Your First Virtual Private Cloud

Table of Contents

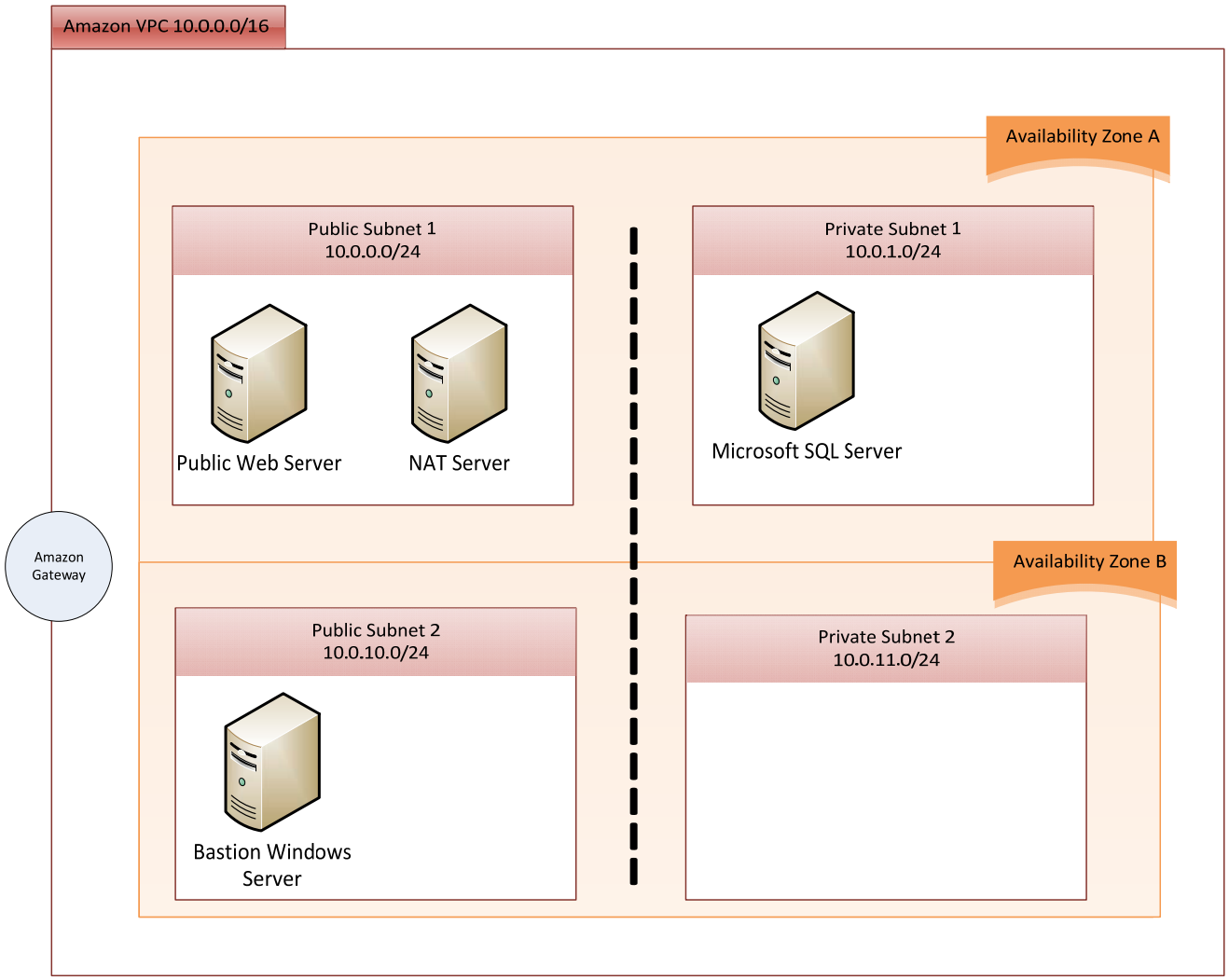
Introduction

Amazon Virtual Private Cloud (Amazon VPC)

Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the Amazon Web Services (AWS) Cloud where you can launch AWS resources in a virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways.

In this lab session, we are going to create a basic VPC using one of the web console wizards, and then to extend it manually to configure it to our needs.

The diagram below represents the network infrastructure that we will build.



This VPC is designed to incorporate several basic features:

- It spans two Availability Zones (AZs). This allows deploying applications across the AZs, therefore ensuring application's durability and availability.
- Within each Availability Zone (AZ) there are two subnets: one "public" subnet is connected directly to the Internet. The other "private" subnet is able to communicate with any other subnet within the VPC; however there is no access to private subnets from the Internet. The dashed line indicates this isolation. Each subnet has its own IP addresses range.
- We'll walk through one way to allow external access to servers that are in the private subnets: the technique of bastion hosts. (Another technique would be to use a dedicated VPN server in the public subnet)

Start your *qwikLAB™*

1. Start your qwikLAB™
   Use the 'Start Lab' button to start your lab.
   (Hint: If you are prompted for a token, please use one you've been given or have purchased.)

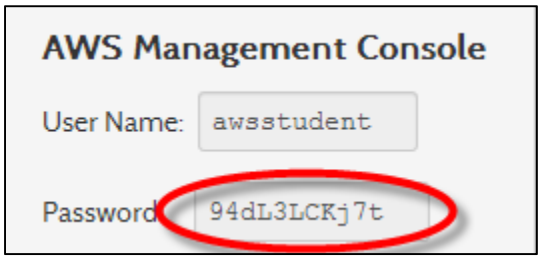
   You will see the lab creation in progress.

2. Note a few properties of the lab.

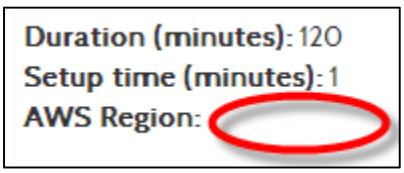
   a. **Duration -** The time the lab will run for before shutting itself down.
   b. **Setup Time -** The estimated lab creation time on starting the lab.
   c. **AWS Region** - The AWS Region the lab resources are being created in.

3. Copy the Password provided.

   d. Hint: selecting the value shown and using Ctrl+C works best



4. Note the AWS Region set for your lab in *qwikLAB™*



5. Click the ‘Open Console’ button.

6. Make sure that you are not logged into any other instances of the AWS console (in a student account or your own account), as this may cause conflicts when you open the console and log in below for this lab.

7. Login to the AWS Management Console

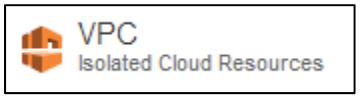
   Enter the User Name ‘**awsstudent**’ and paste the password you copied from the lab details in *qwikLAB*™ into the Password field.

   Click on the ‘Sign in using our secure server’ button.

   In this step you logged into the AWS Management Console using login credentials for a user provisioned via AWS Identity Access Management in an AWS account by *qwikLAB*™.

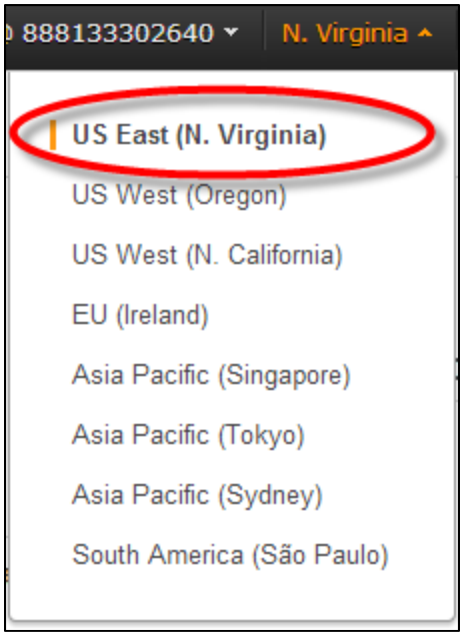
AWS Management Console

8. Once logged in, select "VPC" as from the service console.



Confirm your AWS Region

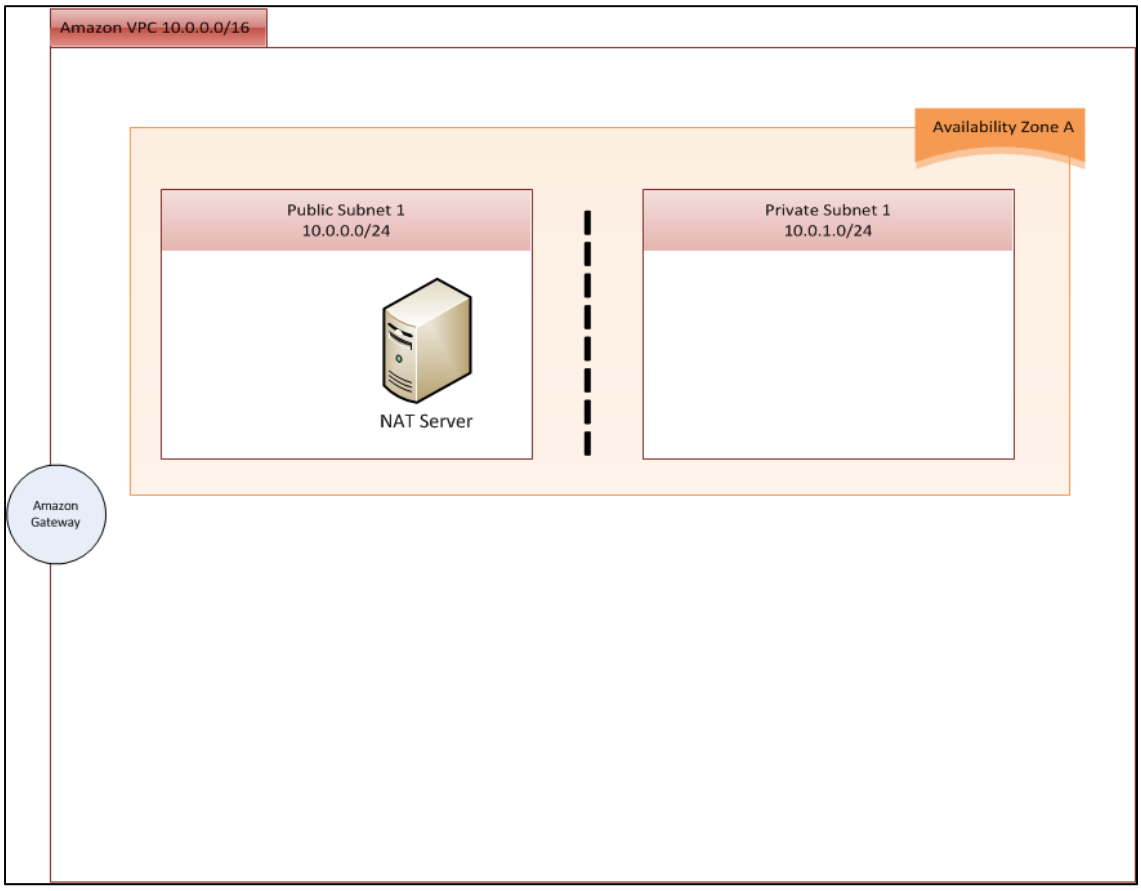
9. Select or confirm that the same AWS Region is already set in the AWS Management Console



Create the Base VPC

We'll use a wizard to set up the initial VPC, and then we'll extend the result manually.

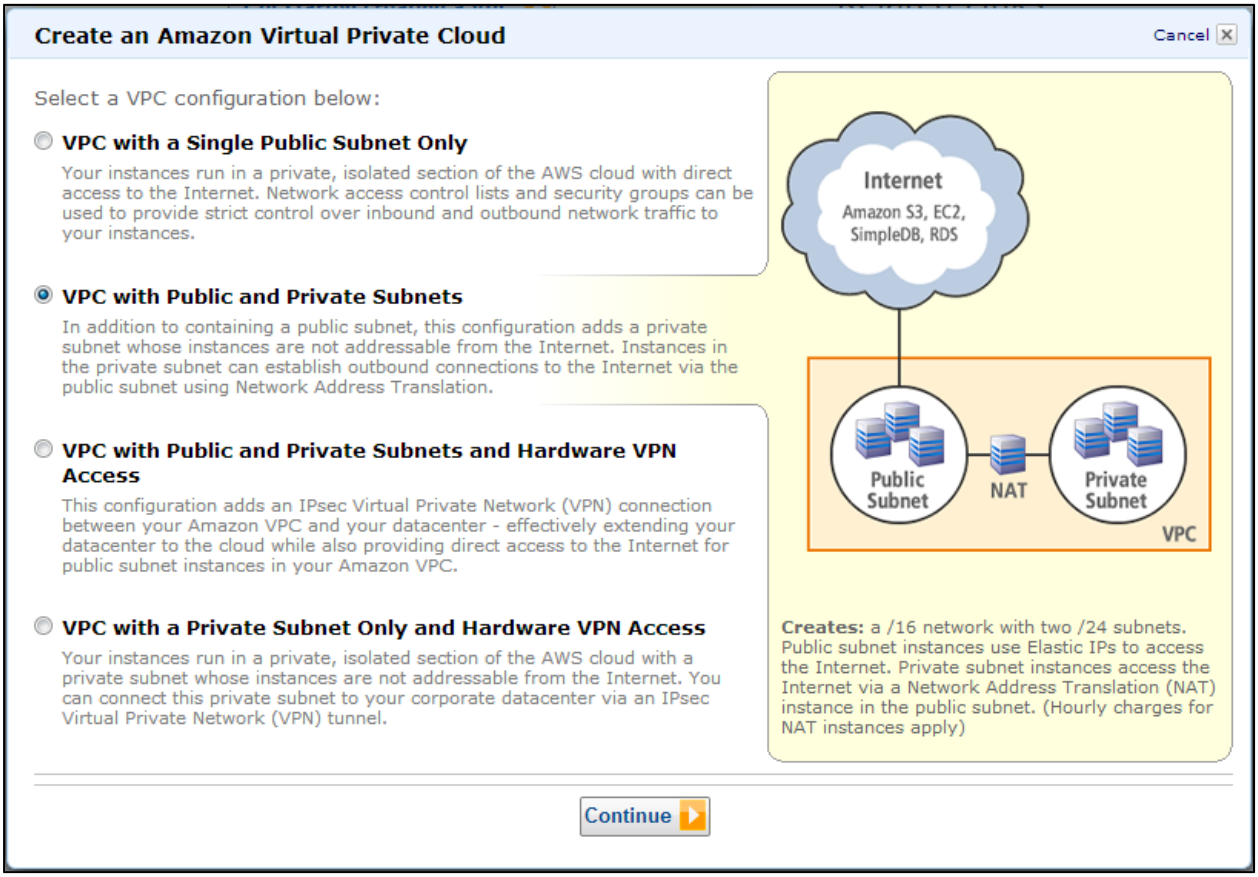
Initially we will create this:



We'll use the wizard to set up the initial network, which is fast and easy, but that won't teach you very much about VPC.

1. Click the Get started creating a VPC button.

2. Choose the second option (VPC with Public and Private Subnets) on the list and click Continue.



This screen contains a lot of parameters. Depending on your professional background, the notation may appear different than what you are used to. This notation is commonly known as CIDR block notation, so, for example, 10.0.1.0/24 can also be expressed as 10.0.1.0 with a subnet mask of 255.255.255.0.

The VPC itself is a Class B network in the 10.0.0.0 space. If you are familiar with the IPv4 address space, this will sound familiar as one of the non-routable address blocks.

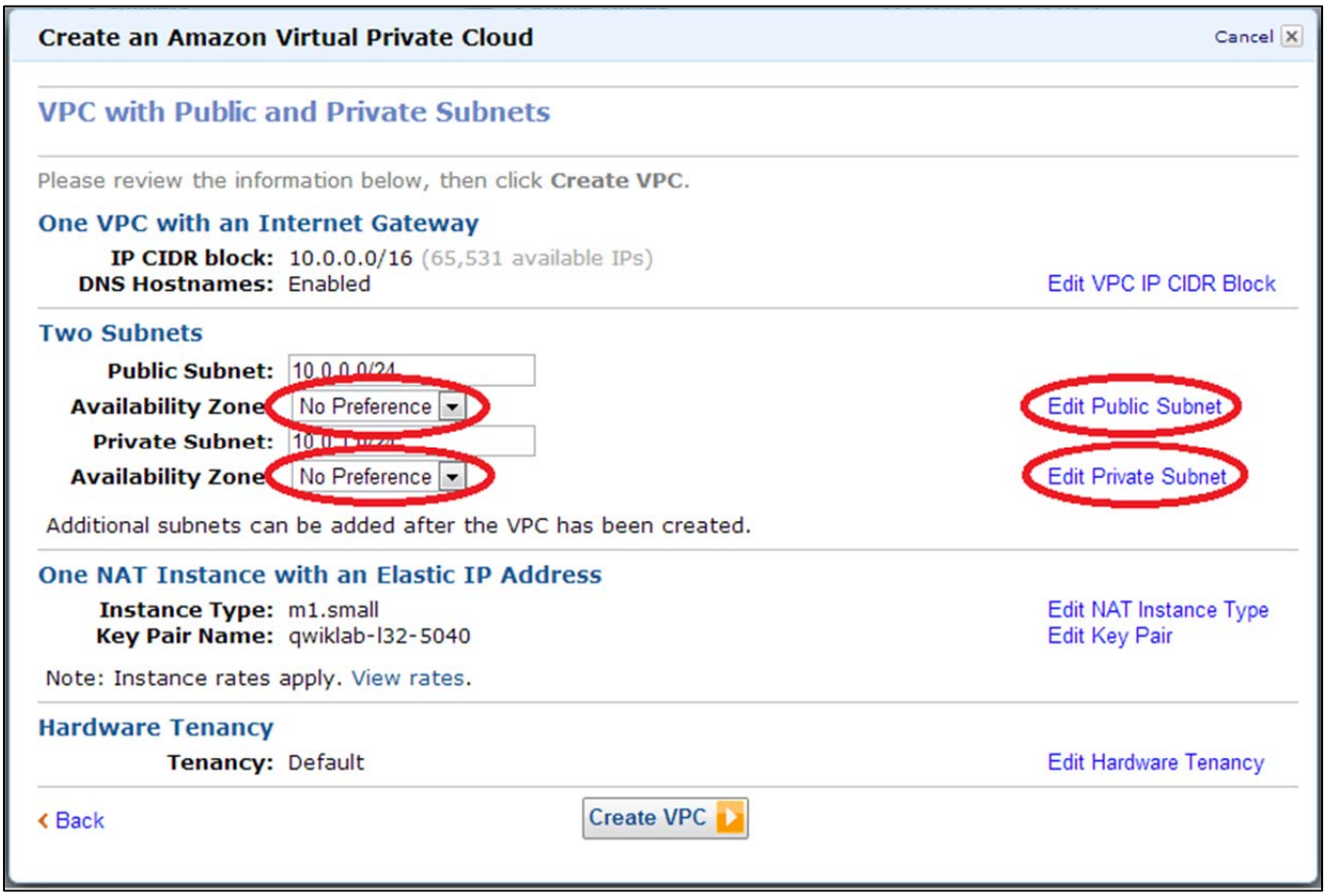
The overall address space uses an IP CIDR block of 10.0.0.0/16, which is the equivalent of a subnet mask of 255.255.0.0 (a full Class B network).

We're going to leave most of this set to the default values, except for two settings.

