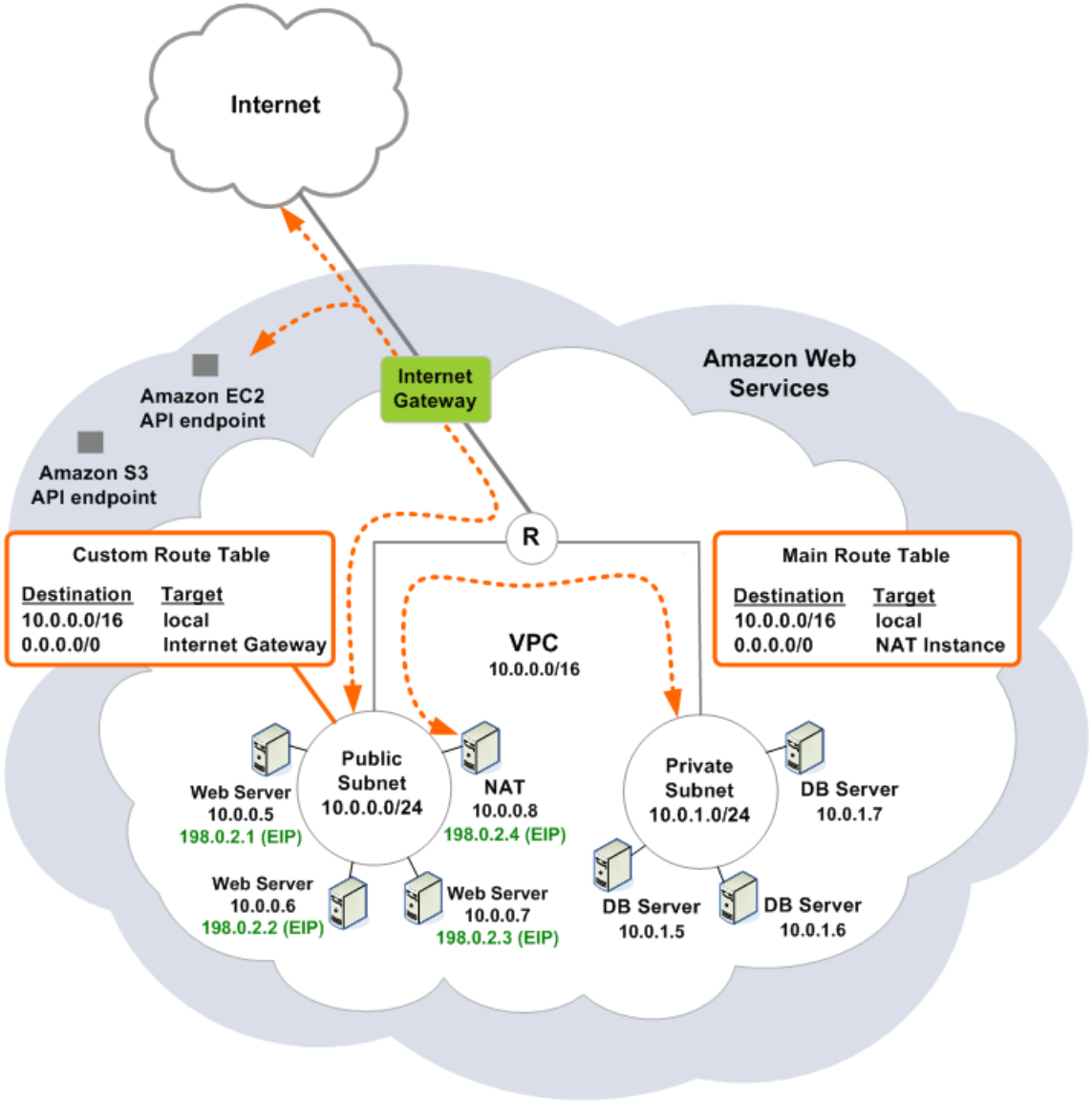
Topics

This guide uses the term NAT instance to refer to an instance that's configured to perform Network Address Translation.

> **Note**
>
> The NAT instance's primary role is actually Port Address Translation (PAT). However, we use the more widely known term *NAT* when referring to the instance. For information about PAT, go to the Wikipedia article about PAT.

You can optionally use a NAT instance in your VPC if you want to enable private instances (those with only a private IP address in a private subnet) to initiate outbound traffic to the Internet, but to keep them from receiving inbound traffic initiated by someone on the Internet. For an example of a VPC with a NAT instance, see Scenario 2: VPC with Public and Private Subnets (p. 16)).

The following figure illustrates the purpose of the NAT instance. The main route table points the traffic from the instances in the private subnet to the NAT instance. The NAT instance forwards the traffic to the Internet gateway so that the source of the traffic appears to be the NAT instance's Elastic IP address. The NAT specifies a high port number for the response; if a response comes back, the NAT knows which instance in the private subnet to forward the response to based on the port number the response came in on.



Amazon provides Amazon Linux AMIs (in 32-bit and 64-bit formats) that have been specially configured to run as NAT instances. The AMI includes the string `ami-vpc-nat` in its name, so you can search for it in the AWS Management Console.

You can log in to a NAT instance and make modifications if you want, and you can create a new AMI from your customized instance (for more information, go to Creating Amazon EBS-Backed AMIs in the *Amazon Elastic Compute Cloud User Guide*.)

Note

The login for Amazon Linux AMIs is ec2-user, and not root.

Alternatively, you can configure your own instance to function as a NAT instance if you'd like.

Required NAT Instance Setup

For your running NAT instance to perform its role in your VPC, you must do the following:

- Disable the `SrcDestCheck` attribute on the instance (see Disabling Source/Destination Checking (p. 134))
- Associate an Elastic IP address with the instance (see Allocating and Associating an Elastic IP Address (p. 131))

Disabling Source/Destination Checking

For a NAT instance to perform network address translation, you must disable source/destination checking on the instance. Each EC2 instance performs source and destination checking by default. This means the instance must be the source or destination of any traffic it sends or receives. However, the NAT instance needs to be able to send and receive traffic where the source or destination is not itself. To enable that behavior, you must disable source/destination checking on the NAT instance.

Tip

Option #2 in the VPC creation wizard (which creates a VPC with public and private subnets) automatically launches a NAT instance for you and disables source/destination checking on that instance.

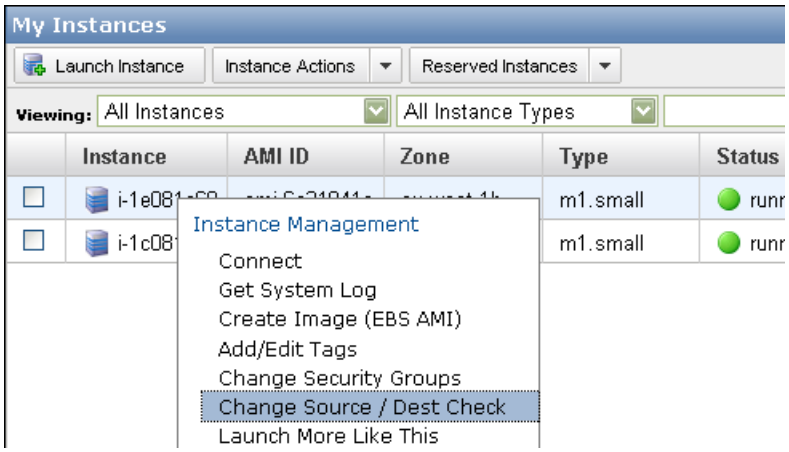
The following procedure explains how to disable the `SrcDestCheck` attribute on an instance. We assume you've already launched the NAT instance and it is either running or stopped.

To disable source/destination checking on the NAT instance

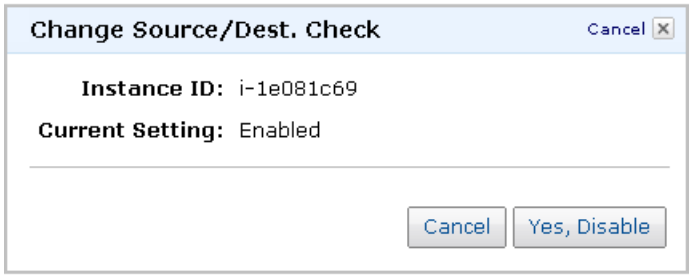
Note

This procedure only works for EC2 instances that are running within a VPC.

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the NAT instance in the list of instances, and select **Change Source / Dest Check**.



The **Change Source/Dest. Check** dialog box opens.



For a regular instance, the value should be *Enabled*, indicating that the instance is performing source/destination checking. For a NAT instance, you want the value to be *Disabled*.

4. Click **Yes, Disable**.

The attribute is disabled. You can enable the attribute at any time using the same basic procedure.

API and Command Overview

The following table summarizes the available commands related to NAT instances and the corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-modify-instance-attribute` `ModifyInstanceAttribute`	Enables or disables the `SrcDestCheck` attribute of an instance, which enables it to perform Network Address Translation (NAT).

Security in Your VPC

Topics

Amazon VPC offers two ways to help provide security for your VPC:

- **Security groups—**Act as an instance firewall, controlling ingress and egress for one or more instances
- **Optional network Access Control Lists (ACLs)—**Act as a subnet firewall, controlling ingress and egress for an entire subnet (as a second layer of defense on top of security groups)

> **Important**
>
> Security groups are a basic Amazon EC2 concept. VPC security groups have different capabilities than EC2 security groups (see EC2 vs. VPC Security Groups (p. 139)).

You can secure your VPC instances using only security groups; however, you might want to use both and take advantage of the additional security that network ACLs provide.

A security group acts as a firewall that controls the traffic allowed in and out of a group of instances. When you launch an instance in a VPC, you can assign the instance to one or more VPC security groups that you've created. The groups act at the instance level, not the subnet level. Therefore, each instance in a subnet in your VPC could belong to a different set of security groups. If you don't specify a particular group at launch time, the instance automatically belongs to the VPC's *default security group*.

Whoever launches the instance must carefully specify which security group or groups the instance should go in. If the person makes a mistake, the instance might be vulnerable. However, if you've created a network ACL that mirrors the desired security group rules, that ACL can provide a second layer of security and protect the instance.

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 VPC Management (p. 198).

Comparison of Security Groups and Network ACLs

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

Security Groups	Network ACLs
They operate at the instance level (i.e., act as an instance firewall—the first layer of defense)	They operate at the subnet level (i.e., act as a subnet firewall—the second layer of defense)
You can specify *allow* rules only, for both ingress and egress	You can specify *allow* rules and *deny* rules, for both ingress and egress
They are stateful: Return traffic is automatically allowed regardless of any security group rules	They are stateless: Return traffic must be explicitly allowed by network ACL rules
We evaluate all applicable rules before deciding whether to allow the traffic in question	We process the ACL's rules in line-number order when deciding whether to allow the traffic in question
A given security group is applicable to an instance only if the person launching the instance specifies the security group in the launch request, or moves the instance into the group (in other words, you must rely on the person to correctly specify the security group)	A given network ACL is automatically applicable to all instances in any subnet it's associated with (in other words, you don't have to rely on the person launching the instance to correctly specify the security group; the ACL therefore acts as a backup layer of defense on top of a security group)

Note

Amazon security groups and network ACLs do not filter traffic to or from link-local addresses (169.254.0.0/16). Link-local addresses in the Amazon VPC support the following services: Domain Name Services (DNS), Dynamic Host Configuration Protocol (DHCP), Amazon EC2 instance-specific metadata, and Key Management Server (KMS—license management for Windows instances). You can implement third party or operating system vendor firewall solutions in your Amazon EC2 instances to block network communication with link-local addresses.

Security Groups

Topics

A security group acts as a firewall that controls the traffic allowed in and out of a group of instances. When you launch an instance in a VPC, you can assign the instance to up to five VPC security groups. The groups act at the instance level, not the subnet level. Therefore, each instance in a subnet in your VPC

could belong to a different set of security groups. If you don't specify a particular group at launch time, the instance automatically belongs to the VPC's *default security group*.

For each group, you add rules that govern the allowed inbound traffic to instances in the group, and a separate set of rules that govern the allowed outbound traffic. This section describes the basics things you need to know about VPC security groups and their rules.

Basic Things to Know about VPC Security Groups

These are the basic characteristics of VPC security groups:

- You can specify *allow* rules, but not *deny* rules.
- You can specify inbound rules and separate outbound rules.
- By default, no ingress is allowed into a security group until you add inbound rules to the group.
- By default, all egress is allowed from the security group until you add outbound rules to the group (then only the egress you specified is allowed).
- Responses to allowed inbound traffic are allowed to egress regardless of outbound rules, and vice versa (security groups are therefore *stateful*).
- Instances in a group can't talk to each other unless you add rules allowing it (exception: instances in the default security group have these rules by default).
- After you launch an instance, you can change which security groups the instance is in.

Your VPC's Default Security Group

Your VPC automatically comes with a *default security group*. All instances in your VPC automatically belong to this group if you don't specify a different security group at instance launch time. These are the default settings for this group:

- Allow no inbound traffic from outside the group
- Allow all outbound traffic from the instances in the group
- Allow all instances in the group to talk to each other (i.e., allow all inbound and outbound traffic between the instances in the group)

You can change the rules for the default security group.

Basic Things to Know about Security Group Rules

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

- For inbound rules only: The source of the traffic (CIDR range or security group) and the destination (i.e., listening) port or port range
- For outbound rules only: The destination (CIDR range or security group) and the destination port or port range
- You can specify 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 add or remove rules from a group, the rules are automatically applied to all instances in the group.

Note

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

Following is an example of the rules for a security group.

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
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
DBServerSG	TCP	1433	Allow outbound MS SQL access to instances in the group called DBServerSG
0.0.0.0/0	TCP	80	Allow outbound HTTP access to servers on the Internet (e.g., for software updates)
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to servers on the Internet (e.g., for software updates)

EC2 vs. VPC Security Groups

If you're already an EC2 user, you might be familiar with security groups. The security groups you've created for EC2 (i.e., *EC2 security groups*) are not available to use in your VPC. You must create a separate set of security groups to use in your VPC (i.e., *VPC security groups*). The rules you create for a VPC security group can't reference a EC2 security group in your account, and vice versa. Also, VPC security groups have additional capabilities not available to EC2 security groups. The differences between the two types of groups are described in the following table.

EC2	**VPC**
Groups control ingress only.	Groups control both ingress and egress.
Groups allow access from security groups in your AWS account or other accounts.	Groups allow access from other security groups in your VPC only.
After an instance is launched, you can't change which groups it's in.	After an instance is launched, you can change which groups it's in.
When you add a rule to a group, you don't have to specify a protocol, and only TCP, UDP, or ICMP are available.	When you add a rule to a group, you must specify a protocol, and it can be any protocol with a standard protocol number, or all protocols (see Protocol Numbers).

EC2	VPC
When you add a rule to a group, you must specify port numbers (for TCP or UDP).	When you add a rule to a 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

Topics

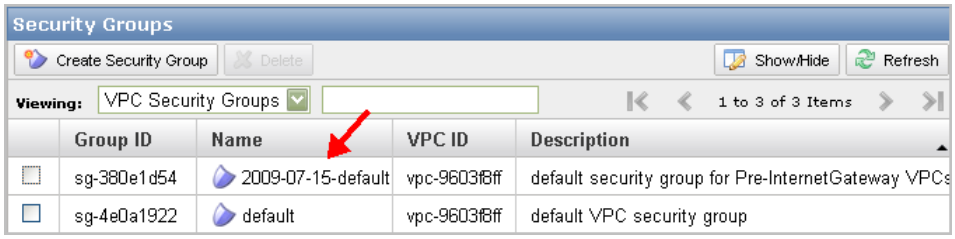
This section gives procedures for working with security groups.

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. One of the scenarios presented in this guide includes instructions for modifying the default security group. The procedure is the same as modifying any other security group. For more information, see Task 5: Update the Default Security Group (p. 98) in scenario 4.

Deleting the Legacy Security Group

We introduced security groups to Amazon VPC with the 2011-01-01 release of Amazon VPC. At the time of the release, each existing VPC automatically received a legacy security group called *2009-07-15-default*, and all existing instances in the VPC were automatically put in this group. This group's rules permit all traffic in each direction so that the existing instances in the VPC are not affected by being placed in this group. The following image shows the legacy 2009-07-15-default group in the first row, and the VPC's default group in the second row.



If you have a VPC with a legacy security group, before you can attach an Internet gateway to the VPC, you must move the existing instances to a different security group of your choice and delete the legacy group. The group you move the instances to can be the default group or one you create. This requirement forces you to consider the security group rules you want to apply to your existing instances before you expose your VPC to the Internet.

For more information, see Adding an Internet Gateway to Your VPC (p. 156).

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. Several of the scenarios presented in this guide include instructions for creating your own security groups. For more information, see Task 6: Create Security Groups and Add Rules (p. 38) in scenario 2.

Adding and Removing Rules

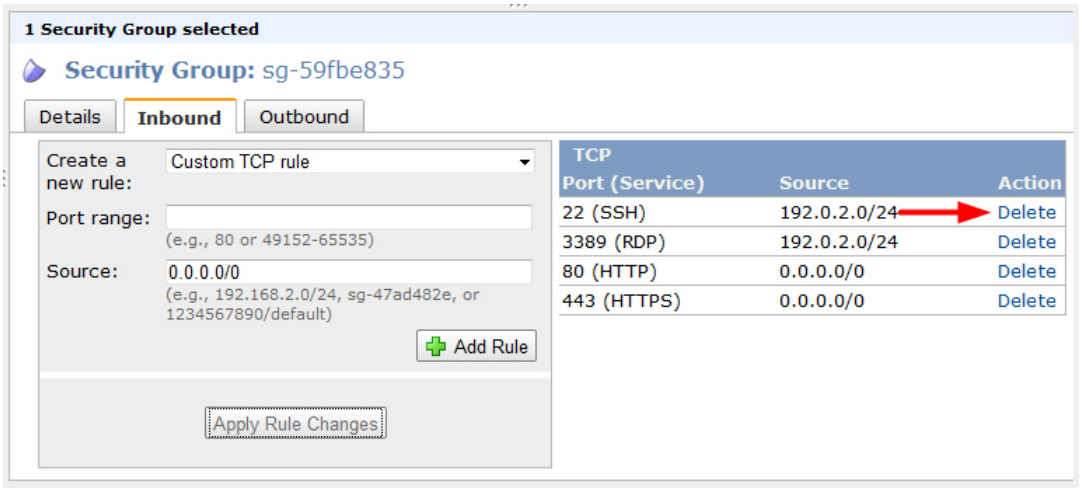
When you add or remove a rule, any instances already in the group are subject to the change. You can't modify rules; you can only add and delete rules.

