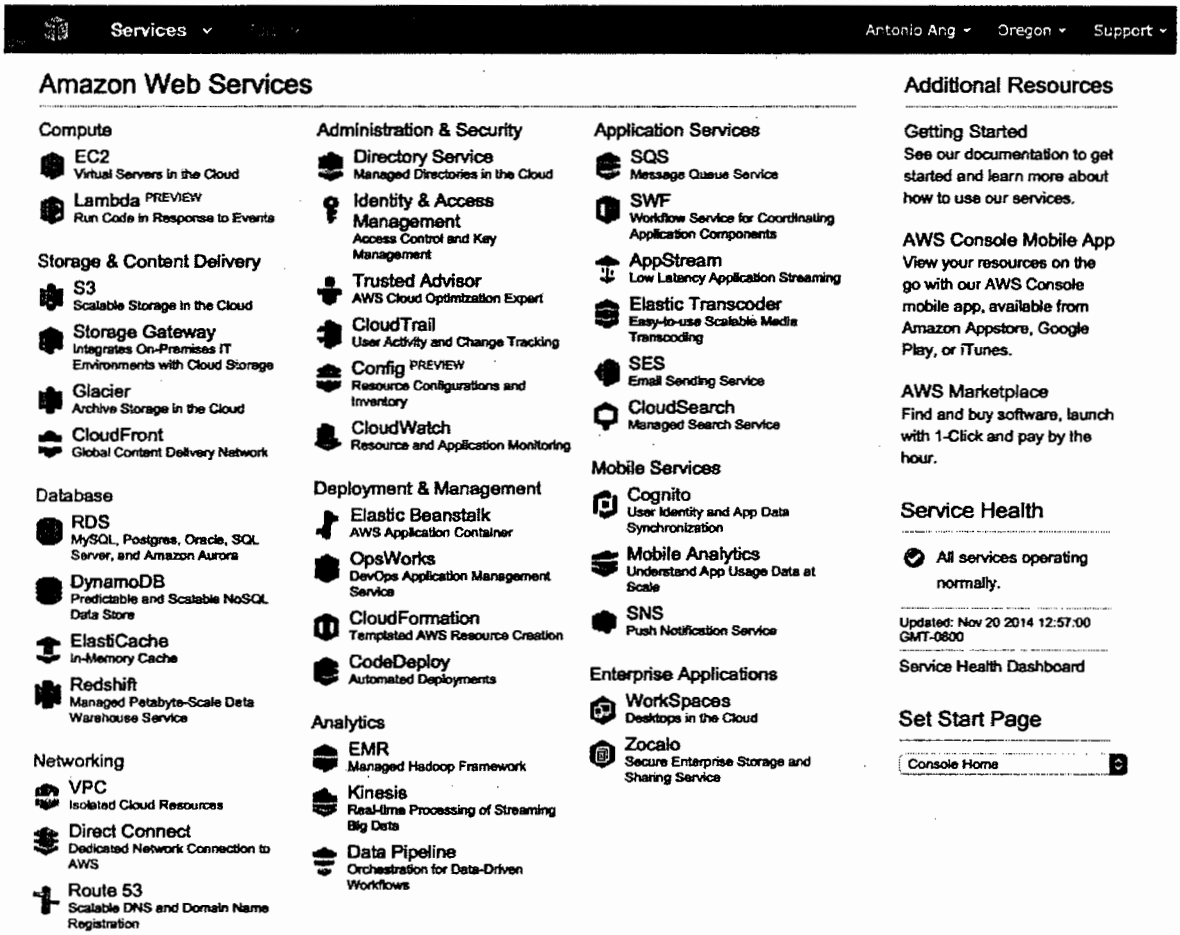
Lab 8

SECURING YOUR VPC USING PUBLIC AND PRIVATE SUBNETS

STEP 1: Log In to the Amazon Web Service Console

This laboratory experience is about Amazon Web Services and you will use the AWS Management Console in order to complete all the lab steps.



The AWS Management Console is a web control panel for managing all your AWS resources, from EC2 instances to SNS topics. The console enables cloud management for all aspects of the AWS account, including managing security credentials, or even setting up new IAM Users.

Log in to the AWS Management Console

In order to start the laboratory experience, open the Amazon Console by clicking this button:

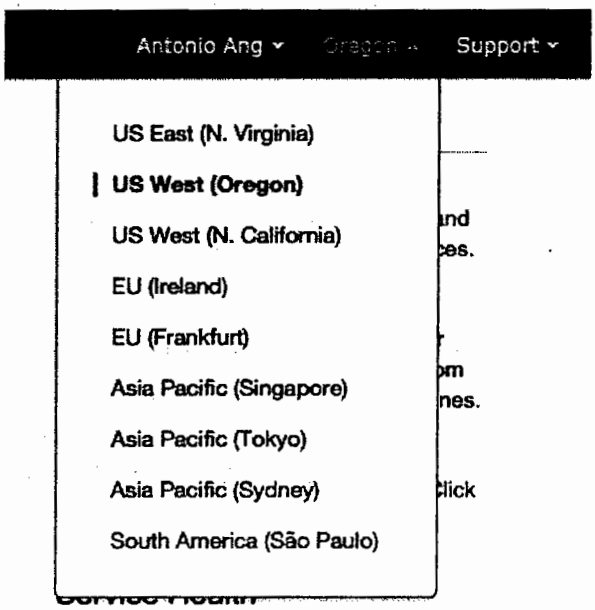
Open AWS Console

Log in with the username **xxxx** and the password **xxxx** .

Select the right AWS Region

Amazon Web Services is available in different regions all over the world, and the console lets you provision resources across multiple regions. You usually choose a region that best suits your business needs to optimize your customer's experience, but you must use the region **US West (Oregon)** for this laboratory.

You can select the **US West (Oregon)** region using the upper right dropdown menu on the AWS Console page.



STEP 2: Create a VPC

Amazon OpsWorks lets you easily orchestrate the different parts of your application using**Chef** to perform the actual automation. It presents the different AWS resources that make up your app as multiple layers, each composed of resources. A typical app might have two layers, an app server layer (where your Ruby/NodeJS/Python/PHP app actually runs) and a database layer (backed by RDS). Typically, you'd manage each instance and RDS installation separately, but with OpsWorks you can manage all instances in the "app server" layer together.

The advantage of using Chef is that you can use AWS' published OpsWorks cookbooks, open source community cookbooks, build your own, or mix and match. AWS publishes cookbooks for typical Rails applications, Nginx proxies, memcached servers, monitoring, haproxy, and more.

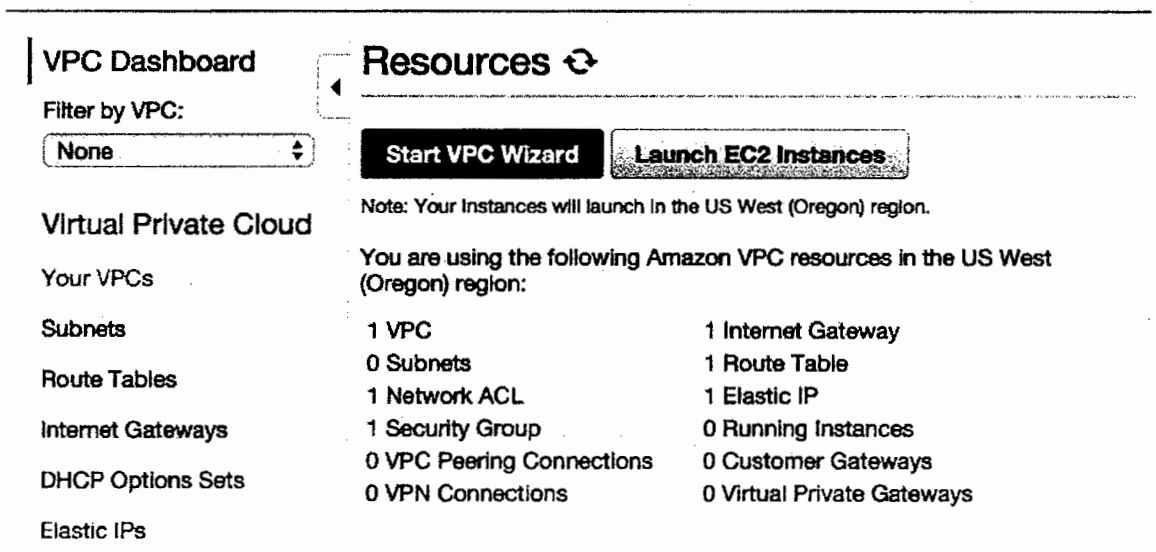
But before we get started building our first OpsWork stack, I'd like to remind you that it is just a collection of resources, and often doesn't create underlying resources like VPC networks automatically do. So we'll need to digress and make a VPC for all our instances to inhabit first.

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

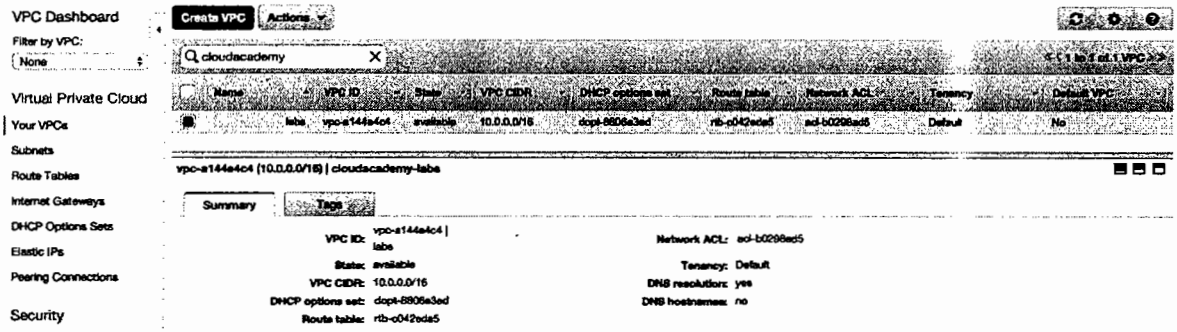
You can create a new VPC using the AWS Management Console.

Select the VPC service from the Management Console dashboard:

From the VPC dashboard, click on **Your VPCs** link in the sidebar menu.

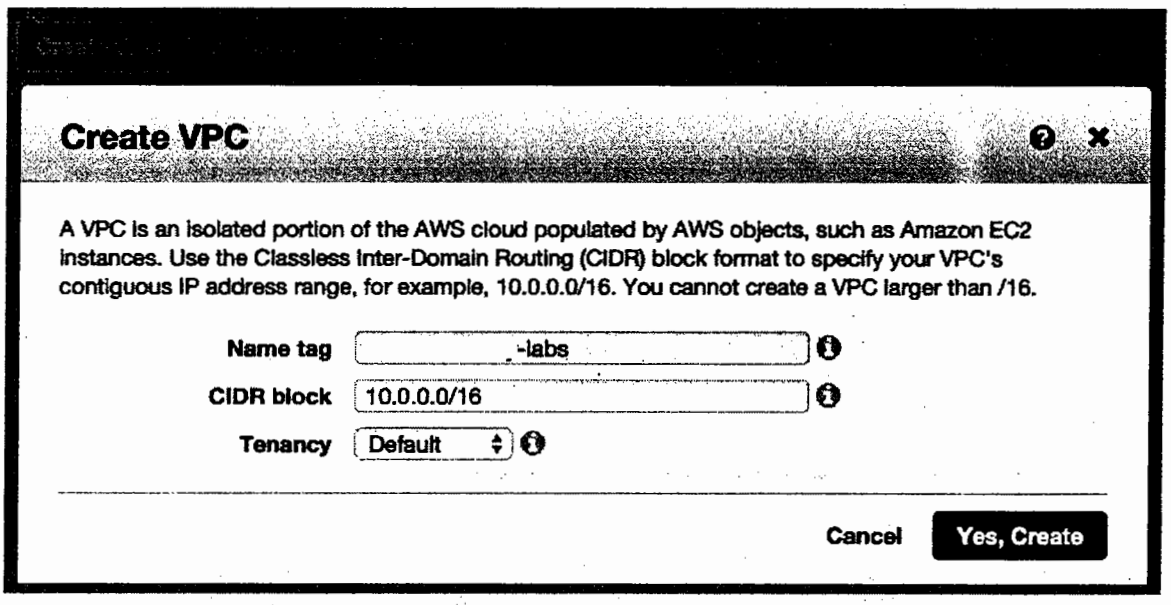


Your VPCs page lists all previously created VPCs (any new AWS account comes with a default fully-working VPC); click on the **Create VPC** blue button to begin creating a new VPC.



