

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 screenshot shows the AWS Management Console interface. At the top, there's a header bar with the AWS logo, a 'Services' dropdown menu, and user information: 'Antonio Ang', 'Oregon', and a 'Support' link. Below the header, the main content area is titled 'Amazon Web Services' and is organized into several columns of service categories, each with an icon and a brief description. The categories include Compute (EC2, Lambda), Storage & Content Delivery (S3, Storage Gateway, Glacier, CloudFront), Database (RDS, DynamoDB, ElastiCache, Redshift), Networking (VPC, Direct Connect, Route 53), Administration & Security (Directory Service, IAM, Trusted Advisor, CloudTrail, Config, CloudWatch), Deployment & Management (Elastic Beanstalk, OpsWorks, CloudFormation, CodeDeploy), Analytics (EMR, Kinesis, Data Pipeline), Application Services (SQS, SWF, AppStream, Elastic Transcoder, SES, CloudSearch), Mobile Services (Cognito, Mobile Analytics, SNS), and Enterprise Applications (WorkSpaces, Zocalo). To the right of these categories, there's a section titled 'Additional Resources' which includes links for 'Getting Started', 'AWS Console Mobile App', 'AWS Marketplace', 'Service Health', and 'Set Start Page'. The 'Service Health' section shows a status of 'All services operating normally' as of Nov 20 2014 12:57:00 GMT-0800. At the bottom of the 'Set Start Page' section, there's a 'Console Home' button.

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 **xxxxx** and the password **xxxxx**.



Account:

User Name:

Password:

 I have an MFA Token ([more info](#))

Sign In

[Sign in using root account credentials](#)

[Terms of Use](#) [Privacy Policy](#)
© 1996-2014, Amazon Web Services, Inc. or its affiliates.

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.

Antonio Ang ▾ Oregon ▾ Support ▾

US East (N. Virginia)

| US West (Oregon)

US West (N. California)

EU (Ireland)

EU (Frankfurt)

Asia Pacific (Singapore)

Asia Pacific (Tokyo)

Asia Pacific (Sydney)

South America (São Paulo)

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:

Networking



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

VPC Dashboard
Filter by VPC:

None

Virtual Private Cloud
Your VPCs
Subnets
Route Tables
Internet Gateways
DHCP Options Sets
Elastic IPs

Resources

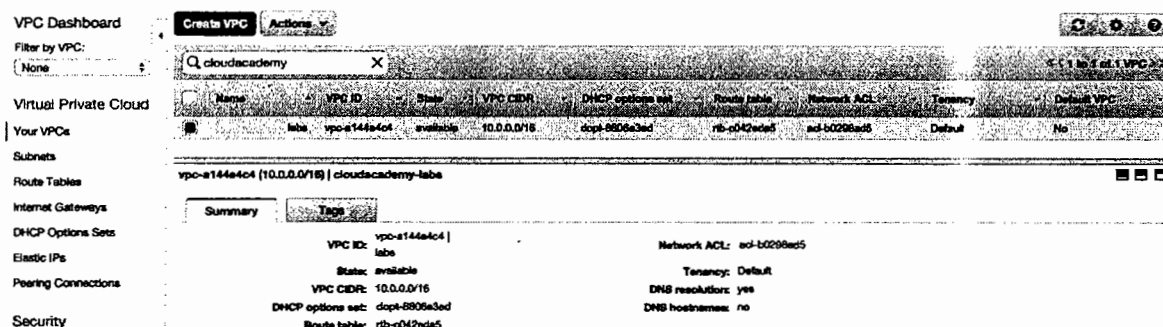
Start VPC Wizard**Launch EC2 Instances**

Note: Your instances will launch in the US West (Oregon) region.

You are using the following Amazon VPC resources in the US West (Oregon) region:

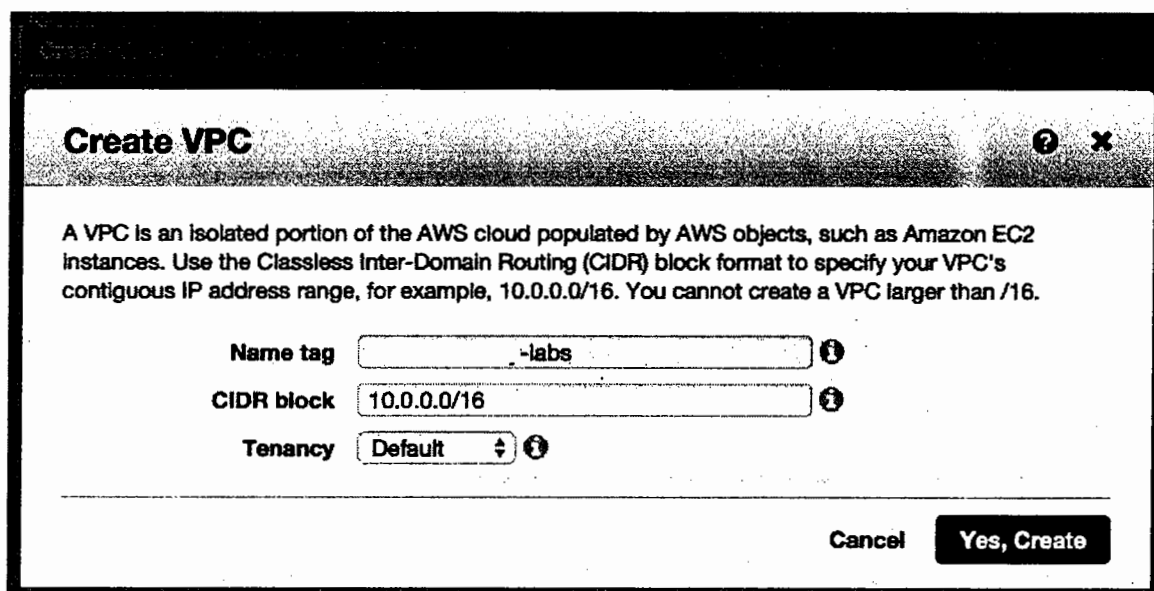
1 VPC	1 Internet Gateway
0 Subnets	1 Route Table
1 Network ACL	1 Elastic IP
1 Security Group	0 Running Instances
0 VPC Peering Connections	0 Customer Gateways
0 VPN Connections	0 Virtual Private Gateways