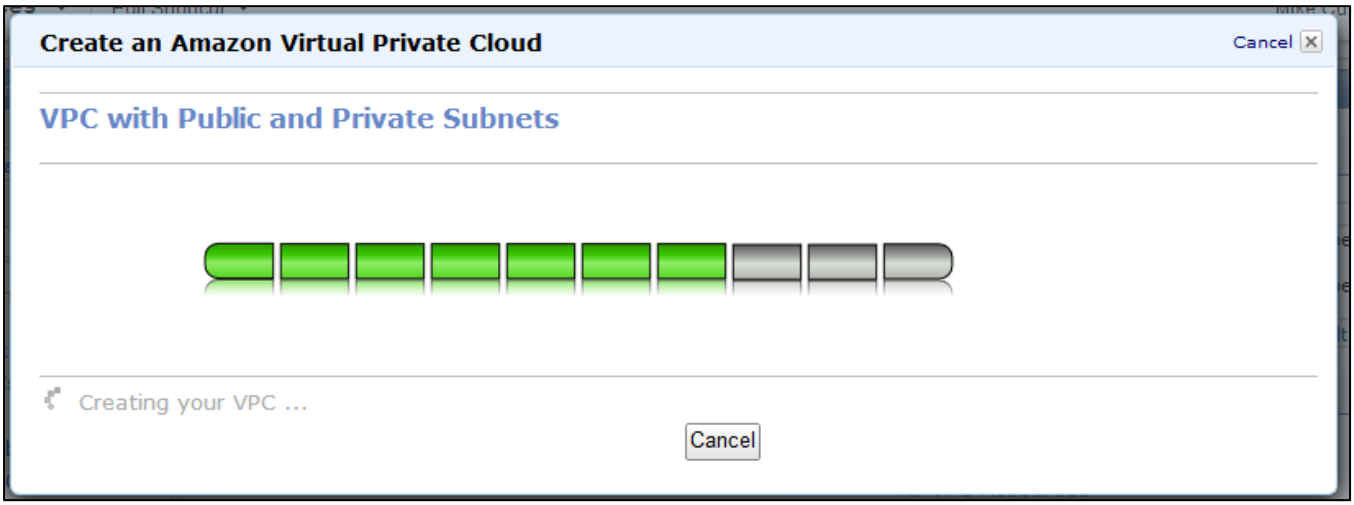
3. Click on Edit Public Subnet. Select any Amazon EC2 Availability Zone.

4. Click on Edit Private Subnet. Select the same Availability Zone as you selected for the Public Subnet.

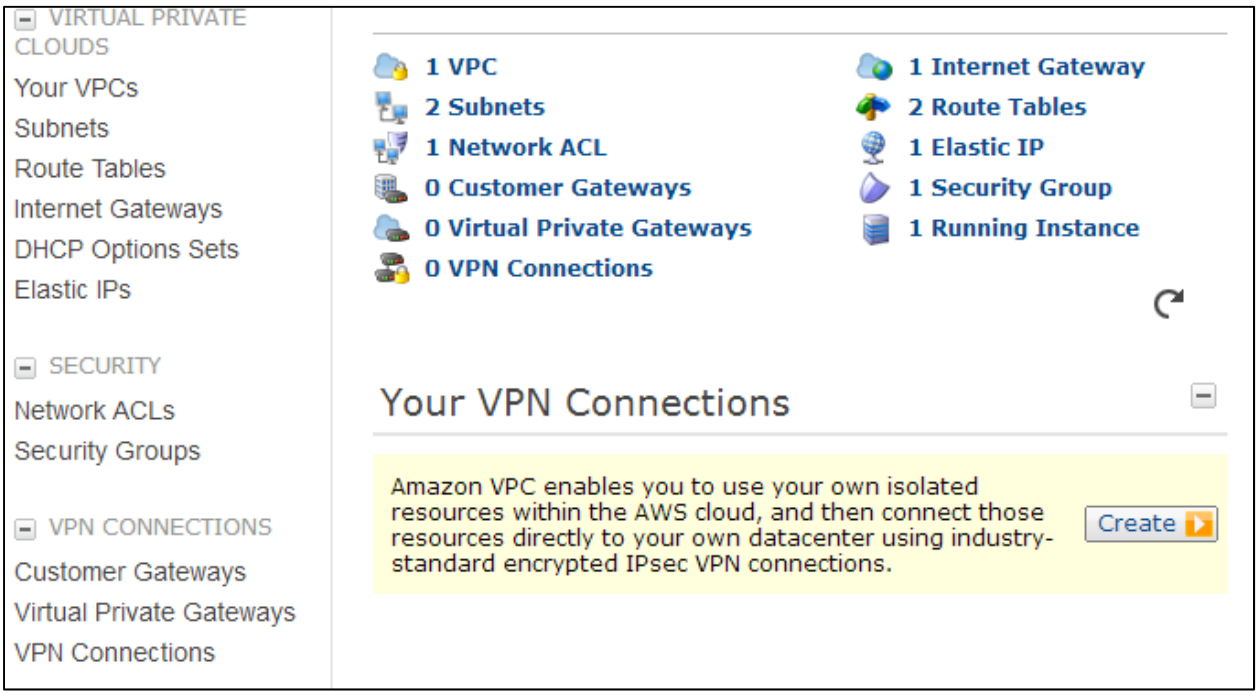
We want to make certain that the subnets are both in the same Amazon EC2 Availability Zone.



5. After you've selected one and the same Availability Zone for both the Public and Private Subnet, create the VPC by clicking on the Create VPC button.



Once finished, you may close the dialog. Back on the VPC Dashboard, we can see the VPC, two subnets, and several other features such as network ACLs and route tables, etc. For the moment all that matters is that the network environment is ready to use.



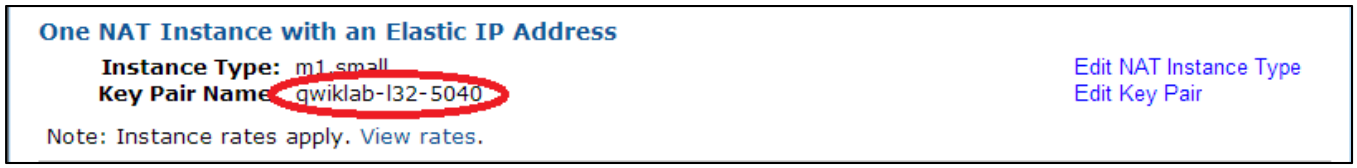
Your VPC does, however, have an important characteristic: everything is in a single Availability Zone. In order to optimize application availability we need to distribute assets across Availability Zones, which means that we'll need to add another pair of subnets. We're going to wait to do that until later in this lab.

NAT Servers are for Outbound Requests

Note that there is already a running Amazon EC2 instance, which is the NAT server that the wizard created. The NAT server is an appliance in the sense that its only purpose is to allow servers in the private subnet to communicate with the Internet in order to get updates, software packages, and so forth. It does not allow Internet clients to make connections to servers in the private subnet. Also note that it is assigned an *Elastic IP*, or NAT (Network Address Translation), address in order to facilitate Internet communication.

The web console wizard has created this NAT Amazon EC2 instance when you created your VPC. It is created from a dedicated AMI provided by Amazon.

By default the instance type is an `m1.small` and the Amazon EC2 Key Pair Name associated with it is one that was generated for you by *qwikLAB*™.



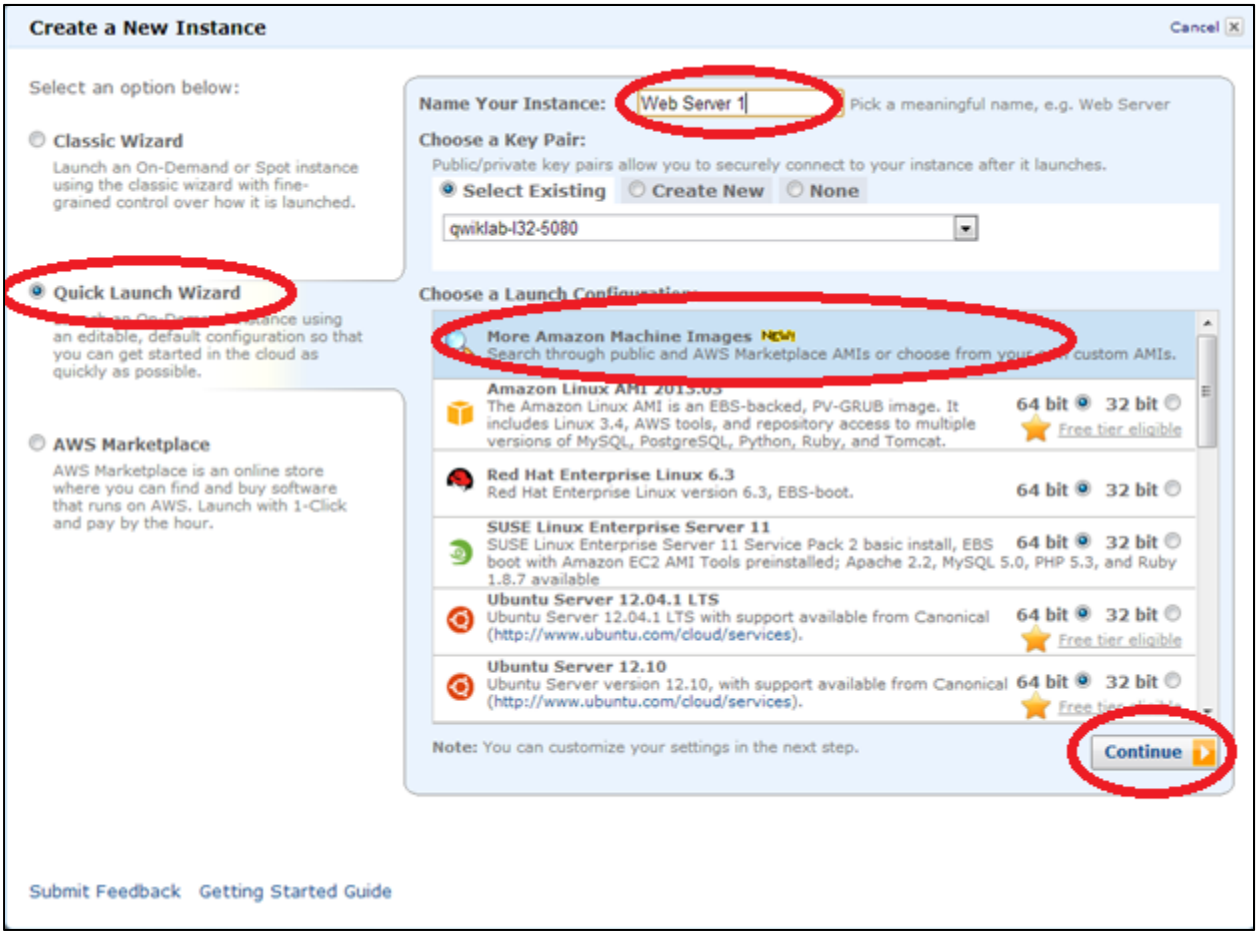
Launch a Web Server

1. Switch to the EC2 Service by clicking on Launch EC2 Instances.

In this lab we're going to launch a BitNami web server as the front-end of our environment. The advantage of this particular AMI is that (a) it was created by a trusted partner, and (b) the Web server will respond to requests with it's default configuration.

2. Click Launch Instance.

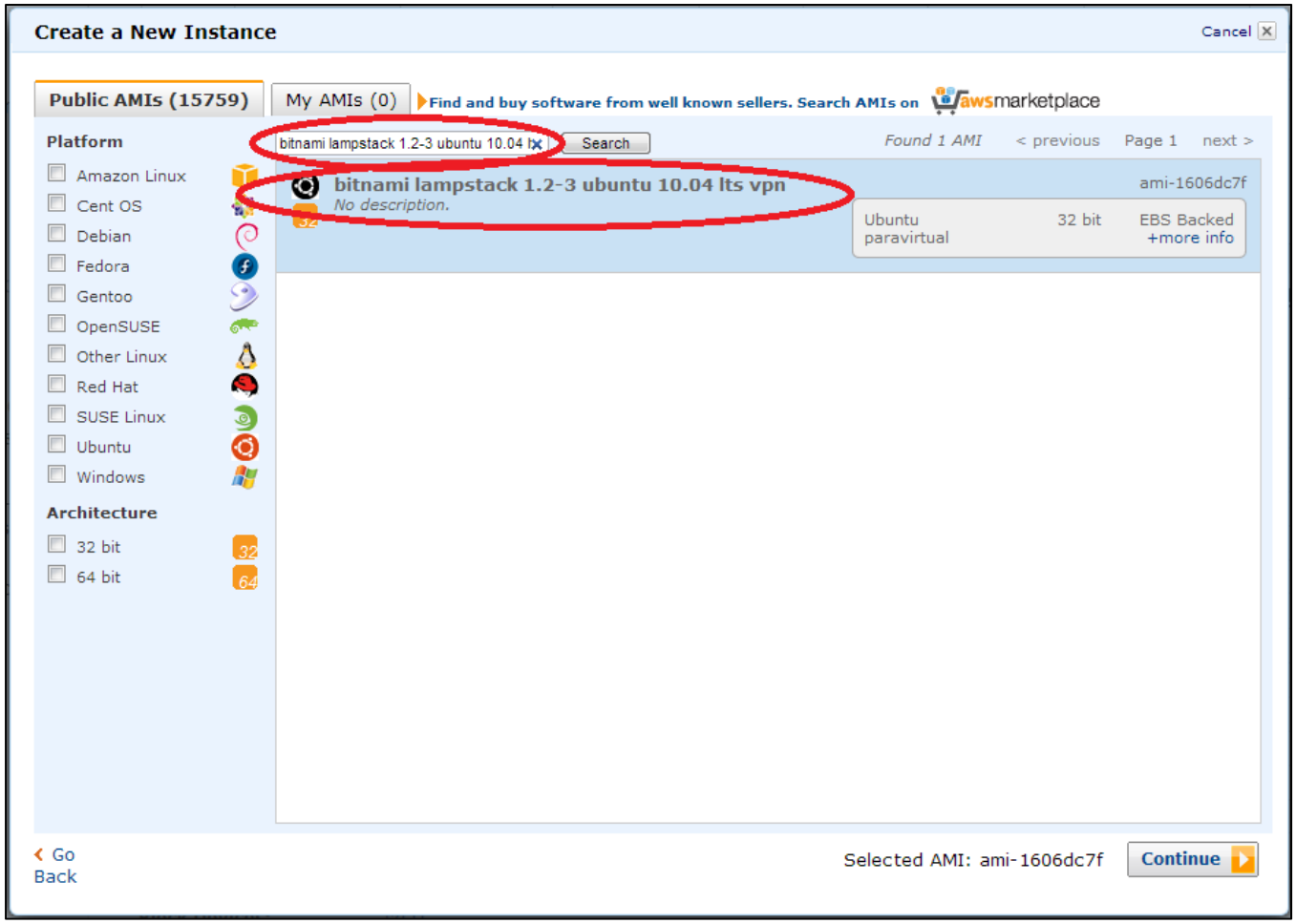
3. Use the Quick-Launch Wizard, with the following additional choices:
    - Name your Instance **Web Server 1**
    - Click on **More Amazon Machine Images**, and then click **Continue**



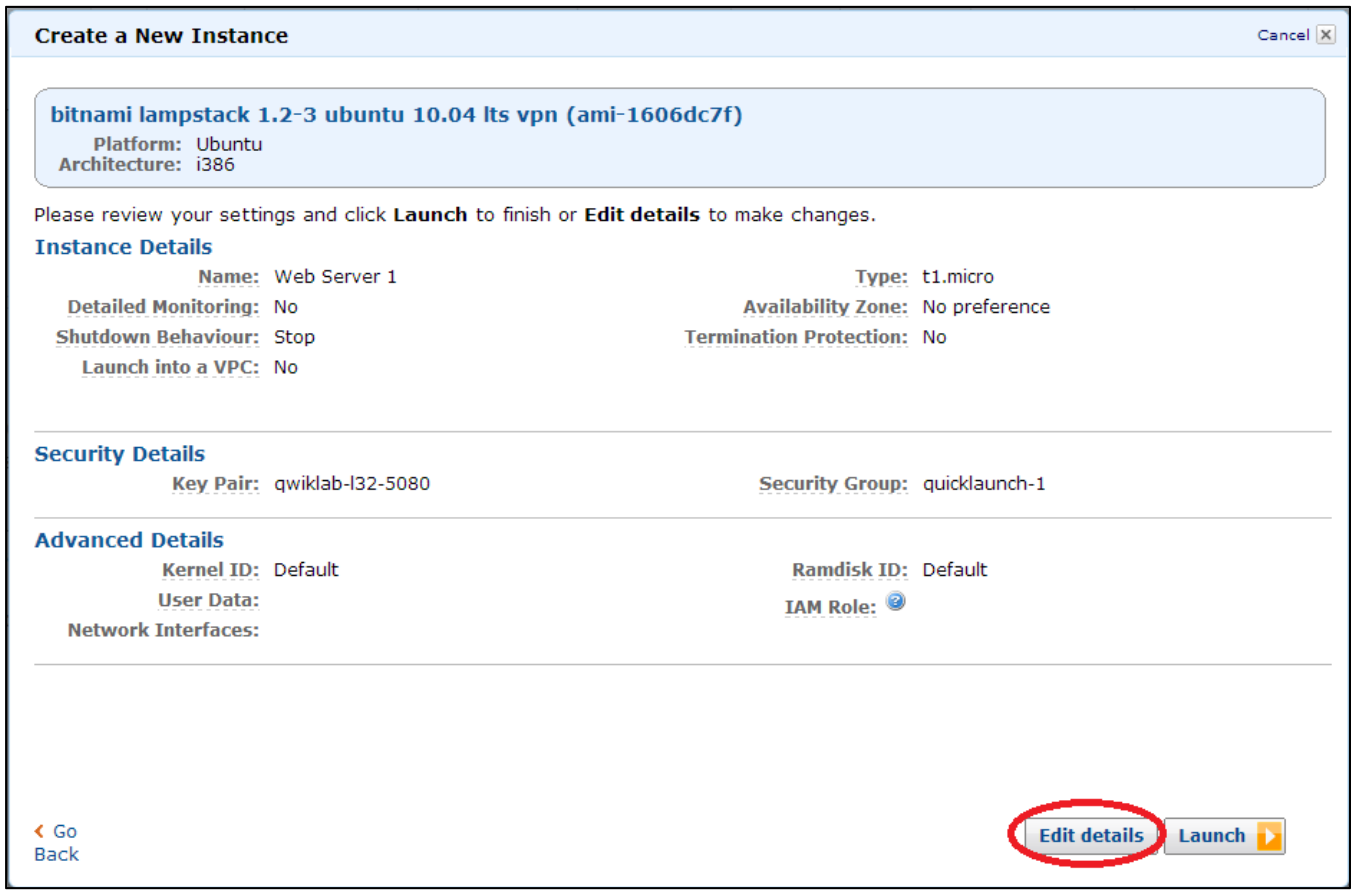
4. On the next screen, copy –paste the following into the search box. You won't even need to click on the search button, because the AMI will simply appear.

   bitnami lampstack 1.2-3 ubuntu 10.04 lts vpn

5. Select the AMI by clicking on the search result, and then click on Continue



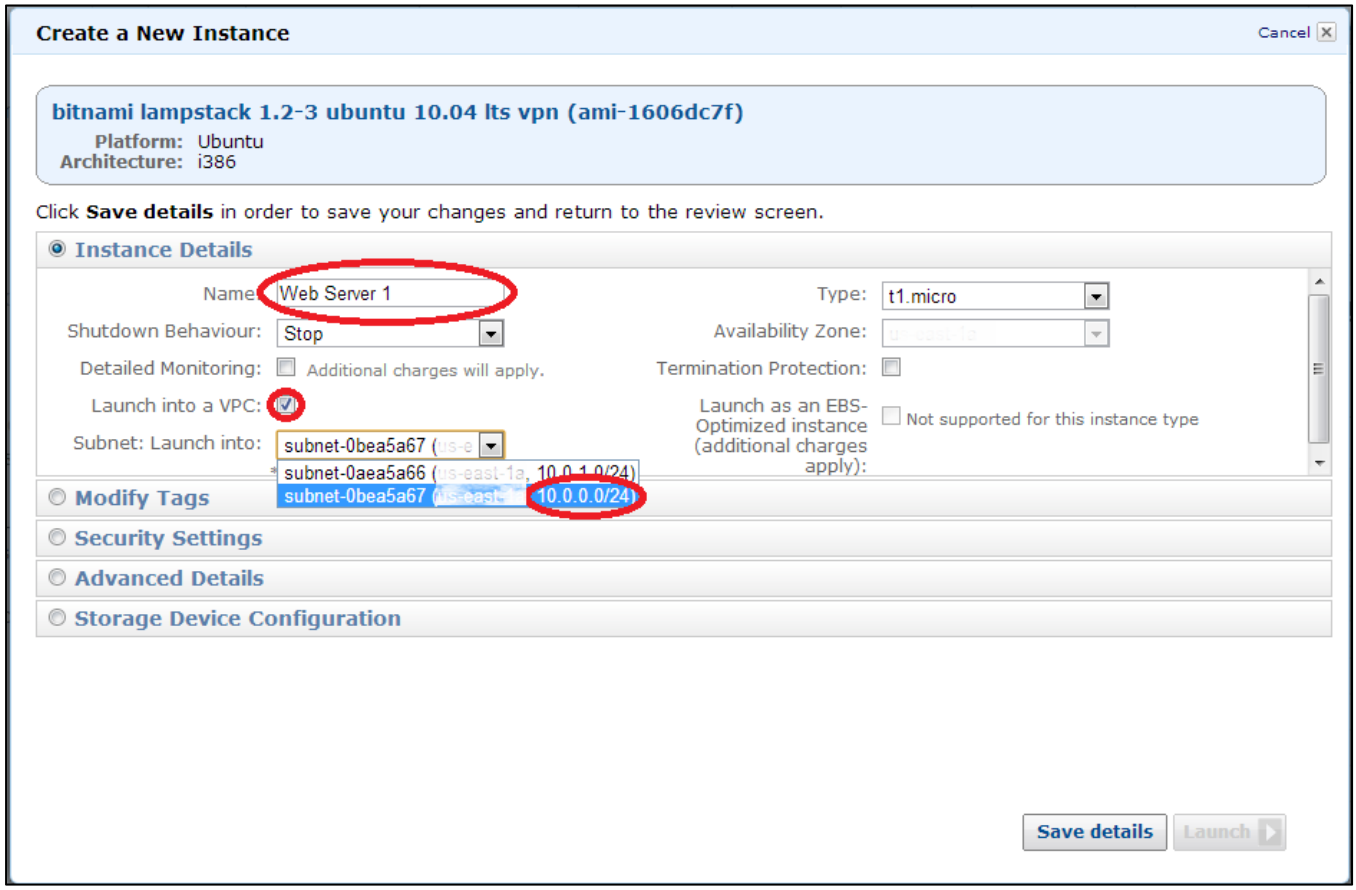
6. Next, Click Edit Details



There are three important things to pay attention to on this screen:

- Enter the server name if for some reason it's blank (it shouldn't)
- Check "**Launch into a VPC**"
- Choose the public Subnet (**10.0.0.0/24**)

7. Do not click on "Save Details". Instead, click on Security Settings.

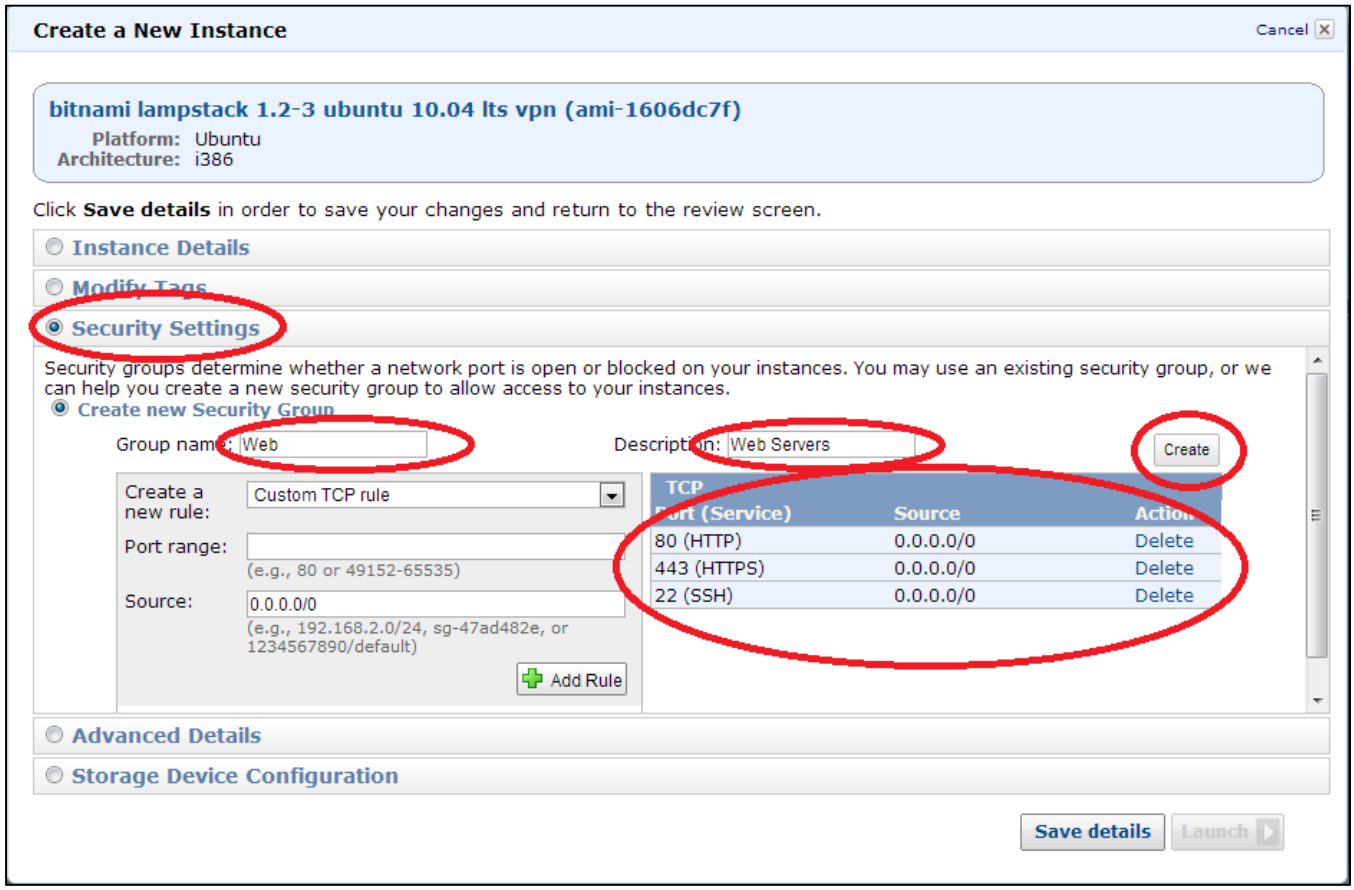


We recommend that you create a custom security group, based on the role of the instance, instead of "selecting an existing Security Group, which is the default.

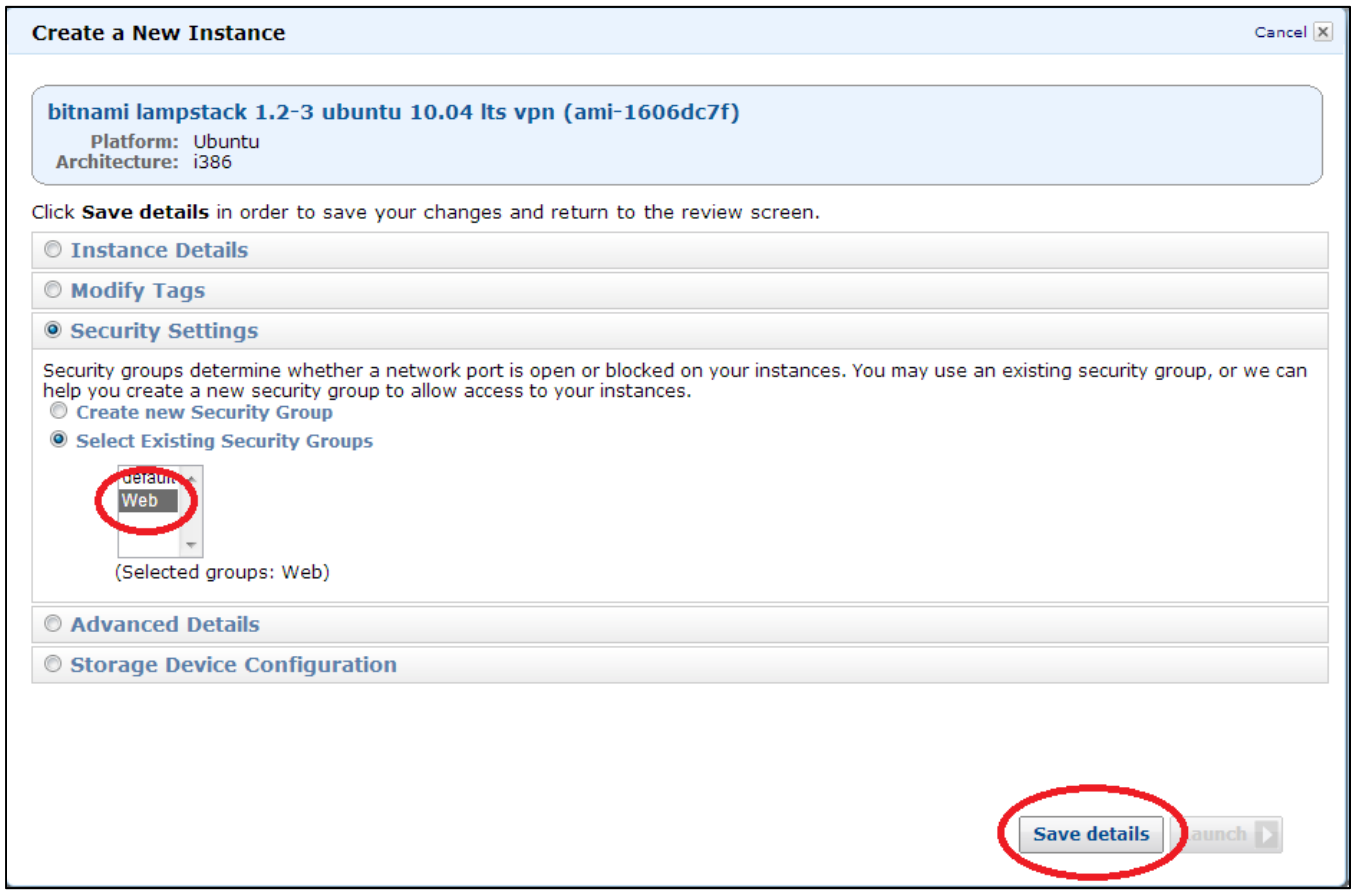
8. Create a new group named "Web", and then add a description.
9. Next, open inbound access for ports 22, 80, and 443. In real life you should restrict Port 22 access to just your own IP address range, or even better, to only allow SSH from a bastion server.

You will need to add each rule, one at a time, which we do not illustrate here. You can choose SSH, HTTP, HTTPS from the dropdown and enter 0.0.0.0/0 in Source.

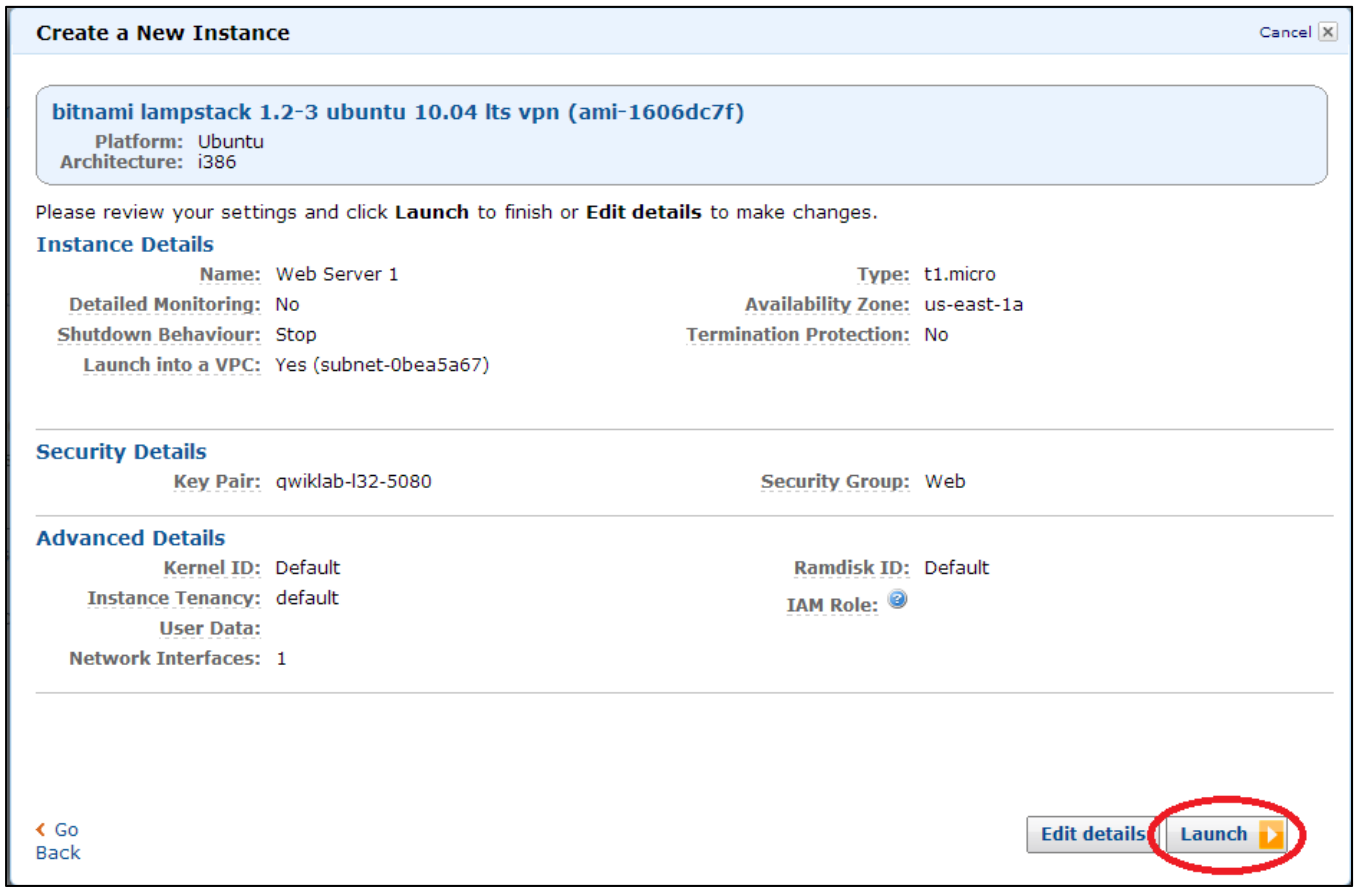
10. Click **Create**.



11. Make sure you select the Security Group you just created and click **Save Details**.



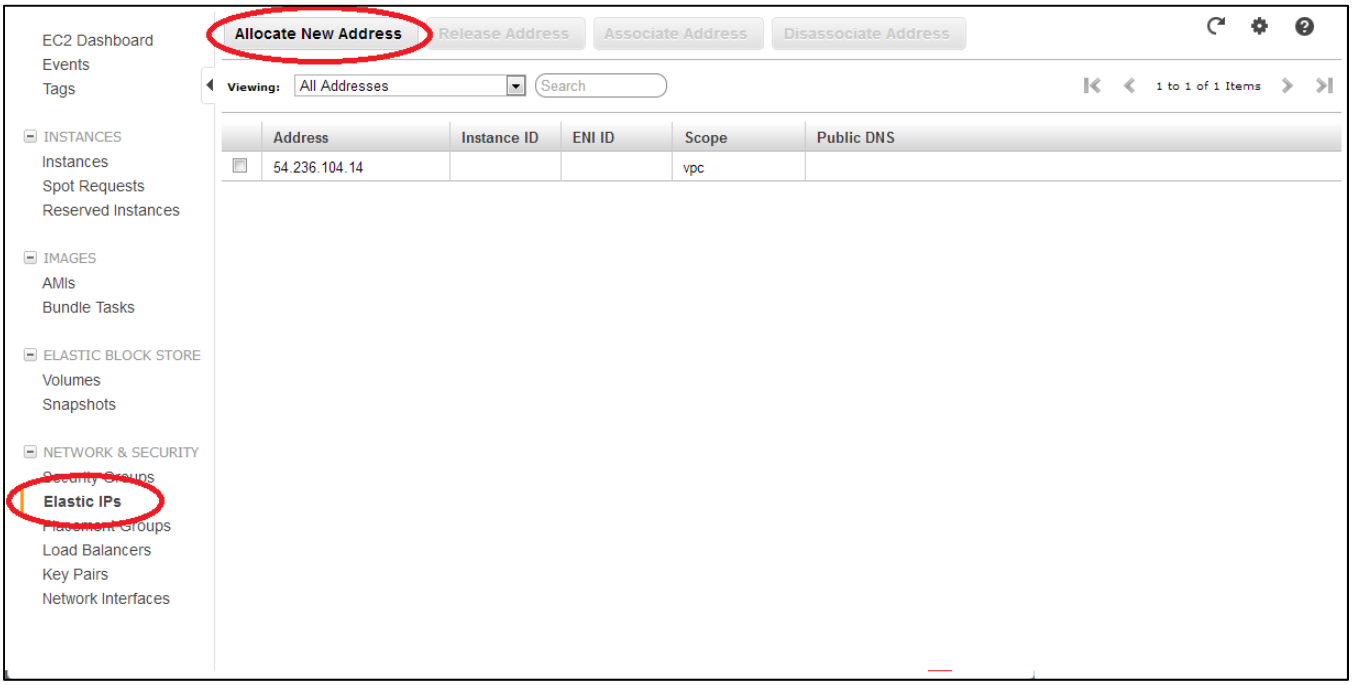
12. Click **Launch** and then **Close**.

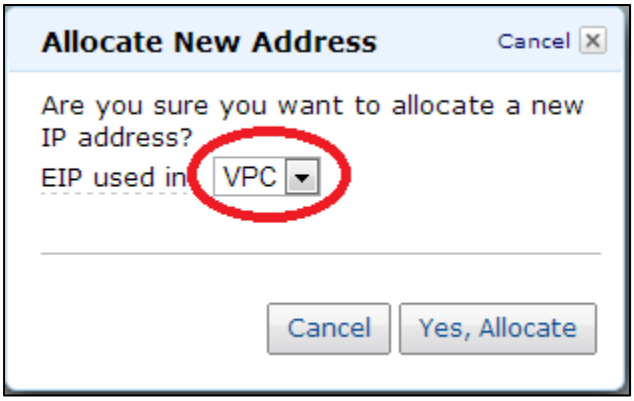


Create and Assign an Elastic IP Address

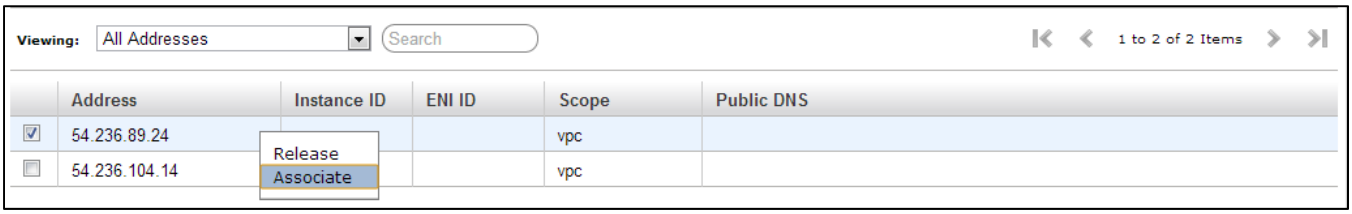
By default instances in the VPC do not have a public IP address. Because this Web server is meant to be public, we need to allocate an Elastic IP address (EIP) and associate it with the server.

1. In the *Elastic IPs* section of the EC2 Console, click on **Allocate New Address**, and make certain that you allocate one in the VPC.