Several of the scenarios presented in this guide include instructions for adding rules to security groups. For more information, see Task 6: Create Security Groups and Add Rules (p. 38) in scenario 2.

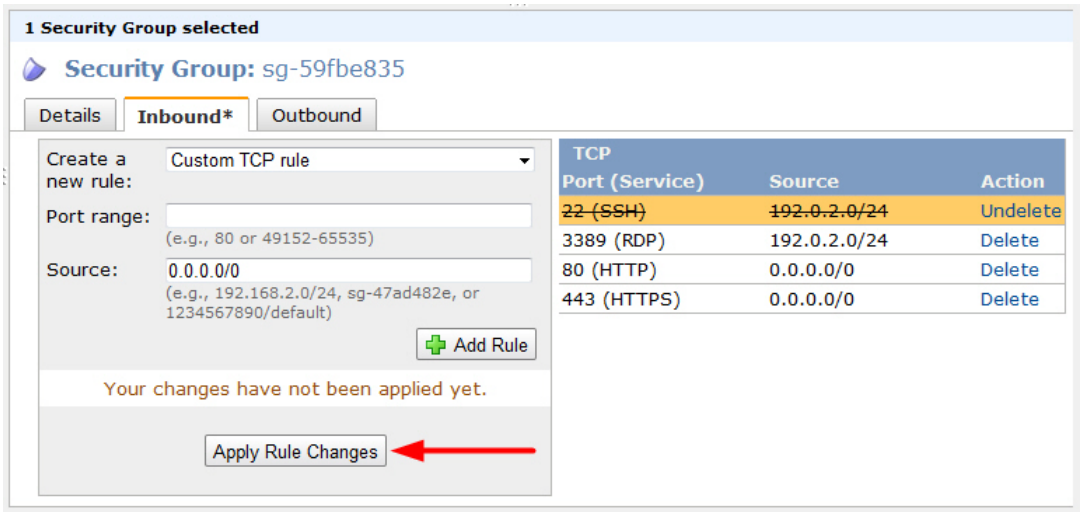
> **Note**
>
> When adding and deleting security group rules using the Amazon VPC console in the AWS Management Console, your changes don't take effect until you click the **Apply Rule Changes** button.

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. In the lower pane, select the check box for the security group.
   Its details are displayed in the lower pane.
4. For the rule you want to delete, click **Delete**.



5. Click **Apply Rule Changes** to delete the rule.



The rule is deleted from the security group. The change affects any instances in the group.

Deleting a Security Group

You can delete a security group only if there are no instances in the group (either running or stopped). You can move the instances into another security group first if you want (see Changing an Instance's Security Groups (p. 142)).

To delete 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. Select the check box for the security group and click **Delete**.
4. In the **Delete Security Group** dialog box, click **Yes, Delete**.

The security group is deleted.

Changing an Instance's Security Groups

You can change an instance's VPC security group membership after the instance is launched. When you make the change, the instance can be either running or stopped.

> **Note**
>
> An instance can be in a maximum of five VPC security groups.

To change an instance's group membership

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the instance that you want to change in the list of instances, and select **Change Security Groups**.

4. In the **Change Security Groups** dialog box, on the **Security Groups** list box, select one or more security groups, and then click **Yes, Change**.

The new list of groups you selected replaces the instance's original list of groups.

API and Command Overview

The following table summarizes the available security group commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Important

You use the same set of commands and actions for both the EC2 security groups and VPC security groups. You must provide your VPC ID when referring to VPC security groups.

Command and API Action	Description
`ec2-add-group` `CreateSecurityGroup`	Creates a new security group.

`ec2-authorize`
`AuthorizeSecurityGroupIngress`
`AuthorizeSecurityGroupEgress`
	Adds rules to a security group.

`ec2-describe-group` `DescribeSecurityGroups`	Returns information about security groups.

`ec2-revoke`
`RevokeSecurityGroupIngress`
`RevokeSecurityGroupEgress`
	Removes rules from a security group.

`ec2-delete-group` `DeleteSecurityGroup`	Deletes a security group.
`ec2-modify-instance-attribute` `ModifyInstanceAttribute`	Changes the security groups an instance belongs to.

Network ACLs

Topics

A network 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 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. 137).

Basic Things to Know about Network ACLs

Following are the basic things you need to know about network ACLs:

- A network ACL is a numbered list of rules that Amazon VPC evaluates in order starting with the lowest numbered rule to determine whether traffic is allowed in or out of any subnet associated with the ACL
- A network ACL has inbound rules and separate 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 ingress and egress in your VPC
- You can create optional custom network ACLs; each custom ACL starts out closed (i.e., permits no traffic) until you add rules to change that behavior
- Each subnet must be associated with an ACL; if you don't associate a subnet with a particular ACL, the subnet is automatically associated with the default ACL
- ACLs are *stateless*: responses to allowed ingress traffic are subject to the ACL's egress rules (and vice versa)

For information about the number of network ACLs you can create, see Appendix B: Limits (p. 216).

Basic Things to Know about Network ACL Rules

You have the option to add or remove rules from the default ACL or create new ACLs for your VPC. When you add or remove rules from an ACL, the changes are automatically applied to the subnets associated with the ACL.

Following are the parts of a network ACL rule:

- Rule number
- A protocol: You can specify 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.
- For inbound rules only: The source of the traffic (CIDR range) and the destination (i.e., listening) port or port range
- For outbound rules only: The destination (CIDR range) and the destination port or port range
- Choice of allow or deny

To help you understand what ACL rules look like, here's what the default network ACL looks like in its initial state.

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

The default network ACL is configured to allow all traffic to flow in and out of each subnet. In other words, the default ACL doesn't block any network traffic.

Every ACL automatically includes a final 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 from any ACL.

Note

You can have up to 32,766 rules in an ACL. However, we recommend you condense your rules as much as possible for easier management.

Example Custom Network ACL

The following table shows an example 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 information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 147).

The 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 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 (i.e., for requests that originate in the subnet) For information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 147).
*	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 (e.g., serving web pages to people visiting the web servers in the subnet). For information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 147).
*	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 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 32768-61000. Windows operating systems through Windows Server 2003 use 1025-5000. Windows Server 2008 uses 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 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, etc.).

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

Topics

This section gives procedures for working with network ACLs.

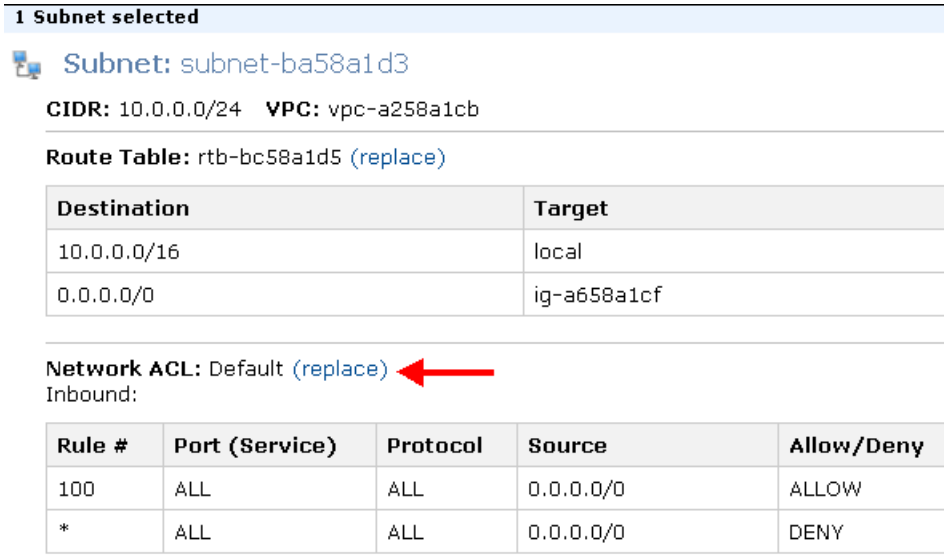
Determining Which Network ACL a Subnet Is Associated With

You can determine which network ACL a subnet is associated with by looking at the subnet's details in the AWS Management Console.

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 check box for the subnet.

   Its details are displayed in the lower pane. The network ACL associated with the subnet is included in the details, along with the ACL's rules.



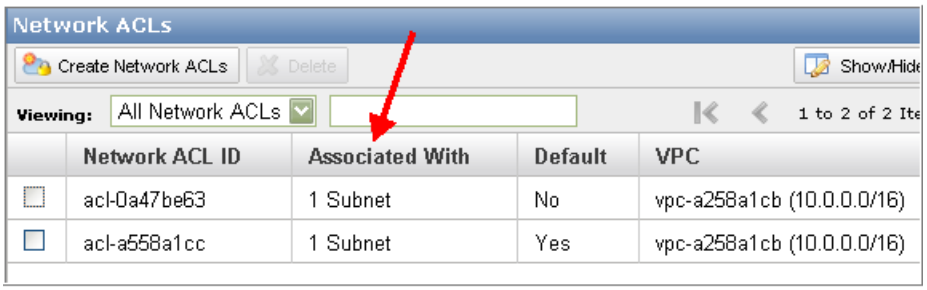
Determining Which Subnets Are Associated with an ACL

You can determine how many and which subnets are associated with a network ACL.

To determine how many subnets are associated

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

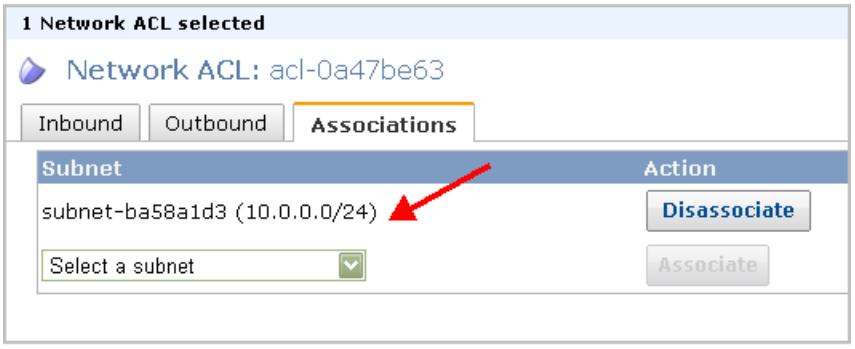
Your VPC's ACLs are listed. The list includes an **Associated With** column that indicates the number of associated subnets.



To determine which subnets are associated

1. Select the check box for the network ACL.
   Its details are displayed in the lower pane.
2. Click the **Associations** tab.

   The subnets associated with the network ACL are listed on the tab.



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**, and then click **Create Network ACL**.
3. In the **Create Network ACL** dialog box, select your VPC's ID from the **VPC** menu, and then click **Yes, Create**.
   The ACL is created and appears on the **Network ACLs** page. Notice that it has an ID (e.g., acl-xxxxxxxx).

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

No subnets are yet associated with your 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.

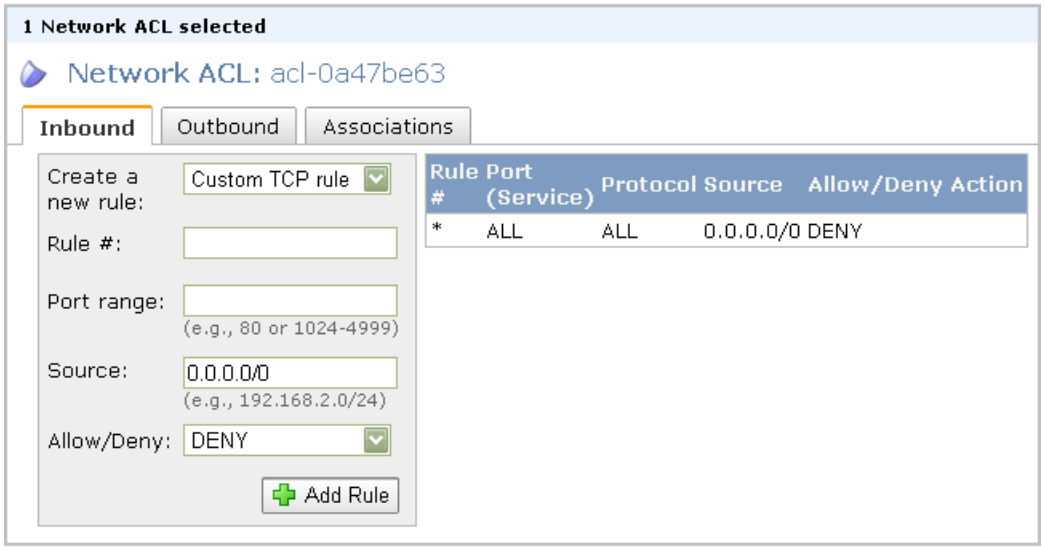
You can't modify rules; you can only add and delete rules. If you need to change the order of a rule in the ACL, you must add a new rule with the new rule number, and then delete the original rule.

Tip

The process for adding a rule to a network ACL is very similar to adding a rule to a security group, except that you must provide a rule number and whether the ACL rule should allow or deny the specified traffic. Also, there's no **Apply Rule Changes** button that you must click as there is for security groups.

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 list of ACLs, select the check box for the ACL you want to add a rule to.
4. In the lower pane, select either the **Inbound** or **Outbound** tab depending on the type of rule you want to add.



5. From the **Create a new rule** drop-down list, select an option that fits the rule you want to add. For example, if you want to add a rule for HTTP, select the **HTTP** option. If you want to add a rule to allow all TCP traffic, select **All TCP**. For some of these options (e.g., HTTP), the console automatically fills in the port for you. If you want to use a protocol not listed in the menu, select **Custom protocol rule**.
6. Provide the rule's details:

   a. In the **Rule #** field, enter a number that specifies where the rule should appear in the list of rules (e.g., 110). The rule number must not already be used in the ACL. We process the rules in order starting with the lowest number.

Important

We recommend that you leave room between the rule numbers (e.g., 100, 110, 120, etc.), and not number them one right after the other (e.g., 101, 102, 103, etc.). This allows you to easily add a new rule between existing ones without having to renumber the rules.

b. If you're creating a custom protocol rule, enter the protocol's number or name (e.g., 47 or GRE) in the **Protocol** field. For a list, go to IANA List of Protocol Numbers.

c. If the protocol you've selected requires a port number, enter the port number or port range separated by a hyphen (e.g., 49152-65535).

d. In the **Source** or **Destination** field (depending on whether this is an inbound or outbound rule), enter the CIDR range you want the rule to apply to (e.g., 172.16.0.0/8).

7. From the **Allow/Deny** menu, select **ALLOW** if you want the rule to allow the specified traffic, or select **DENY** if you want the rule to deny the specified traffic.
8. Click **Add Rule**.
   The list of the rules on the right updates to include the new rule.

You've added a rule to the ACL.

To delete a rule from a network ACL

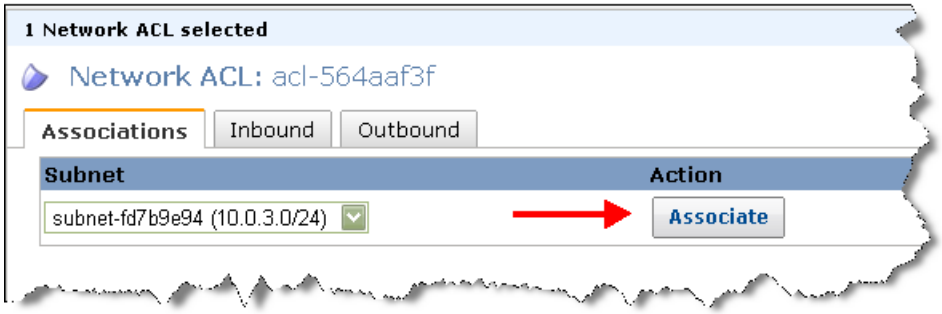
1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Network ACLs**.
3. In the list of ACLs, select the check box for the ACL you want to delete a rule from.
4. In the lower pane, select either the **Inbound** or **Outbound** tab, and then click **Delete**.
5. In the **Delete Network ACL Rule** dialog box, click **Yes, Delete**.

Associating a Subnet with a Network ACL

To apply a network ACL's rules to a particular subnet, you must associate the subnet with the ACL. An ACL can be associated with multiple subnets; however, a subnet can be associated with only one 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 check box for the network ACL.
3. In the lower pane, on the **Associations** tab, select the subnet to associate with the table, and then click **Associate**.



4. In the **Associate Network ACL** dialog box, click **Yes, Associate**.

The subnet is now associated with the ACL and is subject to the ACL's rules.

Disassociating a Network ACL from a Subnet

You might want to disassociate a subnet from its ACL. For example, you might have a subnet that is associated with a custom ACL, and you instead want it associated with the default network ACL. By disassociating the subnet from the custom 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**.
3. Select the ACL you want to disassociate, and then in the lower pane, click its **Associations** tab. On the tab, you can verify the association with the subnet.
4. Click **Disassociate**.
5. In the **Disassociate Network ACL** dialog box, click **Yes, Disassociate**.
   The subnet is no longer associated with the ACL, and is instead associated with the default network ACL. You can confirm this association by looking at the subnet's details on the **Subnets** page.