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**.

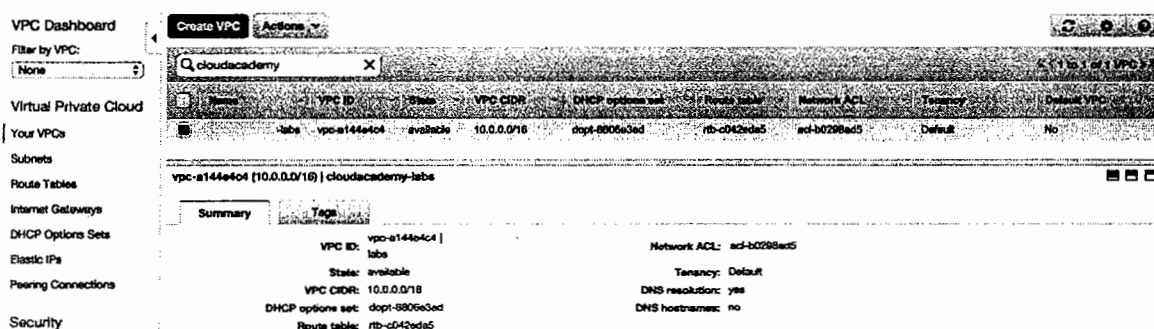
- ✓ **Name tag:** **vepsun-labs**. This is the name for your VPC; doing so creates a tag with a key of Name and the value that you specify.
- ✓ **CIDR block:** **10.0.0.0/16**. You should specify a CIDR block from the private (non-publicly routable) IP address ranges as specified in RFC 1918.
- ✓ **Tenancy:**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

Networking



VPC

Isolated Cloud Resources

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.

The screenshot shows the AWS VPC Dashboard. On the left is a navigation menu with options: VPC Dashboard, Filter by VPC: (set to None), Virtual Private Cloud, Your VPCs, Subnets, Route Tables, Internet Gateways (selected), DHCP Options Sets, Elastic IPs, and Peering Connections. The main area has a top bar with buttons: Create Internet Gateway (highlighted in blue), Delete, Attach to VPC, and Detach from VPC. Below this is a search bar labeled 'Search Internet Gateways and X'. A table lists existing gateways with columns: Name, ID, State, and VPC. One gateway is listed: igw-5718df32, attached, vpc-82a606e7 (172.31.0.0/16). At the bottom, it says 'Select an Internet gateway above'.

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**.

The screenshot shows a modal dialog box titled 'Create Internet Gateway'. It contains the text: 'An Internet gateway is a virtual router that connects a VPC to the Internet.' Below this is a 'Name tag' label followed by a text input field containing 'labs-gw'. To the right of the input field is an information icon. At the bottom right of the dialog are two buttons: 'Cancel' and '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**.

The screenshot shows the AWS VPC Dashboard. On the left, the 'VPC Dashboard' menu is open, and 'Internet Gateways' is selected. The main area displays a table of Internet Gateways. The gateway 'igw-0ca66e69' is highlighted, and the 'Attach to VPC' button is visible. Below the table, the 'Summary' tab for 'igw-0ca66e69 | labs-gw' is shown, indicating it is currently 'detached' from any VPC.