In the Create VPC dialog box, specify the following VPC details as necessary, then click **Yes, Create**.

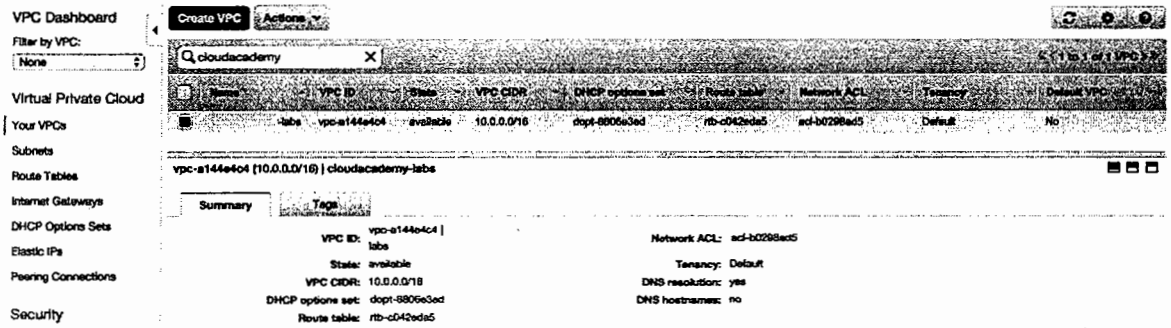
- ✓ **Name tag**: **vepsun-labs**. This is the name for your VPC; doing so creates a tag with a key of Name and the value that you specify.
- ✓ **CIDR block**: **10.0.0.0/16**. You should specify a CIDR block from the private (non-publicly routable) IP address ranges as specified in RFC 1918.
- ✓ **Tenancy**: **default**. Dedicated tenancy ensures your instances run on single-tenant hardware.



Amazon creates the requested VPC and the following linked services:

- ✓ a **DHCP options set** (this set enables DNS for instances that need to communicate over the VPC's Internet gateway)
- ✓ a **Route Table** (it contains a set of rules, called *routes*, that are used to determine where network traffic is directed)
- ✓ a **Network ACL** (it is a list of rules to determine whether traffic is allowed in or out of any subnet associated with the network ACL)

Note that no Subnets or Internet Gateways are automatically created -- you need to add them autonomously.



Now you are ready to create your VPC subnets and customize the routing table.

STEP 3: Create a VPC Internet Gateway

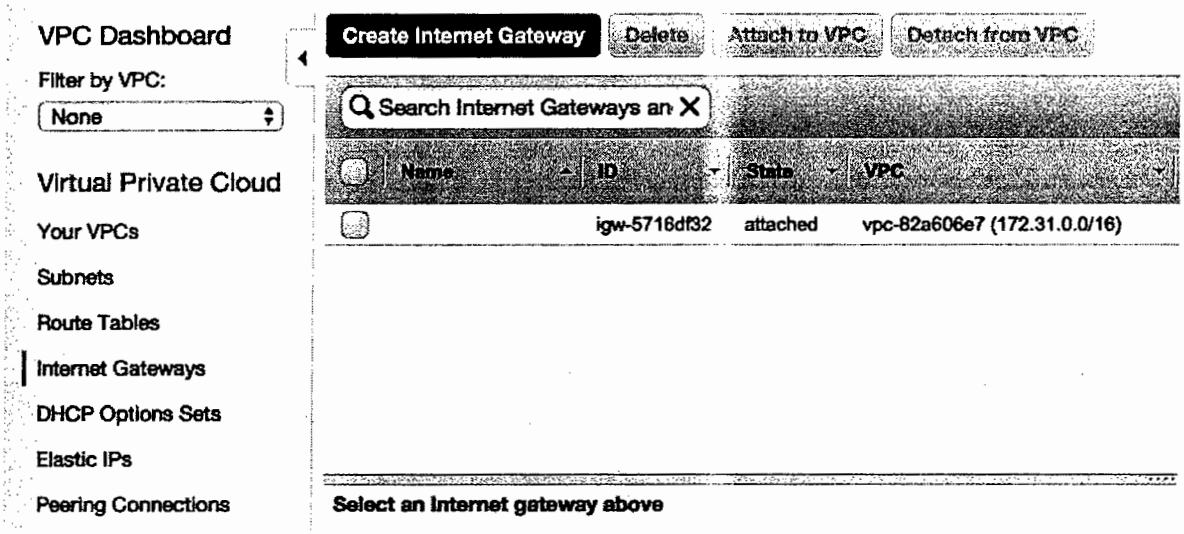
An **Internet Gateway** is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the Internet. It imposes no availability risks or bandwidth constraints on your network traffic. An Internet gateway serves two purposes: to provide a target in your VPC route tables for Internet-routable traffic, and to perform network address translation (NAT) for instances that have been assigned public IP addresses.

You can create a new **Internet Gateway** for your previously created VPC using the AWS Management Console.

Select the VPC service from the AWS Management Console dashboard:

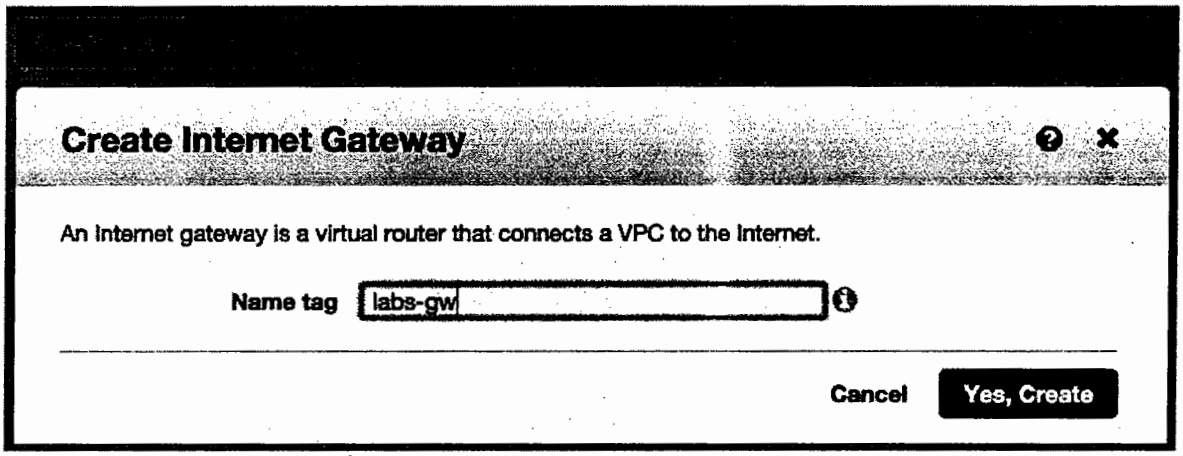
From the VPC dashboard, click the **Internet Gateways** link in the sidebar menu.

The **Internet Gateways** page lists all previously created gateways. Click on the **Create Internet Gateway** blue button to begin creating a new gateway.



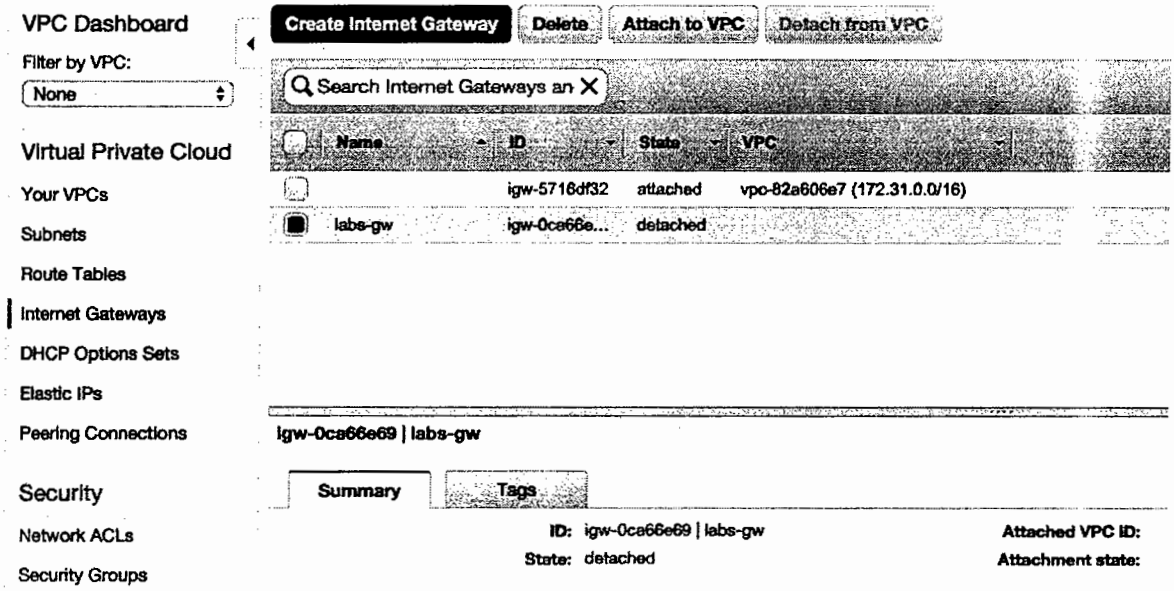
Creating a gateway is a one-step operation, you only need to choose a meaningful name.

Use `labs-gw` as **Name tag** and then click **Yes, Create**.