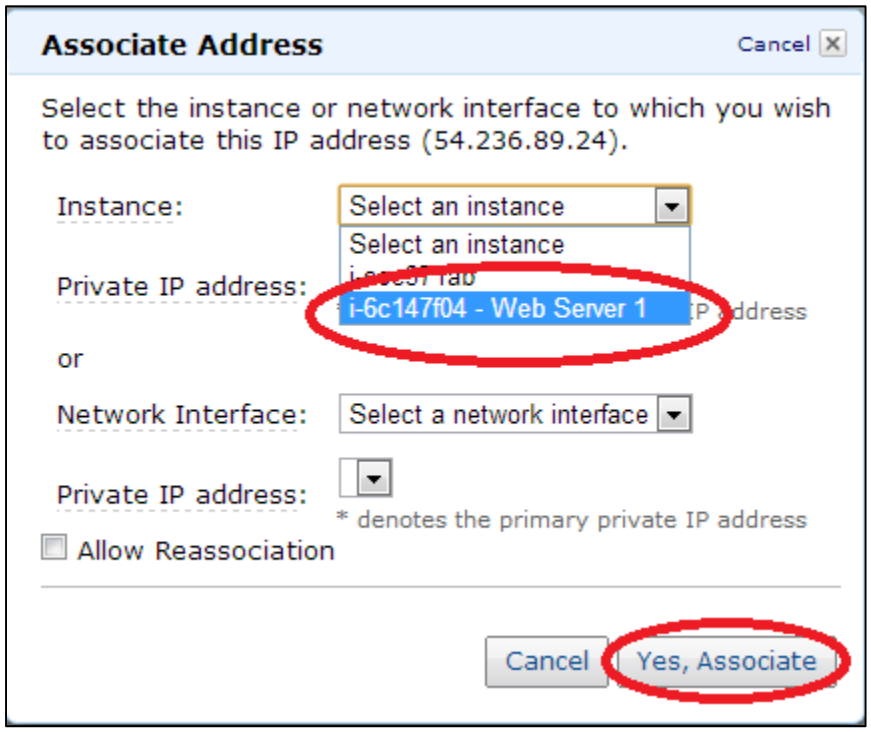


2. Right-click on the new address and click on **Associate**.

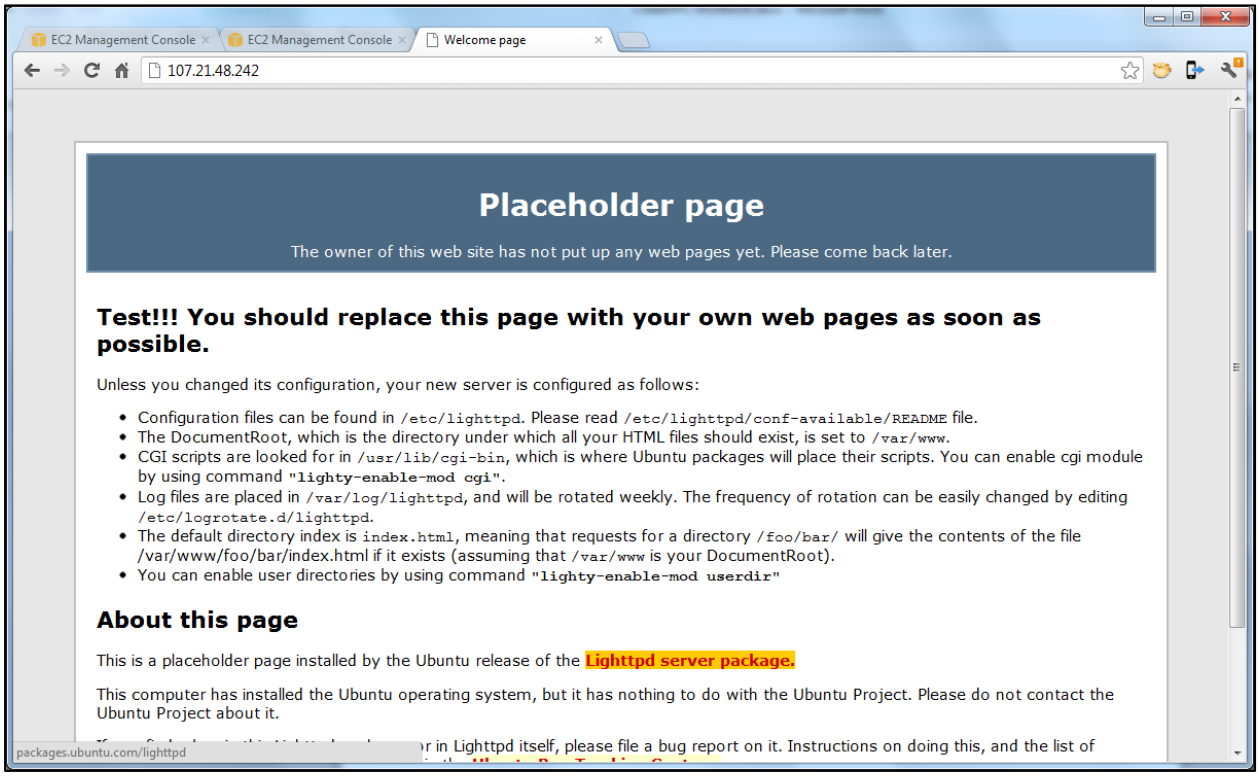


3. Select your **Web Server 1** instance. Be certain that you select the appropriate server, because the options below are unlikely to remember what context you are working with.

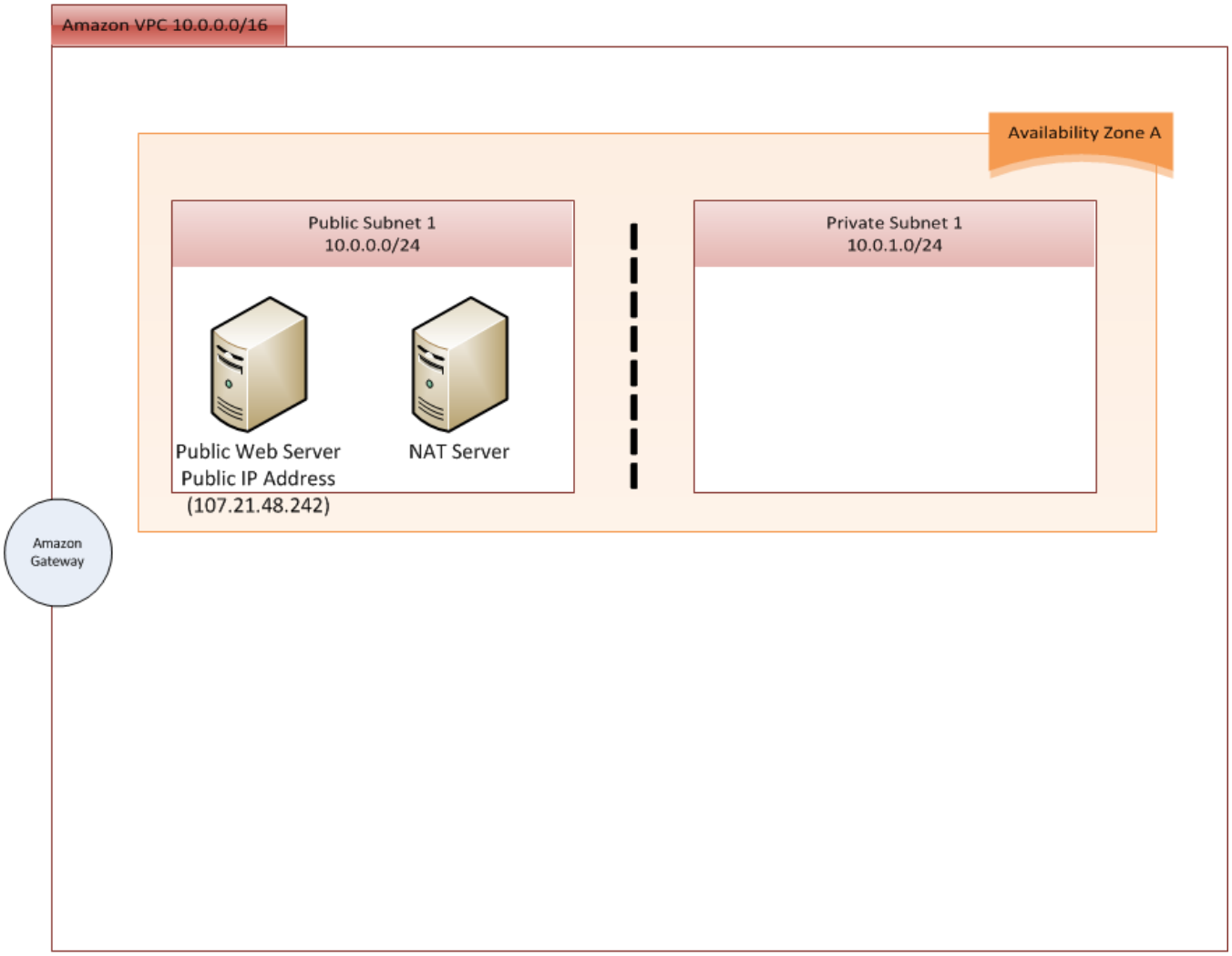
   Make note of this page. Later in the lab exercise you will be asked to do this again, except next time we will not provide screen shots.



4. Try connecting to the Web server from a browser connection by typing the IP address into the page. You will connect to a page similar to the one in this screen shot. We won't modify the Web site, and instead will focus on the network portion of this exercise.



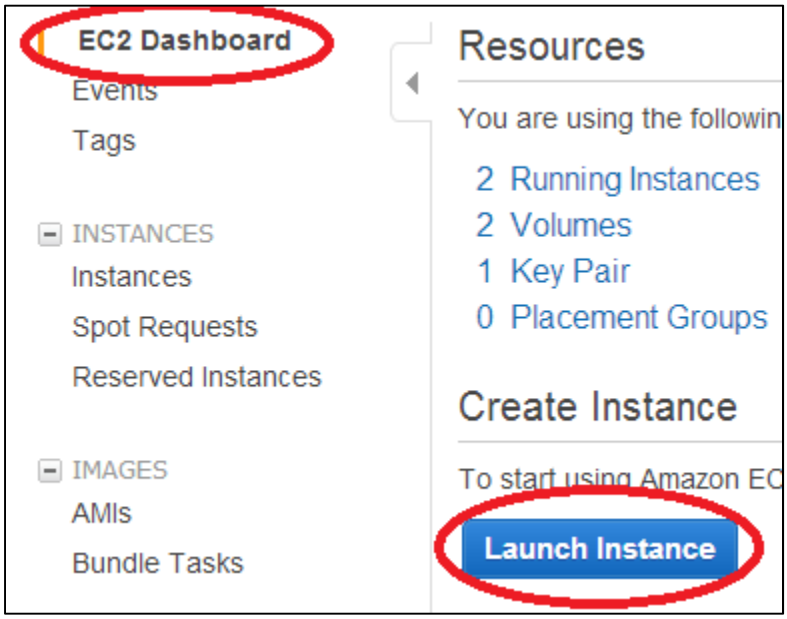
Here are the results so far:



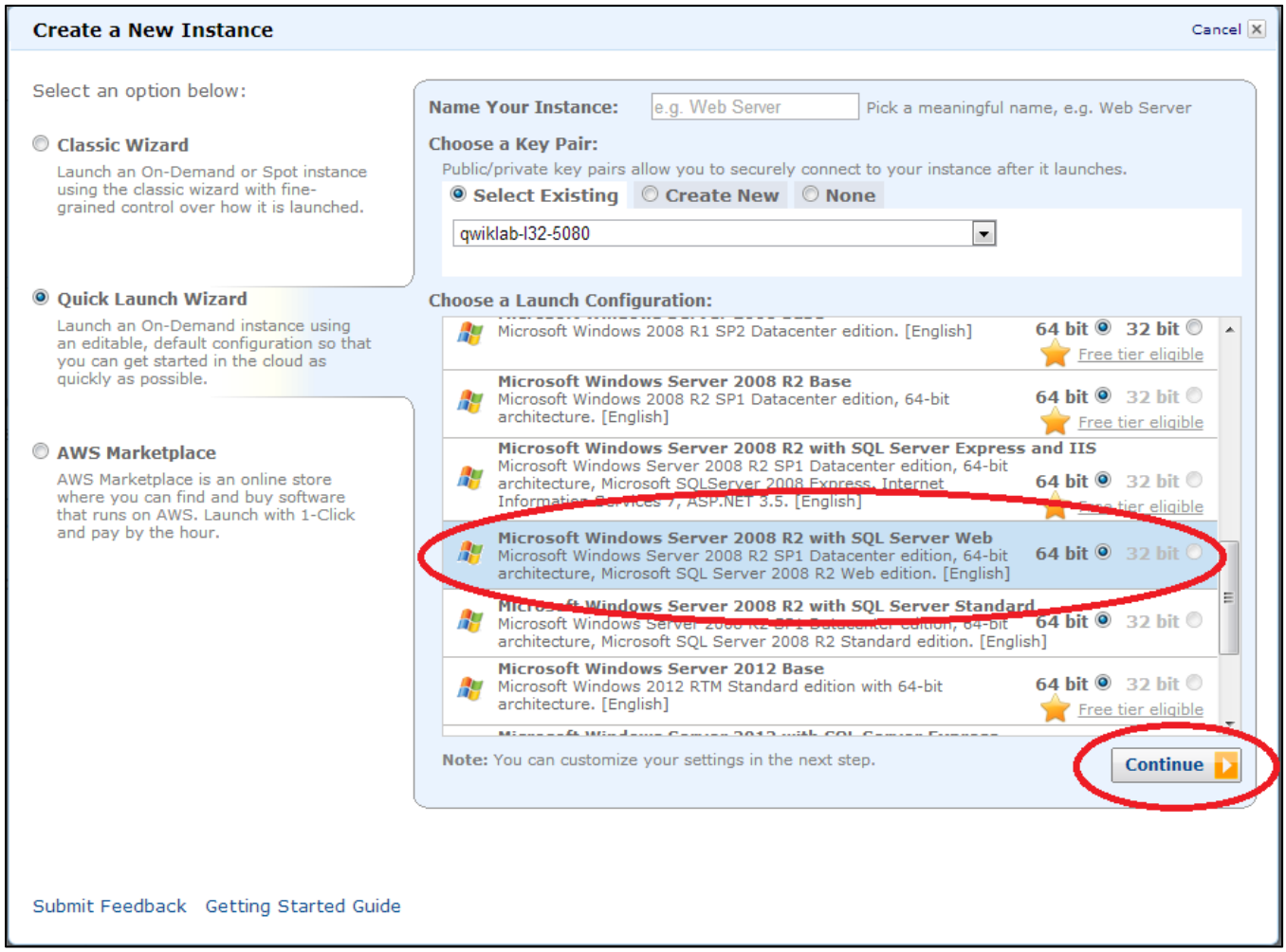
Launch a Back-End Windows SQL Server

In order to increase the security of our database, we're going to place our database in a private subnet, away from Internet traffic.

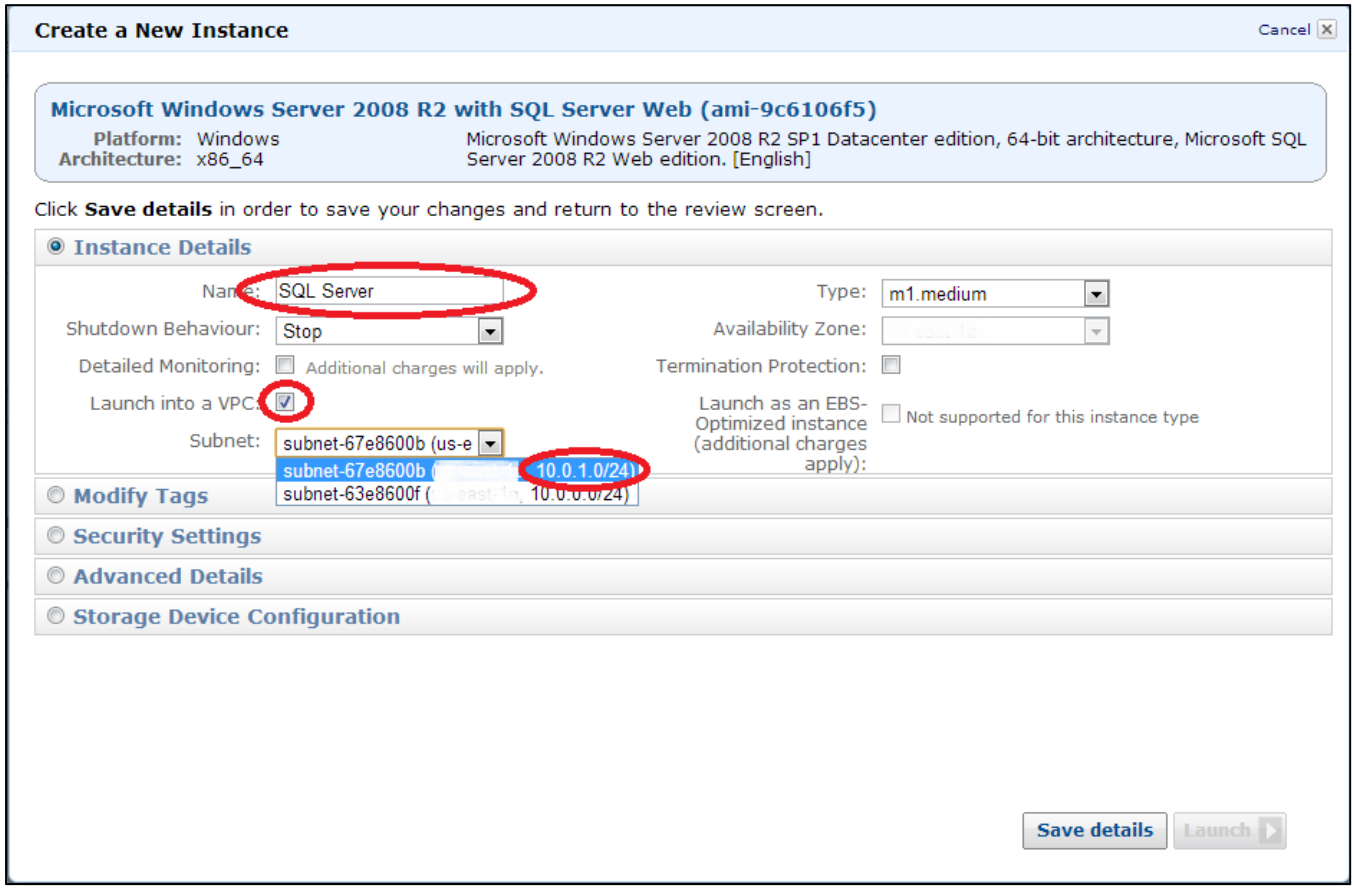
1. In the EC2 Dashboard part of the AWS Management Console, click on the "**Launch Instance**" button and choose the Quick Launch Wizard.



2. Then select the **Microsoft Windows Server 2008 R2 with SQL Server Web** AMI.
   We won't actually use the database in this lab. Rather, our objective is to create a "target server" in the sense that this server will be reachable via RDP under only a select set of conditions.