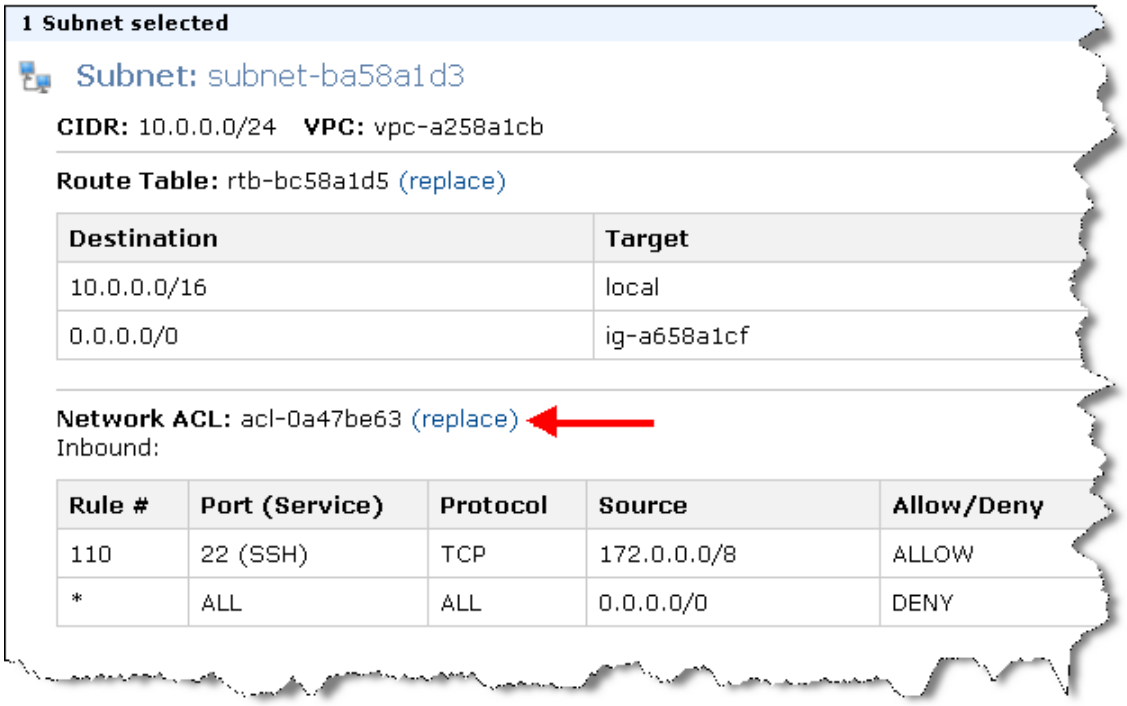
Changing a Subnet's Network ACL

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 ACL you've created.

After changing a subnet's 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 check box for the subnet.
3. In the lower pane, next to the ID of the network ACL associated with the subnet, click **Replace**.



1 Subnet selected

Subnet: subnet-ba58a1d3

CIDR: 10.0.0.0/24 **VPC:** vpc-a258a1cb

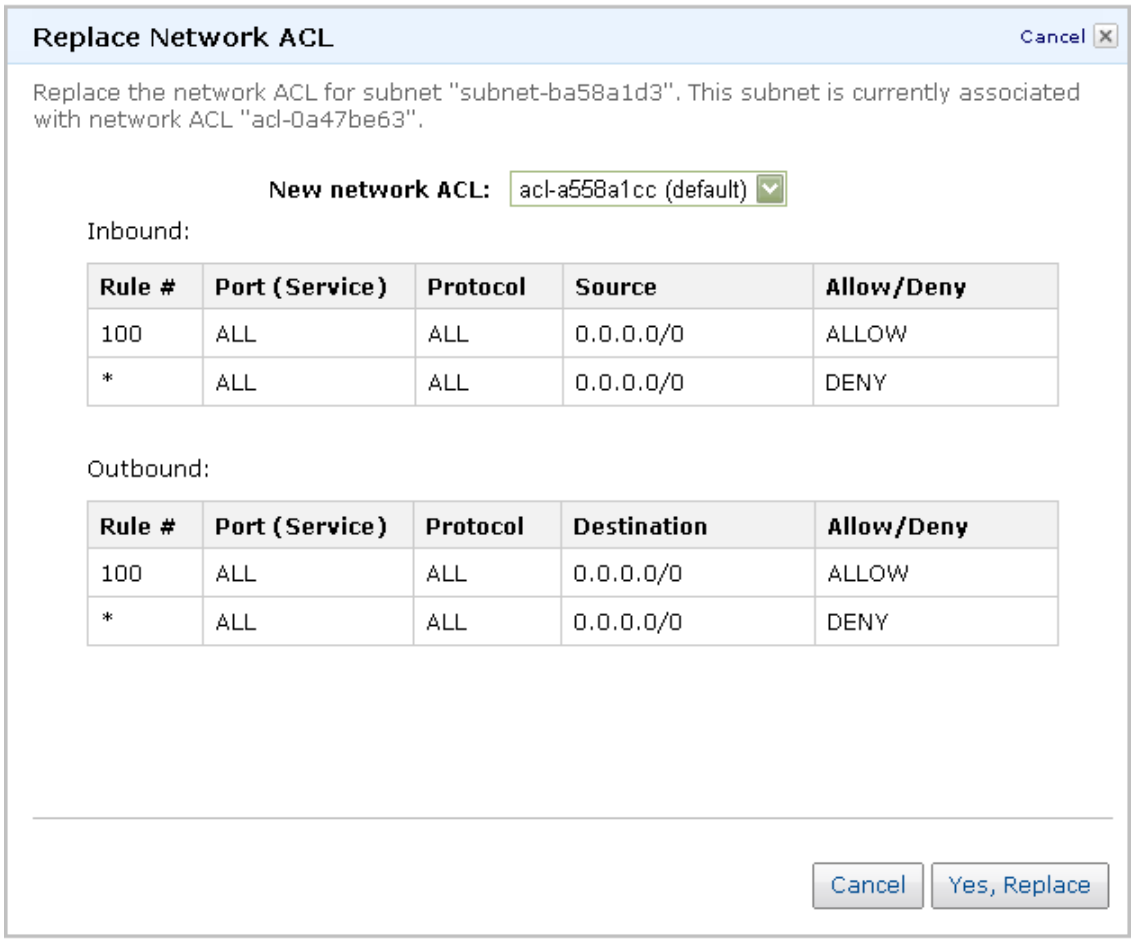
Route Table: rtb-bc58a1d5 (replace)

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	ig-a658a1cf

Network ACL: acl-0a47be63 (replace)

Inbound:

Rule #	Port (Service)	Protocol	Source	Allow/Deny
110	22 (SSH)	TCP	172.0.0.0/8	ALLOW
*	ALL	ALL	0.0.0.0/0	DENY



4. In the **Replace Network ACL** dialog box, in the drop-down list, select the network ACL to associate the subnet with and click **Yes, Replace**.

The subnet is now associated with the new ACL and is subject to the ACL's rules.

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 at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Network ACLs**.
3. Select the check box for the network ACL and click **Delete**.
4. In the **Delete Network ACL** dialog box, click **Yes, Delete**.

You've deleted the network ACL.

API and Command Overview

The following table summarizes the available network ACL commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-create-network-acl` `CreateNetworkAcl`	Creates a new network ACL for your VPC.
`ec2-describe-network-acls` `DescribeNetworkAcls`	Returns information about your network ACLs.
`ec2-delete-network-acl` `DeleteNetworkAcl`	Deletes a network ACL.
`ec2-create-network-acl-entry` `CreateNetworkAclEntry`	Adds a rule to a network ACL.
`ec2-delete-network-acl-entry` `DeleteNetworkAclEntry`	Deletes a rule from a network ACL.
`ec2-replace-network-acl-entry` `ReplaceNetworkAclEntry`	Replaces an existing rule in a network ACL.
`ec2-replace-network-acl-association` `ReplaceNetworkAclAssociation`	Changes which network ACL a subnet is associated with.

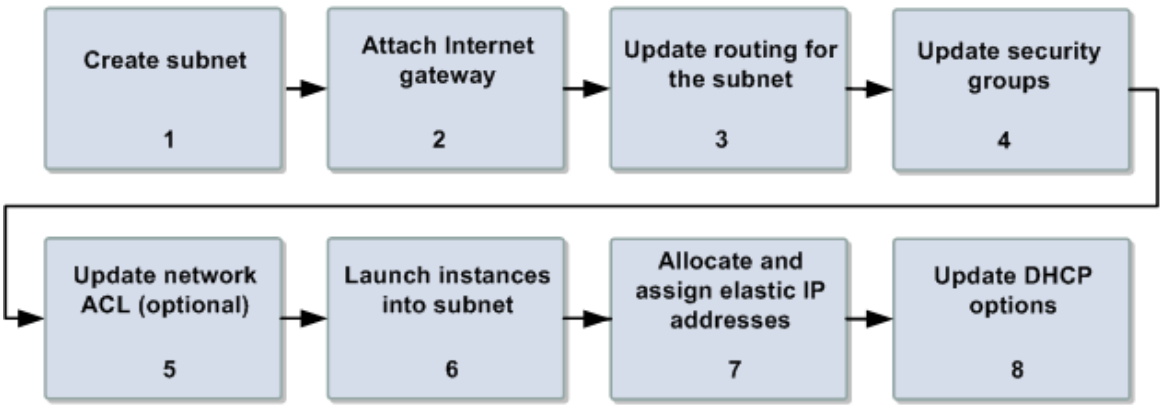
Adding an Internet Gateway to Your VPC

The VPC creation wizard in the AWS Management Console automatically adds an Internet gateway to your VPC (depending on which scenario you select). However, you might have an existing VPC with only a virtual private gateway, and you might want to add an Internet gateway. This section describes the process. The general layout of a VPC with both types of gateways is covered earlier in this guide in scenario 3 (see Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 44)).

When you add an Internet gateway to your VPC, your goal is to have a subnet that contains public instances (instances with public IP addresses, such as web servers or a NAT instance). If you've currently got a VPC with one or more subnets containing private instances, you could do one of the following:

- Add a new subnet for your public instances
- Add your public instances to an existing private subnet

This section assumes you want the first option. The following diagram and table describe the process.



Process for Adding an Internet Gateway

1	Create a new subnet (see Create a Subnet (p. 157)).

2	Attach an Internet gateway to your VPC (see Attach an Internet Gateway (p. 157)).
3	Update the routing to send the subnet's traffic to the Internet gateway (see Update Routing (p. 160)).
4	Create any new security groups you need for the instances in the subnet, or modify existing ones as needed (see Update Security Groups (p. 161)).
5	If you use network ACLs in your VPC, set up an ACL for the subnet, or modify an existing one as needed (for more information about ACLs, see Network ACLs (p. 144)).
6	Launch instances into the subnet.
7	Add Elastic IP addresses to the instances that you want to be public (see Add Elastic IP Addresses (p. 161)).
8	Update the VPC's DHCP options to include Amazon's DNS server (see Update DHCP Options (p. 161)).

Create a Subnet

For information about subnets and how to choose a size for a subnet, see Your VPC and Subnets (p. 105).

To create a subnet

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 and Availability Zone, enter the CIDR range you want for your subnet (e.g., 10.0.0/24), and then click **Yes, Create**.
   The subnet is created and appears on the **Subnets** page.

Attach an Internet Gateway

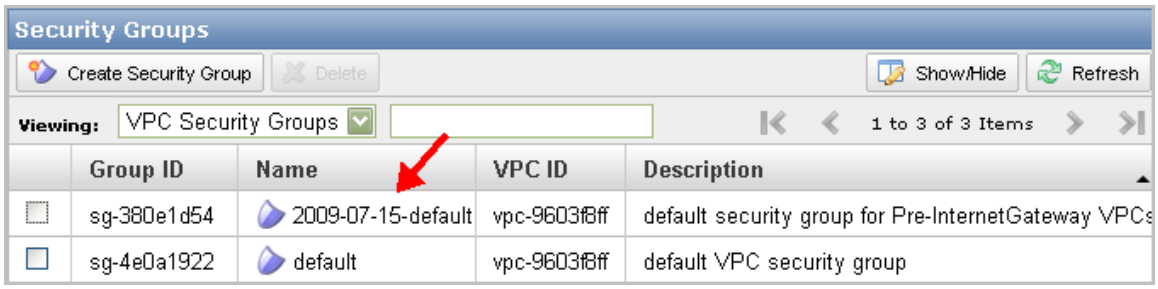
The process of attaching an Internet gateway includes several tasks summarized in the following table.

Process for Attaching an Internet Gateway

A	Determine if your VPC has a security group called *2009-07-15-default*. If yes, continue to task B. If no, skip to task D.
B	Move any existing instances from the *2009-07-15-default* group to a different security group.
C	Delete the legacy *2009-07-15-default* security group.
D	Attach an Internet gateway (see Task D: Attach the Internet Gateway (p. 159)).

Task A: Determine If Your VPC Has the 2009-07-15-default Security Group

The following image shows an example of what the 2009-07-15-default group looks like in the list of security groups.



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

If your VPC has the 2009-07-15-default security group, you must move any instances that are in that group into a different security group of your choice, and then delete the 2009-07-15 default group. This process forces you to consider the security group rules you want to apply to any existing instances before you expose your VPC to the Internet.

If your VPC doesn't have the 2009-07-15-default security group, then you can immediately attach the Internet gateway. See Task D: Attach the Internet Gateway (p. 159).

Task B: Move Instances to Different Security Group

The group you move the instances to can be the default group or one you create.

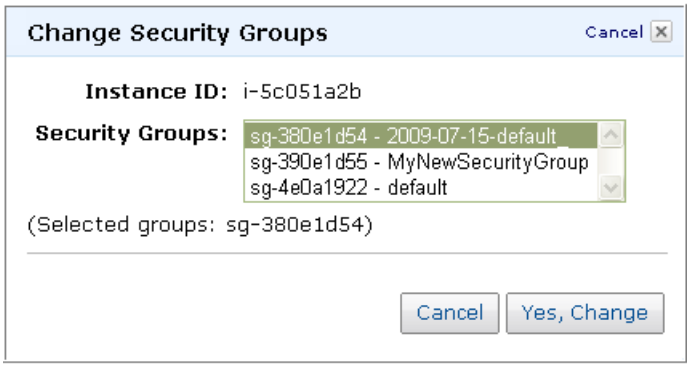
> **Note**
>
> The initial settings of your VPC's default security group deny all inbound traffic, allow all outbound traffic, and allow all instances in the group to talk to each other. If you plan to move the instances to the default group, make sure to modify the group's rules as necessary to maintain the instances' ability to communicate.

For an example of creating a group and adding rules to it, see Task 6: Create Security Groups and Add Rules (p. 38) in scenario 2. For general information about security groups, see Security Groups (p. 137).

After you determine a group to move the instances to, you can move each instance to the group. The following procedure moves an instance to a different security group (i.e., changes the security group or groups that an instance belongs to). You must repeat this procedure for each instance in the group.

To move an instance to a different security group

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the **Navigation** pane, click **Instances**.
3. Right-click the instance that you want to move in the list of instances, and select **Change Security Groups**.



4. In the **Change Security Groups** dialog box, in the **Security Groups** list, select the security group to move the instance to, and then click **Yes, Change**.

   Tip

   When changing an instance's group membership, you can select multiple groups from the list. The new list of groups you select replaces the instance's current list of groups.

5. Repeat the preceding steps for each instance you need to move.

After you complete the preceding procedure, all your instances are now in a different security group, and you can delete the legacy 2009-07-15-default group.

Task C: Delete the 2009-07-15-default Security Group

Now that the 2009-07-15-default security group is empty, you can delete it.

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

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Security Groups**.
3. Select the check box for the 2009-07-15-default security group and click **Delete**.
4. In the **Delete Security Group** dialog box, click **Yes, Delete**.

The legacy 2009-07-15-default security group is deleted. You can now attach an Internet gateway.

Task D: Attach the Internet Gateway

The following procedure creates an Internet gateway and attaches it to your VPC.

To attach the Internet gateway

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, click **Yes, Create**.
   The Internet gateway is created and appears on the page. Notice that it has an ID (e.g., igw-xxxxxxxx).
4. Select the Internet gateway and click **Attach to VPC**.

5. In the **Attach to VPC** dialog box, click **Yes, Attach**.

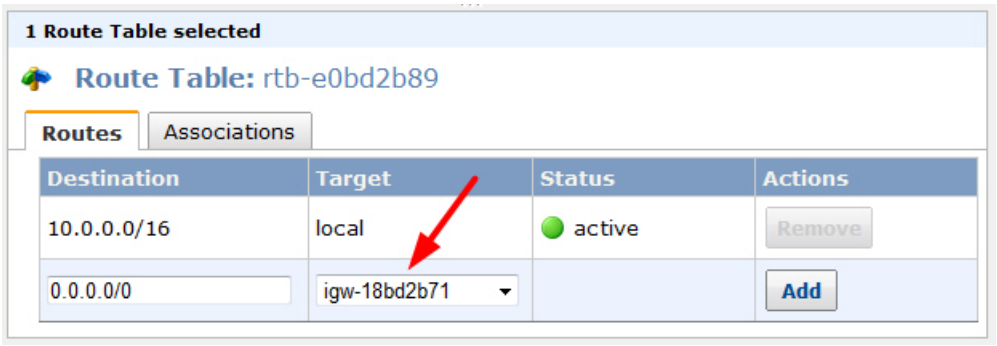
Your VPC now has an Internet gateway attached.

Update Routing

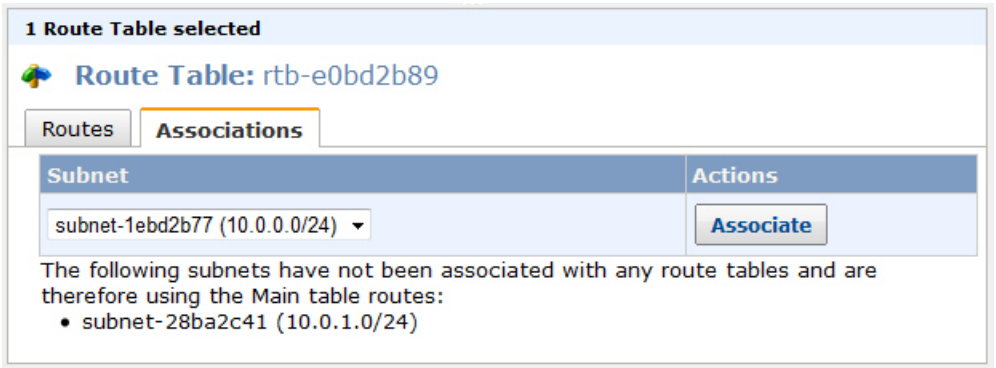
Your new subnet is automatically associated with the VPC's main route table. The subnet needs to have its traffic routed to the Internet gateway. This section describes how to create a custom route table with the necessary route and associate the subnet with that table. For more information about route tables, see Route Tables (p. 111).

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, make sure your VPC is selected, and then click **Yes, Create**. The new route table is created and appears on the page. Notice that it has an ID (e.g., rtb-xxxxxxxx).
4. Select the check box for the custom route table.
5. In the lower pane, on the **Routes** tab, enter `0.0.0.0/0` in the **Destination** field, select the Internet gateway's ID in the **Target** drop-down list, and then click **Add**.



6. On the **Associations** tab, select the ID of the subnet and click **Associate**.



The subnet is now associated with the custom route table. Any traffic leaving the subnet goes the Internet gateway.