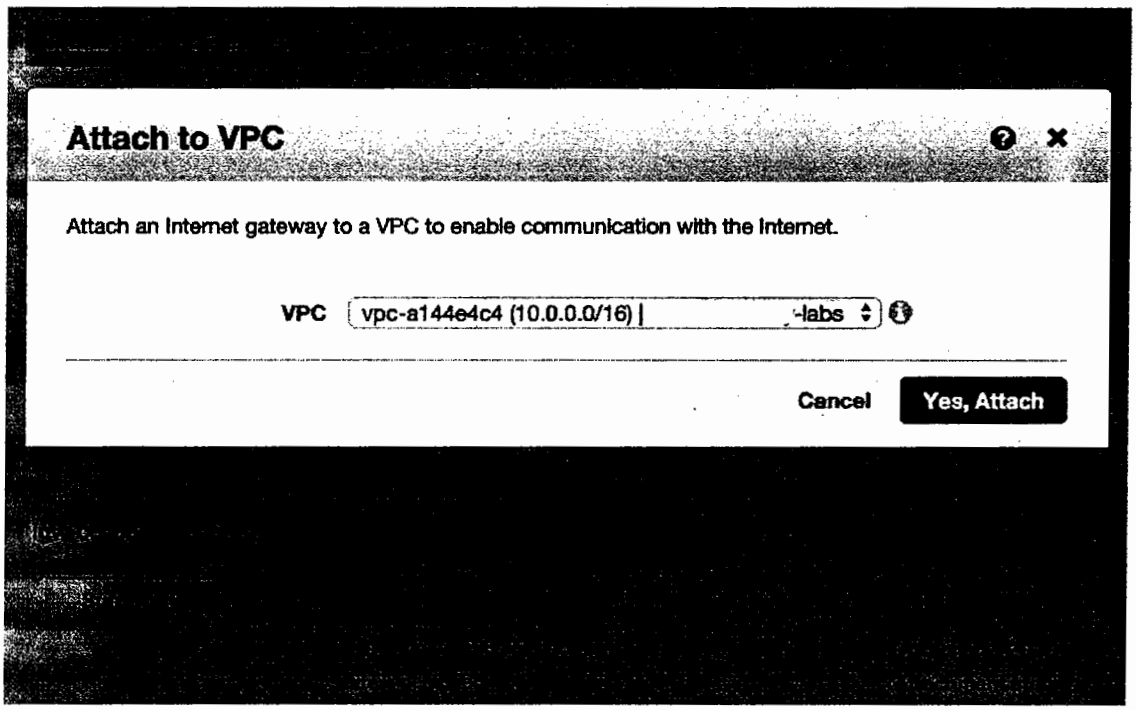


How to attach the Internet Gateway to a VPC

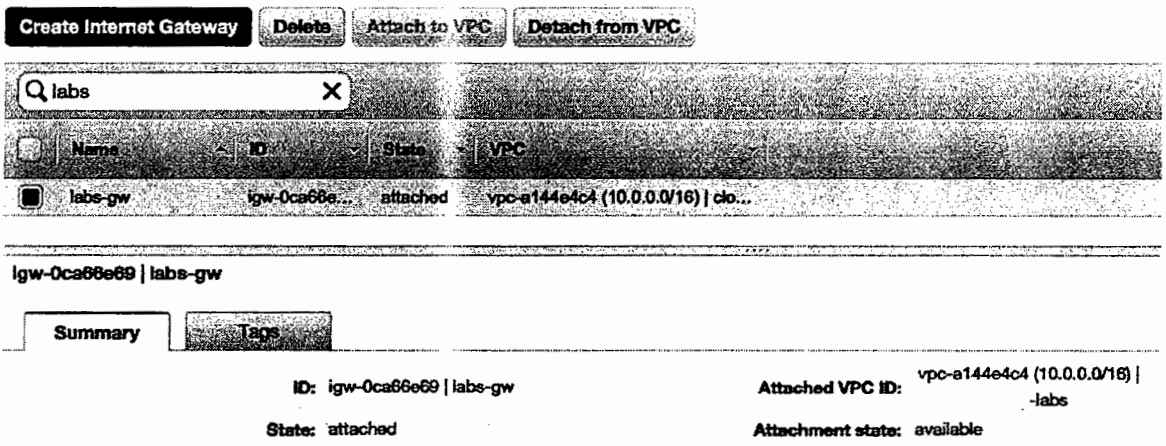
Select the Internet gateway that you just created, and then click **Attach to VPC**.



In the Attach to VPC dialog box, select the VPC **vepsun-labs** from the list, and then click **Yes, Attach**.



Your new Internet Gateway is ready to be used by the EC2 instances of the selected VPC.



STEP 4: Create a Public Subnet

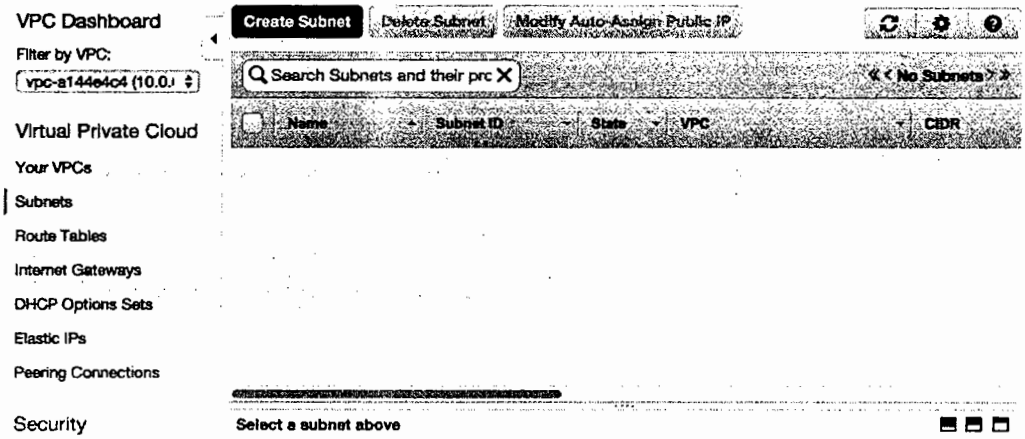
You can create a subnet for your VPC using the AWS Management Console.

Select the VPC service from the Management Console dashboard:

From the VPC dashboard, click on **Subnets** link in the sidebar menu.

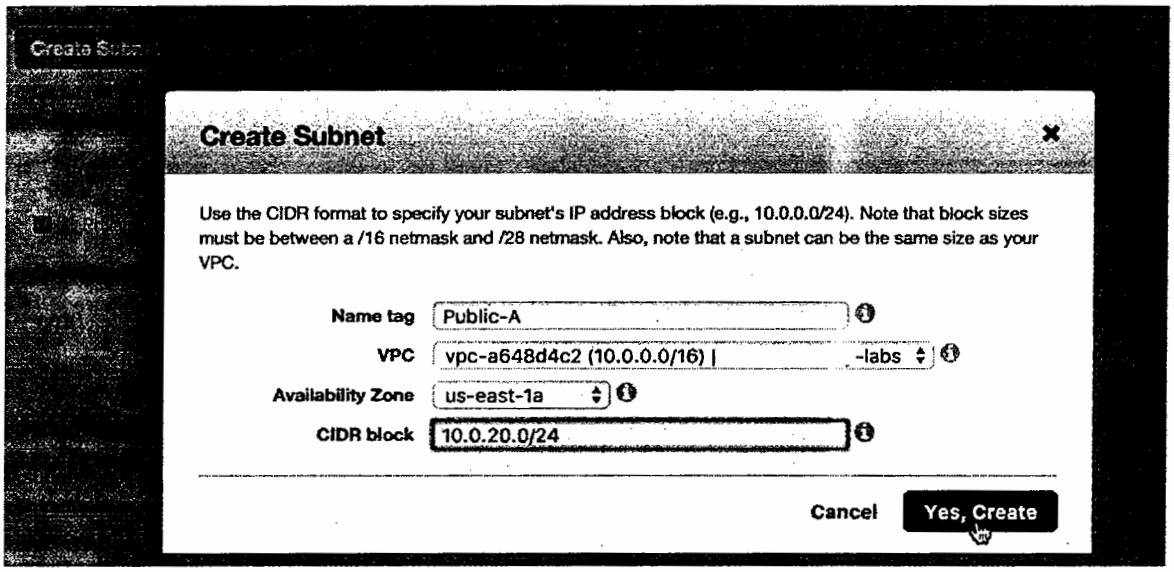
Your Subnets page lists all previously created subnets, you can use the **Filter by VPC** feature for listing only the services linked to a specific VPC.

Click on the **Create Subnet** blue button for starting the creation of a new subnet.



In the Create Subnet dialog box, specify the following Subnet details then click **Yes, Create**.

- ✓ **Name tag**: **Public-A**. This is the name for your subnet; doing so creates a tag with a key of Name and the value that you specify.
- ✓ **VPC**: **vepsun-labs**.
- ✓ **Availability Zone**: **us-west2a**.
- ✓ **CIDR block**: **10.0.20.0/24**. You should specify a CIDR block in the selected VPC.



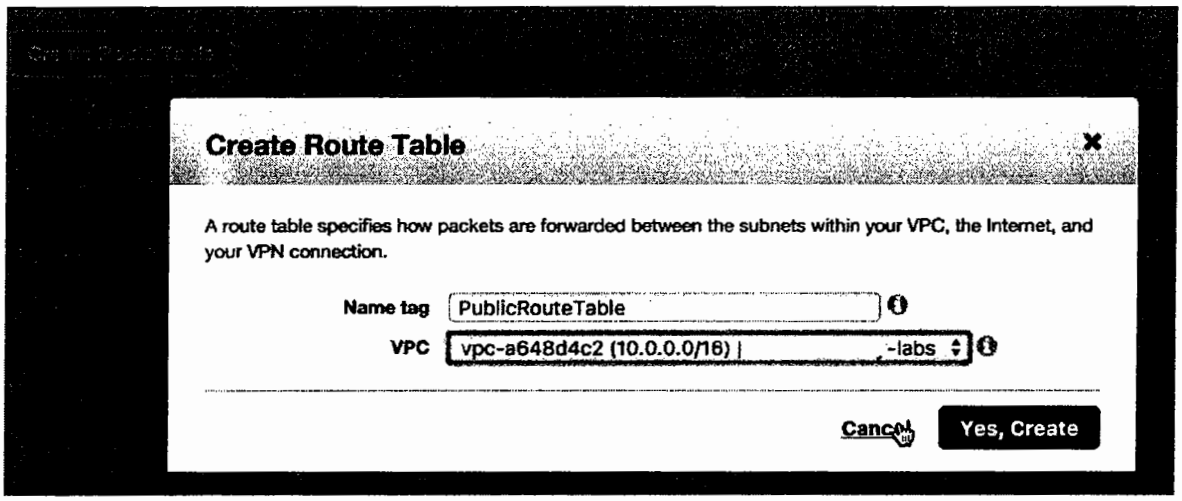
As you can see, the created subnet is automatically attached to the default VPC Route table and the default Network ACL.

A route table contains a set of rules, called routes that are used to determine where network traffic is directed. Each route in a table specifies a destination CIDR and a target (for example, traffic destined for 172.16.0.0/12 is targeted for the virtual private gateway). If a subnet have a route with the destination (0.0.0.0/0) and target the Internet Gateway, the subnet is known as a **public subnet.**

We can create a custom route table for VPC using the Amazon VPC console.

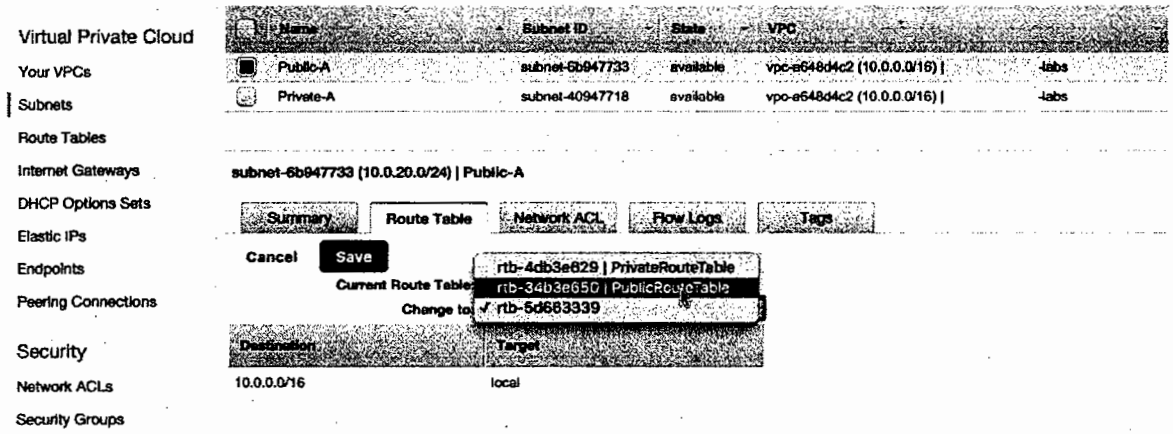
In the navigation pane, choose Route Tables.

1. Choose **Create Route Table**.
2. In the Create Route Table dialog box, you can optionally name your route table with **Name tag**: **Public-RouteTable**. This is the name for your subnet; doing so creates a tag with a key of Name and the value that you specify.
3. Select **VPC**: **vepsun-labs** from the VPC list, and then choose **Yes, Create**.



Now we must change the default route table of public subnet with the new route table. To change a subnet route table association

1. In the navigation pane, choose **Subnets**, and then select the subnet **Public-A**
2. In the Route Table tab, choose **Edit**.
3. Select the route table **Public-RouteTable** from the Change to list, and then choose Save.



As you can see, the subnet only has local routing (Destination 10.0.0.0/16 with Target local) so we need a route to internet with destination the Internet Gateway associated to our VPC.