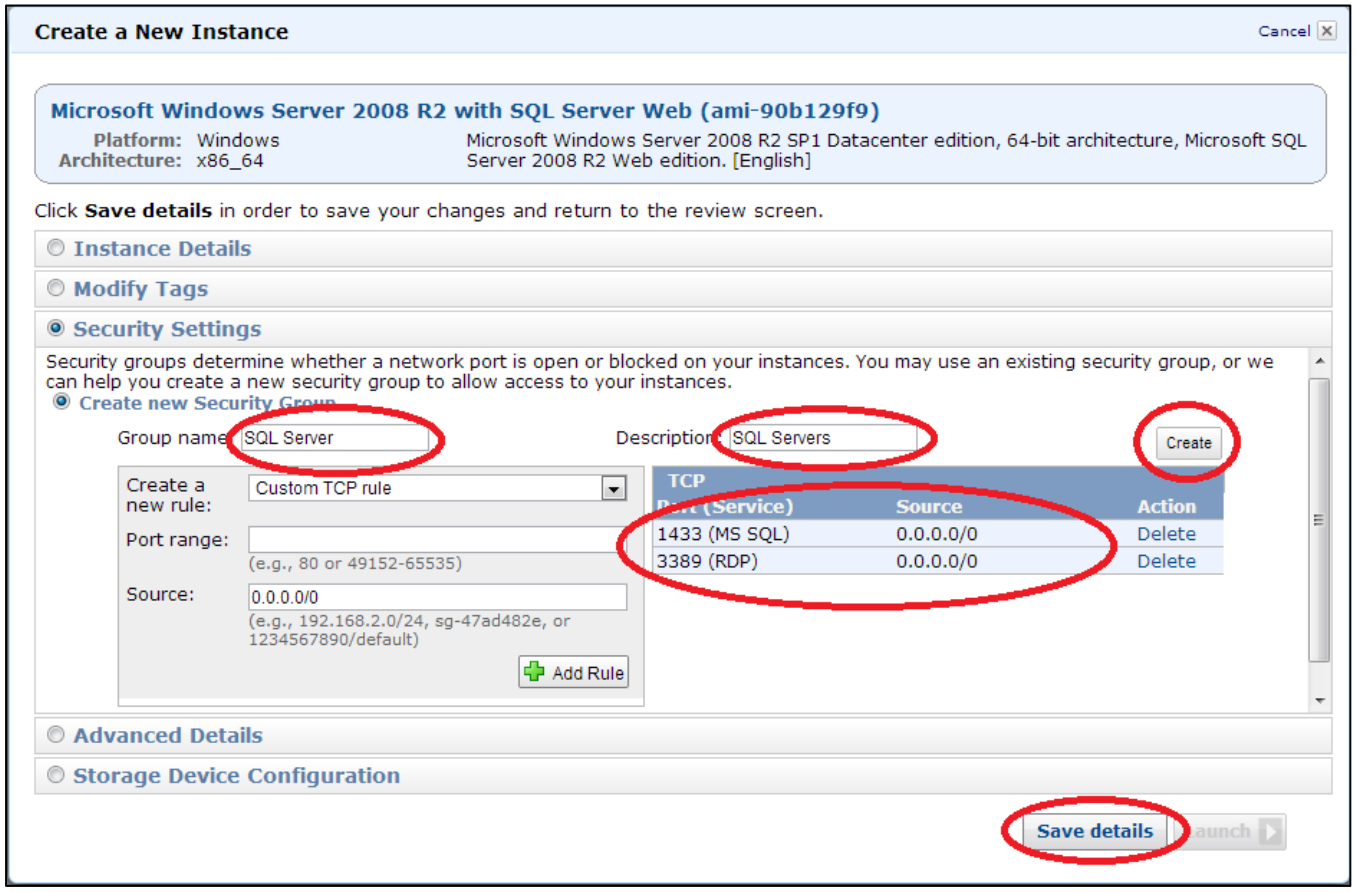

Create a New Instance
Cancel
Select an option below:
Classic Wizard
Launch an On-Demand or Spot instance using the classic wizard with fine-grained control over how it is launched.
Quick Launch Wizard
Launch an On-Demand instance using an editable, default configuration so that you can get started in the cloud as quickly as possible.
AWS Marketplace
AWS Marketplace is an online store where you can find and buy software that runs on AWS. Launch with 1-Click and pay by the hour.
Name Your Instance:
e.g. Web Server
Pick a meaningful name, e.g. Web Server
Choose a Key Pair:
Public/private key pairs allow you to securely connect to your instance after it launches.
Select Existing
Create New
None
qwiklab-l32-5080
Choose a Launch Configuration:
Microsoft Windows 2008 R1 SP2 Datacenter edition. [English]
64 bit
32 bit
Free tier eligible
Microsoft Windows Server 2008 R2 Base
Microsoft Windows 2008 R2 SP1 Datacenter edition, 64-bit architecture. [English]
Microsoft Windows Server 2008 R2 with SQL Server Express and IIS
Microsoft Windows Server 2008 R2 SP1 Datacenter edition, 64-bit architecture, Microsoft SQLServer 2008 Express, Internet Information Services 7, ASP.NET 3.5. [English]
Microsoft Windows Server 2008 R2 with SQL Server Web
Microsoft Windows Server 2008 R2 SP1 Datacenter edition, 64-bit architecture, Microsoft SQL Server 2008 R2 Web edition. [English]
Microsoft Windows Server 2008 R2 with SQL Server Standard
Microsoft Windows Server 2008 R2 SP1 Datacenter edition, 64-bit architecture, Microsoft SQL Server 2008 R2 Standard edition. [English]
Microsoft Windows Server 2012 Base
Microsoft Windows 2012 RTM Standard edition with 64-bit architecture. [English]
Note: You can customize your settings in the next step.
Continue
Submit Feedback
Getting Started Guide

3. Edit the Instance Details.
    a. Name the instance **SQL Server**.
    b. Change the instance type to something larger than a `t1.micro` (we suggest an `m1.medium`).
    c. Select **Launch into a VPC**
    d. Choose **10.0.1.0/24** as the subnet. This subnet is not reachable directly from the Internet.because it is not attached to a VPC Internet Gateway (more on this later in this lab).



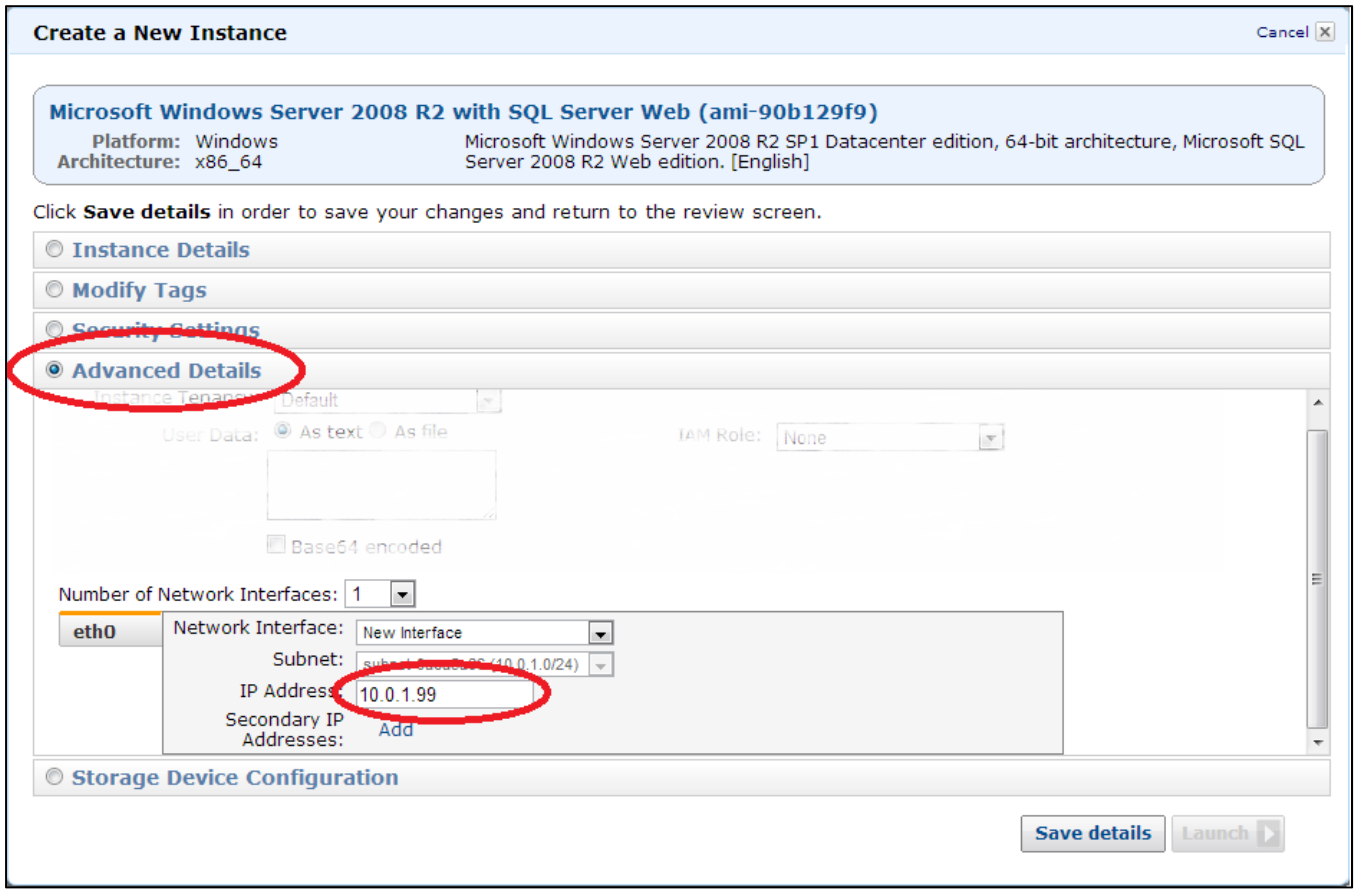
4. Click on **Security Settings** and then **Create new Security Group**. Change the name of the security group to **SQLServer** (no spaces in names) and add a description.

   Note that the source IP address range says 0.0.0.0/0, which means "from anywhere". In fact, the routing restrictions translate this meaning into "from any host, as long as it is on one of the VPC subnets". We'll tighten this rule in a few minutes, when the Bastion Server will be created.



5. Select the **SQLServer** Security Group you've just created and click **Save details**. Do not click the Launch button yet!

6. Click **Edit details** again There will be additional options under **Advanced Details**.
7. In Advanced Details for the instance specify **10.0.1.99** as the address for this server.

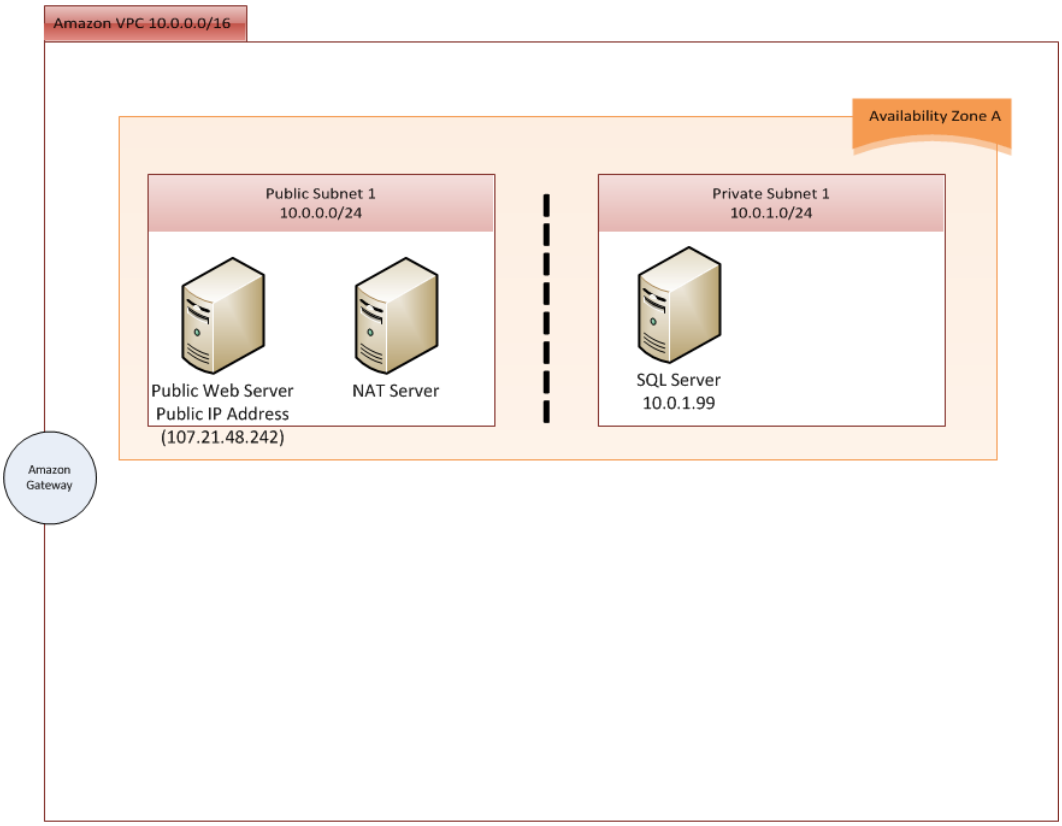


8. Click **Save details** and then **Launch**.

Our network now looks like the diagram below. It's still not ready for production, because the database is not set up to serve the Web server, and we still need a secure way to connect to and to administer the SQL Server. However, the NAT server stands ready to act as a router that allows the SQL Server to make outbound calls to the Internet in order to download Windows Updates, etc.

There is one other very important thing missing from our environment: a second Availability Zone with another Web server and a second database server in it. AWS provides you access to multiple Availability Zones (data centers) at no additional cost to you. A best practice is to mirror servers across at least two zones, and then use load balancing and other techniques to distribute traffic between them.

Amazon operates state-of-the-art, highly available data centers. Although rare, failures can occur that affect the availability of instances that are in the same location. If you host all your instances in a single location that is affected by such a failure, none of your instances would be available.

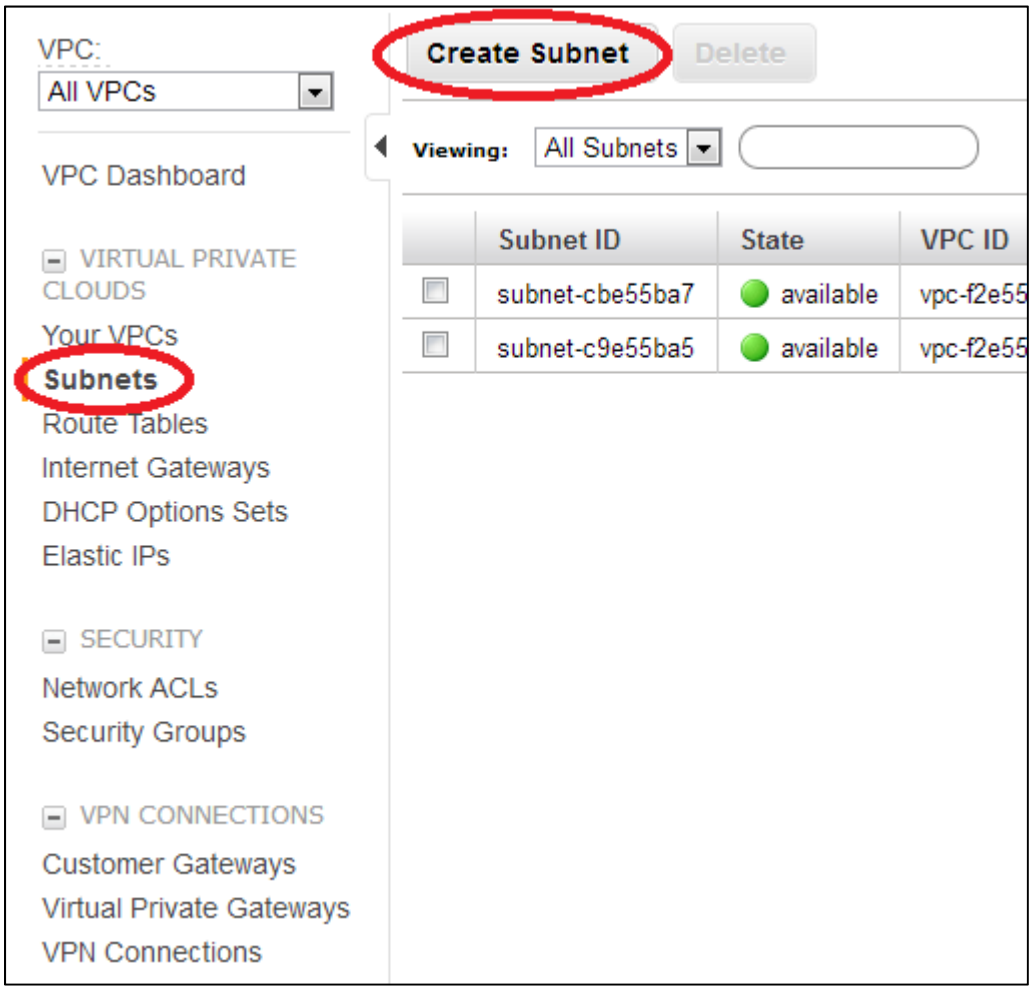


Create Two More Subnets

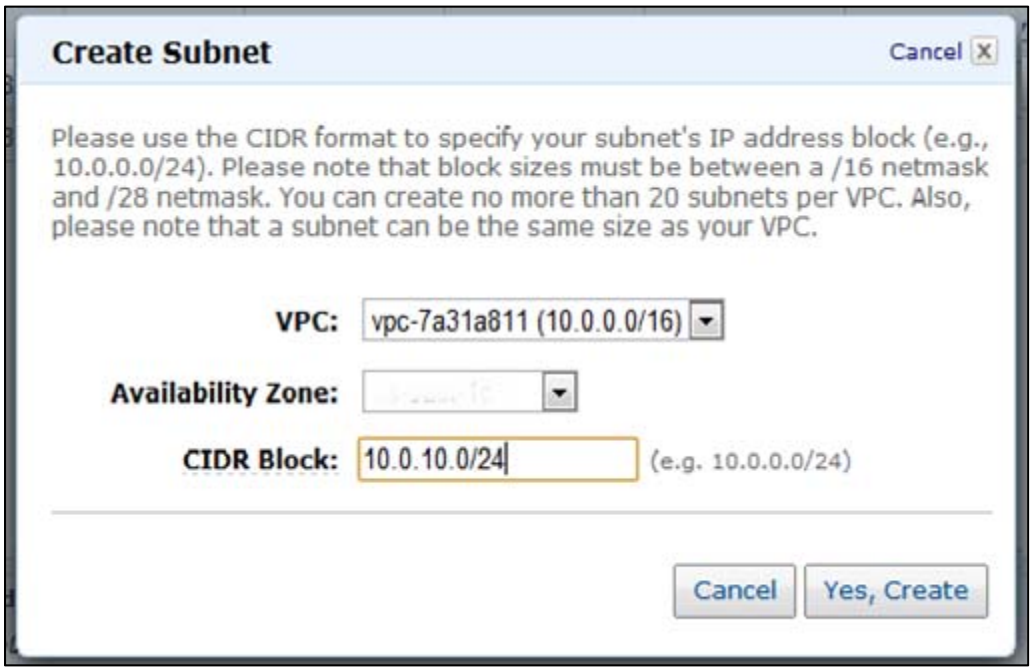
We need to create a public subnet, and also a private subnet in another Availability Zone. Unlike the previous subnets, we'll create these without the assistance of a wizard. Along the way we'll learn a bit more about how they operate.

Manually Create Each Subnet

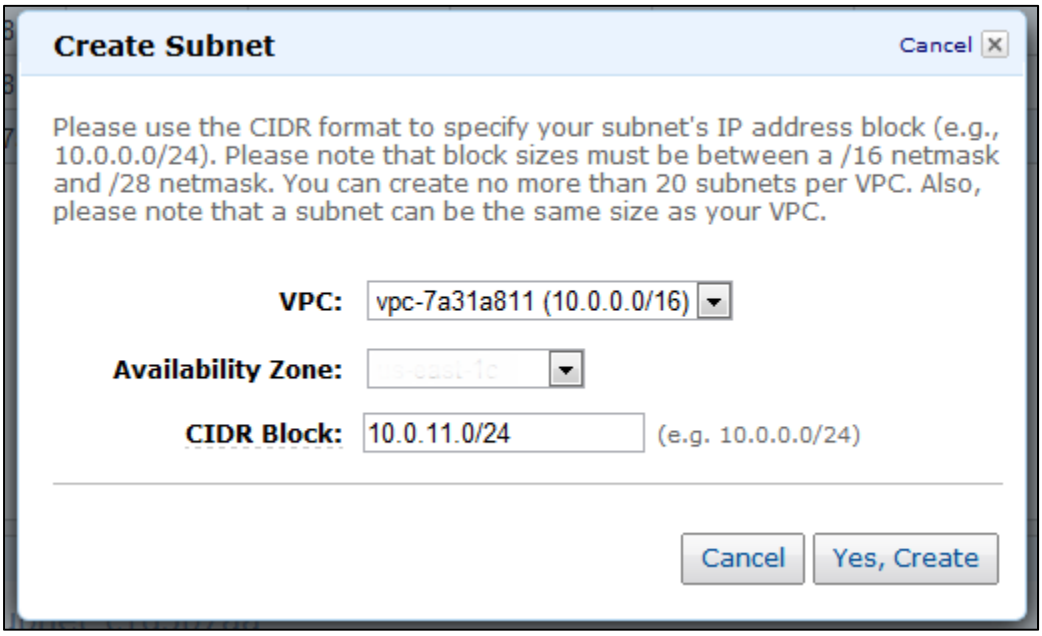
1. Back in the VPC section of the management console, click **Create Subnet**.



Let's refresh our memory. The first two subnets we created were 10.0.0.0/24 (public), and 10.0.1.0/24 (private). Both were in the same Availability Zone. We will now create two subnets in a second Availability Zone. These subnets are 10.0.10.0/24 (public), and 10.0.11.0/24 (private). These two subnets must be in a distinct Availability Zone than the two subnets we already created.



Repeat for 10.0.11.0/24



Be sure to select the same Availability Zones for both the new public subnet (10.0.10.0/24) and the private subnet (10.0.11.0/24).