Update Security Groups

You need to either update an existing security group or create one or more new groups for the public instances you plan to put in the subnet. For example, you might want new security group rules that restrict inbound and outbound traffic to only HTTP and HTTPS (e.g., for web servers). For managing the instances, you might also want rules to allow inbound SSH or RDP access, depending on the type of operating system. For examples of these rules, see Task 7: Create Security Groups and Add Rules (p. 64). For more information about security groups, see Security Groups (p. 137).

Add Elastic IP Addresses

After you've launched instances in the subnet, you must assign elastic IP addresses to any instances that you want to be public. For more information about Elastic IP addresses, see Elastic IP Addresses (p. 129).

To allocate and assign an elastic IP address to an instance

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Elastic IPs**, and then click **Allocate New Address**.
3. In the **Allocate New Address** dialog box, in the **EIP used in:** drop-down list, select VPC and click **Yes, Allocate**.
   The new address is allocated and appears on the page.
4. Right-click the IP address in the list and select **Associate**.
5. In the **Associate Address** dialog box, select the instance you want to associate the address with and click **Yes, Associate**.
   The address is associated with the instance. Notice that the instance ID is displayed next to the IP address in the list.

Update DHCP Options

You need a DNS server that enables your public subnet to communicate with servers on the Internet. Amazon provides a DNS server you can use (AmazonProvidedDNS). To enable your VPC to use it, you must create a new set of DHCP options and associate them with your VPC. You don't have to use Amazon's DNS server; if you have your own, specify the IP address for your server and not Amazon's in the following procedure. For more information about DHCP options, see Using DHCP Options with Your VPC (p. 176).

Tip

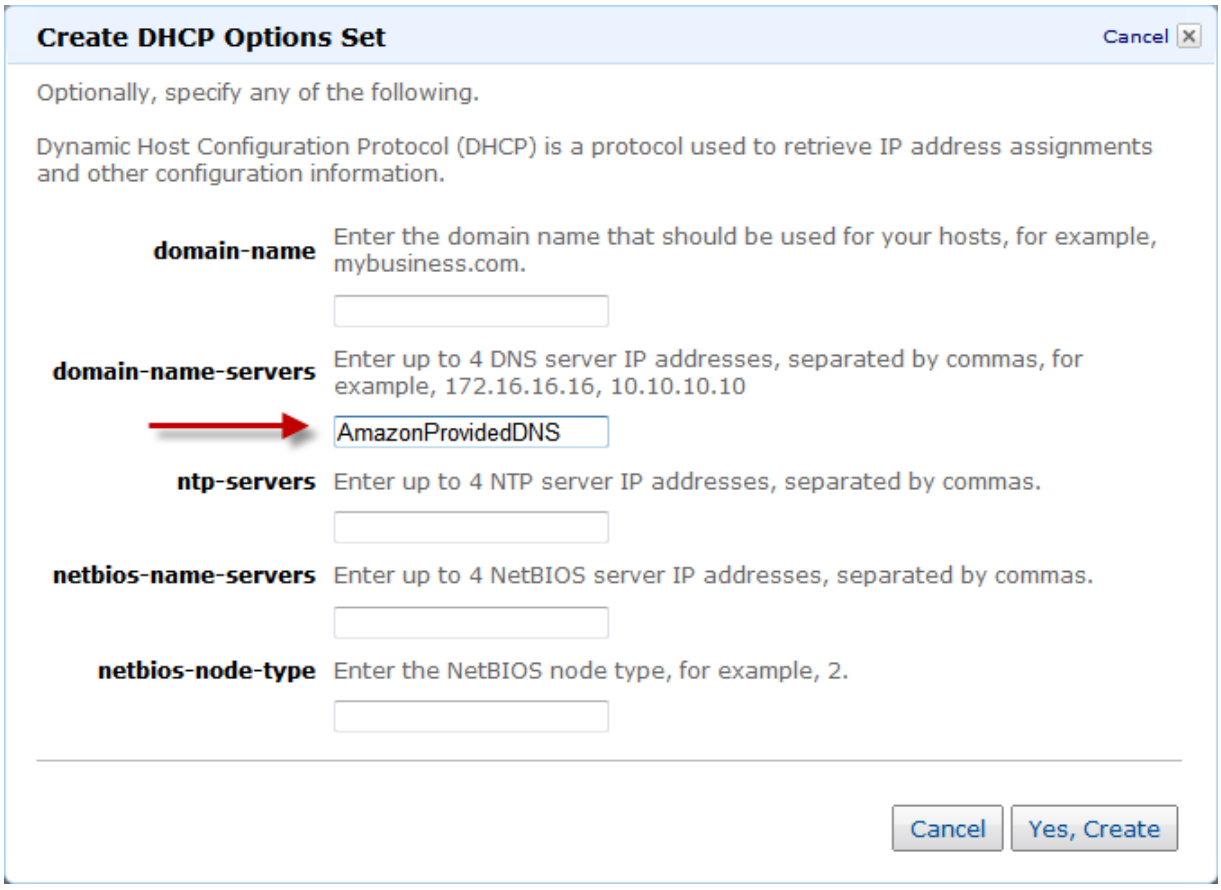
If you create a VPC using the 2011-01-01 API version (or the AWS Management Console), the VPC automatically comes with a set of DHCP options that includes only the Amazon DNS server AmazonProvidedDNS.

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**, and then click **Create DHCP Options Set**.
3. In the **Create DHCP Options Set** dialog box, in the **domain-name-servers** field, enter the Amazon DNS server IP address label (AmazonProvidedDNS).

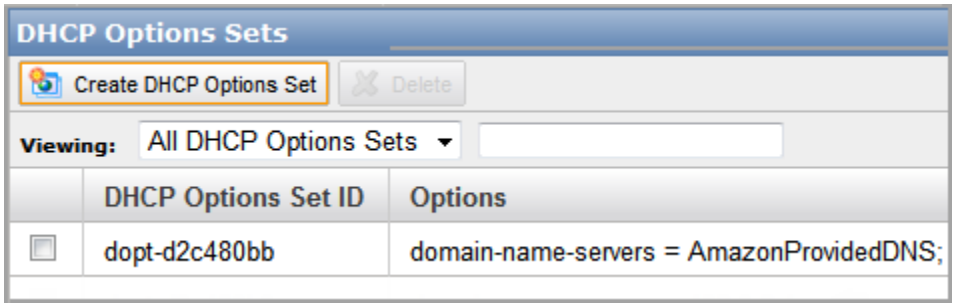
Note

The string `AmazonProvidedDNS` is not case sensitive. For example, the following values are also valid: `amazonprovideddns`, `AmazonProvidedDns`, and `AMAZONPROVIDEDDNS`.



4. Click **Yes, Create**.

   The new set of DHCP options is created.



5. Write down the ID of the new set of options you just created.
6. In the **Navigation** pane, click **Your VPCs**.
7. Select the VPC and click **Change DHCP Options Set**.
8. In the **Change DHCP Options Set** dialog box, select the ID of the new set of options, and then click **Yes, Change**.

The VPC now uses this new set of DHCP options and therefore has access to the Amazon DNS server.

Congratulations! You've added an Internet gateway to your VPC.

Adding a Hardware Virtual Private Gateway to Your VPC

Topics

This section is for Amazon Virtual Private Cloud (Amazon VPC) users who want to use an IPsec hardware VPN gateway with their VPC. You'll get basic information about the components that make up the VPN, and how to set up the VPN connection. Most of the setup is automated for you if you use the AWS Management Console.

Components of Your VPN

If you plan to have a VPN connection between your VPC and home network, you need to be familiar with the following concepts.

VPN Connection

An Amazon VPC *VPN connection* is a connection between your VPC and your data center, home network, or co-location facility. A VPN connection has two endpoints (or anchors): a *customer gateway* (your gateway) and *VPN gateway* (our gateway). Although *VPN connection* is a general term, throughout the documentation we specifically mean the connection between a VPC and your own network.

You can monitor the status of your VPN connections using the VPC console or by using the Amazon EC2 API/CLI. You can view information about your VPN connections including the State (such as Connected, Disconnected, Error), time since last state change (for example, 24 days, 2 hours, 23 minutes since last status change), and descriptive error text.

Virtual Private Gateway

An Amazon VPC *virtual private gateway* is the Amazon side of a VPN connection that maintains connectivity. The virtual private gateway interconnects your VPC (via an attachment) and your customer gateway (via a VPN connection).

Customer Gateway

An Amazon VPC *customer gateway* is your side of a VPN connection that maintains connectivity. The customer gateway can be either a physical device or software appliance. The internal interfaces of the customer gateway connect to your data center and the external interfaces connect to the VPN connection, which leads to the virtual private gateway in the AWS cloud.

VGW Attachment

An Amazon VPC *virtual private gateway attachment* is the connection between the virtual private gateway and the VPC. For information about how many virtual private gateway you can have per region as well as the limits for other components within your VPC, see Appendix B: Limits (p. 216)).

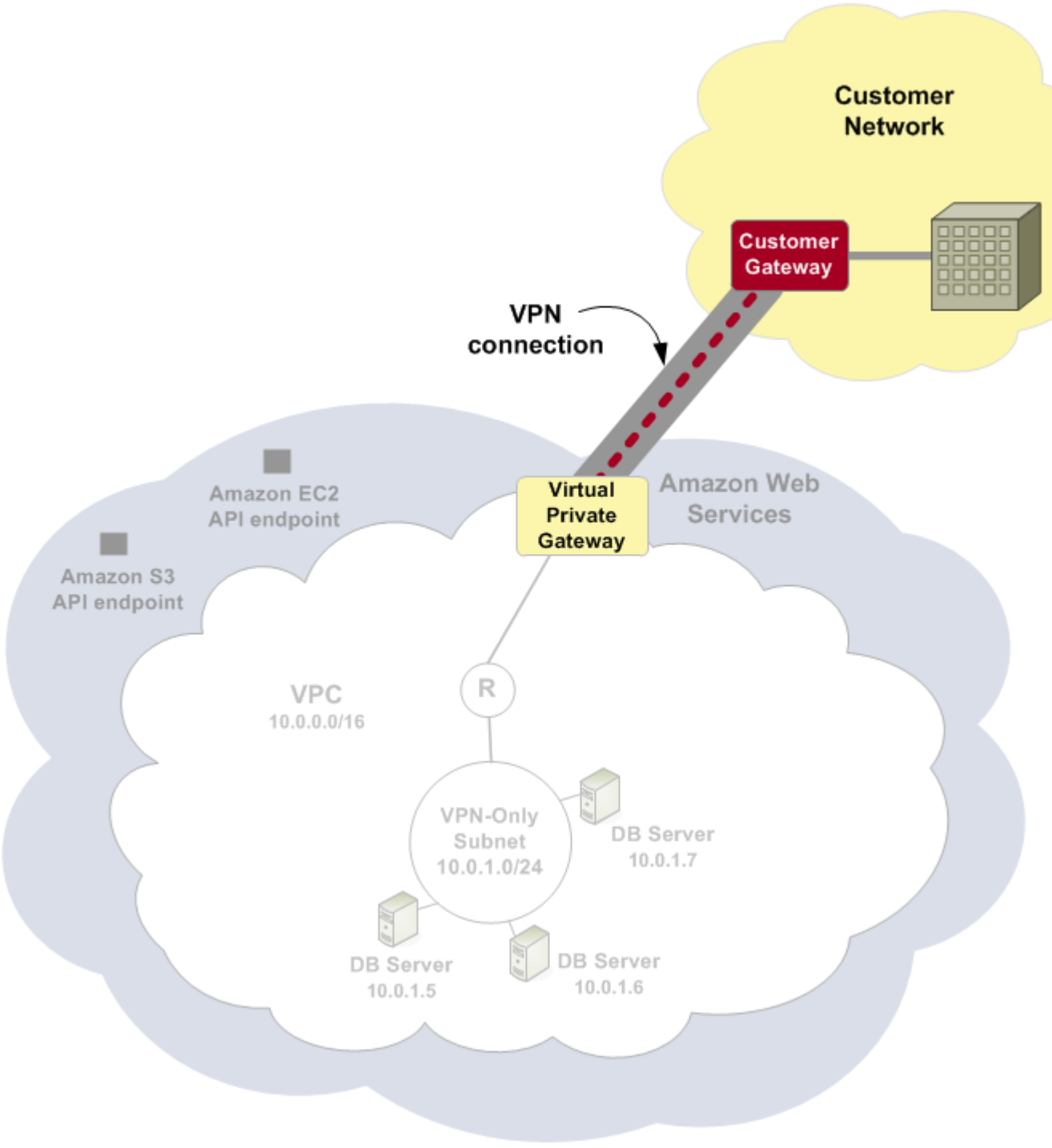
Examples

The following diagrams illustrate single and multiple VPN configuration using the preceding components in your VPC. The VPC has a virtual private gateway attached, and your home 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 home 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

The following diagram shows a single VPN connection:



Multiple VPN connections

The following diagram shows a multiple VPN configuration:

For information about how you're charged for using a VPN connection with your VPC, go to the Amazon VPC product page.

What You Need for the VPN

To use Amazon VPC with a VPN connection, someone on your team must designate a physical appliance as your Amazon VPC customer gateway and configure it. Amazon VPC provides information required for the configuration, including the VPN preshared key and other parameters related to setting up the VPN connection. This configuration is typically performed by a network administrator. For information about the customer gateway requirements and configuration, go to the Amazon Virtual Private Cloud Network Administrator Guide.

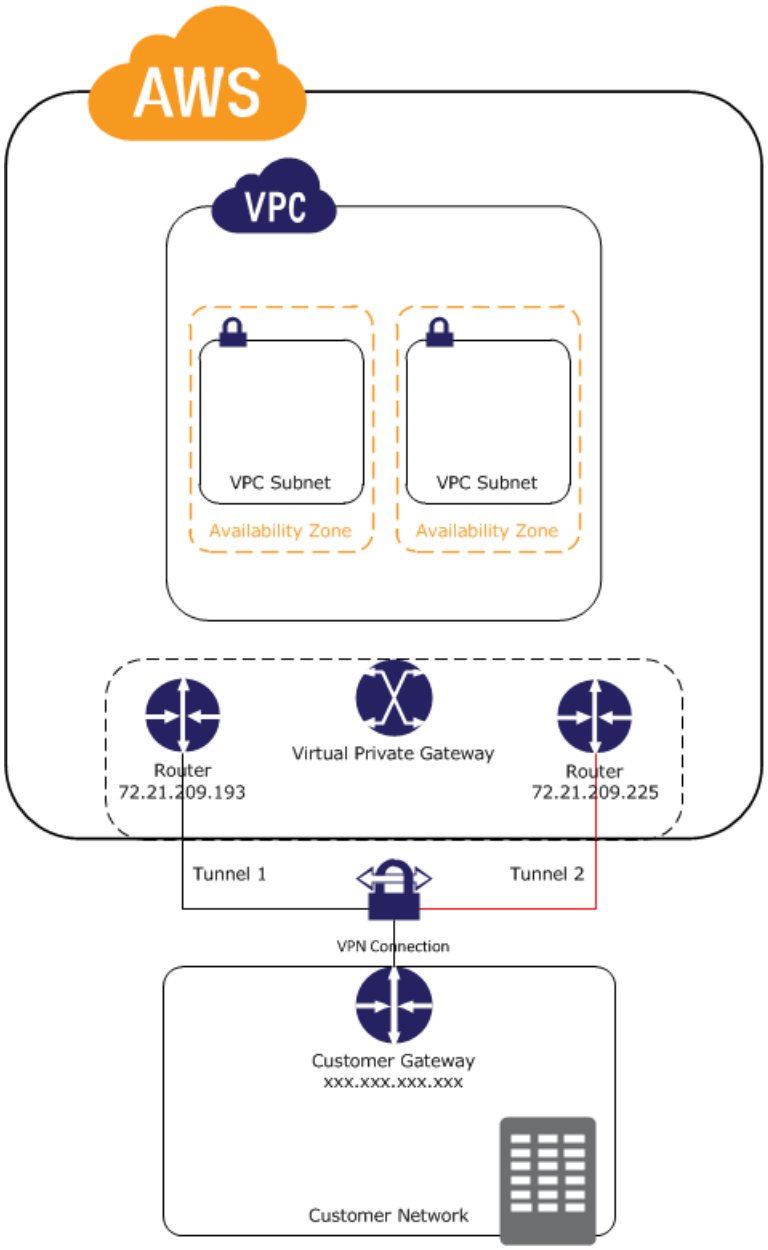
The following table lists information you need to provide so that Amazon VPC can establish your VPN connection.

Item	How Used	Comments	
Type of customer gateway (for example, Cisco ISR, Juniper J-Series, Juniper SSG, Yamaha, or other)	Specifies how to format the returned information that you use to configure the customer gateway.		
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).	
Border Gateway Protocol (BGP) Autonomous System Number (ASN) of the customer gateway	Used to create and configure your customer gateway (it's 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, go to the Wikipedia article. Amazon VPC supports 2-byte ASN numbers.	

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 (e.g., down for maintenance), network traffic is automatically routed to the available tunnel for that specific VPN connection.

The folllowing diagram shows the two tunnels of the VPN connection, as they would be configured for the US East (Northern Virginia) Region.