We can add the route to internet:

1. In the navigation pane, choose **Route Tables**, and then select the route table **Public-RouteTable**

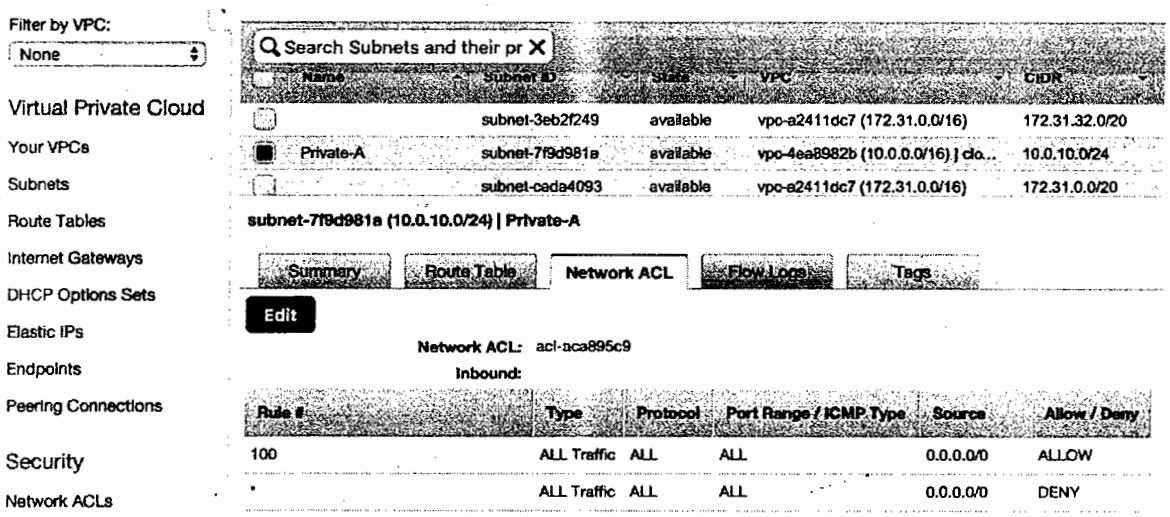
2. In the Routes tab, choose **Edit**.
3. Choose **Add another route** to add more routes
4. In the new rule set destination to **0.0.0.0/0** and Target to **labs-gw** then choose Save when you're done.

Now our subnet is a public subnet with route to Internet.

STEP 5: Create Network ACL for Public

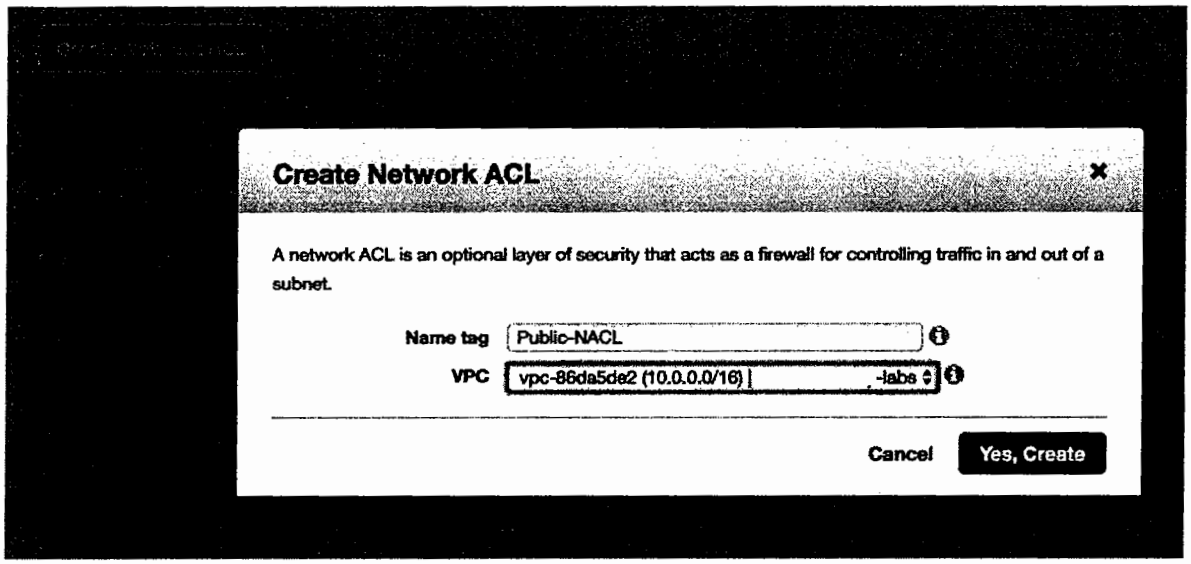
A network access control list (ACL) is an optional layer of security that acts as a firewall for controlling traffic in and out of a subnet. It is a numbered list of rules, evaluated in order to determine if traffic is allowed in or out of any associated subnet.

The VPC comes with a modifiable default network ACL and each subnet must be associated with a network ACL. As you can see from the image below, if you don't explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL that allows all inbound and outbound traffic.



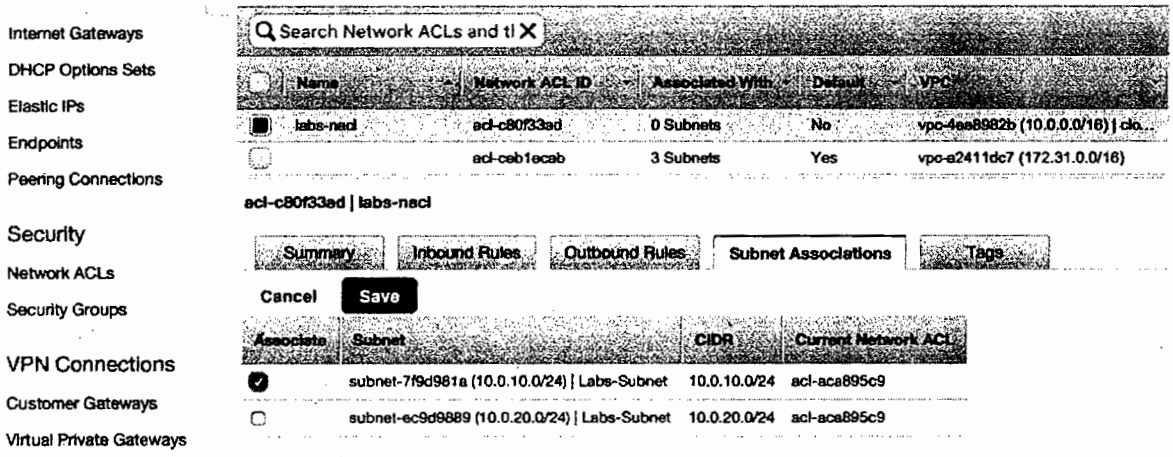
You can create custom network ACL at any time directly from Web Console:

1. In the navigation pane, choose Network ACLs.
2. Choose **Create Network ACL**.
3. In the Create Network ACL dialog box, optionally name your network ACL **Public-NACL** and then select **vepsun-labs** from the VPC list, and choose **Yes, Create**.



By default, a network ACL is not associated with a subnet until you explicitly associate it with one. To associate a subnet with a network ACL

1. In the navigation pane, choose Network ACLs, and then select the network ACL **Public-NACL**
2. In the details pane, on the Subnet Associations tab, choose **Edit**.
3. Select the Associate check box for the subnet **Public-A** to associate with the network ACL, and then choose **Save.**



Now you can configure your rules remembering that a Network ACLs is a numbered list of rules that we evaluate in order and are stateless: responses to allowed inbound traffic are subject to the rules for outbound traffic (and vice versa).

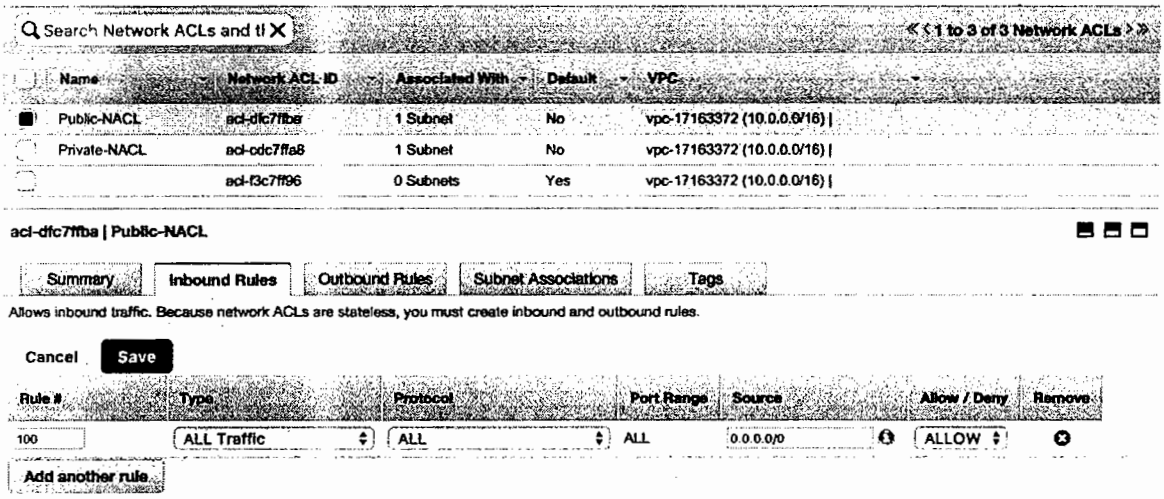
STEP 6: Add rules to Public Network ACL

A network ACL is a numbered list of rules that are evaluated in order, starting with the lowest numbered rule, to determine whether traffic is allowed in or out of any subnet associated with the network ACL. We recommend that you start by creating rules with rule numbers that are multiples of 100, so that you can insert new rules where you need to later on.

To add Inbound rules to a network ACL

1. In the navigation pane, choose **Network ACLs**.
2. From Network ACL list, select the ACL with Name **Public-NACL.**
3. In the details pane, choose the Inbound Rules and then choose **Edit**.
4. In Rule #, enter the rule number **100.**
5. From the Type list, select **ALL Traffic** rule.
6. In the Source field enter the CIDR range **0.0.0.0/0**
7. From the Allow/Deny list, select ALLOW to allow the specified traffic or DENY to deny the specified traffic.
8. Choose **Save** to save the new rule.

To add another rule, choose Add another rule, and repeat steps 4 to 8 as required.

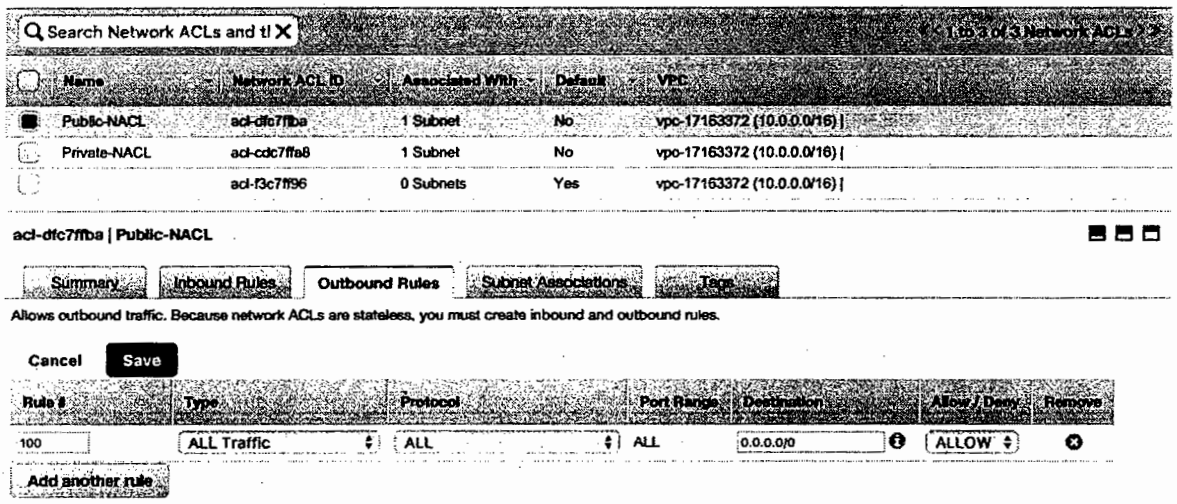


To add Outbound rules to a network ACL

1. In the navigation pane, choose **Network ACLs**.
2. From the Network ACL list, select the ACL with Name **Public-NACL.**

3. In the details pane, choose the Outbound Rules and then choose **Edit**.
4. In Rule #, enter the rule number **100**
5. From the Type list, select **ALL Traffic** rule.
6. In the Destination field enter the CIDR **0.0.0.0/0**
7. From the Allow/Deny list, select ALLOW to allow the specified traffic or DENY to deny the specified traffic.
8. Choose **Save** to save the new rule

To add another rule, choose Add another rule, and repeat steps 4 to 9 as required.