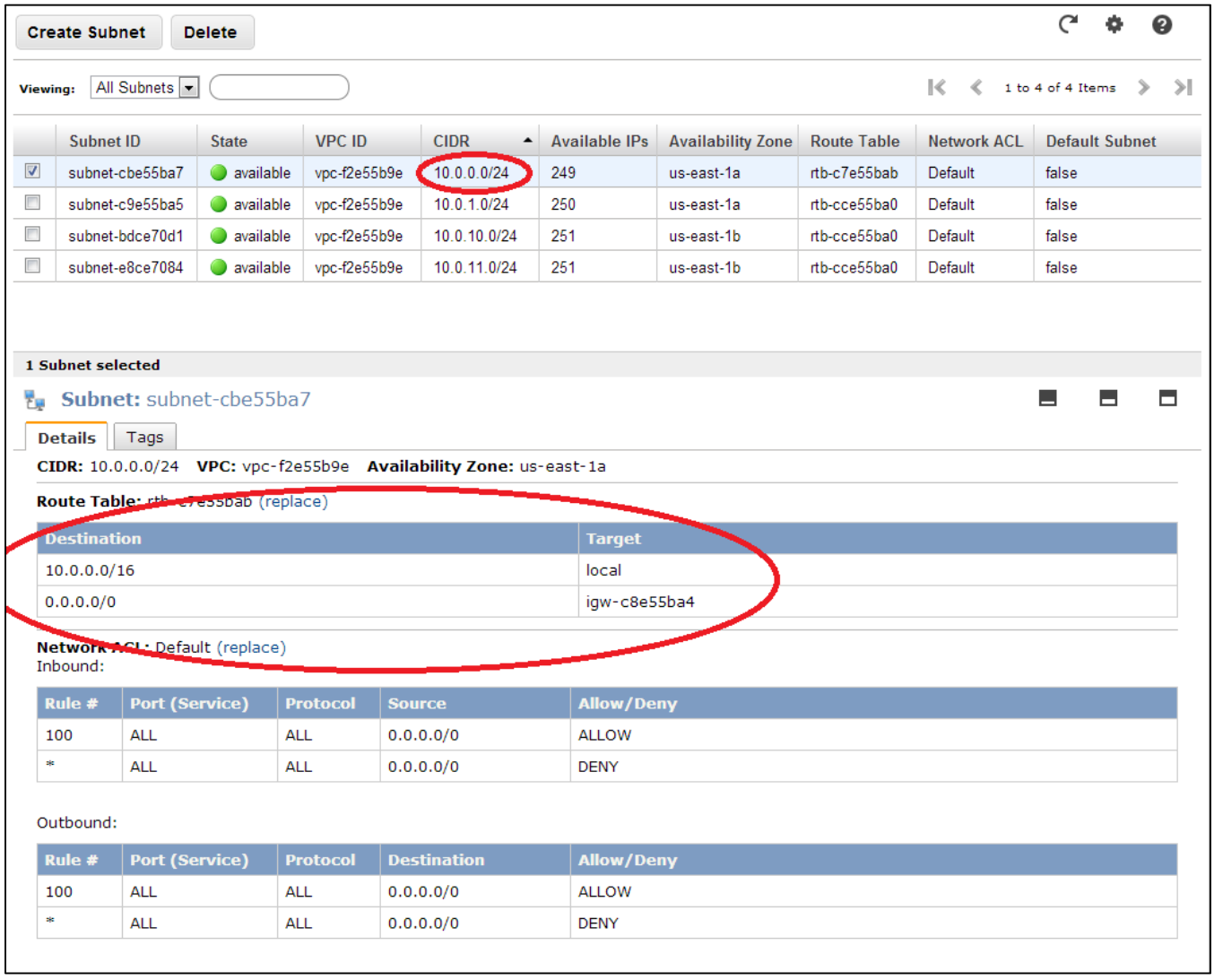
What Determines Whether a Subnet is Public or Private?

Now we have two more subnets, but what makes them private or public? It's the routing rules.

Select **10.0.0.0/24**, and note that there are two routing rules in the Route Table:

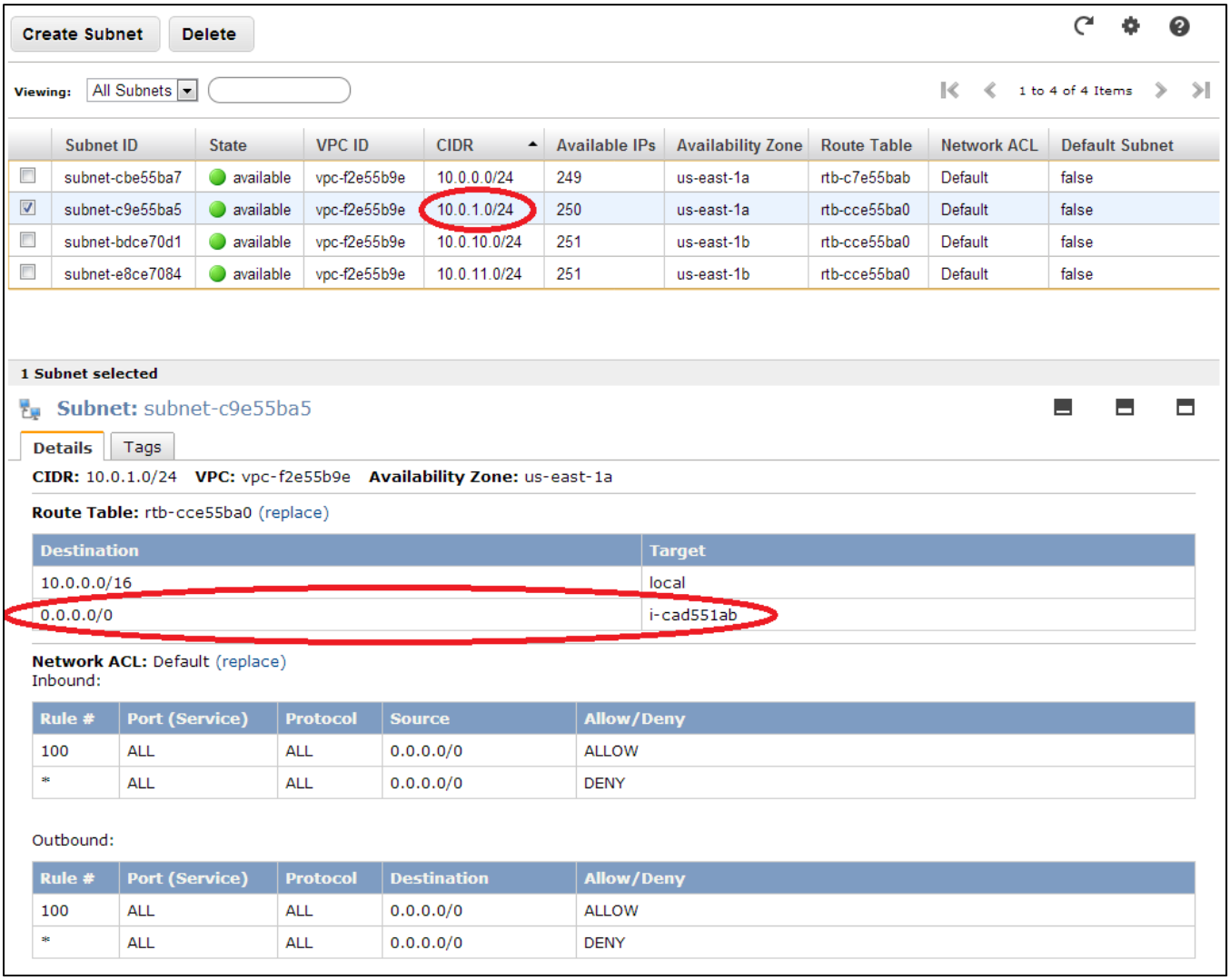
- Any machine in this subnet can communicate with any other machine in 10.0.0.0/16, which is the entire VPC. In other words, communication between all subnets is wide open. Later in this lab we'll look at security groups as a mechanism to restrict traffic.
- Any traffic to/from the Internet (0.0.0.0/0) will be routed thru the Internet Gateway device. We have not looked at that device so far, but think of it as a router on the edge of our VPC. In fact, that's how it is depicted in the network diagrams.

Scroll down and you will see some Network ACLs, which in theory could also control traffic. However the VPC supports a limited number of rules so we will use alternate controls that are even more granular.



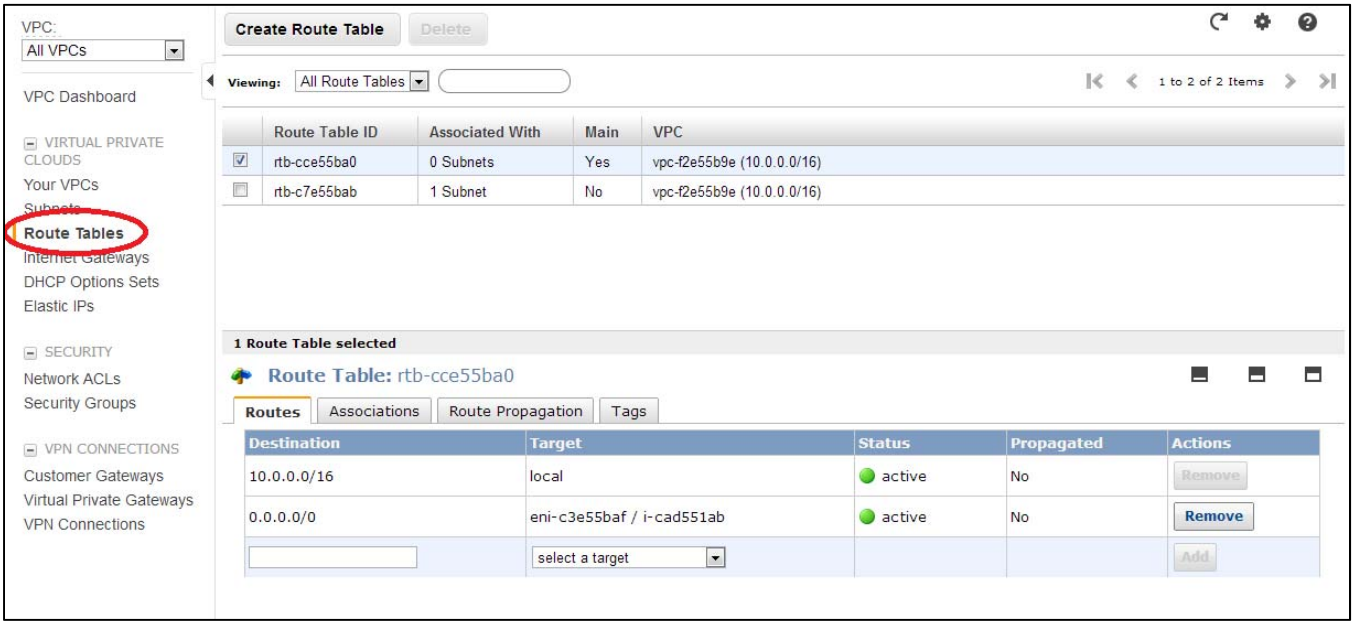
In a similar manner 10.0.1.0/24 also has routing rules:

- Traffic bound for any other subnet in the VPC (10.0.0.0/16) is unrestricted.
- Traffic destined for the Internet will flow to the EC2 Instance, which is the Instance performing NAT. Note that the NAT will not route random requests from the Internet back into this subnet though. It will only route replies made in response to outbound requests from inside this subnet.



Let's switch over to the Route Tables view (**Route Tables** from the left menu) and look at this from the other side. According to this view, only 1 subnet is associated with any routing rule at all, but we have a total of 4 subnets!

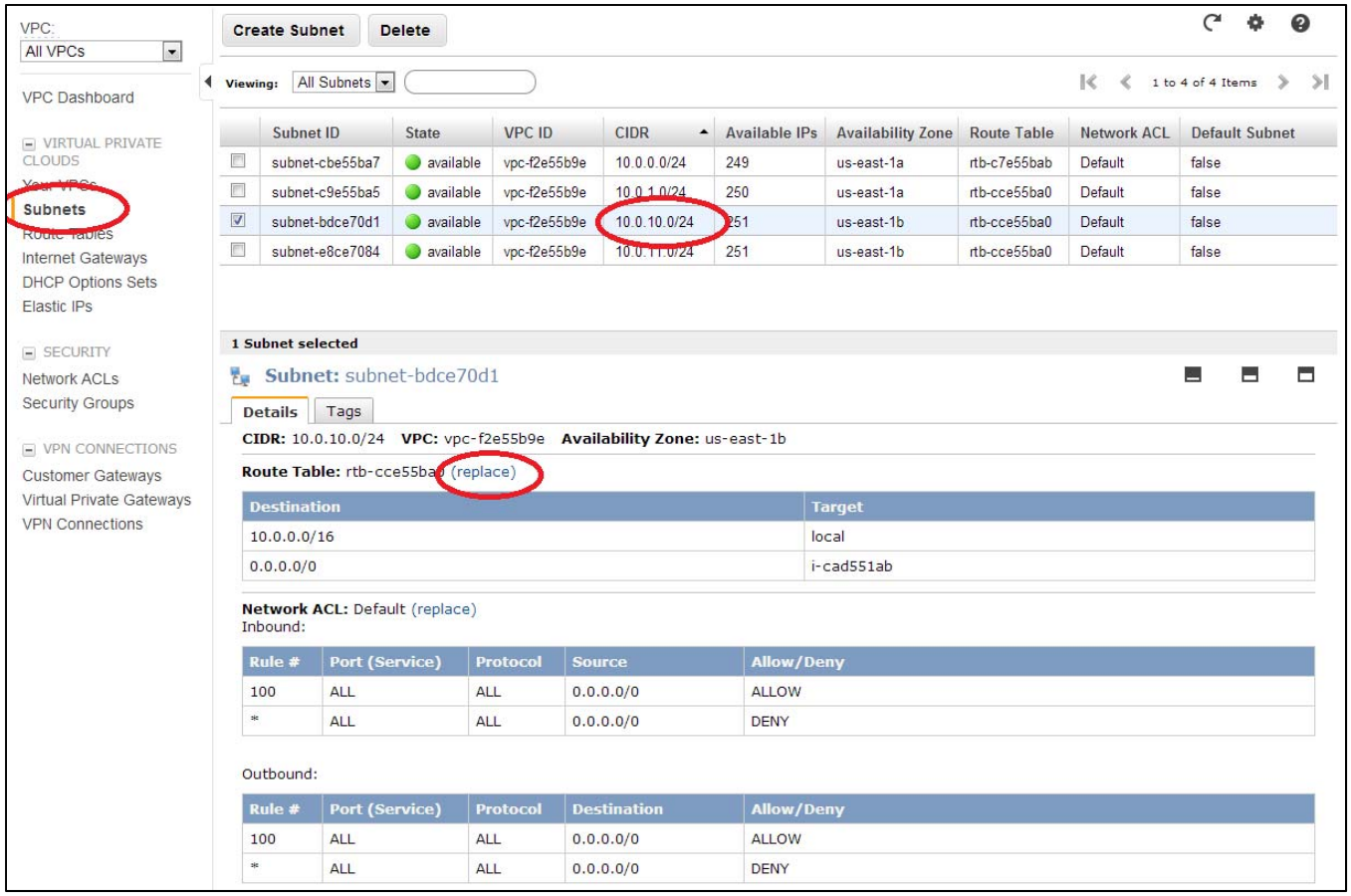
The Amazon VPC operates on a "safety first" principle. Note that one of the rule sets is marked "main". If a subnet is not explicitly associated with a routing ruleset, it uses the Main ruleset, which happens to be the ruleset that does not talk to the Internet. **So by default, no subnet is able to communicate with the Internet** (unless you switch the default.)



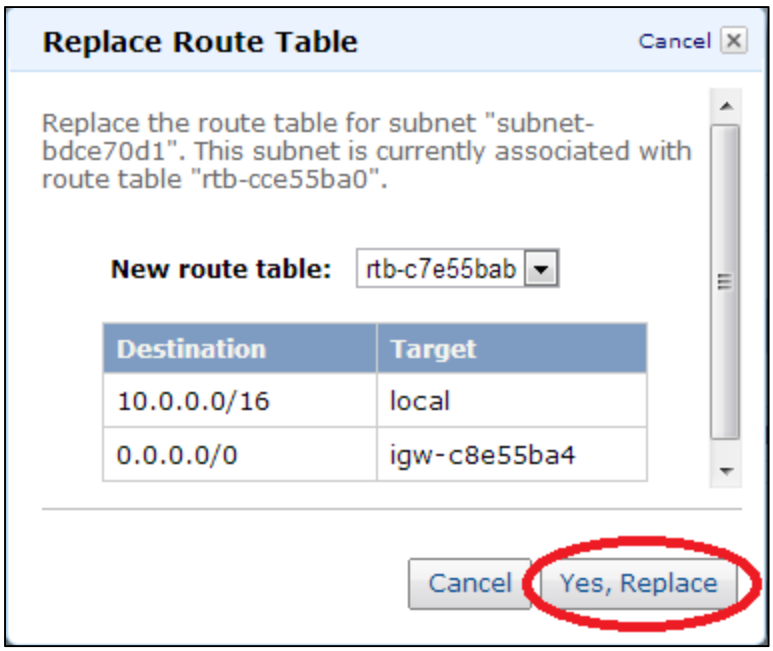
Now that we understand routing mechanisms within VPC, let's make our public subnet really public.

We need to associate the new public subnet (10.0.10.0/24) with the routing ruleset that routes bi-directionally to the Internet.

2. Return to **Subnets** and select the **10.0.10.0/24** subnet and **replace** the ruleset.

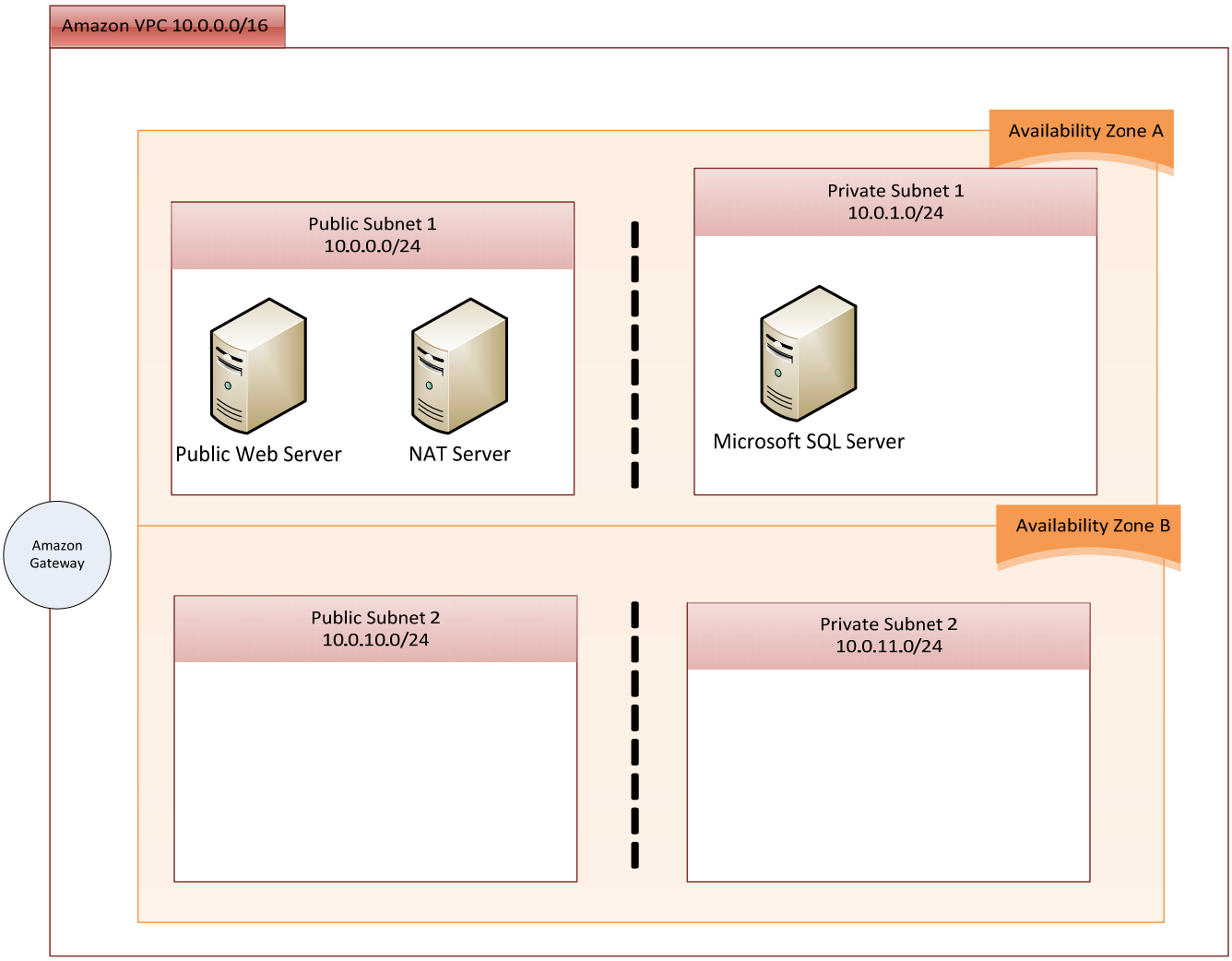


3. There is only one choice in the drop-down list because the console is smart enough to know that you don't want to replace the current routing rules with…..the current routing rules. Click **Yes, Replace**.