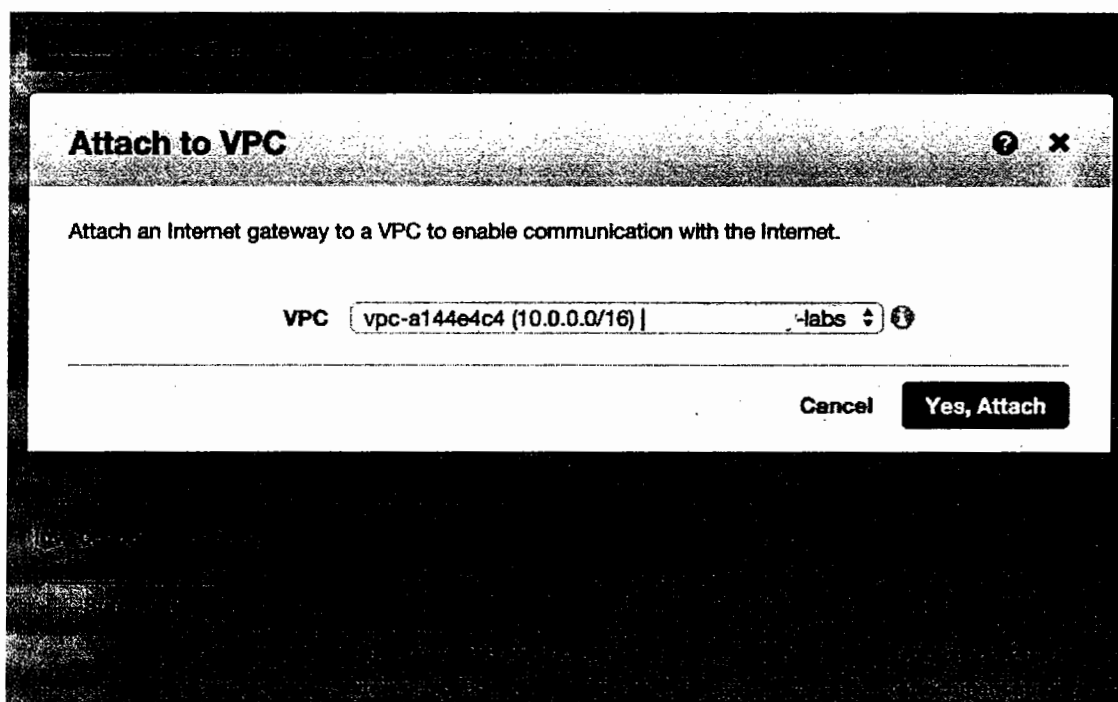
Name	ID	State	VPC
igw-5718df32	igw-5718df32	attached	vpc-82a606e7 (172.31.0.0/16)
igw-0ca66e69	igw-0ca66e69	detached	

igw-0ca66e69 | labs-gw

Summary Tags

ID: igw-0ca66e69 | labs-gw
State: detached
Attached VPC ID:
Attachment state:

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.

Create Internet Gateway Delete Attach to VPC Detach from VPC

Q labs X

Name	ID	State	VPC
labs-gw	igw-0ca68e69	attached	vpc-a144e4c4 (10.0.0.0/16) do...

igw-0ca68e69 | labs-gw

Summary Tags

ID: igw-0ca68e69 | labs-gw Attached VPC ID: vpc-a144e4c4 (10.0.0.0/16) | -labs

State: attached Attachment state: available

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:

Networking



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.

VPC Dashboard

Create Subnet Delete Subnet Modify Auto-Assign Public IP

Filter by VPC: vpc-a144e4c4 (10.0.0.0/16)

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

Elastic IPs

Peering Connections

Security

Search Subnets and their prc X

<< No Subnets >>

Name	Subnet ID	State	VPC	CDR
------	-----------	-------	-----	-----

Select a subnet above

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.

Create Subnet

Use the CIDR format to specify your subnet's IP address block (e.g., 10.0.0.0/24). Note that block sizes must be between a /16 netmask and /28 netmask. Also, note that a subnet can be the same size as your VPC.

Name tag:

VPC:

Availability Zone:

CIDR block:

Cancel **Yes, Create**

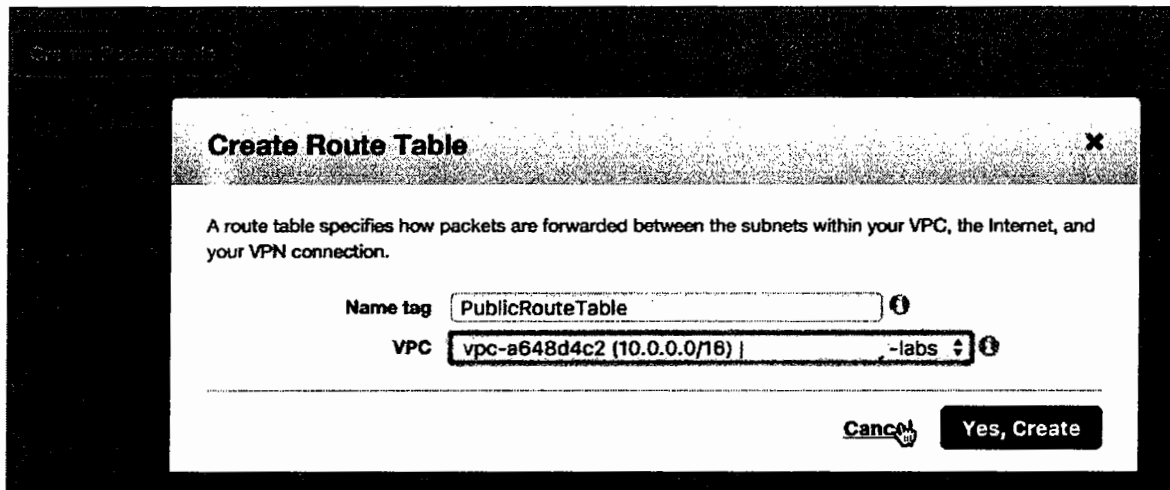
As you can see, the created subnet is automatically attached to the default VPC Route table and the default Network ACL.

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.