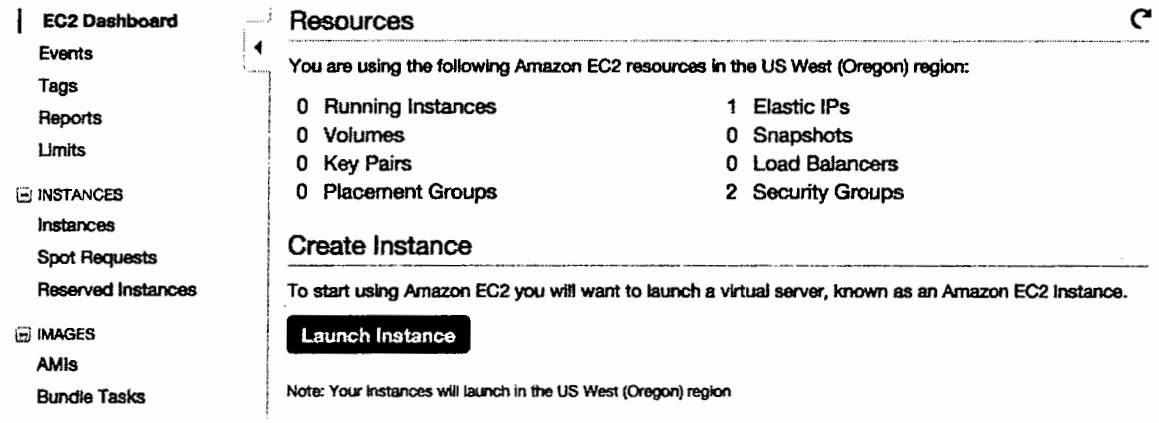
When you add or remove rules from a network ACL, the changes are automatically applied to the subnets it's associated with.

STEP 7: Launch NAT instance

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

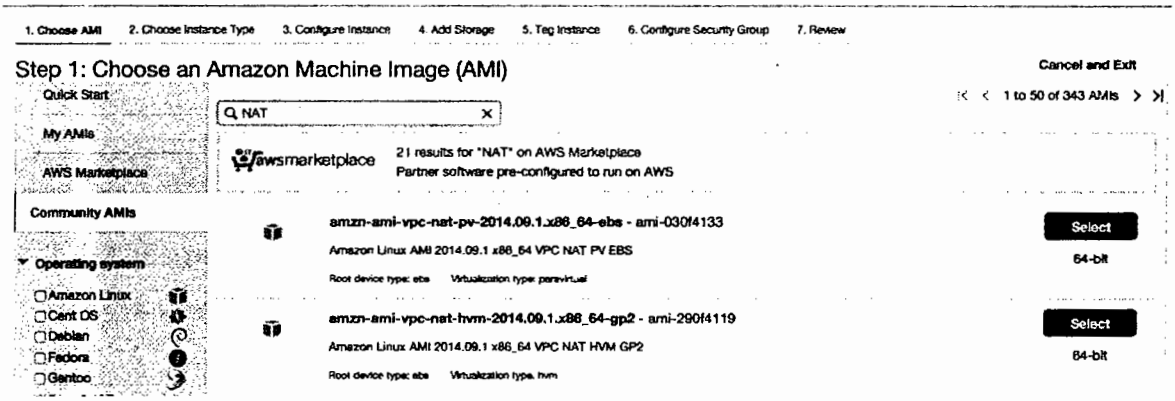
You can launch an EC2 instance using the EC2 launch wizard, selecting the EC2 service from the Management Console dashboard:

From the dashboard, click **Launch Instance**.

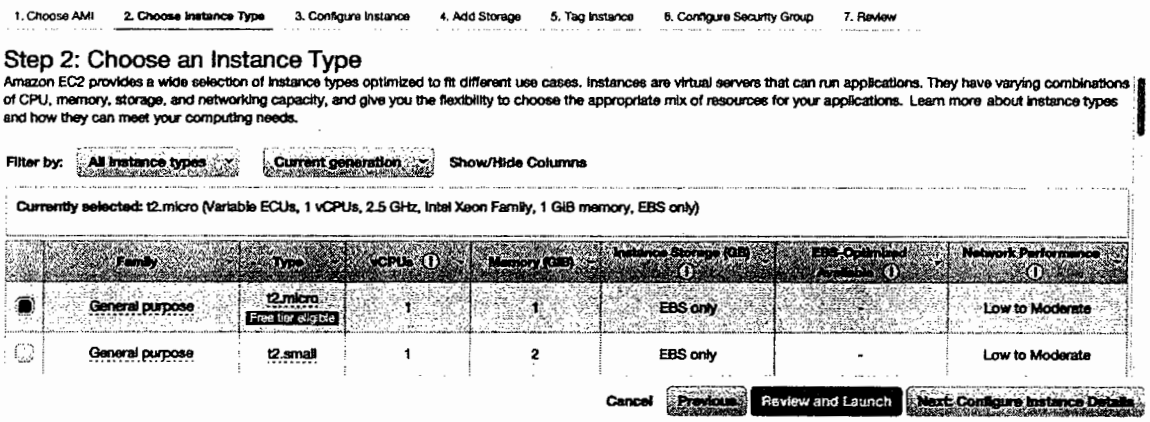


The **Select an Amazon Machine Image (AMI)** page displays a list of basic configurations called **Amazon Machine Images (AMIs)** that serve as templates for your instance.

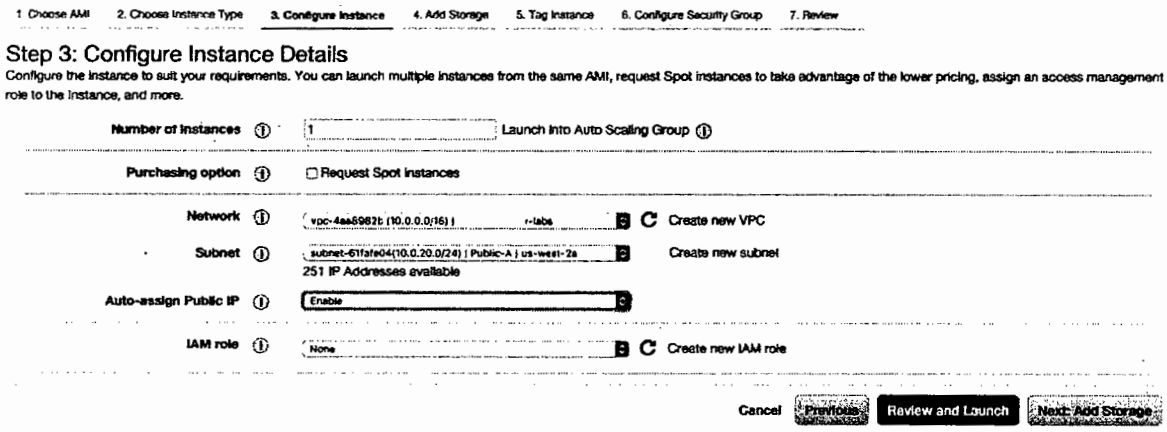
To select an NAT Instance click on **Community AMIs**, then input "NAT" in the search field and **select** the **amzn-ami-vpc-nat-hvm-2014.09.1.x86_64-gp2** AMI.



On the **Select an Instance Type** page, do not change any option and click **Next: Configure Instance Details.**



On the **3. Configure Instance** tab, check the selected **Network (VPC)** **vepsun-labs** and **Subnet** **Public-A** verify that Auto-assign Public IP is set to Enable, then click **Next: Add Storage**.



On the **4. Add Storage** tab, do not change any option and click **Next: Tag Instance** button.

On the **5. Tag Instance** use Key "Name" and Value "NAT Instance" and click **Next: Configure Security Group**

On the **6. Security Group,** select **Create a new security group** using name NAT and descriptions "Only for NAT" furthermore on Rule leave SSH Type and 0.0.0.0/0 on source and insert a new rule with type "ALL Traffic" and source with Custom IP to **10.0.0.0/16** then click **Review and Launch**

On the Review Instance Launch page, click **Launch**.

In the **Select an existing key pair or create a new key pair** dialog box, select **Create a new key pair**, then choose a KeyPair name and download it.

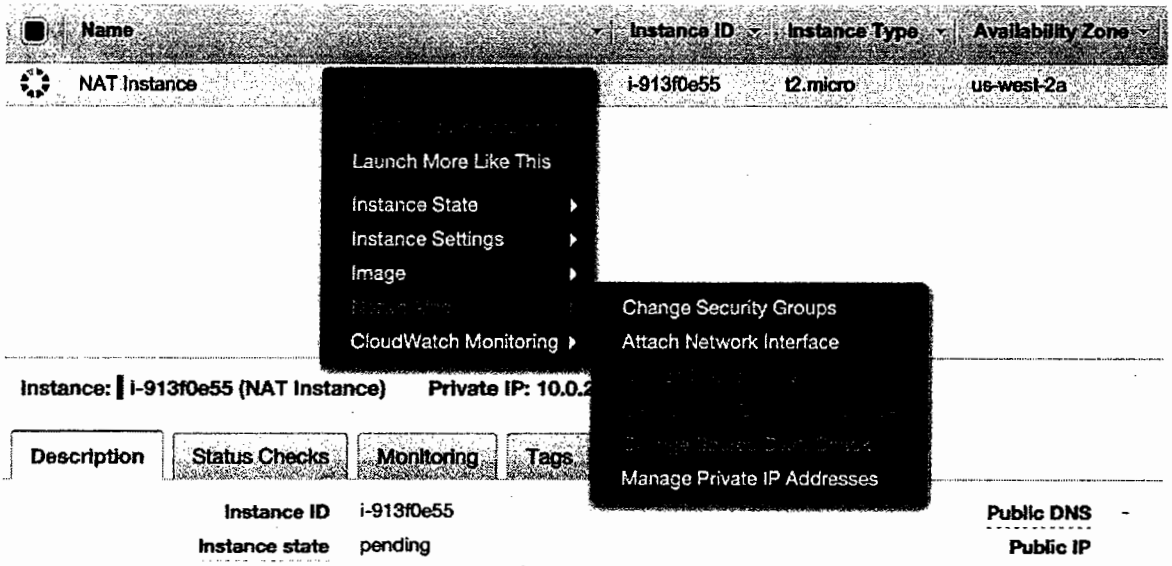
Select the acknowledgment checkbox, and then click **Launch Instances**.

A confirmation page will let you know that your instance is launching. Click **View Instances** to close the confirmation page and return to the console.