In order to test Internet connectivity to our second public subnet, we will need to create an Amazon EC2 instance in this subnet. For the purpose of this lab, we will create a Windows Bastion host in this public subnet.

At this stage, here's the state of our VPC:

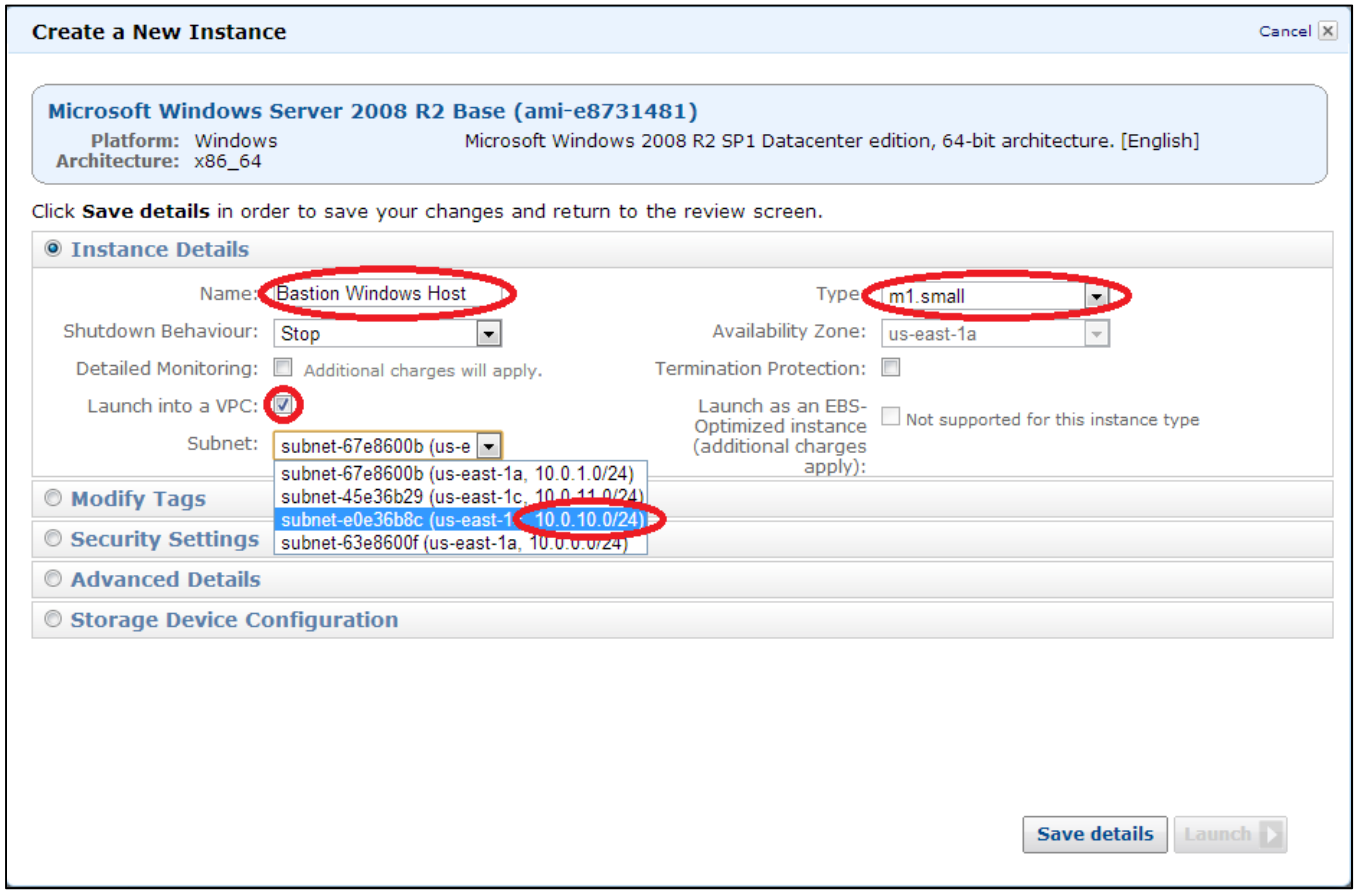


Launch a Bastion Windows Host

Wikipedia's definition of a Bastion Host is "a special purpose computer on a network specifically designed and configured to withstand attacks. The computer generally hosts a single application, for example a proxy server, and all other services are removed or limited to reduce the threat to the computer. It is hardened in this manner primarily due to its location and purpose, which is either on the outside of the firewall or in the DMZ and usually involves access from untrusted networks or computers."

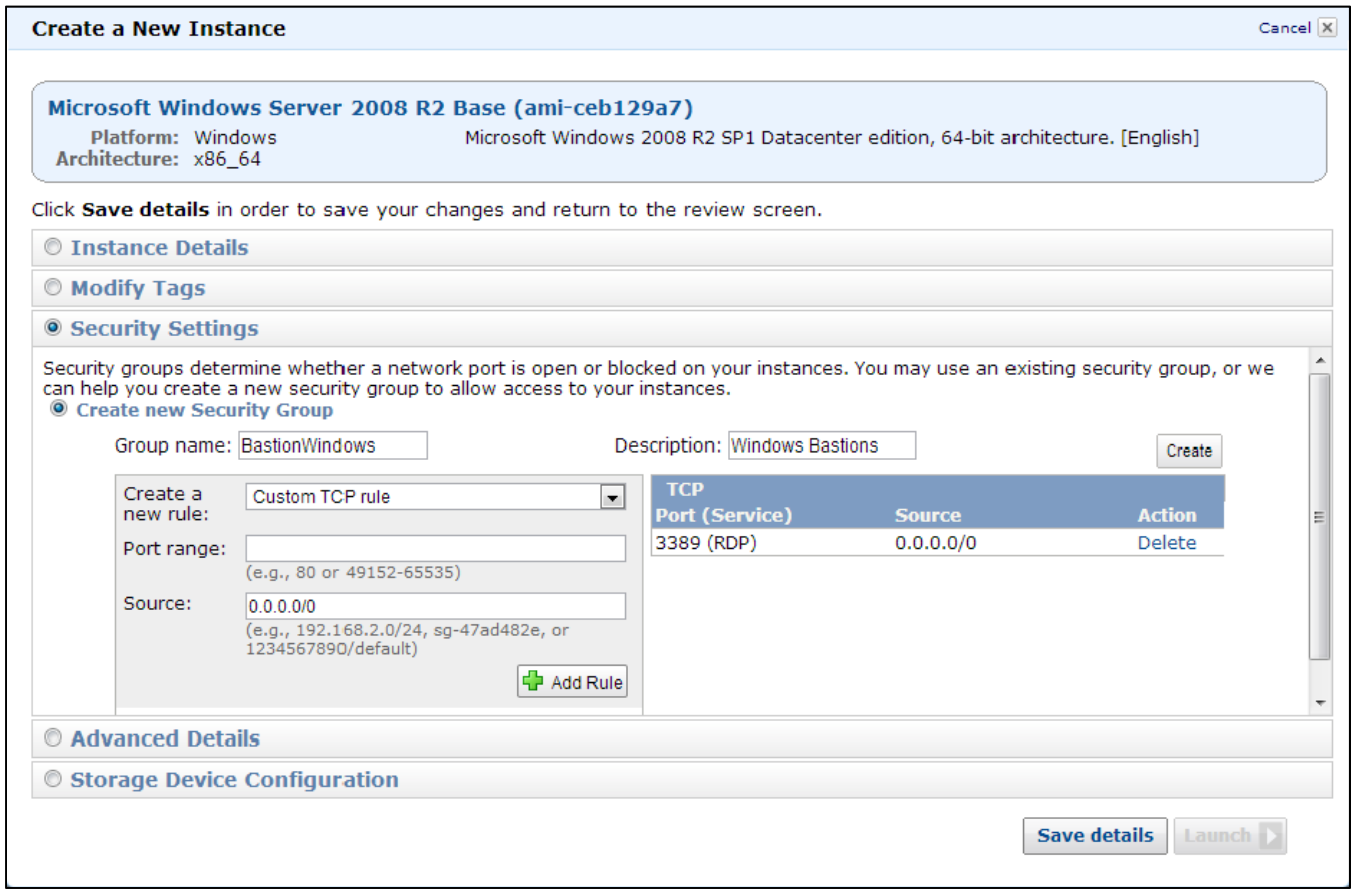
We're going to launch our Bastion Host in the new public subnet, although the original public subnet would work just as well.

1. Back in the VPC Dashboard use the **Launch EC2 Instances** button to take you to the Amazon EC2 Instances section of the EC2 console.
2. Click on **Launch Instance** and choose the **Windows Server 2008 R2 Base** AMI.
3. Name the Instance **Bastion Windows Host**.
4. Launch it into **10.0.10.0/24** as an `m1.small` instance.



There's no need to set a fixed IP address this time.

5. Create another new security group, named **BastionWindows**.
   We are only allowing access to port **3389**, which is the Windows Remote Desktop Protocol (RDP). For this lab we are allowing access from any IP address on the Internet. In real life you will want to restrict access to the address ranges required for administration.



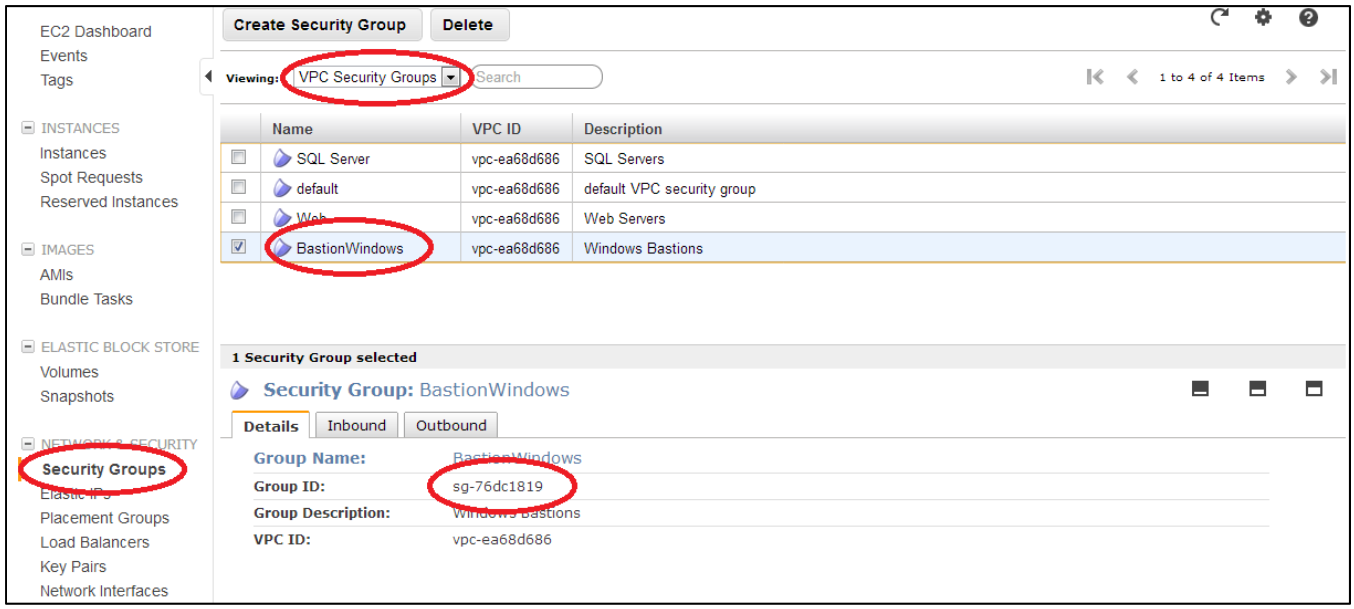
6. Review your options and click on **Launch** to start the Instance.

Now that this new BastionWindows security group exists, let's change the rules for our database server so that the only traffic that it accepts is from the Bastion security group.

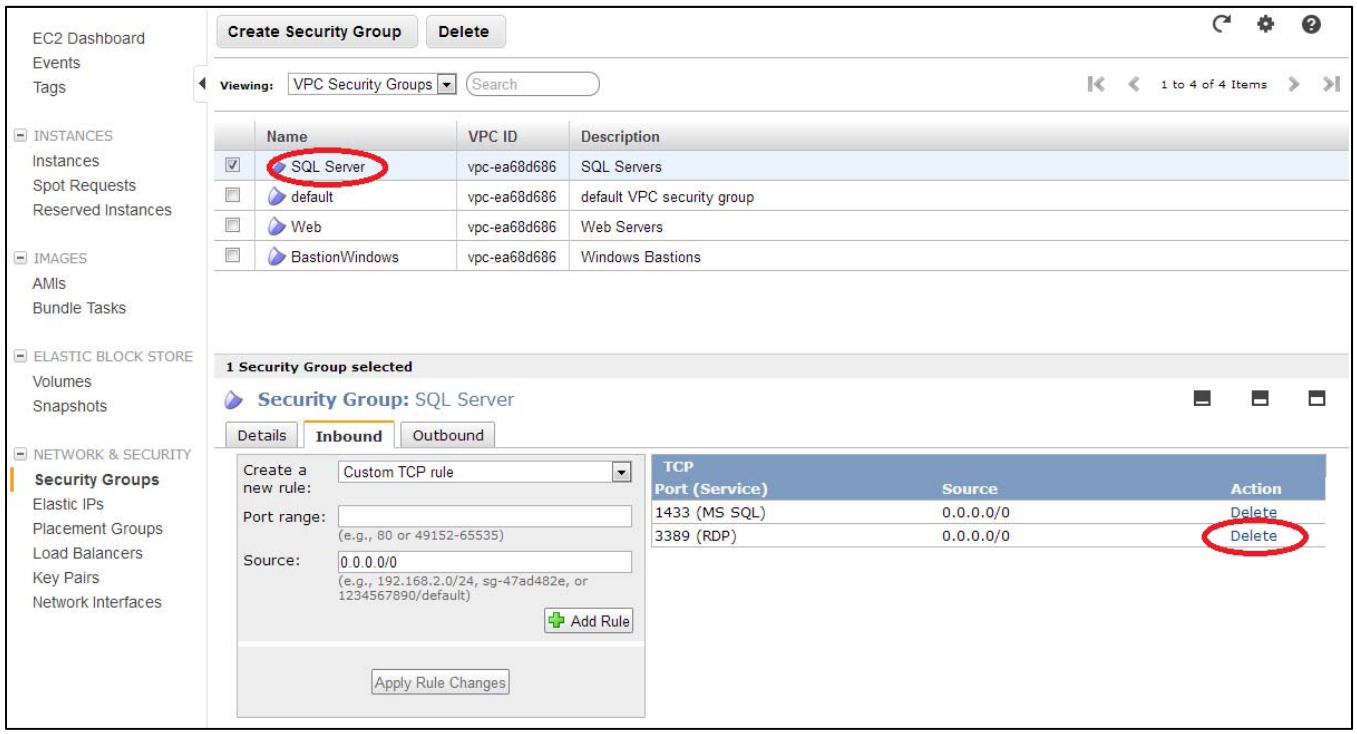
7. Return to VPC / Security Groups.
8. Select **VPC Security Groups** from the drop-down list.

The first step is to make note of the BastionWindows Security Group ID, because you will need it in a moment.

9. Write down your BastionWindows Security Group ID or copy and paste it to notepad or into the clipboard.



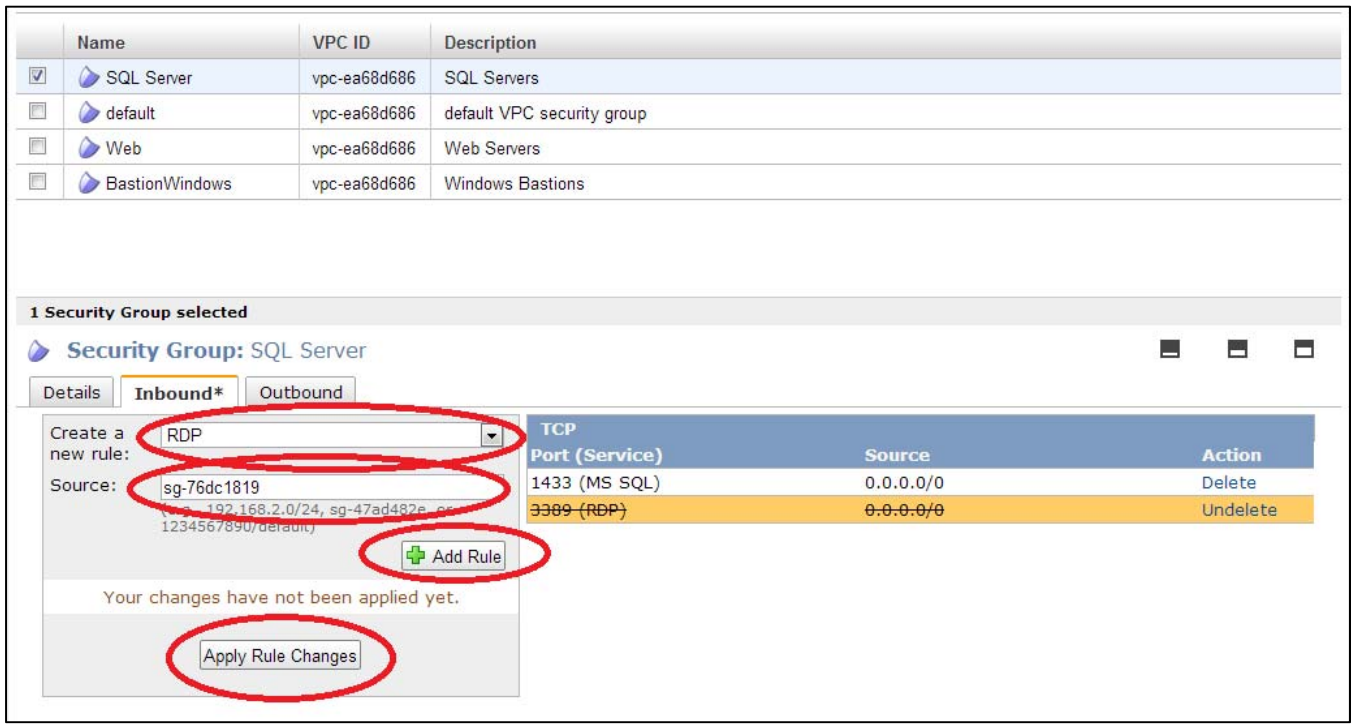
10. Next, **delete** the existing RDP rule out of the SQLServer security group (the one with 0.0.0.0/0 as source)



11. Then add a new rule for RDP that is restricted to the BastionWindows security Group. Use the Security Group ID you pasted into notepad or the clipboard as the source.

    This rule illustrates another powerful way to use security groups. This rule restricts traffic to port 3389 from machines belonging to BastionWindows security group. In other words, our bastion EC2 instance will be the only machine allowed to initiate a RDP connection to the SQL Server instance.

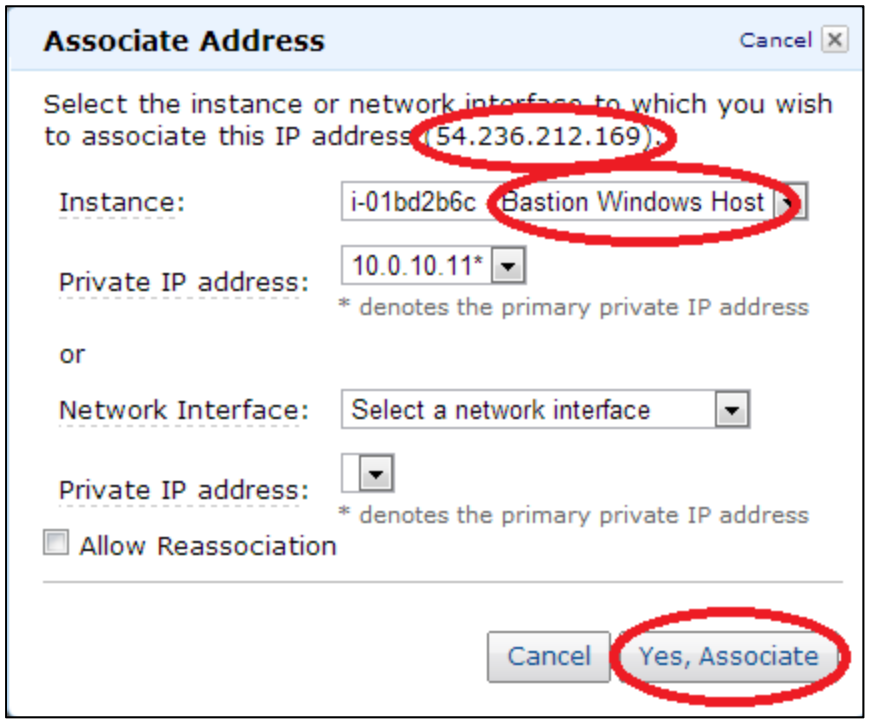
12. Don't forget to apply these rule changes by clicking **Apply Rule Changes**.