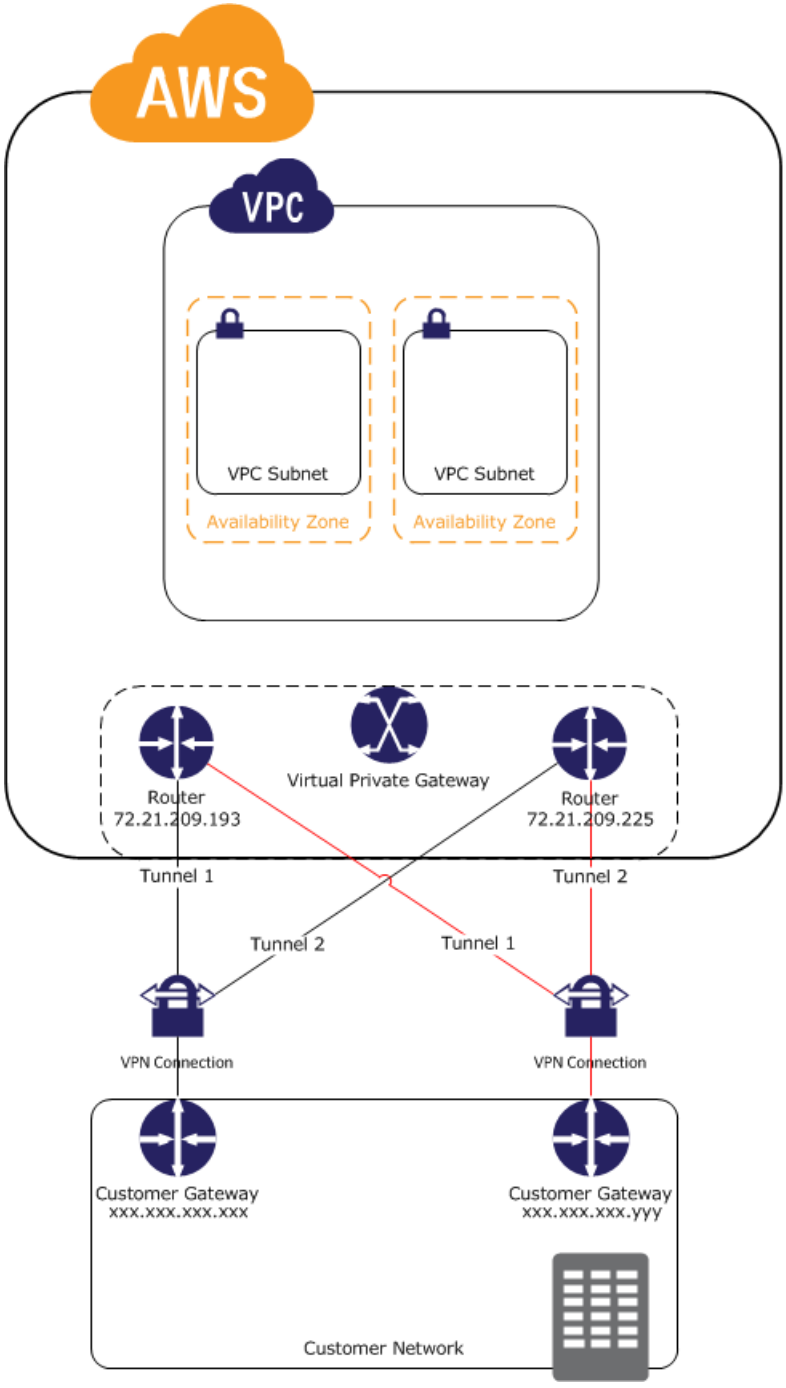


Using Redundant VPN Connections to Provide Failover

As described earlier, a VPN connection consists of two tunnels to Amazon VPC that 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'll need to set up a second VPN connection with Amazon VPC.

The customer gateway IP address for the second VPN connection needs to be publically 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, as they would be configured for the US East (Northern Virginia) Region.



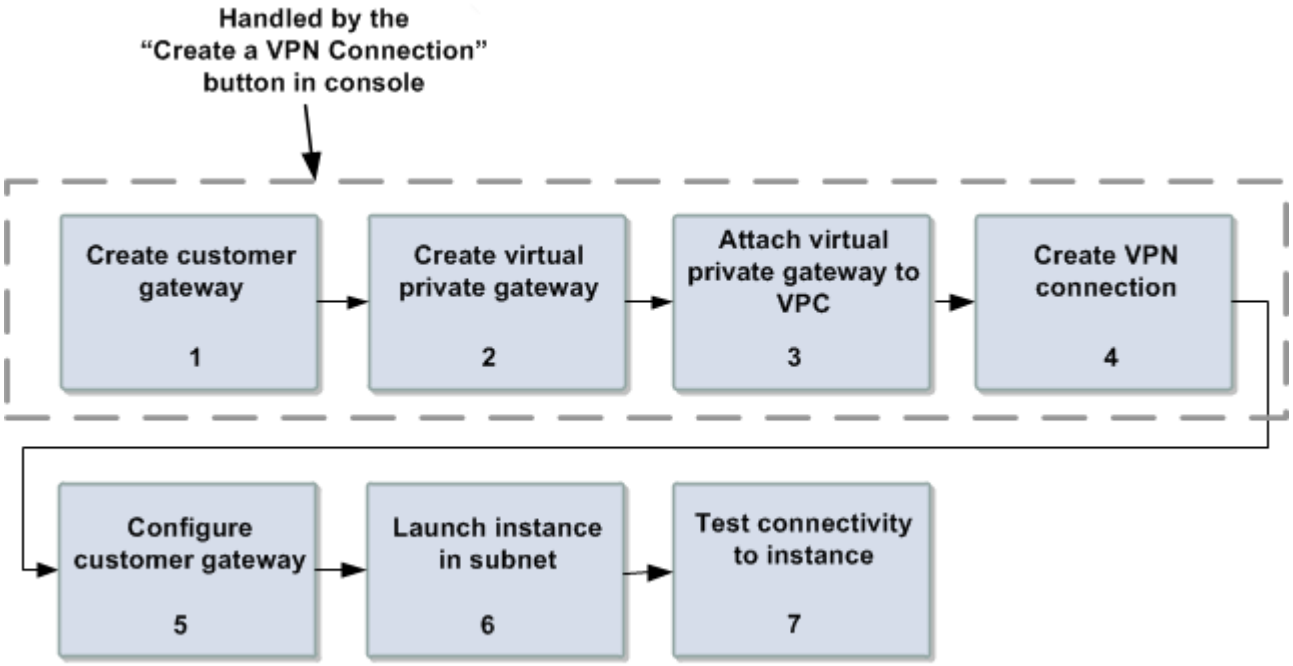
VPN connections use the Border Gateway Protocol (BGP) to exchange routing information between your customer gateways and the virtual private gateways. This allows gateways on both sides to determine which tunnels are available and reroute traffic if a failure occurs. We recommend that you configure your

network to use the routing information provided by BGP to select an available path. The exact configuration will depend on the architecture of your network.

Process for Setting Up the VPN Connection

The following diagram shows the tasks involved in setting up and verifying the VPN connection. We assume you already have a VPC with one or more subnets, and you have the required network information (see What You Need for the VPN (p. 167)).

You can complete the first four tasks in the AWS Management Console by pushing a single button. These tasks are also automatically performed if you use one of the wizard options that includes a VPN connection.



Here's where to get more information about the tasks in the preceding process.

- For information about performing tasks 1-4, see Task 3: Set Up the VPN Connection (p. 96).
- For task 5, you need to give the configuration information you received in the preceding tasks to your network administrator, along with this guide: Amazon Virtual Private Cloud Network Administrator Guide. After the admin configures the customer gateway, the VPN connection will work.

  You can get the configuration information again at any time by going to the VPC Dashboard on the console and clicking **Download Configuration.**
- For information about testing the connectivity, see the next section.

How to Test the End-to-End Connectivity of Your Instance

After you set up your VPN connection, you can launch an instance and test the connection by pinging the instance. You just need to use an AMI that responds to ping requests. We recommend you use one of the Amazon Linux AMIs.

You can monitor the status of your VPN connections using the VPC console or by using the Amazon EC2 API/CLI. You can view information about your VPN connections including the State (such as Connected, Disconnected, Error), Time since last state change (for example, 24 days, 2 hours, 23 minutes since last status change), and descriptive error text.