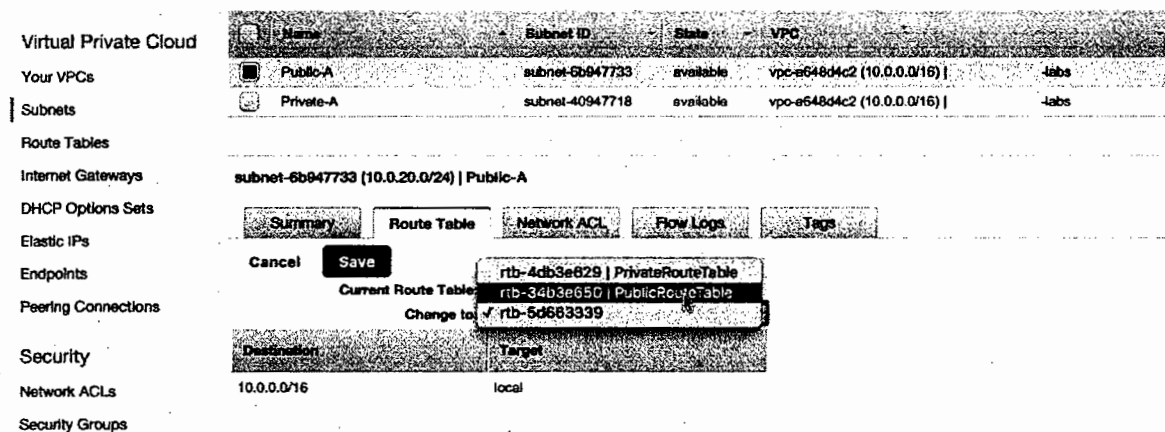
In the navigation pane, choose Route Tables.

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.

Filter by VPC: **None**

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Peering Connections

Security

Network ACLs

Search Subnets and their pr X

Name	Subnet ID	Status	VPC	CIDR
	subnet-3eb2f249	available	vpc-a2411dc7 (172.31.0.0/16)	172.31.32.0/20
<input checked="" type="checkbox"/> Private-A	subnet-7f9d981a	available	vpc-4ea8982b (10.0.0.0/16) clo...	10.0.10.0/24
	subnet-cada4093	available	vpc-a2411dc7 (172.31.0.0/16)	172.31.0.0/20

subnet-7f9d981a (10.0.10.0/24) | Private-A

Summary Route Table Network ACL Flow Logs Tags

Edit

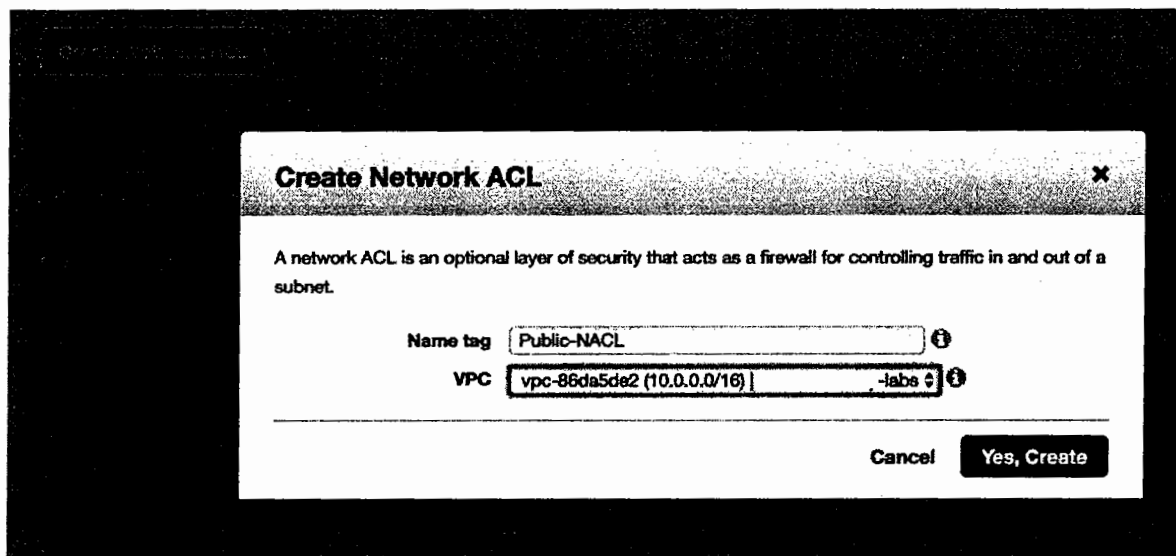
Network ACL: acl-aca895c9

Inbound:

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

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**.

Associate	Subnet	CIDR	Current Network ACL
<input checked="" type="checkbox"/>	subnet-7f9d981a (10.0.10.0/24) Labs-Subnet	10.0.10.0/24	acl-aca895c9
<input type="checkbox"/>	subnet-ec9d9889 (10.0.20.0/24) Labs-Subnet	10.0.20.0/24	acl-aca895c9

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.

The screenshot shows the AWS Network ACL console. At the top, there is a search bar and a link to view all Network ACLs. Below this is a table listing the Network ACLs:

Name	Network ACL ID	Associated With	Default	VPC
Public-NACL	acl-dfc7f1ba	1 Subnet	No	vpc-17163372 (10.0.0.0/16)
Private-NACL	acl-cdc7f1ba	1 Subnet	No	vpc-17163372 (10.0.0.0/16)
	acl-f3c7f1ba	0 Subnets	Yes	vpc-17163372 (10.0.0.0/16)

Below the table, the details for the selected Network ACL (acl-dfc7f1ba | Public-NACL) are shown. The 'Inbound Rules' tab is selected, displaying a list of rules. The first rule is shown with the following details:

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

At the bottom of the console, there is a button to 'Add another rule'.