Now it's important to disable the Source destination Check on NAT Instance

1. Select the NAT instance, choose Actions, select **Networking**, and then select **Change Source/Dest. Check**.

2. Verify that this attribute is disabled. Otherwise, choose **Yes, Disable.**



Now you can use this instance for NAT

STEP 8: Create a Private Subnet

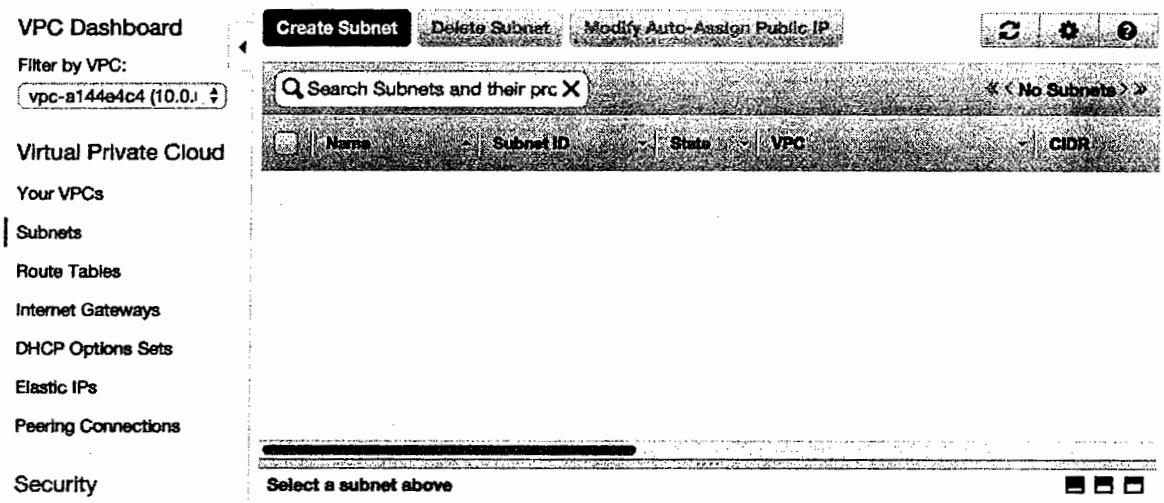
You can create a subnet for your VPC using the AWS Management Console.

Select the VPC service from the Management Console dashboard:

From the VPC dashboard, click on the **Subnets** link in the sidebar menu.

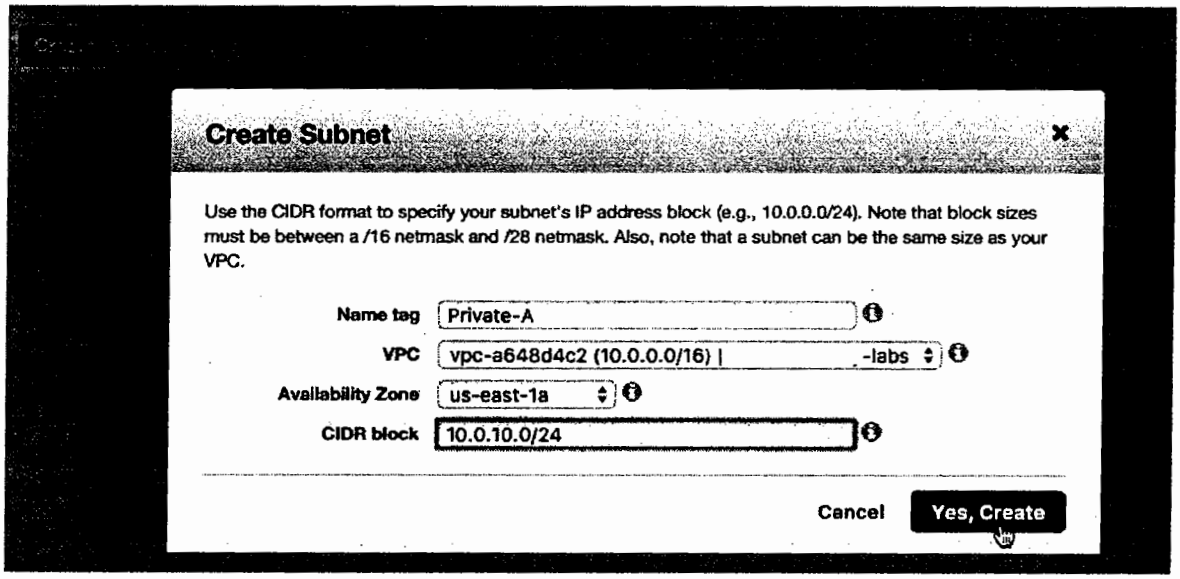
Your Subnets page lists all previously created subnets, you can use the **Filter by VPC** feature for listing only the services linked to a specific VPC.

Click the **Create Subnet** blue button for starting the creation of a new subnet.



In the Create Subnet dialog box, specify the following Subnet details, then click **Yes, Create**.

- ✓ **Name tag**: **Private-A**. This is the name for your subnet.
- ✓ **VPC**: **vepsun-labs**. This is your VPC.
- ✓ **Availability Zone**: **us-west2a**. The Availability Zone of Subnet.
- ✓ **CIDR block**: **10.0.10.0/24**. You should specify a CIDR block in the selected VPC.



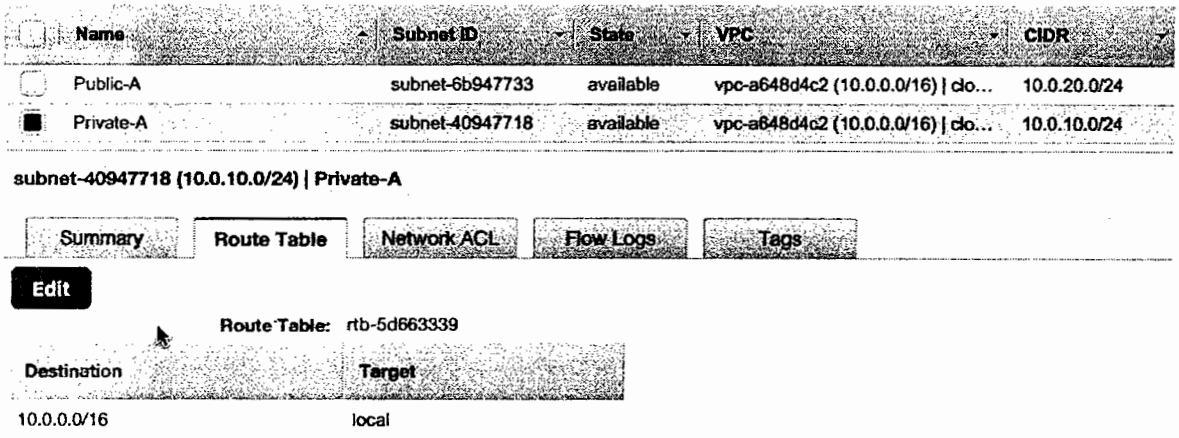
As you can see, the created subnet is automatically attached to the default VPC Route table and the default Network ACL.

A route table contains a set of rules, called routes, that are used to determine where network traffic is directed. Each route in a table specifies a destination CIDR and a target (for example, traffic destined for 172.16.0.0/12 is targeted for the virtual private gateway). If a subnet doesn't have a route to the Internet (0.0.0.0/0) through a gateway, the subnet is known as a **private subnet.**

We can create a custom route table for VPC using the Amazon VPC console.

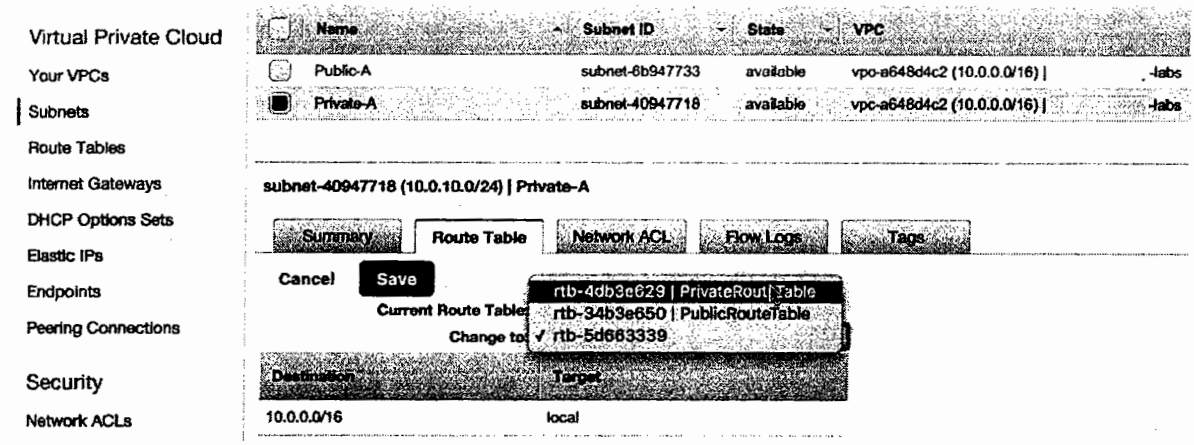
In the navigation pane, choose Route Tables.

1. Choose **Create Route Table**.
2. In the Create Route Table dialog box, you can optionally name your route table with **Name tag**: **Private-RouteTable**. This is the name for your subnet; doing so creates a tag with a key of Name and the value that you specify.
3. Select **VPC**: **vepsun-labs** from the VPC list, and then choose **Yes, Create**.



Now we should change the default route table of our private subnet with the new route table. To change a subnet route table association:

1. In the navigation pane, choose **Subnets**, and then select the subnet **Private-A.**
2. In the Route Table tab, choose **Edit**.
3. Select the route table **Private-RouteTable** from the Change to list, and then choose **Save.**



As you can see, the subnet only has local routing (Destination 10.0.0.0/16 with Target local).

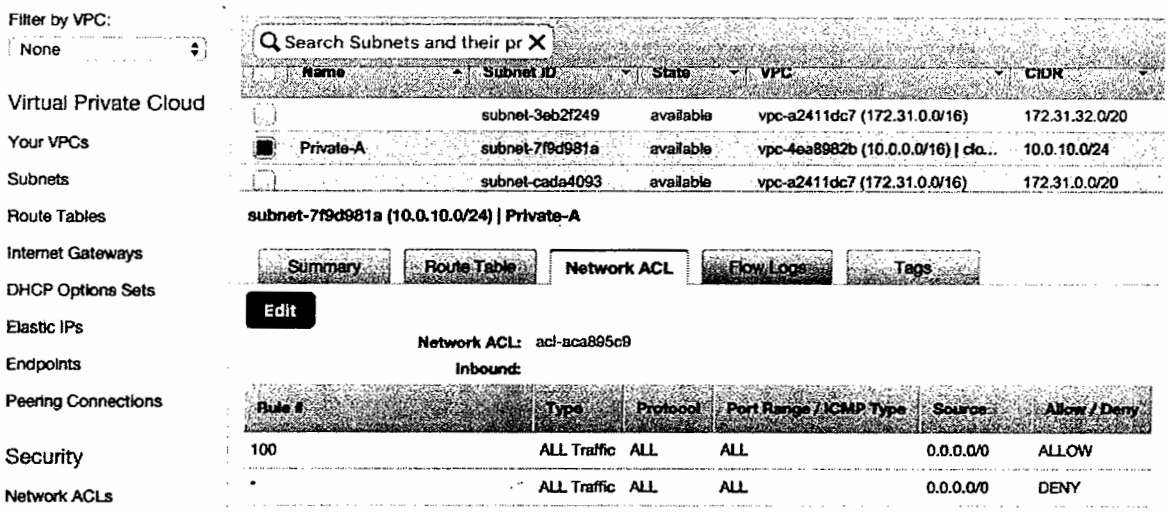
In the navigation pane, choose Route Tables, and then select the route table **Private-RouteTable:**

1. In the Routes tab, choose Edit.
2. choose Add another route and specify **0.0.0.0/0** in the Destination box.
3. select the target **NAT** from the Target list and then choose **Save.**