Important

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

To test the end-to-end connectivity

1. Launch an instance of one of the Amazon Linux AMIs. They're available in the Quick Start menu when you use the instance launch wizard in the AWS Management Console. For more information, go to the Amazon Elastic Compute Cloud Getting Started Guide.
2. After the instance is running, get its private IP address (e.g., 10.0.0.4). The AWS Management Console displays the address as part of the instance's details.
3. On a system in your home network, use the ping command with the instance's IP address. Make sure the computer you ping from is behind the customer gateway. A successful response should be 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
```

Compromised Credentials

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

Note

This procedure requires assistance from your network administrator group.

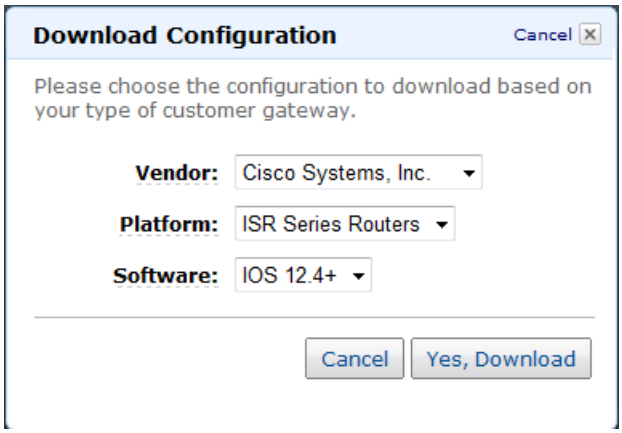
To change the IKE preshared key

1. Delete the VPN connection. You don't need to delete the VPC or the virtual private gateway.
   a. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
   b. In the **Navigation** pane, click **VPN Connections**.
   c. Select the VPN connection and click **Delete**.
   d. In the **Delete VPN Connection** dialog box, click **Yes, Delete**.
   The VPN connection is deleted.

2. Create a new VPN connection.
   a. On the same page, click **Create VPN Connection**.
   The **Create VPN Connection** dialog box is displayed with your virtual private gateway and customer gateway already selected.
   b. Click **Yes, Create**.
   A new VPN connection is created.

3. Download a new customer gateway configuration, which your network administrator must implement. This new configuration replaces the previous gateway configuration that used the old IKE preshared key.
   a. On the same page, select the VPN connection you just created and click **Download Configuration**.

      The **Download Configuration** dialog box is displayed.



   b. Select the customer gateway's vendor, platform, and software version, and click **Yes, Download**.
   The console responds with a text file containing the configuration.
   c. Save the file and give it to your network administrator along with the Amazon Virtual Private Cloud Network Administrator Guide.

After the network administrator implements the new configuration information, your VPN connection starts to use the new credentials, and the network connection to your instances in the VPC resumes.

Communicate Securely Between Sites Using AWS VPN CloudHub

If you have multiple VPN connections, you can securely communicate from one site to another 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.

To use the AWS VPN CloudHub, you must create a virtual private gateway, set up your customer gateways, and create the VPN connections with unique Border Gateway Protocol (BGP) Autonomous System Numbers (ASNs) for the customer gateway. After the routes are advertised over the VPN connections, each site can send data to and receive data from the other sites. The routes 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.

In the following diagram, which shows the AWS VPN CloudHub architecture, the blue dashed lines indicate network traffic sent over the VPN connections.

Sites using 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, Chicago, 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 will 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.

Using DHCP Options with Your VPC

Topics

This section describes DHCP options and how to specify the options you want to use with your VPC.

About 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 Option Sets are associated with your AWS account so that you can use them across all of your Amazon VPCs.

The Amazon EC2 instances you launch inside your VPC are private; they're not assigned a public IP address. By default, all instances in the VPC receive an unresolvable host name that AWS assigns (e.g., 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, go to RFC 2132.

DHCP Option Name	Description
domain-name	A domain name of your choice (e.g., example.com).
domain-name-servers	The IP address of a domain name server. You can specify up to four addresses.

DHCP Option Name	Description
ntp-servers	The IP address of a Network Time Protocol (NTP) server. You can specify up to four addresses.
netbios-name-servers	The IP address of a NetBIOS name server. You can specify up to four addresses.
netbios-node-type	The NetBIOS node type (1, 2, 4, or 8). For more information about the values, see RFC 2132. We recommend you only use 2 at this time (broadcast and multicast are currently not supported).

Amazon DNS Server

When you create your VPC, we automatically create a set of DHCP options and associate them with the VPC. This set includes only a single option: `domain-name-servers=AmazonProvidedDNS`. This 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 VPC 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.

Note

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

Important

When you create a new set of options, make sure to specify your own DNS server or `domain-name-servers=AmazonProvidedDNS`. Otherwise, if your VPC has an Internet gateway, the instances won't have access to DNS.

After you associate a new set of 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.

Working with DHCP Options Sets

Topics

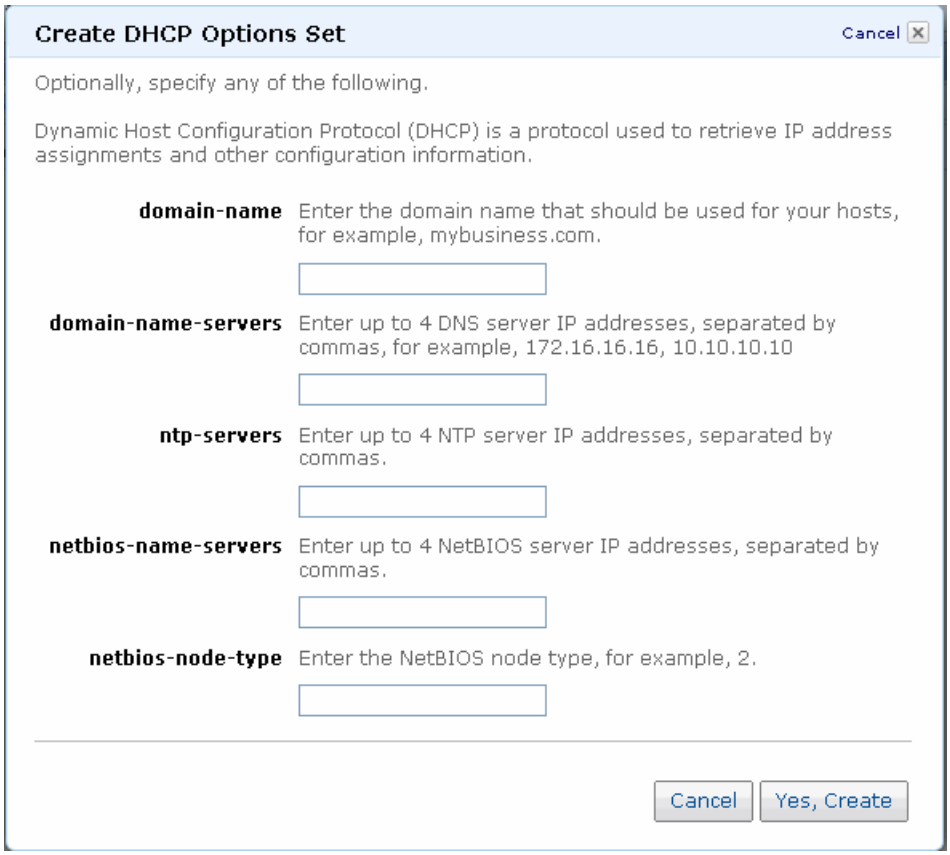
This section gives procedures for working with DHCP options sets.

Creating a DHCP Option Set

Use the following procedure to create a DHCP option set. You can create as many sets as you want, but your VPC can be associated with only a single set at a time. After you create the set, you must configure your VPC to use the new set. For more information, see Changing the Set of DHCP Options a VPC Uses (p. 179).

To create a DHCP option set

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **DHCP Options Set**, and then click **Create DHCP Options Set**.

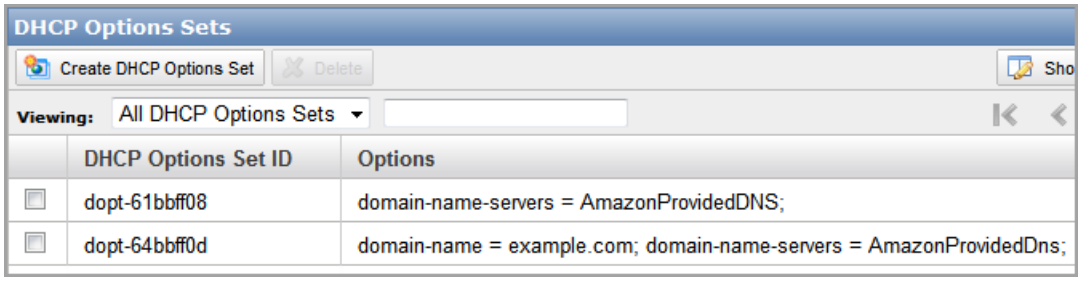


Important

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.

3. In the **Create DHCP Option Set** dialog box, enter values for the options you want to use, and then click **Yes, Create**.

The new set of DHCP options appears in your list of DHCP options. The following image shows an example of the list, with both a new set of options and the set that automatically comes with your VPC (with only domain-name-servers=AmazonProvidedDNS).



4. Make a note of the ID of the new set of options (e.g., dopt-xxxxxxxx). 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 them with your VPC for the options to take effect. You can create multiple sets of options, but you can associate only one set of options with your VPC at a time.

Changing the Set of DHCP Options a VPC Uses

The following procedure changes which set of DHCP options your VPC uses. If you want the VPC to use no DHCP options, see Changing the VPC to use No DHCP Options (p. 179).

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 option set now. For more information, see Creating a DHCP Option Set (p. 178).

To change the DHCP options set associated with a VPC

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Your VPCs**.
3. Select the VPC and click **Change DHCP Options Set**.
4. In the **Change DHCP Options Set** dialog box, select the set of options you want to use, and then click **Yes, Change**.

After you associate a new set of 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 the VPC to use No DHCP Options

If you want, you can set up your VPC to use no set of DHCP options.

1. Right-click the VPC and select **Change DHCP Options Set**.
   The **Change DHCP Options Set** dialog opens.
2. Select **None** from the drop-down list, and click **Yes, Change**.

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 option set

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **DHCP Options Set**.
3. Select the set of options you want to delete, and then click **Delete**.
4. In the **Delete DHCP Options Set** dialog box, click **Yes, Delete**.

API and Command Overview

The following table summarizes the available DHCP options set commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-create-dhcp-options` `CreateDhcpOptions`	Creates a new set of DHCP options in your VPC.
`ec2-associate-dhcp-options` `AssociateDhcpOptions`	Specifies which set of DHCP options the VPC should use, or changes the VPC to use no DHCP options.
`ec2-describe-dhcp-options` `DescribeDhcpOptions`	Returns information about your sets of DHCP options.
`ec2-delete-dhcp-options` `DeleteDhcpOptions`	Deletes a set of DHCP options from your VPC.

Using Auto Scaling with Your VPC

Auto Scaling creates and terminates Amazon EC2 instances based on criteria you set up. For example, you could configure an Auto Scaling policy that starts 10 new instances when the transfer rate of the running instances reaches 80 percent of capacity. You'd also create a second Auto Scaling policy to reduce the number of instances when, for example, the transfer rate of the running instances falls below 40 percent of capacity.

Auto Scaling is available for Amazon EC2 instances running in Amazon VPC. When you create an Auto Scaling group, you can specify a subnet in a VPC for the instances to run in by using the *VPCZoneIdentifier* parameter, which is in the `CreateAutoScalingGroup` Auto Scaling action. You can get the subnet ID from the AWS Management Console when you create the subnet. Auto Scaling stores the subnet ID as part of the Auto Scaling group's metadata, which you can update with calls to the API.

Using Auto Scaling in Amazon VPC

1	Create all of the VPC objects you normally would (VPC, subnets, etc.).
2	Create an Auto Scaling launch configuration that specifies the kind of instances to launch and terminate automatically.
3	Create an Auto Scaling group (using `CreateAutoScalingGroup`) that represents the entire group of instances (minimum size, maximum size, etc.). Include not only the launch configuration from the previous step, but also the subnet ID in *VPCZoneIdentifier*.
4	Optionally, create Amazon CloudWatch alarms and related Auto Scaling policies that tell Auto Scaling when to create new instances or remove existing ones.

Auto Scaling automatically starts (and terminates) Amazon EC2 instances for you in the subnet you created. For more information, go to the Auto Scaling Developer Guide. For more information about Amazon CloudWatch, go to the Amazon CloudWatch Developer Guide.

Using Elastic Network Interfaces with Your VPC

Each Amazon Elastic Compute Cloud (Amazon EC2) instance has a default network interface that is assigned a private IP address on your Amazon VPC network. You can create and attach an additional network interface, known as an elastic network interface (ENI), to any Amazon EC2 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's Guide*.

ENIs have several attributes including, a private IP address, an elastic IP address (optional), a MAC address, membership in specified security groups, a description, and a source/destination check flag. You can create an elastic network interface, attach it to an instance, detach it from an instance, and attach it to another instance. An ENI's attributes, including the private IP address, elastic IP addresses, and MAC address, will follow the ENI as it is attached or detached from an instance and reattached to another instance.

You can attach an ENI to an instance during the launch process (cold attach), when an instance is stopped (warm attach), and when an instance is running (hot attach).

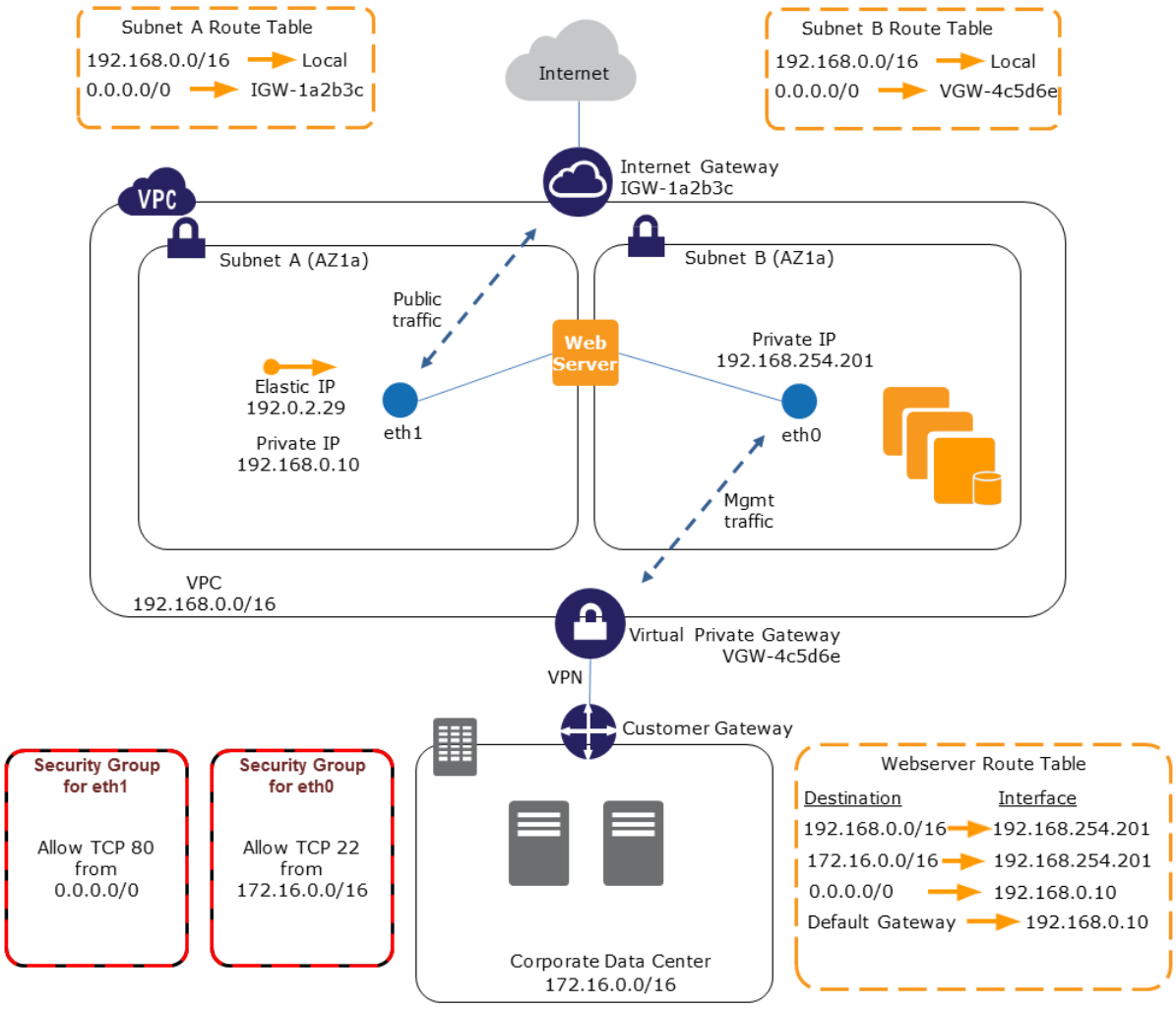
Attaching more than one network interface to an instance is useful when you want to:

- 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.

Creating a Management Network

You can create a management network by utilizing ENIs. In a management network scenario the secondary network interface on the instance handles public-facing traffic and the primary network interface handles back-end management traffic and is connected to a separate subnet in your VPC that has more restrictive access controls. The public facing interface, which may or may not be behind a load balancer, will have an associated security group allowing access to the server from the Internet (e.g.; allow TCP port 80 and 443 from 0.0.0.0/0, or from the load balancer) while the private facing interface will have an associated security group allowing SSH access only from an allowed range of IP addresses either within the VPC or from the Internet, a private subnet within the VPC or a virtual private gateway.

To ensure failover capabilities, consider using a secondary private IP for incoming traffic on a network interface. In the event of an instance failure, you can move the interface and/or secondary private IP address to a standby instance.



Use Network and Security Appliances in Your VPC

Some third party network and security appliances such as load balancers, network address translation (NAT) servers, and proxy servers prefer to be configured with multiple network interfaces. You can create and attach additional network interfaces to instances in a VPC that are running these types of applications and configure the additional interfaces with their own public and private IP addresses, security groups, and source/destination checking.

Creating Dual-homed Instances with Workloads/Roles on Distinct Subnets

You can place an ENI on each of your web servers that connects to a mid-tier network where an application server resides. The application server can also be dual-homed to a backend network (subnet) where the database server resides. Instead of routing network packets through the dual-homed instances, each dual-homed instance receives and processes requests on the front end, initiates a connection to the backend, and then sends requests to the servers on the backend network.

Create a Low Budget High Availability Solution

If one of your instances serving a particular function fails, its network interface can be attached to a replacement or hot standby instance pre-configured for the same role in order to rapidly recover the service. For example, you can use an ENI as your primary or secondary network interface to a critical service such as a database instance or a NAT instance. If the instance fails, you (or more likely, the code running on your behalf) can attach the ENI to a hot standby instance. Because the interface maintains its private IP addresses, elastic IP addresses, and MAC address, network traffic will begin flowing to the standby instance as soon as you attach the ENI to the replacement instance. Users will experience a brief loss of connectivity between the time the instance fails and the time that the ENI is attached to the standby instance, but no changes to the VPC route table or your DNS server are required.

For more information about ENIs, see Elastic Network Interfaces in the Amazon Elastic Compute Cloud User Guide.

Using Multiple IPs with Your VPC

In Amazon Virtual Private Cloud each EC2 instance has a default network interface that is assigned a primary private IP address on your Amazon VPC network. When you launch an instance in a VPC, you can optionally specify an IP address for the instance. If you don't specify a primary private IP address, a primary IP address in the subnet's range is automatically assigned. The assigned address stays with the instance until the instance is terminated. Even if you stop and restart the instance, it retains the same primary IP address.

Beginning with API version 2012-06-15, you can assign additional IP addresses, known as secondary private IP addresses, to Amazon EC2 instances that are running in Amazon VPC. Unlike primary private IP addresses, secondary private IP addresses can be reassigned from one network interface to another and from one instance to another.

You can associate an elastic IP address wtih any primary or secondary private IP address (so the instance can accept external traffic on the ports you specify).

Note

The number of secondary private IP addresses that you can assign varies by instance type. For more information, go to Instance Families and Types.

Use Cases

Assigning multiple private IP addresses to an EC2 instance in your VPC is useful when you want to:

- Host multiple websites on a single server by doing either of the following:
  - Using multiple SSL certificates on a single server and associating each certificate with a specific IP address.
  - Configuring multiple virtual hosts on a Web server.
- Operate network appliances, such as firewalls or load balancers, that have multiple IP addresses for each network interface.
- Redirect internal traffic to a standby Amazon EC2 instance in case your instance fails, by reassigning the secondary private IP address to the Amazon EC2 instance.

Requirements for Multiple IP Addresses

The following list explains how multiple IP addresses work with other components. It also describes the requirements for managing multiple IP addresses.

- Although you cannot move the primary network interface from an instance, you can reassign the secondary private IP address of the primary network interface to another network interface.
- You can move any additional network interface from one instance to another.
- You can assign a secondary private IP addresses to any elastic network interface (eth0 to ethn)
- Secondary private IP addresses must belong to the subnet CIDR block in which the elastic network interface exists
- Security groups apply to interfaces, not to IP addresses. IP addresses are subject to the Security group of the interface to which they're assigned.
- Secondary private IP addresses can be assigned and unassigned to elastic network interfaces which are attached or unattached to instances.
- Secondary private IP addresses can be assigned and unassigned to elastic network interfaces attached to running or stopped instances.
- Secondary private IP addresses that are assigned to one interface can be reassigned to another elastic network interface if you explicitly allow this using the console, the command line tools or the API.
- Each private IP address can only be associated with a single elastic IP address, and vice versa.
- When a secondary private IP address is reassigned to another interface, the secondary private IP address retains its association with an Elastic IP address.
- When a secondary private IP address is unassigned from an interface, the associated Elastic IP address (if it exists) is automatically disassociated from the secondary private IP address.
- When assigning multiple secondary private IP addresses to an network interface using command line tools or API, the entire operation will fail if one of the secondary private IP addresses cannot be assigned.
- Primary private IP addresses, secondary private IP addresses and any associated Elastic IP addresses remain with the interface when the interface is detached from an instance or attached to another instance.

For more information, go to Using Instance IP Addresses in the *Amazon Elastic Compute Cloud User Guide*.

Using EC2 Dedicated Instances Within Your VPC

Topics

When you need to launch instances that are physically isolated at the host hardware level, you can use Amazon Elastic Compute Cloud (Amazon EC2) Dedicated Instances. Launched within your Amazon Virtual Private Cloud (Amazon VPC), Dedicated Instances let you take advantage of Amazon VPC and the AWS cloud while isolating your Amazon EC2 compute instances at the hardware level.

This section discusses the basics of Dedicated Instances, identifies the tools you need to use them, and walks you through the processes of implementing them.

We assume you are familiar with the following concepts:

- Amazon Virtual Private Cloud
- Launching and Using Instances

Dedicated Instance Basics

Amazon EC2 instances launched into a VPC have a tenancy attribute. Setting the instance's tenancy attribute to `dedicated` specifies that your instance will run on single-tenant hardware. Amazon VPCs have a related attribute called *instance tenancy*. Setting this instance tenancy attribute to `dedicated` specifies that only Dedicated Instances can be launched into the VPC.

> **Note**
>
> We have a separate pricing model for running instances that have dedicated tenancy. For more information, go to the Amazon EC2 Dedicated Instances product page.

In planning your VPC and the instances you want to launch into the VPC, consider these two approaches to implementing Dedicated Instances:

- Specify that only Dedicated Instances are launched into your VPC.

To do this, you create your VPC with the instance tenancy set to **Dedicated**. When you launch instances into this VPC, the tenancy will be automatically set to Dedicated.

Note

If you plan to implement Auto Scaling on your Dedicated Instances, the Dedicated Instances must be launched into VPCs that have instance tenancy set to Dedicated.

- Specify that a specific instance launched into your VPC is a Dedicated Instance.

  To do this, you leave the instance tenancy of your VPC set to **Default** when you create it. This way, you can launch instances with dedicated tenancy and instances with default tenancy into that VPC. You specify the tenancy of the instance when you launch it.

Important

You set the tenancy of instances and the instance tenancy of VPCs when you first launch or create them. You cannot change their tenancy or instance tenancy after you set them. If you want your VPC to be dedicated and you didn't specify dedicated as its tenancy value when you created the VPC, you'll have to delete the VPC, recreate it, and relaunch the instances. Likewise, if you want your instance to run on single-tenant hardware and you didn't specify dedicated tenancy when you launched the instance, you must stop the running instance and relaunch it as dedicated.

Note

Although you can launch Amazon EBS-backed Dedicated Instances, the EBS volume will not run on hardware dedicated to your account.

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, go to 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 the price break in the hourly charge if you later launch a Dedicated Instance.

In addition, you can't change the tenancy of a Reserved Instance after you've purchased it. So, if you purchase a Reserved Instance and later you want to use that capacity for launching a Dedicated Instance, you cannot change that Reserved Instance into a Dedicated Reserved Instance.

Dedicated Tenancy Options in the AWS EC2 Tools

You can launch Dedicated Instances or create VPCs with an instance tenancy of dedicated using the AWS Management Console, the API, or the command line tools. In the AWS Management Console, you specify the **Dedicated** option using the **Tenancy** drop-down box.

If you use the API or command line tools, you specify the dedicated tenancy option when you create the VPC using the `CreateVPC` call or the `ec2-create-vpc` command. You specify the instance you launch as dedicated using the `RunInstances` call or the `ec2-run-instances` command. The following table lists the new API actions and commands for Dedicated Instances, and describes them. For more information, go to:

- Amazon Elastic Compute Cloud Command Line Reference
- Amazon Elastic Compute Cloud API Reference.

Command and API Actions	Description
`ec2-create-vpc` `CreateVpc`	The supported tenancy of instances launched into the VPC. A value of `default` means instances can be launched with any tenancy; a value of `dedicated` means all instances launched into the VPC will be launched as dedicated tenancy instances regardless of the tenancy assigned to the instance at launch. Setting the instance's tenancy attribute to `dedicated` specifies that your instance will run on single-tenant hardware.
`ec2-describe-instances` `DescribeInstances`	Returns a tenancy value of default or dedicated.
`ec2-describe-reserved-instances` `DescribeReservedInstances`	Returns a tenancy value of default or dedicated.
`ec2-describe-reserved-instances-offerings` `DescribeReservedInstancesOfferings`	Returns a tenancy value of default or dedicated.
`ec2-describe-vpc` `DescribeVpc`	Includes the supported tenancy options for instances launched into the VPC. If the tenancy value is set to `dedicated`, then only instances with a tenancy of `dedicated` can be launched into the VPC regardless of the tenancy assigned to the instance at launch.
`ec2-run-instances` `RunInstances`	Includes a tenancy value that you specify for the instances you launch into your VPC.

Using Dedicated Instances

In this section, we walk you through launching Dedicated Instances, changing the tenancy of an instance or the instance tenancy of a VPC, and getting tenancy information using the different AWS EC2 tools.

Creating a VPC with an Instance Tenancy of Dedicated

When you create an Amazon 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.

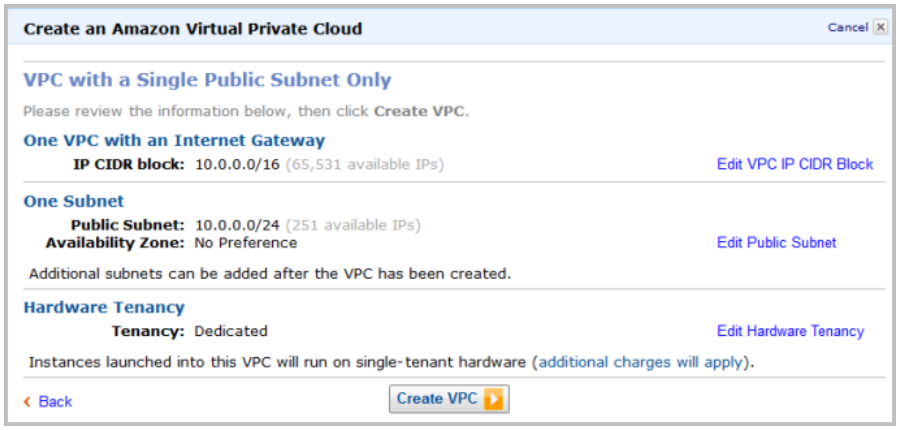
AWS Management Console

To create a VPC with an instance tenancy of dedicated

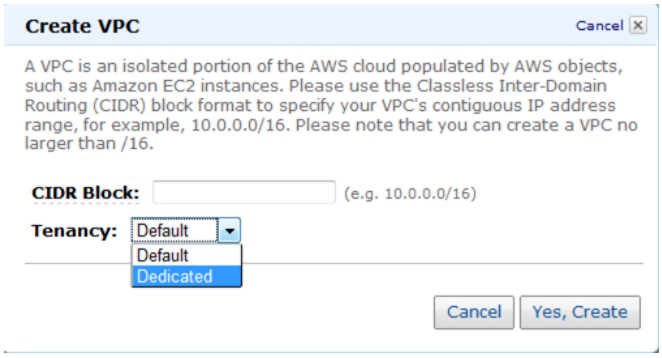
1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.

   Here, you create a VPC either through the **Create an Amazon Virtual Private Cloud** wizard or the **Create VPC** button.
2. When you use the wizard, after selecting your VPC configuration, you will get a confirmation page showing the CIDR blocks, subnets, the size of the NAT instance, key pair, and the instance hardware tenancy of the VPC. You can change any of these values if you want.

   Click **Edit Hardware Tenancy** and select **Dedicated**.



3. Alternatively, use the **Create VPC** button, which you can find when you select **Your VPCs** in the **Navigation** pane.
4. In the **Create VPC** dialog box, click the **Tenancy** drop-down box, and then select **Dedicated**.



5. Specify the **CIDR Block** and click **Yes, Create**. For information on specifying CIDR block, go to the Wikipedia article about Classless Inter-Domain Routing.

   Proceed with the rest of the wizard as you would when creating a VPC with an instance tenancy of default. For more information, go to Task 2: Create the VPC and Subnet (p. 95).

Command Line Tools

To create a VPC with an instance tenancy of dedicated

- Use `ec2-create-vpc` and specify `dedicated` for the optional `tenancy` option.

  Your request will look like this:

```
ec2-create-vpc 10.0.0.0/16 --tenancy dedicated
```

  The command returns a table that includes the new VPC's instance tenancy. The response will look similar to the following example.