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.

The screenshot shows the AWS Management Console interface for Network ACLs. At the top, there is a search bar and a breadcrumb trail: "Home > VPC > Network ACLs". Below this is a table listing Network ACLs:

Name	Network ACL ID	Associated With	Default	VPC
Public-NACL	acl-dfc7f7ba	1 Subnet	No	vpc-17163372 (10.0.0.0/16)
Private-NACL	acl-cdc7ffa8	1 Subnet	No	vpc-17163372 (10.0.0.0/16)
	acl-f3c7ff96	0 Subnets	Yes	vpc-17163372 (10.0.0.0/16)

Below the table, the details for "acl-dfc7f7ba | Public-NACL" are shown. There are tabs for "Summary", "Inbound Rules", "Outbound Rules", "Subnet Associations", and "Tags". The "Outbound Rules" tab is selected. Below the tabs, a message states: "Allows outbound traffic. Because network ACLs are stateless, you must create inbound and outbound rules." At the bottom of the details pane are "Cancel" and "Save" buttons.

Below the details pane is a table for adding or editing rules:

Rule #	Type	Protocol	Port Range	Destination	Allow / Deny	Remove
100	ALL Traffic	ALL	ALL	0.0.0.0/0	ALLOW	X

At the bottom left of the rule table is a button labeled "Add another rule".

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:

Compute



EC2

Virtual Servers in the Cloud

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

EC2 Dashboard

- Events
- Tags
- Reports
- Limits
- INSTANCES**
 - Instances
 - Spot Requests
 - Reserved Instances
- IMAGES**
 - AMIs
 - Bundle Tasks

Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

0 Running Instances	1 Elastic IPs
0 Volumes	0 Snapshots
0 Key Pairs	0 Load Balancers
0 Placement Groups	2 Security Groups

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 Instance.

Launch Instance

Note: Your instances will launch in the US West (Oregon) region

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.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

Quick Start

My AMIs

AWS Marketplace

Community AMIs

Operating system

- ☐ Amazon Linux
- ☐ CentOS
- ☐ Debian
- ☐ Fedora
- ☐ Gentoo

Search: NAT

21 results for "NAT" on AWS Marketplace
Partner software pre-configured to run on AWS

	amzn-ami-vpc-nat-pv-2014.09.1.x86_64-efs - ami-030f4133 Amazon Linux AMI 2014.09.1 x86_64 VPC NAT PV EBS Root device type: ebs Virtualization type: paravirtual	Select 64-bit
	amzn-ami-vpc-nat-hvm-2014.09.1.x86_64-gp2 - ami-290f4119 Amazon Linux AMI 2014.09.1 x86_64 VPC NAT HVM GP2 Root device type: ebs Virtualization type: hvm	Select 64-bit

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

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

Filter by: **All instance types** **Current generation** Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs ①	Memory (GB)	Instance Storage (GB) ①	EBS-Optimized Instance ①	Network Performance ①
<input checked="" type="radio"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate
<input type="radio"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate

Cancel Previous **Review and Launch** 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**.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of Instances ① Launch into Auto Scaling Group ①

Purchasing option ① ☐ Request Spot Instances

Network ① **r-labs** ☒ Create new VPC

Subnet ① **251 IP Addresses available** ☒ Create new subnet

Auto-assign Public IP ① ☒ Enable

IAM role ① ☒ Create new IAM role

Cancel Previous **Review and Launch** 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**.

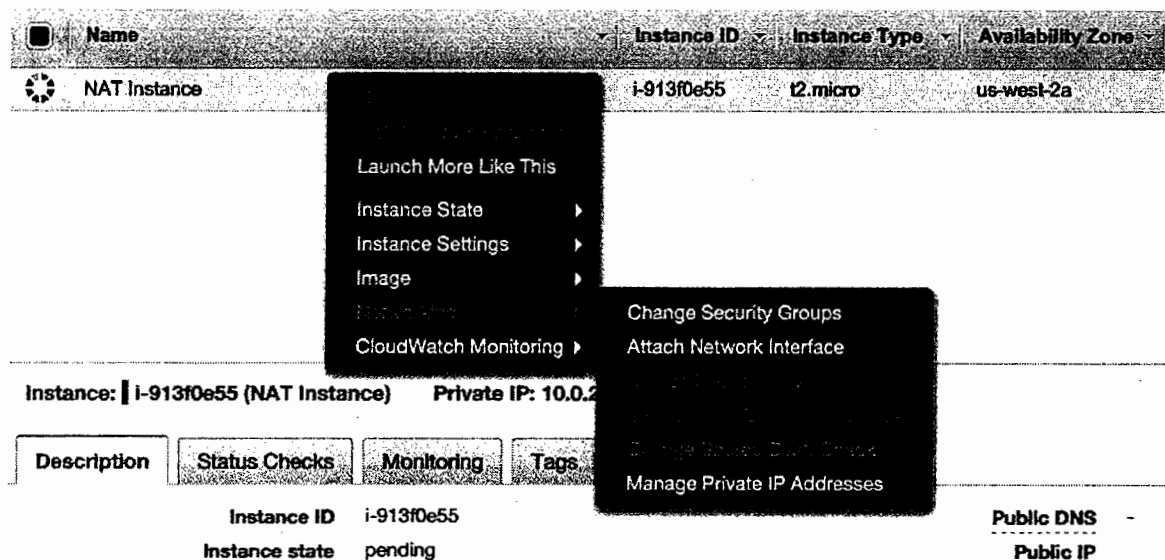
In the **Select an existing key pair or create a new key pair** dialog box, select **Create a new key pair**, then choose a KeyPair name and download it.