In order to use the Bastion server, you will need a public IP address to connect to.

13. Assign an Elastic IP Address. If you forget how to create an Elastic IP address and how to assign it to an Amazon EC2 instance, you can go back to the “Create and Assign an Elastic IP Address” section of this lab on page 20.

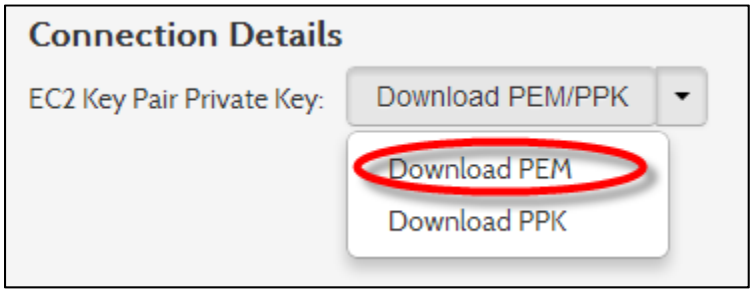
    Once assigned, the address will appear as part of the details for the Bastion host.



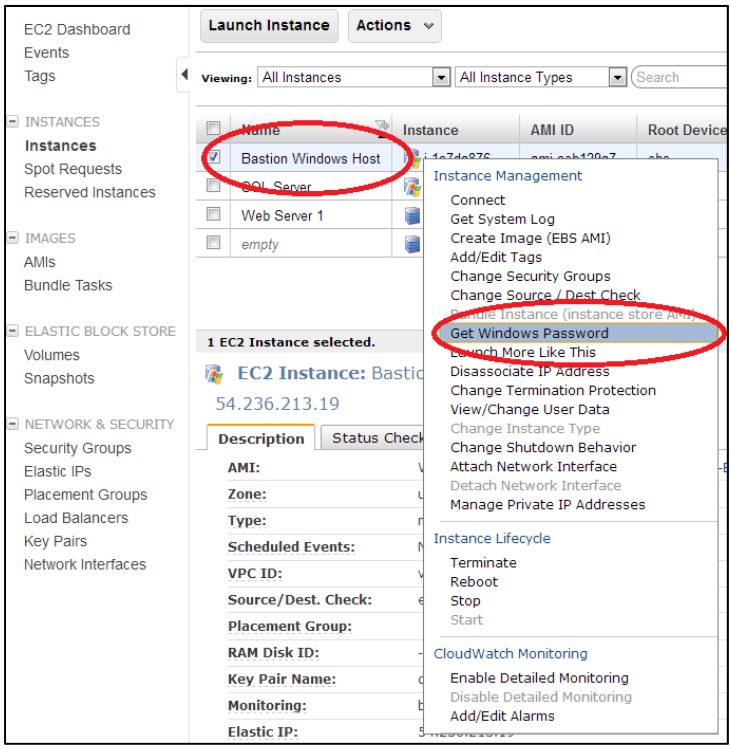
Note the public IP address that you allocated and associated to the Bastion host. This is the one you will use to connect to the instance via RDP in the next section.

Get the Password for your Windows Instance

14. Go back to your lab in *qwikLAB*™.
15. Download the *qwikLAB*™ provided EC2 Key Pair private key file in the PEM format by clicking on Download PEM option in the “Download PEM/PPK” drop-down.



1. Save the file to your computer’s Downloads folder or directory or some other folder or directory of your choice.
2. Go back to the AWS Management Console.
3. Locate the Bastion Host instance in the Amazon EC2 section.
4. Right-click on the Instance in the AWS Management Console.
5. Click Get Windows Password.



6. Click on Choose File and navigate to your Downloads folder (or another place you choose).
7. Select the EC2 Key Pair private key file that you downloaded from *qwikLAB*™.

If the Instance is still starting up you'll see a message the password is not available yet. Close the message and wait awhile and try again if you do see it.

8. Click Decrypt Password.

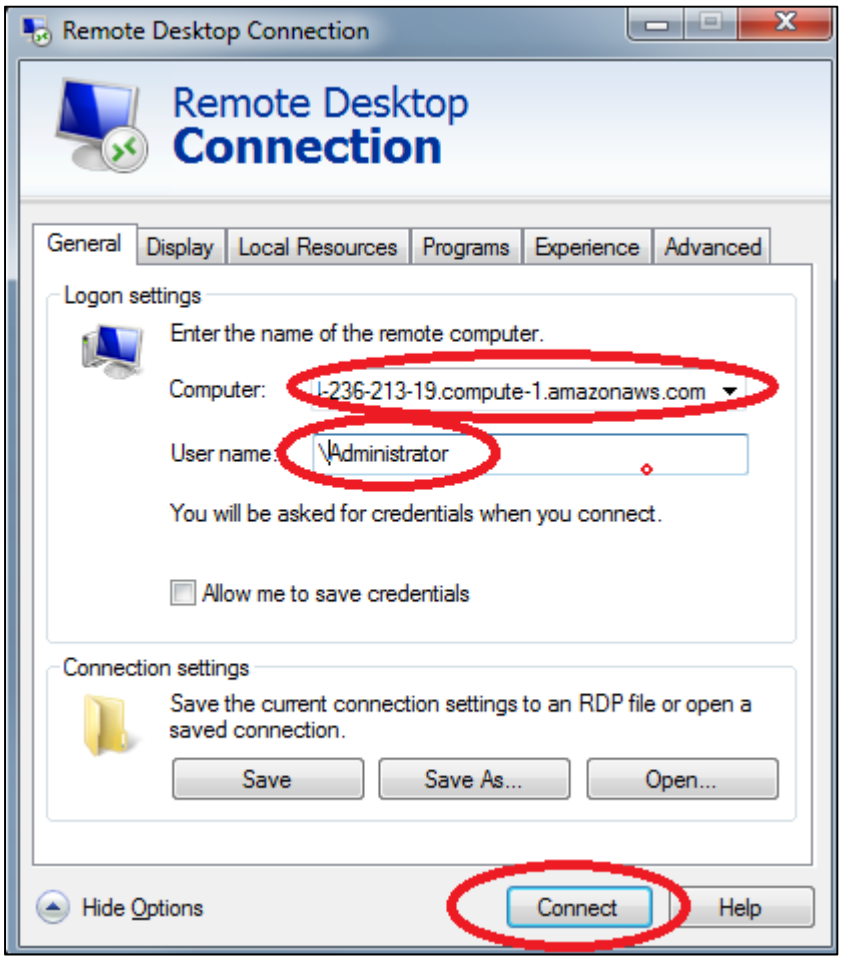
9. Make a note of the Computer, User, and Decrypted Password. You might copy-paste them to a text file on your computer.

Connect to the Bastion Server (Windows)

(Hint: Go to the section Connect to the Bastion Server (OS X) or (Linux) if you are using one of those where you will run a Remote Desktop Protocol (RDP) client.)

1. On your local computer Start -> Run, and then type in MSTSC to start the local RDP client.
2. Click Show Options.
3. Enter the Computer and Username you noted and then Connect.

You'll be signing in as another account - Administrator, and may need to specify the user name as "\Administrator" (with a leading backslash) in order to differentiate from the Administrator user on your local computer.



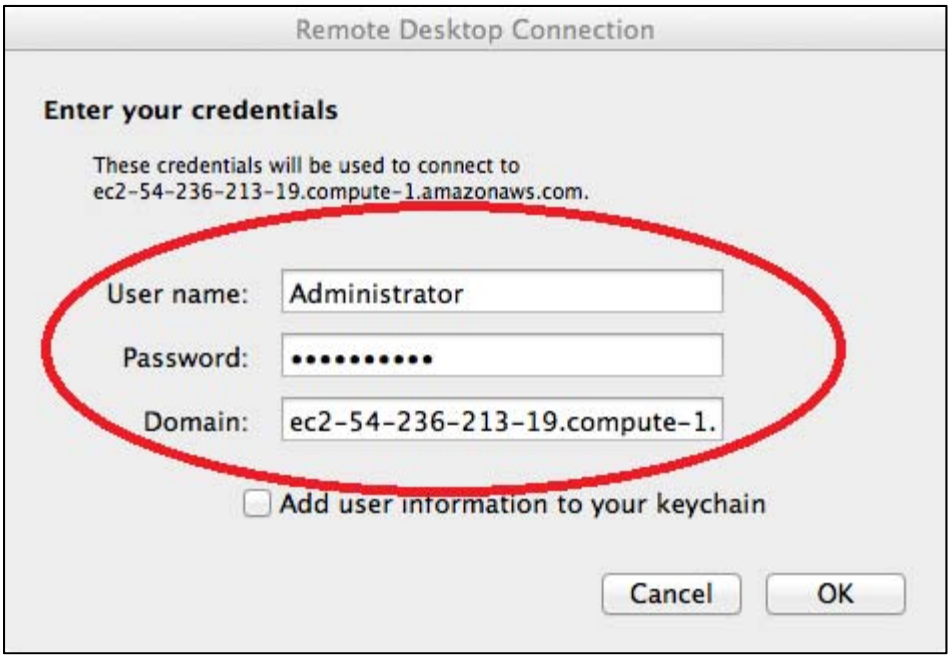
4. When prompted, enter the Password you noted.
5. Click Yes when you see a certificate verification message similar to this one:

Connect to the Bastion Server (OS X)

(Hint: Go to the section Connect to the Bastion Server (Windows) or (Linux) if you are using one of those where you will run a Remote Desktop Protocol (RDP) client.)

1. Open the Remote Desktop Connection for Mac application. Enter the Bastion Host Computer DNS hostname you noted or copied down above and click Connect.

2. When prompted, enter the Username and Password you noted. The Domain will auto-populate with the Amazon EC2 Instance DNS and you can ignore it.
3. Click OK.

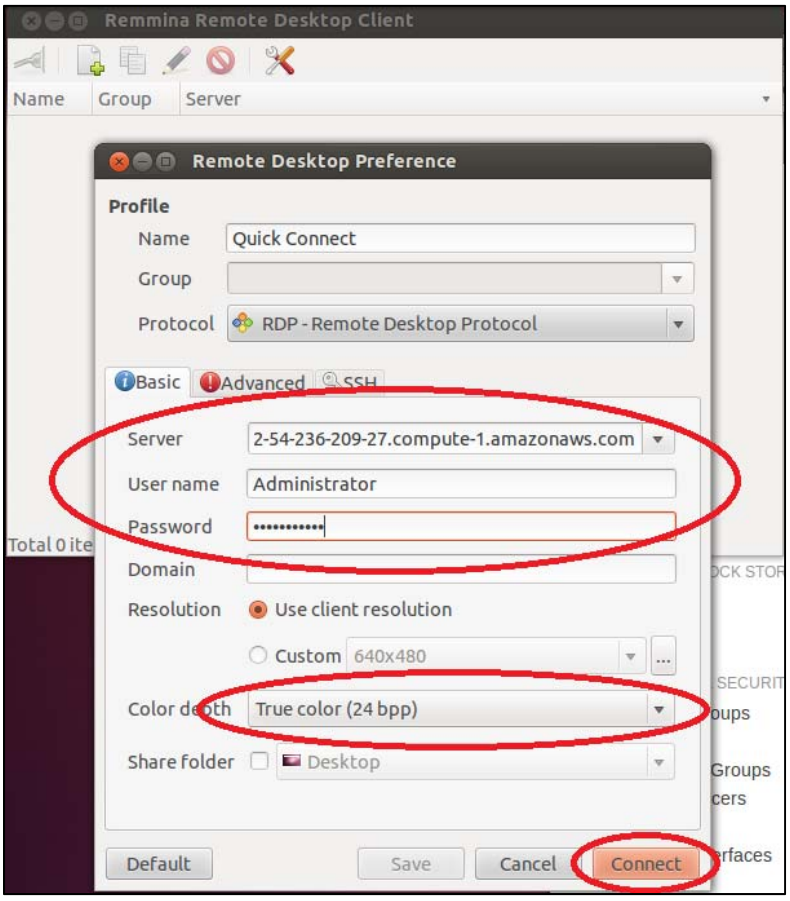


4. Click Yes when you see a certificate verification message similar to this one:

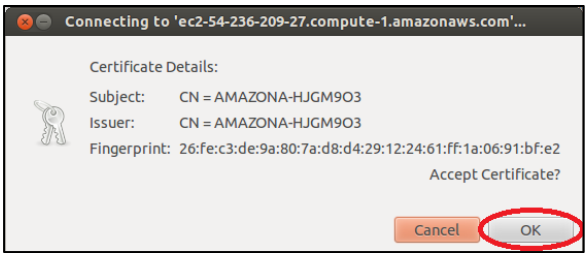
Connect to the Bastion Server (Linux)

(Hint: Go to the section Connect to the Bastion Server (Windows) or (OS X) if you are using a Remote Desktop Protocol (RDP) client.)

5. Open the Remmina Remote Desktop Client.
6. Enter the Bastion Host Computer DNS hostname you noted or copied down above in Server.
7. Enter the Username and Password. Optionally, set Color depth to something that your bandwidth supports (in this example 'True color (24 bpp)') for a nicer remote desktop.
8. Click Connect.



9. Click OK when prompted to accept the remote certificate.



Log in to the Database Server

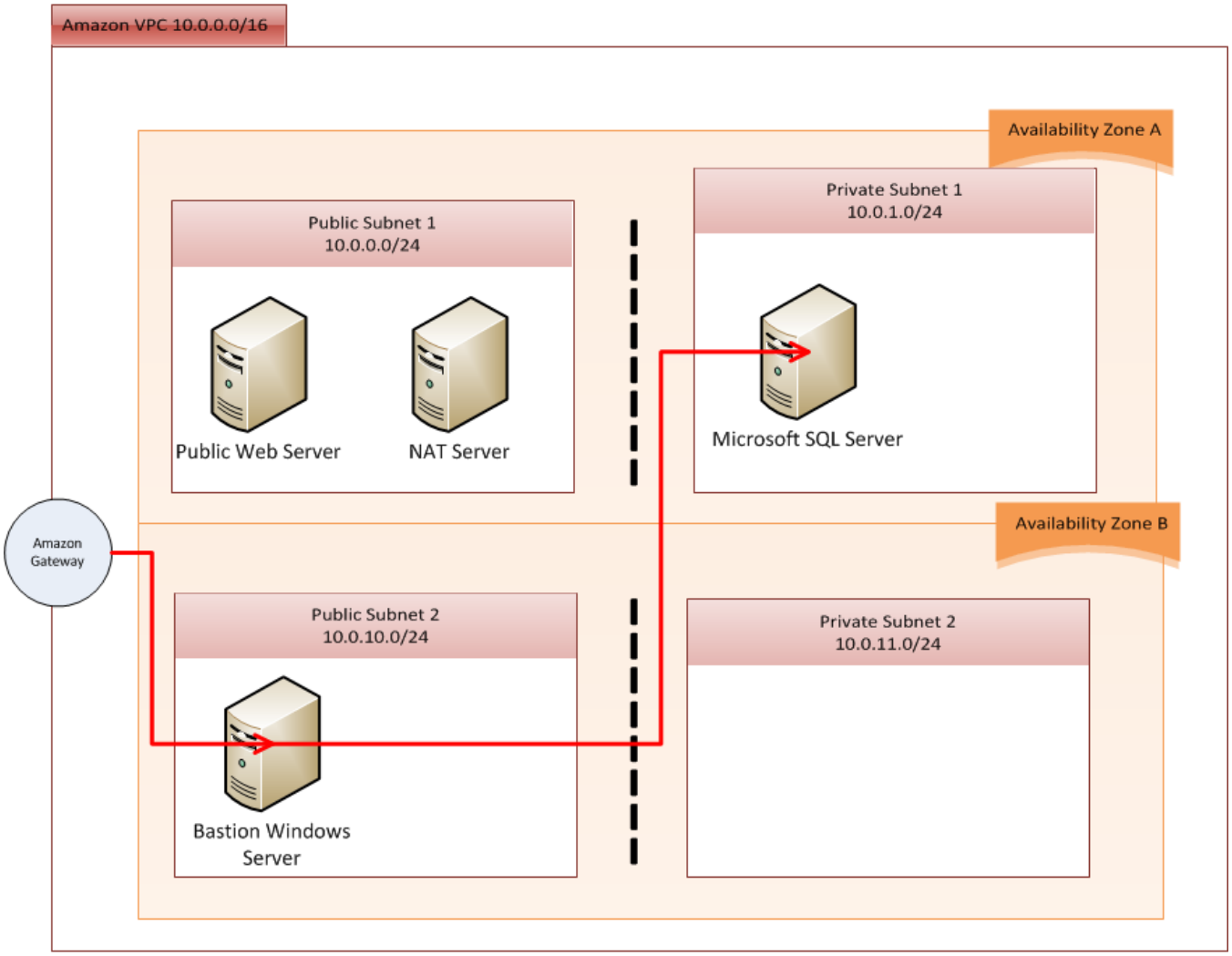
1. Now that we have logged in to the Bastion Host, repeat the process to log in to the SQL Server Instance from Windows.

   Hint: use the above ‘Connect to the Bastion Server (Windows)’ even if you’re using OS X or Linux yourself as this time, you’ll be running the RDP client from the remote Windows Bastion Host.

2. Connect to 10.0.1.99 via RDP in the remote Windows Bastion Host. You will need to repeat the process to retrieve the password for the SQL Server.

Here's our environment, now that all these pieces are in place. The line from the gateway device to the SQL Server illustrates traffic flow from the edge of the VPC network, through the Bastion Host, and to the SQL Server.

You might wonder why we created the Private Subnet 2 (10.0.11/0/24) subnet. It is because, in real life scenario, you will probably deploy a slave, replica SQL Server Instance for the SQL Server in Private Subnet 1 (10.0.1.0/24).



Conclusion

Amazon networking is secure by default, and as you just learned there are multiple ways to safely connect to servers that are kept in private subnets.

In order to ensure that your network is secure, pay attention to which subnet you place servers in. Bastion hosts and VPN tunnels are two different techniques to allow external access to private subnets. Each technique has its own advantages.

Bastion hosts are good if you need to log in to manage servers, especially if only a few people need to perform this activity. Bastion hosts can be shutdown (Terminated or Stopped) when not in use, allowing you to reduce AWS costs and to add an extra level of security.

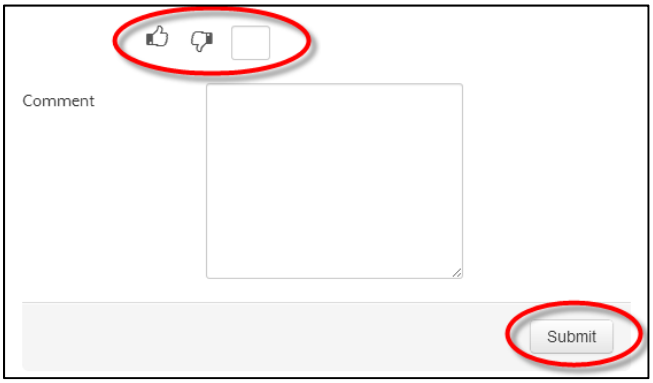
If you want the VPC to act as a virtual extension to your corporate network, then a VPN might make more sense.

Finally, we learned how Security Group rules can be either very precise, or quite loose. Ensure that your Security Groups are as restrictive as possible, but not so restrictive that there are unintended side effects.

End Lab

1. Sign-out of the AWS Management Console.
2. Click the End Lab button in *qwikLAB*™.

3. Give the lab a thumbs-up/down, or enter a comment and click Submit



Errors in this lab may be reported to aws-course-feedback@amazon.com.