```
VPC     vpc-1773ec7e     pending     10.0.0.0/16     dopt-eb73ec82     ded
icated
```

API

To create a VPC with an instance tenancy of dedicated

- Use `createvpc` and specify `dedicated` for the optional `instancetenancy` option.

  Your request will look like this:

```
https://ec2.amazonaws.com/
?SignatureMethod=HmacSHA256
&SignatureVersion=2
&Version=2011-02-28
&Expires=2011-03-26T07:43:41Z
&Action=CreateVpc
&CidrBlock=10.32.0.0/16
&InstanceTenancy=dedicated
&AWSAccessKeyId=YOUR_ACCESS_ID
&Signature=YOUR_SIGNATURE
```

  The following is an example response.

```
<CreateVpcResponse xmlns="http://ec2.amazonaws.com/doc/2011-02-28/">
    <requestId>a9e49797-a74f-4f68-b302-a134a51fd054</requestId>
    <vpc>
        <vpcId>vpc-11a63c78</vpcId>
        <state>pending</state>
        <cidrBlock>10.32.0.0/16</cidrBlock>
        <dhcpOptionsId>dopt-27fd624e</dhcpOptionsId>
        <instanceTenancy>dedicated</instanceTenancy>
    </vpc>
</CreateVpcResponse>
```

Launching Dedicated Instances into a VPC

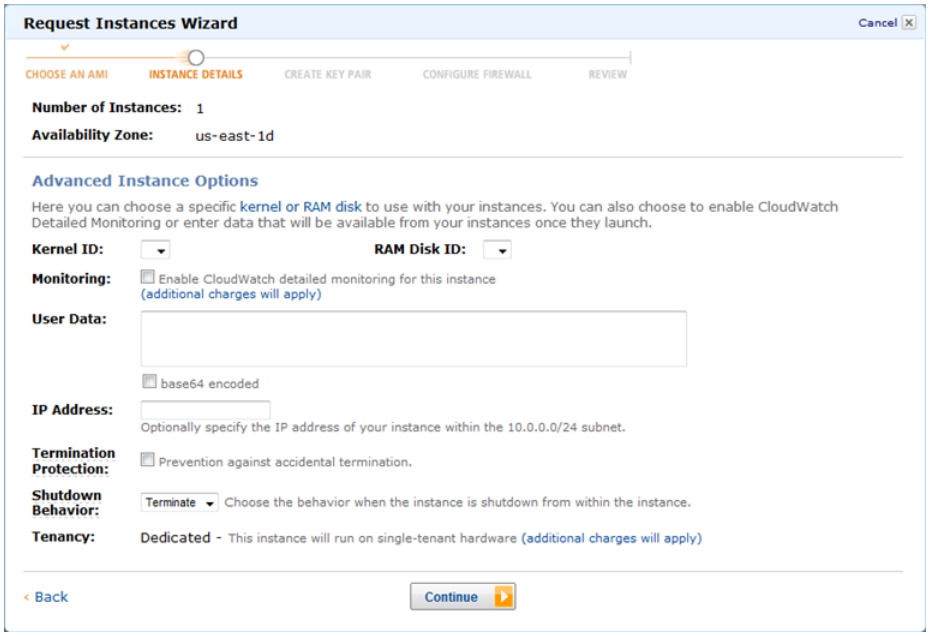
You can launch Dedicated Instances into a VPC that has an instance tenancy of either default or dedicated. Dedicated Instances and instances that have default tenancy can be launched into VPCs that have default instance tenancy. In contrast, all instances launched into dedicated tenancy VPCs will be launched as dedicated instances, regardless of the tenancy assigned to the instance at launch.

AWS Management Console

To launch Dedicated Instances into a VPC

1. Sign in to the AWS Management Console, open the VPC console, and create a VPC or use a VPC that you previously created.
2. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
3. Click the **Launch Instance** button.
4. In the **Request Instances Wizard**, select an AMI from **My AMIs**, if you have AMIs of your own, or one of the available AMIs in the **Community AMIs** tab.
5. On the **INSTANCE DETAILS** page, select the **Launch Instances Into Your Virtual Private Cloud** radio button.
6. Select a **Subnet ID** you want to attach the instance to and click **Continue**.

   The **INSTANCE DETAILS** page shows an **Advanced Instance Options** section.



7. Select **Dedicated** from the **Tenancy** drop-down box and click **Continue**.
8. Proceed with the rest of the wizard as you would when launching an instance. For information about launching instances, go to Running an Instance.

Command Line Tools

To launch Dedicated Instances into a VPC

1. Create your VPC using the `ec2-create-vpc` command or use a VPC that you previously created.
2. To launch a Dedicated Instance, you specify the `tenancy` value as `dedicated`.

   Your request will look something like this example.

```
ec2-run-instances ami-546c983d --tenancy dedicated -s subnet-726cf31b
```

API

To launch Dedicated Instances into a VPC

1. Create your VPC using the `CreateVpc` API function or use a VPC that you previously created.
2. Launch a Dedicated Instance by specifying the `Placement.Tenancy` value as `dedicated`.

   Specify the `SubnetId` of the dedicated VPC you want to launch the instance into.

   Your request will look something like this example.

```
https://ec2.amazonaws.com/
?SignatureMethod=HmacSHA256
&SignatureVersion=2
&Version=2011-02-28
&Expires=2011-03-26T07:53:11Z
&Action=RunInstances
&ImageId=ami-2a1fec43
&SubnetId=subnet-dea63cb7
&Placement.Tenancy=dedicated
&MinCount=1
&MaxCount=1
&AWSAccessKeyId=YOUR_ACCESS_ID
&Signature=YOUR_SIGNATURE
```

   The response will look something like this example.

```
<RunInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-02-28/">
    <requestId>65c0a512-c9ae-4022-9f83-f596fcd002fd</requestId>
    <reservationId>r-a4337bc9</reservationId>
    <ownerId>YOUR_OWNER_ID</ownerId>
    <groupSet/>
    <instancesSet>
        <item>
            <instanceId>i-aca17cc3</instanceId>
            <imageId>ami-2a1fec43</imageId>
            <instanceState>
                <code>0</code>
                <name>pending</name>
            </instanceState>
            <privateDnsName/>
            <dnsName/>
```

```
            <reason/>
            <amiLaunchIndex>0</amiLaunchIndex>
            <productCodes/>
            <instanceType>m1.small</instanceType>
            <launchTime>2011-03-26T07:48:13.000Z</launchTime>
            <placement>
                <availabilityZone>us-east-1a</availabilityZone>
                <groupName/>
                <tenancy>dedicated</tenancy>
            </placement>
            <kernelId>aki-407d9529</kernelId>
            <monitoring>
                <state>disabled</state>
            </monitoring>
           <subnetId>subnet-dea63cb7</subnetId>
            <vpcId>vpc-11a63c78</vpcId>
            <privateIpAddress>10.32.16.192</privateIpAddress>
            <sourceDestCheck>true</sourceDestCheck>
            <groupSet>
                <item>
                    <groupId>sg-d39e8dbf</groupId>
                    <groupName>default</groupName>
                </item>
            </groupSet>
            <stateReason>
                <code>pending</code>
                <message>pending</message>
            </stateReason>
            <rootDeviceType>instance-store</rootDeviceType>
            <blockDeviceMapping/>
            <clientToken/>
            <hypervisor>xen</hypervisor>
        </item>
    </instancesSet>
</RunInstancesResponse>
```

Changing Tenancy

You set the tenancy of your instance when you launch it; you set the instance tenancy of your VPC when you create it. The following procedures outline what to do if you must change the tenancy of your instance, or the instance tenancy of your VPC.

To change the tenancy of your instance

1. Stop the running instance, if it's EBS-backed, using `ec2-stop-instances`.

   Or terminate it, if it's S3-backed, using `ec2-terminate-instances`.
2. Launch the instance using `ec2-run-instances`.

To change the instance tenancy of your VPC

1. Terminate all running instances, using the previous procedure.
2. Delete or detach all objects that are dependent on the VPC. Such objects include security groups and route tables.

Note

Understand that deleting your VPC involves deleting all other components related to it. For more information, go to Deleting Your VPC (p. 109).

3. Delete the VPC, using `ec2-delete-vpc`.

Obtaining Tenancy Information

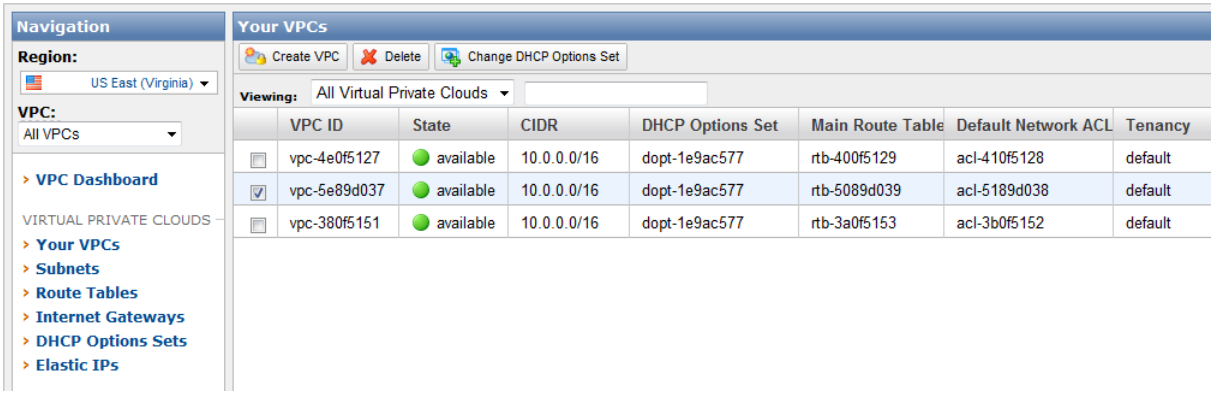
You can determine the tenancy of the instances and the instance tenancy of the VPCs that you have access to by using the AWS Management Console, the API, or the command line tools.

AWS Management Console

To obtain tenancy information

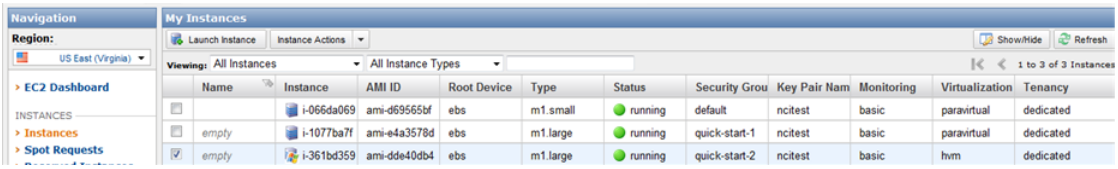
1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the **Navigation** pane, click **Your VPCs**.

   A row showing information about your VPC displays **Tenancy** information.



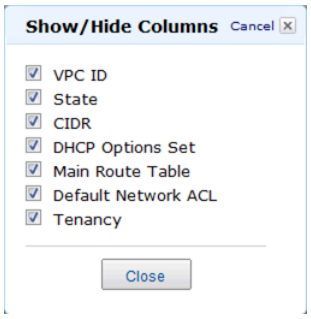
3. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
4. In the **Navigation** pane, click **Instances**.

   A table showing information about your instances displays **Tenancy** information.

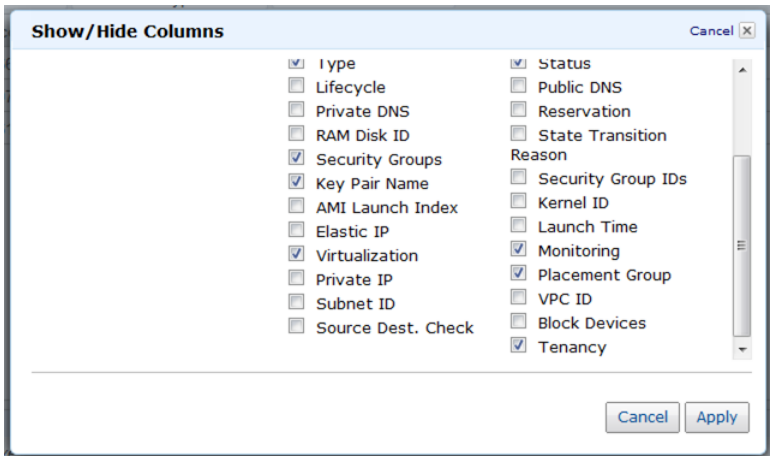


5. If **Tenancy** or other information is not showing up in the VPC or Instances table, click the **Show/Hide** button on the top-right of the right pane and select the items in the **Show/Hide Columns** box that you want the console to display. Click **Close**.

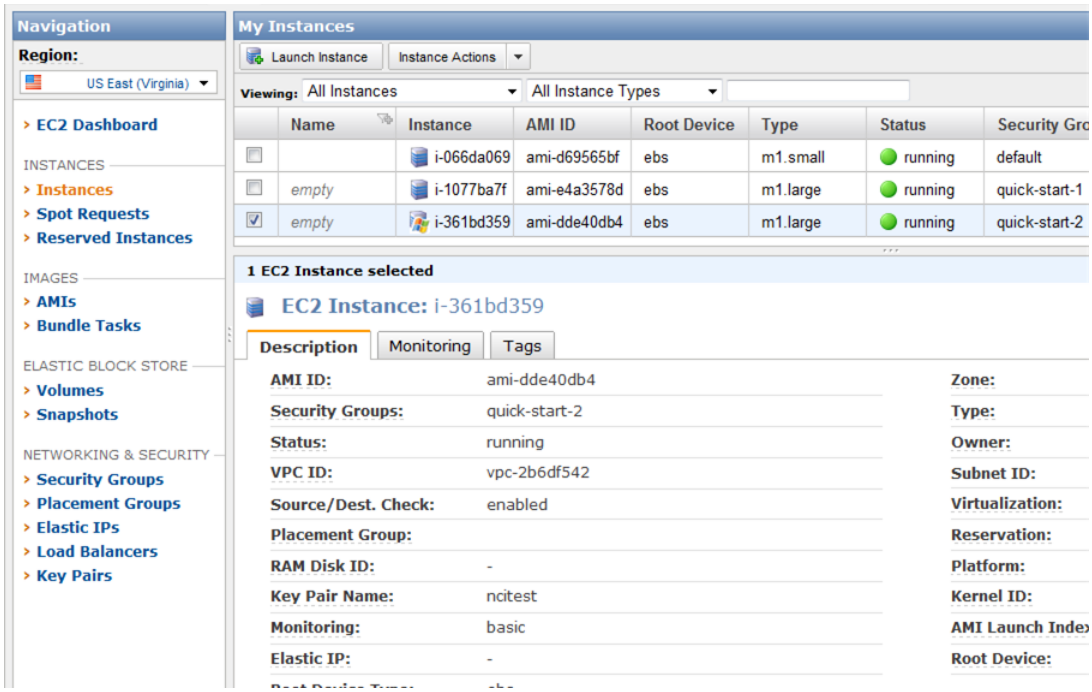
   Here is the **Show/Hide Columns** box for VPCs.



Here is the **Show/Hide Columns** box for Instances.



6. Alternatively, for instances, select the instance you want information about in the table on the right pane. A tabbed page opens below the table. The **Description** tab displays information about your instance.



Command Line Tools

Use the following describe commands to obtain information about instances and VPCs.

- `ec2-describe-instances`
- `ec2-describe-reserved-instances`
- `ec2-describe-reserved-instances-offerings`
- `ec2-describe-vpcs`

API

Use the following describe commands to obtain information about instances and VPCs.

- `DescribeInstances`
- `DescribeReservedInstances`
- `DescribeReservedInstancesOfferings`
- `DescribeVPCs`

Controlling VPC Management

Do you want to control who can set up and manage your Amazon Virtual Private Cloud (VPC)? Do you want to control who can do tasks such as attaching an Internet gateway or defining security groups and network ACLs? You can use AWS Identity and Access Management (IAM) to create and manage users in your account. A *user* is either a person or an application that needs to interact with AWS. With IAM, you can centrally manage your account's users, their security credentials such as access keys, and permissions that control which AWS resources the users can access.

For Amazon VPC and Amazon EC2, you can use IAM to control which API actions a user has access to. For example, you could create a *network administrators* group of users in IAM, and then give only that group the permission to call actions in the Amazon EC2 API related to VPC creation and management. Therefore, not just anyone in your organization can make changes to the layout, routing, and security in your VPC.

Note

Currently, you can't use IAM to limit a user's access to a specific Amazon EC2 or Amazon VPC resource. You can only limit users' access to individual API actions. For example, you can't use IAM to prevent a user from accessing a particular instance or security group; the IAM permission applies to *all* instances or security groups.

IAM uses *policies* in JSON format to specify permissions for users. You create a policy and then attach it to the group of users you want the permissions to apply to. The next sections show some example policies you might find useful.

Note

IAM policies control access regardless of the interface. For example, you could provide a user with a login to the AWS Management Console, and the policies for that user would control what the user can do in the console. Or, you could provide the user with AWS access keys for making API calls to AWS, and the policies would control what actions the user could call through a library or client that uses those access keys for authentication.

For detailed information about setting up users in your account, policies, and IAM, go to Using AWS Identity and Access Management.

Using AWS IAM with Amazon VPC

With IAM, you can manage groups and their access to your VPC resources programmatically using the JSON format or through the AWS Management Console. With both tools, you can create a group, such as Administrator, and grant it full access to your VPC. That group can perform a whole range of tasks such as creating and deleting VPCs and subnets, associating and disassociating route tables, and revoking security group access. Or, you can create a group with access limited to viewing a defined set of VPC resources.

This section shows you examples of IAM policies you can define using JSON and the AWS Management Console. In addition, this section also discusses what you can and cannot do, and how to work around current limitations.

Note

In the future, Amazon VPC might add new actions that should logically be included in one of the following policies, based on the policy's stated goals.

Managing a VPC

Following is an example policy you might give to a network administrator group that needs to create and manage your VPC. This policy gives the group access to 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 the account's resources. For a complete list of the possible actions for Amazon EC2 and Amazon VPC, go to Amazon Elastic Compute Cloud API Reference.

Note

The policy uses wildcards (e.g., `*SecurityGroup*`) to specify all actions for each type of object. You could instead 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 give the group access to those new actions.

To manage a VPC using JSON

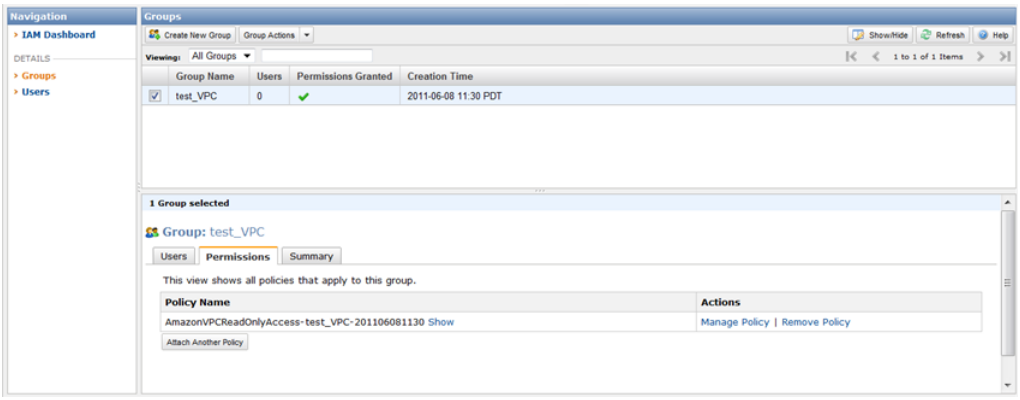
- Use the following sample code -- replacing the * wildcard with specific actions such as "create," "delete," "describe," etc., as appropriate.