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**.

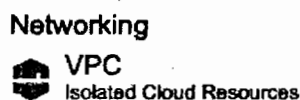


Now you can use this instance for NAT

STEP 8: Create a Private Subnet

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

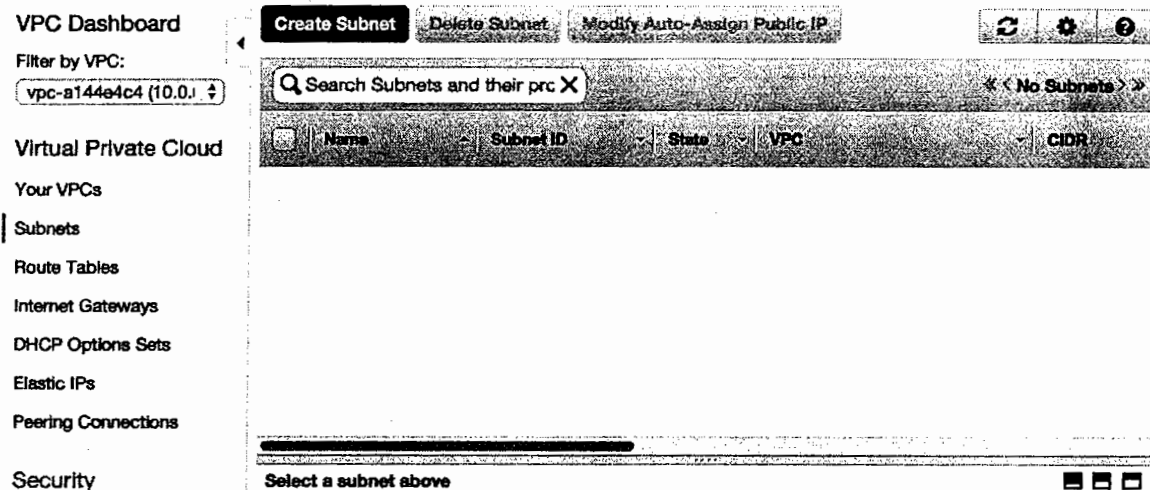
Select the VPC service from the Management Console dashboard:



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

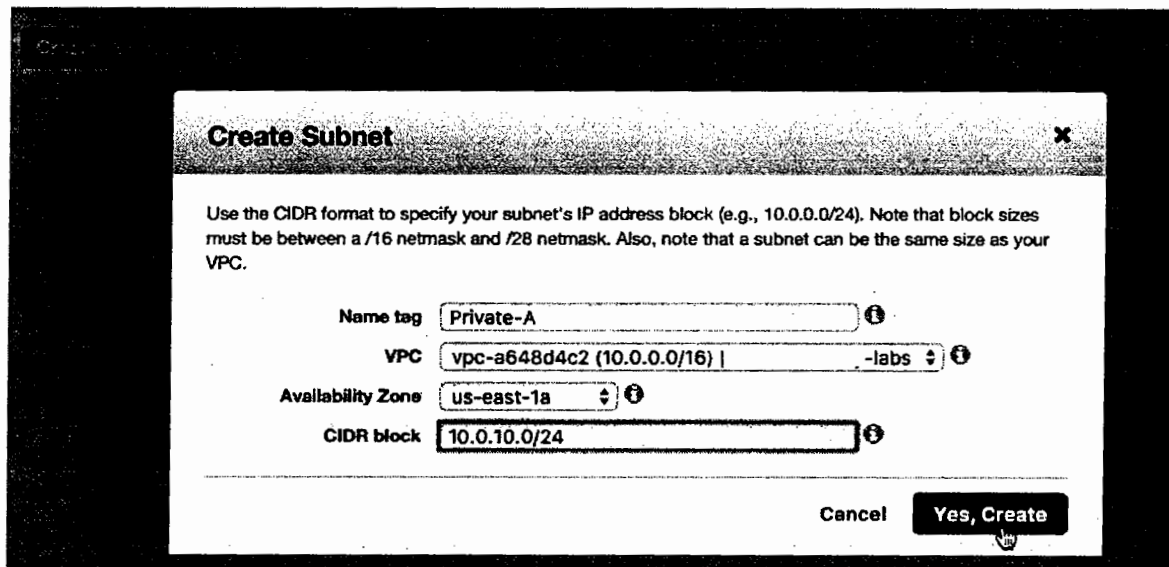
Your Subnets page lists all previously created subnets, you can use the **Filter by VPC** feature for listing only the services linked to a specific VPC.

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**.

