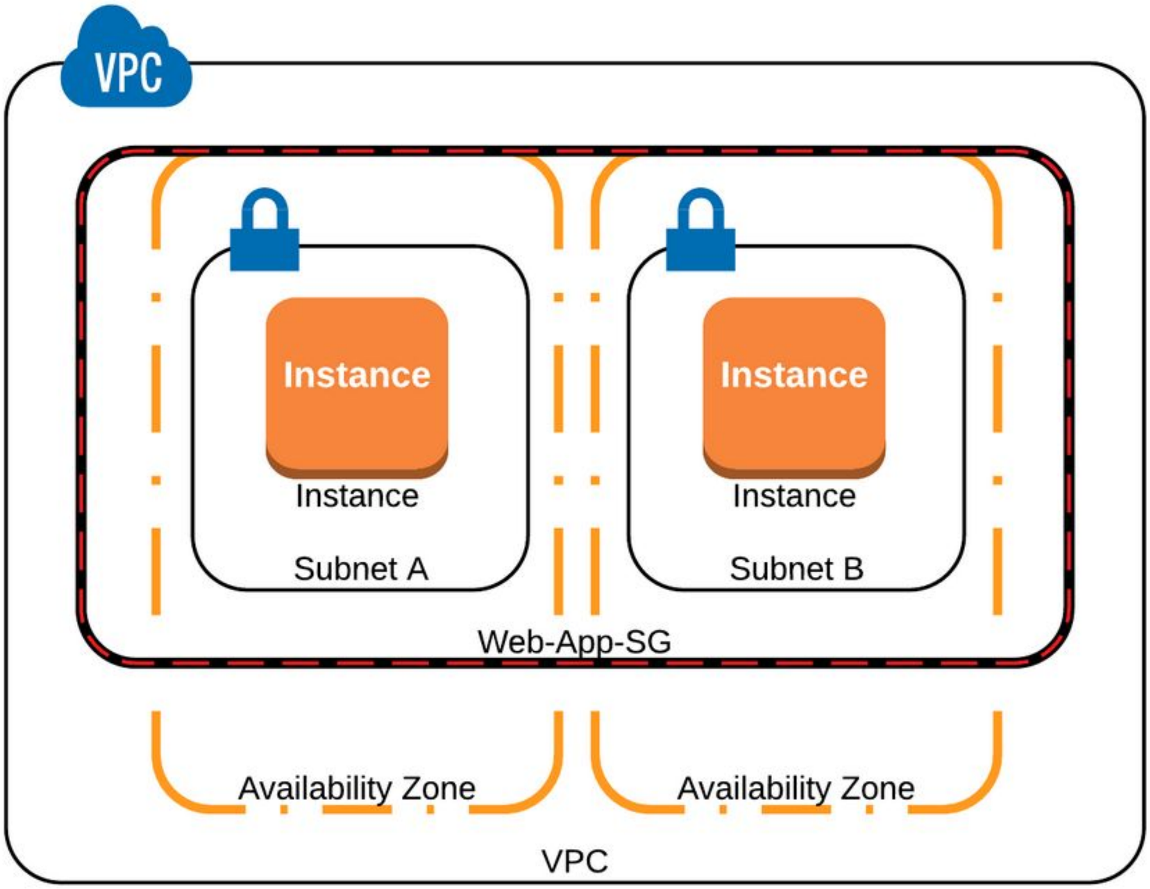
Classic Load Balancer planning

The Elastic Load Balancing Service automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables you to achieve fault tolerance in your applications, seamlessly providing the required amount of load balancing capacity needed to route application traffic.

Elastic Load Balancing offers two types of load balancers both of which feature high availability, automatic scaling, and robust security. These include the Classic Load Balancer that routes traffic based on either application or network level information, and the Application Load Balancer that routes traffic based on advanced application level information that includes the content of the request. The Classic Load Balancer is ideal for simple load balancing of traffic across multiple EC2 instances, while the Application Load Balancer is ideal for applications needing advanced routing capabilities, microservices, and container-based architectures. Application Load Balancer offers the ability to route traffic to multiple services or load balance across multiple ports on the same EC2 instance. In this lab, we are going to focus on the **Classic Load Balancer**.