STEP 9: Create Network ACL for Private

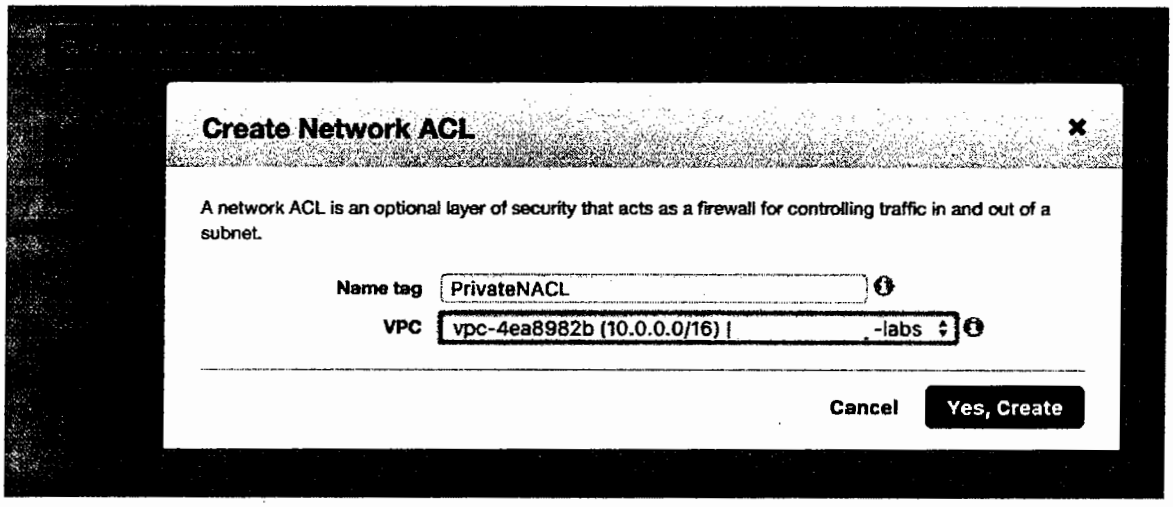
A network access control list (ACL) is an optional layer of security that acts as a firewall for controlling traffic in and out of a subnet. It is a numbered list of rules, evaluated in order, to determine whether traffic is allowed in or out of any associated subnet.

The VPC comes with a modifiable default network ACL and each subnet must be associated with a network ACL. As you can see from the image below, if you don't explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL that allows all inbound and outbound traffic.



You can create custom network ACL at any time directly from Web Console:

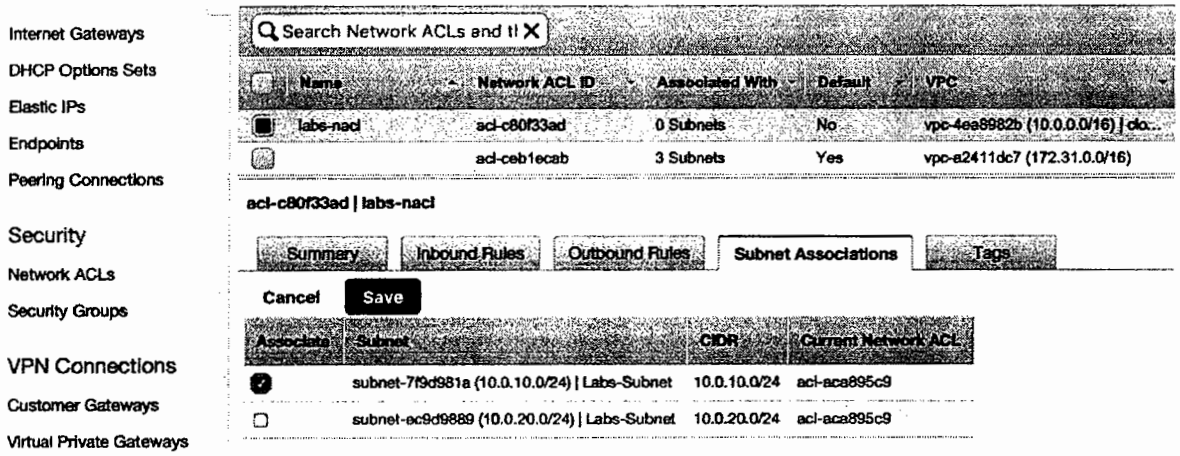
1. In the navigation pane, choose Network ACLs.
2. Choose **Create Network ACL**.
3. In the Create Network ACL dialog box, optionally name your network ACL **Private-NACL** and then select **vepsun-labs** from the VPC list, and choose **Yes, Create**.



By default, a network ACL is not associated with a subnet until you explicitly associate it with one. To associate a subnet with a network ACL:

1. In the navigation pane, choose Network ACLs, and then select the network ACL **Private-NACL.**
2. In the details pane, on the Subnet Associations tab, choose **Edit**.

3. Select the Associate check box for the subnet **Private-A** to associate with the network ACL, and then choose **Save.**



Now you can configure your rules. Remember that a Network ACLs is a numbered list of rules that we evaluate in order and is stateless: responses to allowed inbound traffic and is subject to the rules for outbound traffic (and vice versa).

STEP 10: Add rules to Private Network ACL

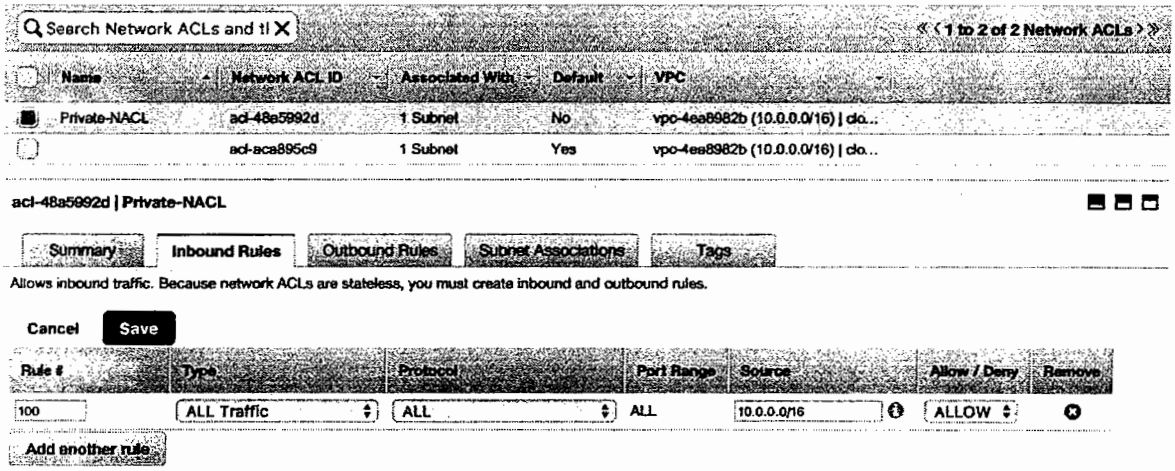
.A network ACL is a numbered list of rules that are evaluate in order, starting with the lowest numbered rule, to determine whether traffic is allowed in or out of any subnet associated with the network ACL. We recommend that you start by creating rules with rule numbers that are multiples of 100, so that you can insert new rules where you need to later on.

To add Inbound rules to a network ACL:

1. In the navigation pane, choose **Network ACLs**.
2. From the Network ACL list, select the ACL with Name **Private-NACL.**
3. In the details pane, choose the Inbound Rules and then choose **Edit**.
4. In Rule #, enter the rule number **100.**
5. From the Type list, select **ALL Traffic** rule.
6. In the Source field, enter the CIDR range **10.0.0.0/16.**
7. From the Allow/Deny list, select ALLOW to allow the specified traffic or DENY to deny the specified traffic.

8. Choose **Save** to save the new rule.

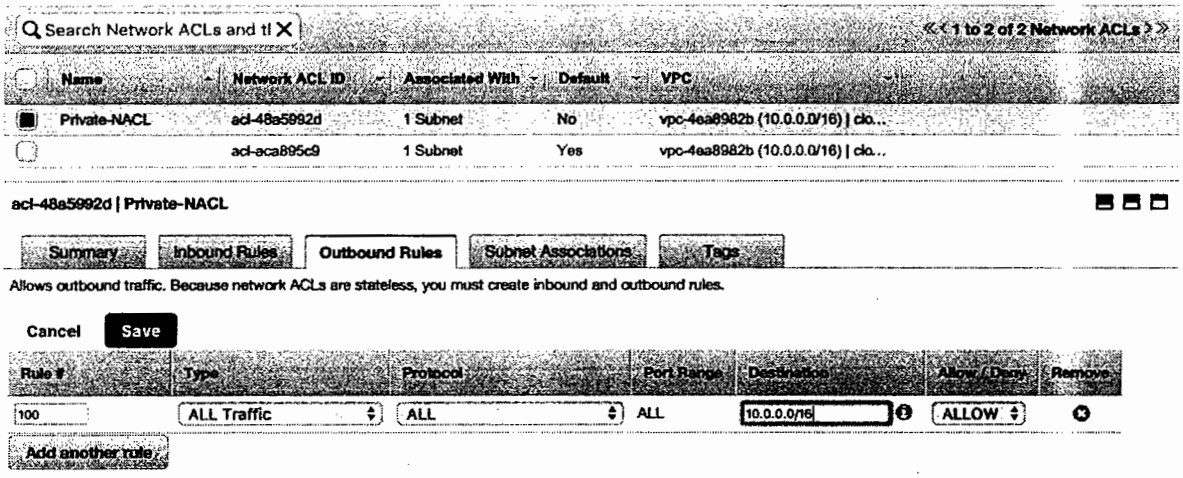
To add another rule, choose Add another rule, and repeat steps 4 to 9 as required.



To add Outbound rules to a network ACL:

1. In the navigation pane, choose **Network ACLs**.
2. From the Network ACL list, select the ACL with Name **Private-NACL.**
3. In the details pane, choose the Outbound Rules and then choose **Edit**.
4. In Rule #, enter the rule number **100.**
5. From the Type list, select **ALL Traffic** rule.
6. In the Destination field enter the CIDR **10.0.0.0/16.**
7. From the Allow/Deny list, select ALLOW to allow the specified traffic.
8. Choose **Save** to save the new rule.

To add another rule, choose Add another rule, and repeat steps 4 to 9 as required.



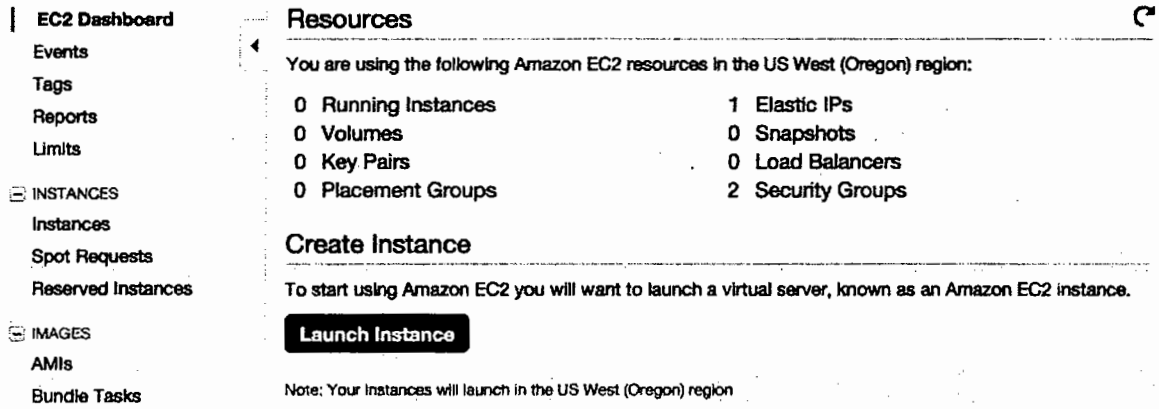
When you add or remove rules from a network ACL, the changes are automatically applied to the subnets it's associated with.

STEP 11: Launch EC2 instance on private subnet

You can launch an EC2 instance using the EC2 launch wizard.