Name	Subnet ID	State	VPC	CIDR
Public-A	subnet-6b947733	available	vpc-a648d4c2 (10.0.0.0/16) clo...	10.0.20.0/24
Private-A	subnet-40947718	available	vpc-a648d4c2 (10.0.0.0/16) clo...	10.0.10.0/24

subnet-40947718 (10.0.10.0/24) | Private-A

Summary Route Table Network ACL Flow Logs Tags

Edit

Route Table: rtb-5d663339

Destination	Target
10.0.0.0/16	local

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**.

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Peering Connections

Security

Network ACLs

Name	Subnet ID	State	VPC
Public-A	subnet-6b947733	available	vpc-a648d4c2 (10.0.0.0/16) -labs
Private-A	subnet-40947718	available	vpc-a648d4c2 (10.0.0.0/16) -labs

subnet-40947718 (10.0.10.0/24) | Private-A

Summary Route Table Network ACL Flow Logs Tags

Cancel Save

Current Route Table: rtb-4db3e629 | PrivateRouteTable

Change to: rtb-34b3e850 | PublicRouteTable

rtb-5d663339

Destination	Target
10.0.0.0/16	local

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.

Filter by VPC: None

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Peering Connections

Security

Network ACLs

Search Subnets and their pr X

Name	Subnet ID	State	VPC	CIDR
	subnet-3eb2f249	available	vpc-a2411dc7 (172.31.0.0/16)	172.31.32.0/20
Private-A	subnet-7f9d981a	available	vpc-4ea8982b (10.0.0.0/16) clo...	10.0.10.0/24
	subnet-cada4093	available	vpc-a2411dc7 (172.31.0.0/16)	172.31.0.0/20

subnet-7f9d981a (10.0.10.0/24) | Private-A

Summary Route Table Network ACL Flow Logs Tags

Edit

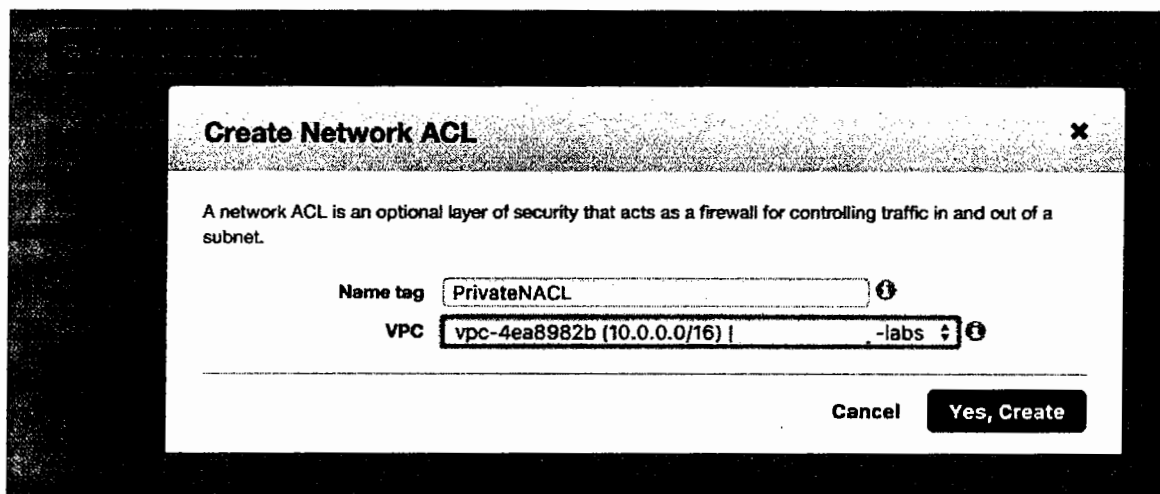
Network ACL: acl-aca895c9

Inbound:

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

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**.

The screenshot shows the AWS Management Console interface for configuring a Network ACL. On the left is a navigation pane with categories like Internet Gateways, DHCP Options Sets, Elastic IPs, Endpoints, Peering Connections, Security, Network ACLs, Security Groups, VPN Connections, Customer Gateways, and Virtual Private Gateways. The main panel displays a table of Network ACLs. Below the table, the 'Subnet Associations' tab is active, showing a table with columns: Associate, Subnet, CIDR, and Current Network ACL. The first row shows 'subnet-7f9d981a (10.0.10.0/24) | Labs-Subnet' with the 'Associate' checkbox checked. The second row shows 'subnet-ac9d9889 (10.0.20.0/24) | Labs-Subnet' with the 'Associate' checkbox unchecked. At the top of the details pane, there are buttons for 'Summary', 'Inbound Rules', 'Outbound Rules', 'Subnet Associations', and 'Tags'. Below these are 'Cancel' and 'Save' buttons. The 'Save' button is highlighted.

Name	Network ACL ID	Associated With	Default	VPC
labs-nacl	acl-c80f33ad	0 Subnets	No	vpc-4ea9982b (10.0.0.0/16) do...
	acl-ceb1ecab	3 Subnets	Yes	vpc-e2411dc7 (172.31.0.0/16)

acl-c80f33ad | labs-nacl

Summary Inbound Rules Outbound Rules **Subnet Associations** Tags

Cancel Save

Associate	Subnet	CIDR	Current Network ACL
<input checked="" type="checkbox"/>	subnet-7f9d981a (10.0.10.0/24) Labs-Subnet	10.0.10.0/24	acl-aca895c9
<input type="checkbox"/>	subnet-ac9d9889 (10.0.20.0/24) Labs-Subnet	10.0.20.0/24	acl-aca895c9

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.