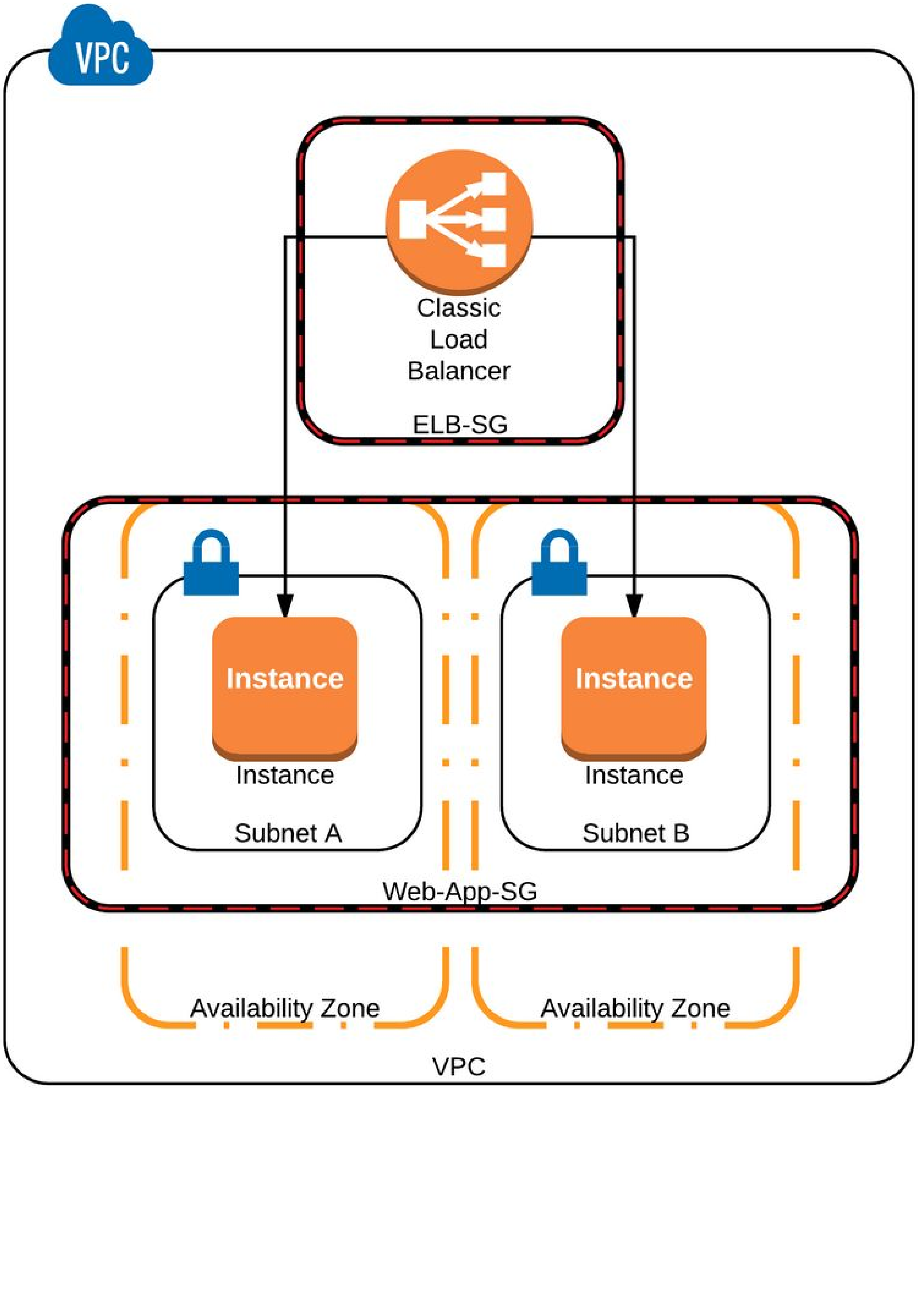
Planning

Before going ahead and creating a Load Balancer (LB), let's take a look at an overview of our current infrastructure. When you connected to the AWS account provided in the former step, you had a few things that were already deployed. This is the current infrastructure that was already deployed for you:



You already have a VPC with some subnets and 2 EC2 instances running inside the VPC in different Availability Zones. Both instances are inside the same Security Group called `Web-App-SG`, which is allowing HTTP access from port 80 to anywhere (0.0.0.0/0). Each EC2 instance is running the same web application. We want to **configure an LB to create a central point of access to our application**, and we also want to configure our architecture in a way that **users can only access the application through the ELB**.