Select the EC2 service from the Management Console dashboard:

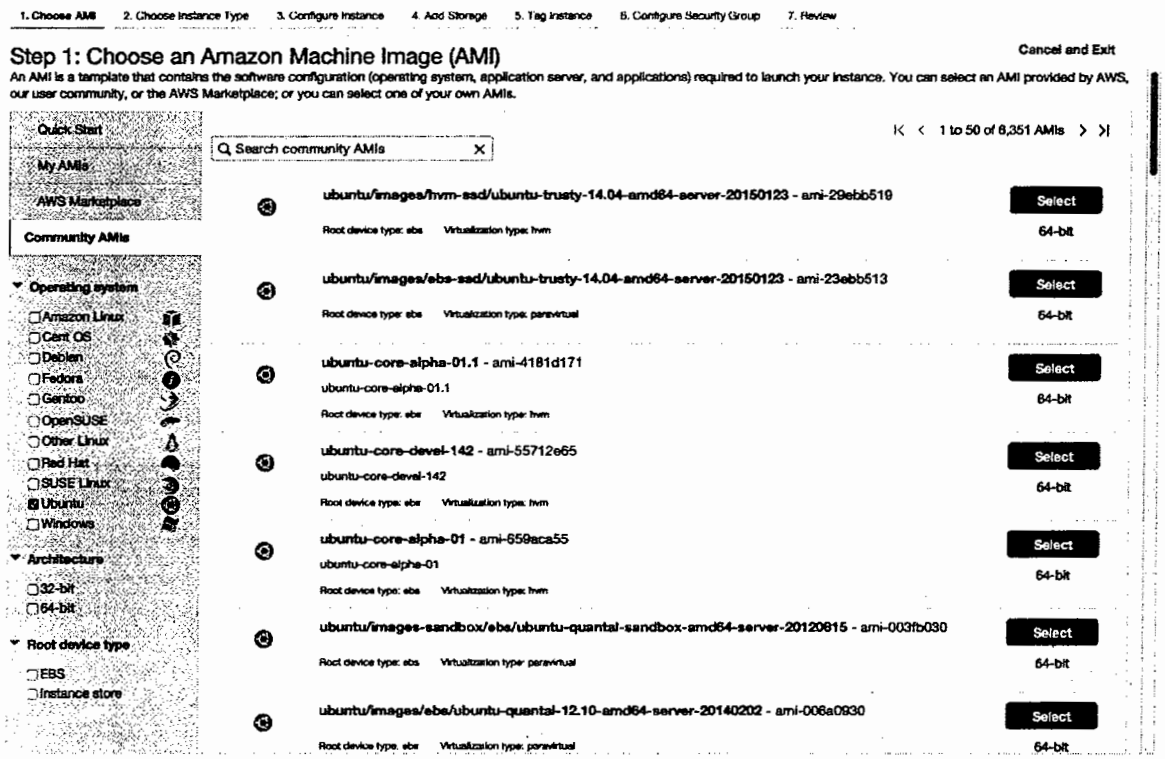
From the dashboard, click **Launch Instance**.



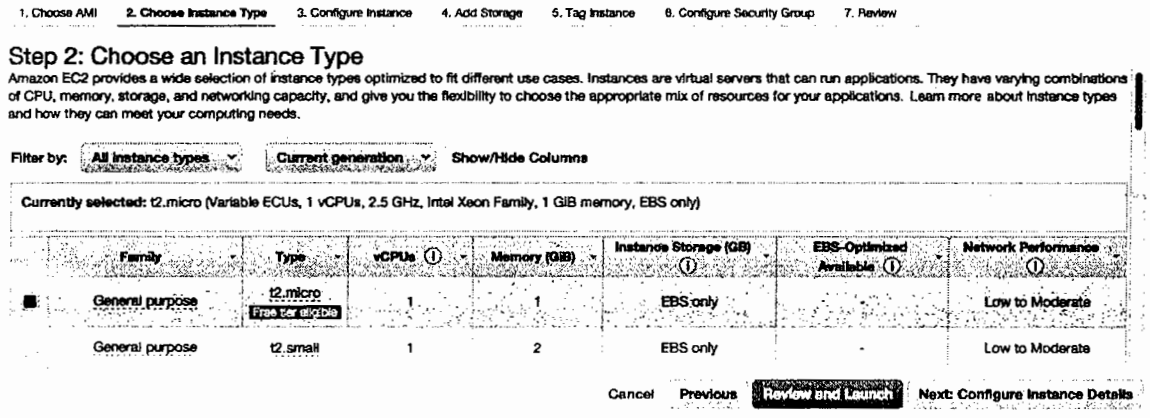
The **Select an Amazon Machine Image (AMI)** page displays a list of basic configurations called **Amazon Machine Images (AMIs)** that serve as templates for your instance.

Click on **Community AMIs**, then select the **Ubuntu** Operating System, **x86_64** Architecture and **ebs** Root device type.

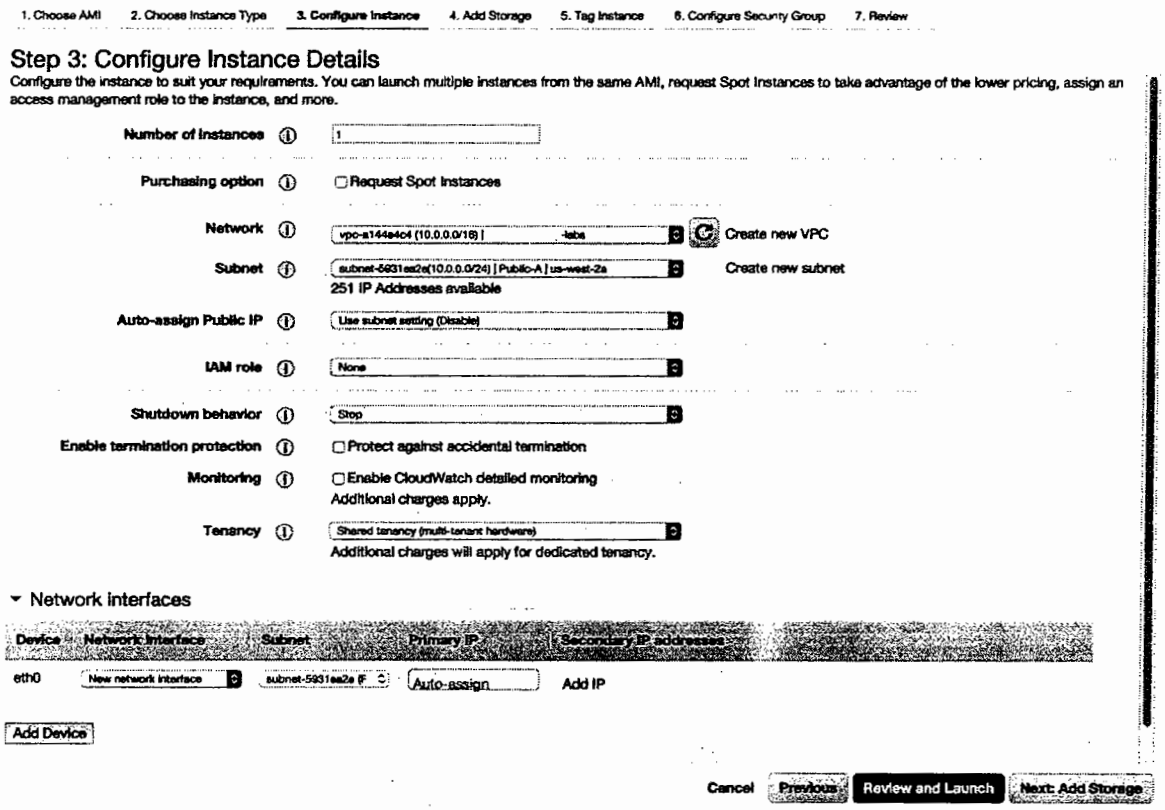
Wait until the AMI list has not been updated and then find and select the AMI.



On the **Select an Instance Type** page, do not change any option and click on **Next, Configure Instance Details.**



On the **3. Configure Instance** tab, select **Network** **10.0.0.0/16** and **Subnet** **Private-A** and then click **Next, Add Storage**.



On the **4. Add Storage** tab, do not change any option and click "**Review and Launch**" button.

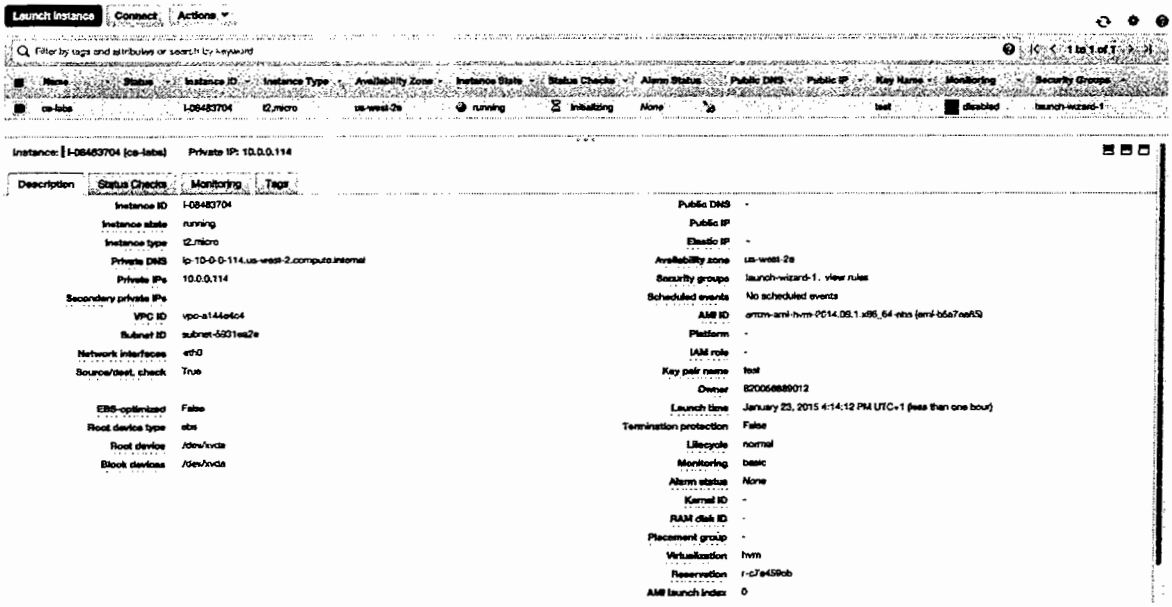
On the Review Instance Launch page, click **Launch**.

In the **Select an existing key pair or create a new key pair** dialog box, select **Create a new key pair**, then choose a KeyPair name and download it.

Select the acknowledgment check box, and then click **Launch Instances**.

A confirmation page will let you know that your instance is launching. Click **View Instances** to close the confirmation page and return to the console.

On the Instances screen, you can view the status of your instance. It will take a short time for your instance to be launched. When you launch an instance, its initial state is **pending**. After the instance starts, its state changes to **running**, and it receives a public DNS name.



STEP 12: Securing your VPC resume

In the laboratory, the VPC has been configured with two subnets, a public subnet, and a private subnet. If a subnet's traffic is routed to an Internet gateway, the subnet is known as a *public subnet*. If a subnet doesn't have a route to the Internet gateway, the subnet is known as a *private subnet*. The Subnets are associated with a specific Route Table and ACL. A Public Route Table, where there is an NAT Instance, is the route table with a route to an Internet Gateway, while the Private Route Table is a table with a route to NAT Instances. The ACL is an optional layer of security that acts as a firewall for controlling traffic in and out of a Public or Private Subnet.

Instances that you launch into a private subnet in a virtual private cloud (VPC) can't communicate with the Internet. You can optionally use a network address translation (NAT) instance in a public subnet in your VPC to enable instances in the private subnet to initiate outbound traffic to the Internet, but prevent the instances from receiving inbound traffic initiated by someone on the Internet. The following figure illustrates how the NAT instance works. The Private route table sends the traffic from the instances in the private subnet to the NAT instance in the public subnet. The NAT instance sends the traffic to the Internet gateway for the VPC. The traffic is attributed to the Elastic IP address of the NAT instance.