The screenshot shows the AWS IAM console interface for Network ACLs. At the top, there is a search bar and a breadcrumb trail indicating the current location. Below this is a table listing two Network ACLs: 'Private-NACL' and 'acl-aca895c9'. The 'Private-NACL' is selected, and its details are shown in the main pane. The details pane has tabs for 'Summary', 'Inbound Rules', 'Outbound Rules', 'Subnet Associations', and 'Tags'. The 'Inbound Rules' tab is active, showing a single rule with the following details: Rule # 100, Type ALL Traffic, Protocol ALL, Port Range ALL, Source 10.0.0.0/16, and Action ALLOW. Below the rule list, there is a button to 'Add another rule'.

Name	Network ACL ID	Associated With	Default	VPC
Private-NACL	acl-48a5992d	1 Subnet	No	vpc-4ea8982b (10.0.0.0/16) do...
	acl-aca895c9	1 Subnet	Yes	vpc-4ea8982b (10.0.0.0/16) do...

acl-48a5992d | Private-NACL

Summary Inbound Rules Outbound Rules Subnet Associations Tags

Allows inbound traffic. Because network ACLs are stateless, you must create inbound and outbound rules.

Cancel Save

Rule #	Type	Protocol	Port Range	Source	Allow / Deny	Remove
100	ALL Traffic	ALL	ALL	10.0.0.0/16	ALLOW	

Add another rule

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.

Search Network ACLs and t1 X << 1 to 2 of 2 Network ACLs >>

Name	Network ACL ID	Associated With	Default	VPC
<input checked="" type="checkbox"/> Private-NACL	acl-48a5992d	1 Subnet	No	vpc-4aa8982b (10.0.0.0/16) do...
<input type="checkbox"/>	acl-aca895c9	1 Subnet	Yes	vpc-4aa8982b (10.0.0.0/16) do...

acl-48a5992d | Private-NACL

Summary Inbound Rules Outbound Rules Subnet Associations Tags

Allows outbound traffic. Because network ACLs are stateless, you must create inbound and outbound rules.

Cancel Save

Rule #	Type	Protocol	Port Range	Destination	Allow/Deny	Remove
100	ALL Traffic	ALL	ALL	10.0.0.0/16	ALLOW	

Add another rule

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:

Compute



EC2

Virtual Servers in the Cloud

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

EC2 Dashboard

Events

Tags

Reports

Limits

INSTANCES

Instances

Spot Requests

Reserved Instances

IMAGES

AMIs

Bundle Tasks

Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

0 Running Instances

0 Volumes

0 Key Pairs

0 Placement Groups

1 Elastic IPs

0 Snapshots

0 Load Balancers

2 Security Groups

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the US West (Oregon) region

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.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Quick Start

My AMIs

AWS Marketplace

Community AMIs

Operating system

Architecture

Root device type

Search community AMIs

ubuntu/images/hvm-ssd/ubuntu-trusty-14.04-amd64-server-20150123 - ami-29ebb519

Root device type: ebs Virtualization type: hvm

Select

64-bit

ubuntu/images/ebs-ssd/ubuntu-trusty-14.04-amd64-server-20150123 - ami-23ebb513

Root device type: ebs Virtualization type: paravirtual

Select

64-bit

ubuntu-core-alpha-01.1 - ami-4181d171

ubuntu-core-alpha-01.1

Root device type: ebs Virtualization type: hvm

Select

64-bit

ubuntu-core-devel-142 - ami-55712e65

ubuntu-core-devel-142

Root device type: ebs Virtualization type: hvm

Select

64-bit

ubuntu-core-alpha-01 - ami-659aca55

ubuntu-core-alpha-01

Root device type: ebs Virtualization type: hvm

Select

64-bit

ubuntu/images-sandbox/ebs/ubuntu-quantal-sandbox-amd64-server-20120815 - ami-003fb030

Root device type: ebs Virtualization type: paravirtual

Select

64-bit

ubuntu/images/ebs/ubuntu-quantal-12.10-amd64-server-20140202 - ami-006a0930

Root device type: ebs Virtualization type: paravirtual

Select

64-bit

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

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
General purpose	t2.micro Free tier eligible	1	1	EBS only		Low to Moderate
General purpose	t2.small	1	2	EBS only		Low to Moderate

Cancel Previous Review and Launch 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**.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of Instances ①

Purchasing option ① ☐ Request Spot Instances

Network ①

Subnet ①
251 IP Addresses available

Auto-assign Public IP ①

IAM role ①

Shutdown behavior ①

Enable termination protection ① ☐ Protect against accidental termination

Monitoring ① ☐ Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy ①

▼ **Network interfaces**

Device	Network Interface	Subnet	Primary IP	Secondary IP addresses
eth0	<input type="button" value="New network interface"/>	<input type="text" value="subnet-5931ee2e F"/>	<input type="text" value="Auto-assign"/>	<input type="button" value="Add IP"/>

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

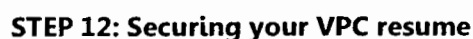
On the Review Instance Launch page, click **Launch**.

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.



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.