```
{
    "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":"*"
   }
 ]
 }
```

To manage a VPC using the AWS Management Console

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the **Navigation** pane, click **Groups**, and then select the group that you want to give full access to your VPC.
3. In the bottom pane, go to the **Permissions** tab and click **Manage Policy**.



4. In the **Manage Group Permissions** page, in the **Policy Name** drop-down menu, select **AmazonVPCFullAccess** and click **Apply Policy**.

   Note

   If you want only a subset of the privileges listed for the policy to apply to your users, edit the list in the **Policy Document** box and click **Apply Policy**.

Read-Only Policy for Amazon VPC

In the following policy, you are giving users permission to view the Amazon VPC console in the AWS Management Console. They can't make any changes; they can only look at information related to your VPC and its components.

To grant read-only access to your VPC using JSON

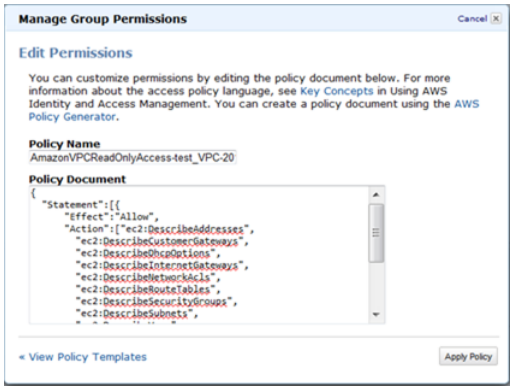
- Use the following sample code to allow a group to look at information about your VPC.

```
{
    "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":"*"
    }
  ]
}
```

To grant read-only access to your VPC using the AWS Management Console

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the **Navigation** pane, click **Groups**, and then select the group that you want to give read-only access to your VPC.
3. In the bottom pane, go to the **Permissions** tab and click **Manage Policy**. The **Manage Group Permissions** page appears.



4. In the **Policy Name** drop-down menu, select **AmazonVPCReadOnlyAccess** and click **Apply Policy**.

   Note

   If you want only a subset of the privileges listed for the policy to apply to your users, edit the list in the **Policy Document** box and click **Apply Policy**.

Custom Policies for Amazon VPC

You can customize the access policies that can be granted to users of your VPC. In the following policy, you are assigning to a group of users permission to launch instances and list the Amazon EC2 and Amazon VPC resources that are available. This policy prevents the users from making any changes to your VPC's layout, routing, or security.

To grant launch instance privileges to your VPC using JSON

- The following policy allows the group to access the desired actions, and denies the group access to any other actions. The users can launch instances, stop instances, start instances, terminate instances, and describe any of the account's resources (i.e., get a list of the resources). The second statement in the policy protects against any other policy that might grant the user access to a wide range of API actions.

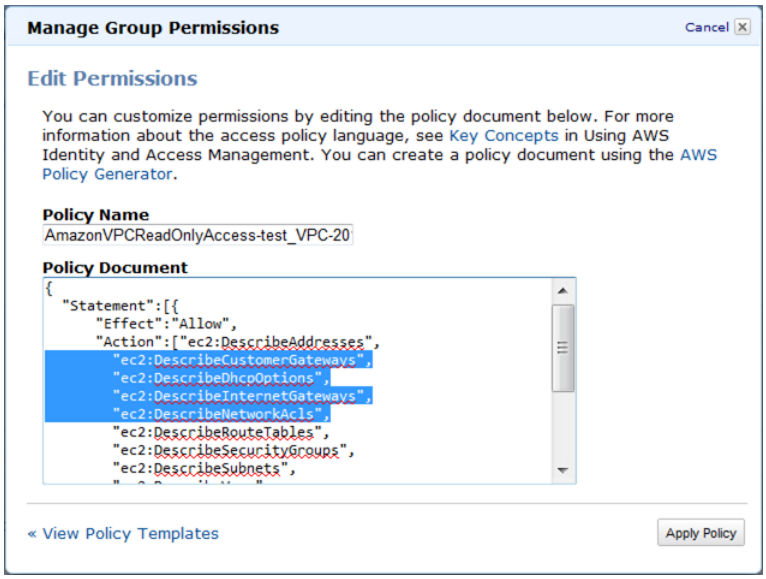
  Note

  The following policy prevents the users from creating or attaching Amazon EBS volumes to instances, or creating snapshots of volumes. It also prevents them from associating Elastic IP addresses with the instances. If the users need those capabilities, you could add the relevant API actions to the policy.

```
{
  "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":"*"
     }
  ]
}
```

To grant launch instance privileges to your VPC using the AWS Management Console

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the **Navigation** pane, click **Groups**, and then select the group that you want to give a defined set of access privileges to your VPC.
3. On the **Permissions** tab in the bottom pane, and click **Manage Policy**. The **Manage Group Permissions** page appears.
4. Click **View Policy Templates**, select **Custom Policy** and click **Select**. The **Manage Group Permissions** page appears.



Note

If you want only a subset of the privileges listed for the policy to apply to your users, edit the list in the **Policy Document** box and click **Apply Policy**.

Working with Current Limitations

You can use IAM policies to specify which Amazon VPC actions a User in your AWS Account can use with Amazon VPC resources in general. However, you can't specify a particular Amazon VPC resource, such as a specific VPC or subnet, in the IAM policy.

Important

Using Amazon VPC with IAM doesn't change how you use Amazon VPC. There are no changes to Amazon VPC actions, and no new Amazon VPC actions related to Users and access control.

For examples of policies that cover Amazon VPC actions, see Managing a VPC (p. 199).

Amazon VPC ARNs

Amazon VPC does not use the Amazon Resource Name (ARN) format because you can't specify a particular Amazon VPC resource in an IAM policy. When writing a policy to control access to Amazon VPC actions, you use the * wildcard as the resource. For more information about ARNs, see ARNs.

Amazon VPC Actions

In an IAM policy, you can specify any actions that Amazon VPC offers. You must prefix them with the lowercase string `ec2:`. For example: `ec2:CreateCustomerGateway`, `ec2:*VpnGateway*`, `ec2:*` (for all Amazon VPC and Amazon EC2 actions). For a list of the actions, refer to the Amazon Elastic Compute Cloud API Reference.

Amazon VPC Keys

Amazon EC2 (and thus Amazon VPC) implements the following AWS-wide policy keys, but no others. For more information about policy keys, see Available Keys.

AWS-Wide Policy Keys

- `aws:CurrentTime` (for date/time conditions)
- `aws:EpochTime` (the date in epoch or UNIX time, for use with date/time conditions)
- `aws:SecureTransport` (Boolean representing whether the request was sent using SSL)
- `aws:SourceIp` (the requester's IP address, for use with IP address conditions)
- `aws:UserAgent` (information about the requester's client application, for use with string conditions)

If you use `aws:SourceIp`, and the request comes from an Amazon EC2 instance, we evaluate the instance's public IP address to determine if access is allowed.

For services that use only SSL, such as Amazon RDS and Amazon Route 53, the `aws:SecureTransport` key has no meaning.

The key names are case insensitive. For example, `aws:CurrentTime` is equivalent to `AWS:currenttime`.

Amazon VPC Resources

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

Resource	Description
Amazon Virtual Private Cloud Getting Started Guide	The getting started guide provides instructions for using the service for the first time.
Amazon Virtual Private Cloud Network Administrator Guide	The network administrator guide gives information a network engineer needs to configure a customer gateway.
Amazon Elastic Compute Cloud Command Line Reference	The command line reference gives complete descriptions of the commands you use with the command line tools.
Amazon Virtual Private Cloud Quick Reference Card	The quick reference card gives a concise listing of the commands you use with the command line tools.
Amazon Elastic Compute Cloud API Reference	The API reference gives the WSDL and schema location; complete descriptions of the API actions, parameters, and data types; and a list of errors that the service returns.
Amazon VPC Release Notes	The release notes give a high-level overview of the current release. They specifically note any new features, corrections, and known issues.
Technical documentation for the Amazon Elastic Compute Cloud	The technical documentation provides a detailed discussion of Amazon EC2. It includes the basics of getting started, an overview of the service, command line reference, programming reference, and API reference.
AWS Developer Resource Center	A central starting point to find documentation, code samples, release notes, and other information to help you build innovative applications with AWS.
Discussion Forums	A community-based forum for developers to discuss technical questions related to Amazon VPC.

Resource	Description
AWS Support Center	The home page for AWS Technical Support, including access to our Developer Forums, Technical FAQs, Service Status page, and Premium Support (if you are subscribed to this program).
AWS Premium Support Information	The primary web page for information about AWS Premium Support, a one-on-one, fast-response support channel to help you build and run applications on AWS Infrastructure Services.
Product information for Amazon VPC	The primary web page for information about Amazon VPC.
Contact Us	A central contact point for inquiries concerning AWS billing, account, events, abuse, etc.
Conditions of Use	Detailed information about the copyright and trademark usage at Amazon.com and other topics.

Appendix A: Recommended Network ACL Rules

Topics

The scenarios presented earlier in this guide use the default network ACL with the default rules. These rules allow all traffic in and out of the subnets, which effectively means you're not using ACLs to provide any additional security for your VPC.

If you want an extra layer of security, this appendix describes the network ACL rules we recommend for the scenarios presented in this guide. For more information about network ACLs and how to use them, see Network ACLs (p. 144).

> **Important**
>
> The following example ACLs list the ephemeral port range as 49152-65535. You might want to use a different range. For more information, see Ephemeral Ports (p. 147).

Recommended Rules for Scenario 1

In scenario 1, you have a single subnet with instances that can receive and send Internet traffic. For a complete discussion of scenario 1, see Scenario 1: VPC with a Public Subnet Only (p. 9).

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

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
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 (e.g., 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

In scenario 2, you have 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 a complete discussion of scenario 2, see Scenario 2: VPC with Public and Private Subnets (p. 16).

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 recommended rules 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

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 (e.g., 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	10.0.0.0/24	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 (e.g., 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

In scenario 3, you have 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 a complete discussion of scenario 3, see Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 44).

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 recommended rules 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 (e.g., 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

In scenario 4, you have a single subnet with instances that can communicate only with your home network over a VPN connection. For a complete discussion of scenario 4, see Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access (p. 84).

The following table shows the recommended rules. 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)

Appendix B: Limits

The following table list the limits for components in your VPC. To request to increase in any of these limits, go to the Amazon VPC Limits form.

Component	Limit	Comments
Number of VPCs per region	5	
Number of subnets per VPC	20	
Number of Internet gateways per region	5	One per VPC
Number of virtual private gateways per region	5	One per VPC
Number of customer gateways per region	50	
Number of VPN connections per region	50	Ten per virtual private gateway
Number of route tables per VPC	10	Including the main route table
Number of entries per route table	20	
Number of VPC Elastic IP addresses per AWS account	5	You have one limit for VPC Elastic IP addresses (5) and another for standard EC2 addresses (5).
Number of VPC security groups per VPC	50	
Number of rules per VPC security group	50	
Number of VPC security groups a VPC instance can be in	5	
Number of network ACLs per VPC	10	
Number of rules per network ACL	20	
Number of BGP Advertised Routes per VPN Connection	100	

Document History

This documentation is associated with the 2011-07-15 release of Amazon VPC. This guide was last updated on 11 September 2012.

The following table describes the important changes since the last release of the Amazon VPC documentation set.

Change	Description	Release Date
AWS VPN CloudHub and redundant VPN connections	With this release, the user guide has been updated with information about AWS VPN CloudHub, which you can use to securely communicate from one site to another with or without a VPC, and updated with information about using redundant VPN connections to provide a fault-tolerant connection to your VPC.	29 September 2011
VPC Everywhere	With this release, the user guide has been rewritten to reflect the new features available in the 2011-07-15 API version.	03 August 2011
Dedicated Instances	With this release, the user guide has been updated with information about Dedicated Instances, what they are, and how you create and use them.	27 March 2011
Redesign of the Guide	With this release, the user guide has been rewritten to reflect the new features available in the 2011-01-01 API version.	11 March 2011

Glossary

Amazon Machine Image (AMI)	An Amazon Machine Image (AMI) is an encrypted machine image stored in the Amazon Simple Storage Service. It contains all the information necessary to boot instances of your software.
Availability Zone	A distinct location within a Region that is engineered to be insulated from failures in other Availability Zones and provides inexpensive, low latency network connectivity to other Availability Zones in the same Region.
BGP ASN	Border Gateway Protocol (BGP) Autonomous System Number (ASN). A unique identifier for a network, for use in BGP routing. Amazon EC2 supports all 2-byte ASN numbers in the range of 1 - 65334, with the exception of 7224, which is reserved.
customer gateway	An Amazon VPC customer gateway is your side of a VPN connection that maintains connectivity. The customer gateway can be either a physical device or software appliance. The internal interfaces of the customer gateway connect to your data center and the external interfaces connect to the VPN connection, which leads to the VPN gateway in the AWS cloud.
default network ACL	The default network ACL is used automatically by any new subnet. You can associate the subnet with a different network ACL of your choice. You cannot change which network ACL in your VPC is the default network ACL.
DHCP options	The Dynamic Host Configuration Protocol (DHCP) provides a standard for passing configuration information to hosts on a TCP/IP network. You can create a set of DHCP options for your VPC instances to use. For example, you can specify the IP addresses of one or more DNS servers that you want your VPC instances to use.
Elastic IP Address	A static, public IP address that you can assign to any instance in a VPC, thereby making the instance public. Elastic IP addresses also enable you to mask instance failures by rapidly remapping your public IP addresses to any instance in the VPC.
instance	After you launch an Amazon Machine Image (AMI), the resulting running system is referred to as an instance.
main route table	The default route table that any new subnet automatically uses for routing. You can associate the subnet with a different route table of your choice. You can also change which of the VPC's route tables is the main route table.
NAT instance	An instance that's configured to perform Network Address Translation (NAT) in a VPC. A NAT instance enables private instances in the VPC to initiate Internet-bound traffic, without those instances being directly reachable from the

	Internet. The NAT instance's primary role is actually Port Address Translation (PAT). However, this guide uses the more widely known term NAT when referring to the instance.
network ACL	An optional layer of security that acts as a firewall for controlling traffic in and out of a subnet. You can associate multiple subnets with a single network ACL, but a subnet can be associated with only one network ACL at a time.
private subnet	A VPC subnet that you've configured for private instances (i.e., instances that do not need to be reachable from the Internet).
public subnet	A VPC subnet that you've configured for public-facing instances (i.e., instances that need to be reachable from the Internet).
Region	A geographical area in which you can launch instances (e.g., US-East (Northern Virginia) Region).
route table	A group of routing rules that controls the traffic leaving any subnet that is associated with the route table. You can associate multiple subnets with a single route table, but a subnet can be associated with only one route table at a time.
security group	A group of firewall rules that control ingress and egress for one or more instances in a VPC.
source/destination checking	Each EC2 instance performs source and destination checking by default. This means the instance must be the source or destination of any traffic it sends or receives. However, the NAT instance needs to be able to send and receive traffic where the source or destination is not itself. To enable that behavior, you must disable source/destination checking on the NAT instance.
subnet	An Amazon VPC subnet is a segment of a VPC's IP address range that Amazon EC2 instances can be attached to. Subnets enable you to group instances based on security and operational needs.
tunnel	Transmission of private network data through a public network (e.g., the Internet) in such a way that the public network's routing nodes are unaware that the transmission is part of a private network.
VPC	A VPC is an isolated portion of the AWS cloud. You define a VPC's IP address space from a range you select. **Note** Wherever the initials VPC stand alone, they refer to a specific network and not the Amazon VPC product.
VPN connection	An Amazon VPC VPN connection is a connection between your VPC and data center, home network, or co-location facility. A VPN connection has two endpoints (or anchors): a customer gateway and VPN gateway. Although VPN connection is a general term, throughout the documentation we specifically mean the connection between a VPC and your own network.
virtual private gateway	An Amazon VPC virtual private gateway is the Amazon side of a VPN connection that maintains connectivity. The internal interfaces of the virtual private gateway connect to your VPC via the VPN attachment and the external interfaces connect to the VPN connection, which leads to the customer gateway.

Index

V

W