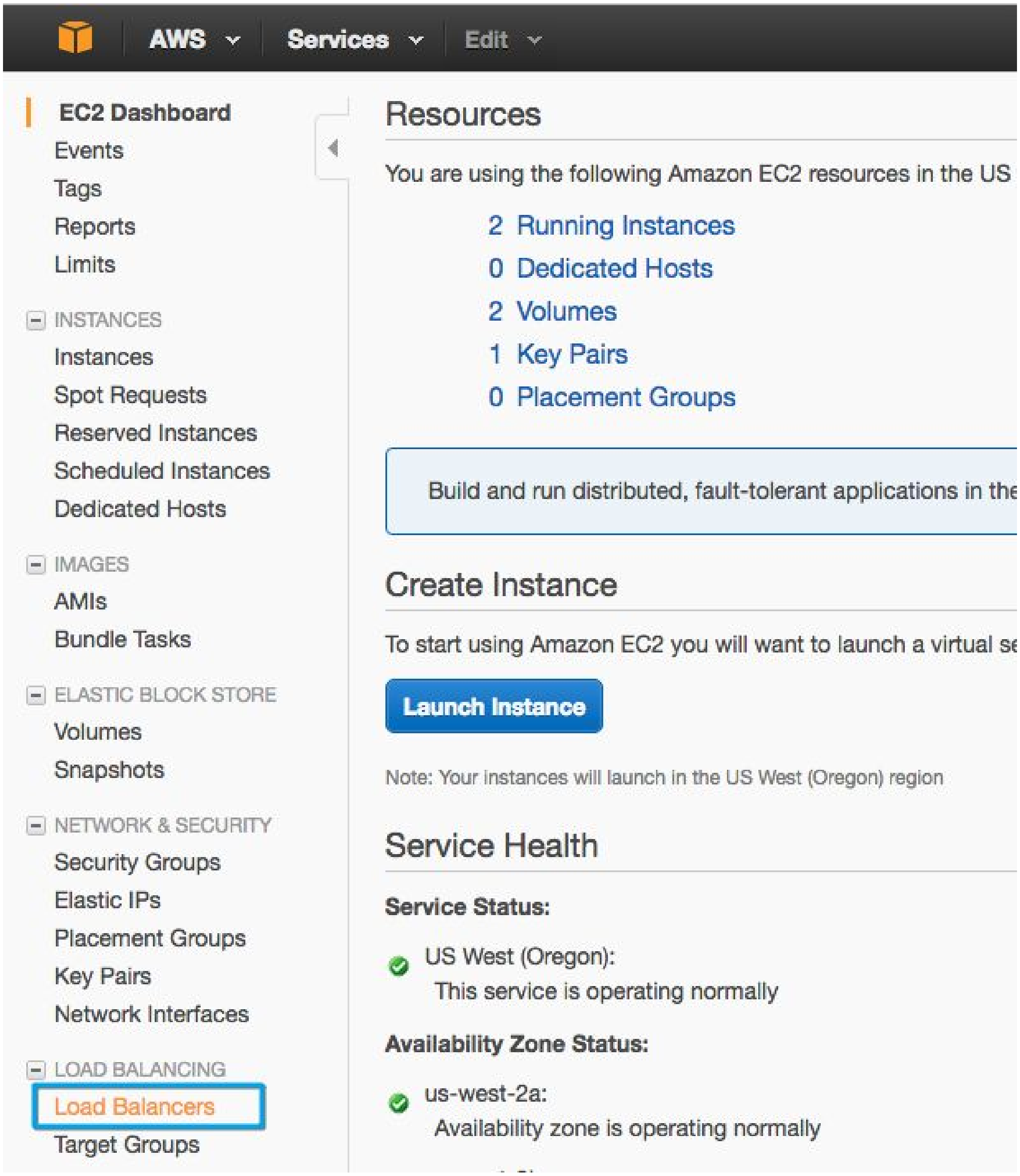
In the end, we should have a solution similar to this one:


VPC
Classic
Load
Balancer
ELB-SG
Instance
Instance
Subnet A
Instance
Instance
Subnet B
Web-App-SG
Availability Zone
Availability Zone
VPC

To do that we will have to create and configure a Classic Load Balancer, and properly configure the needed Security Groups to make sure that our application will work as expected.

Create a Classic Load Balancer and register EC2 instances

Click on Load Balancers:



You are using the following Amazon EC2 resources in the US

2 Running Instances
0 Dedicated Hosts
2 Volumes
1 Key Pairs
0 Placement Groups

Build and run distributed, fault-tolerant applications in the

To start using Amazon EC2 you will want to launch a virtual se

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

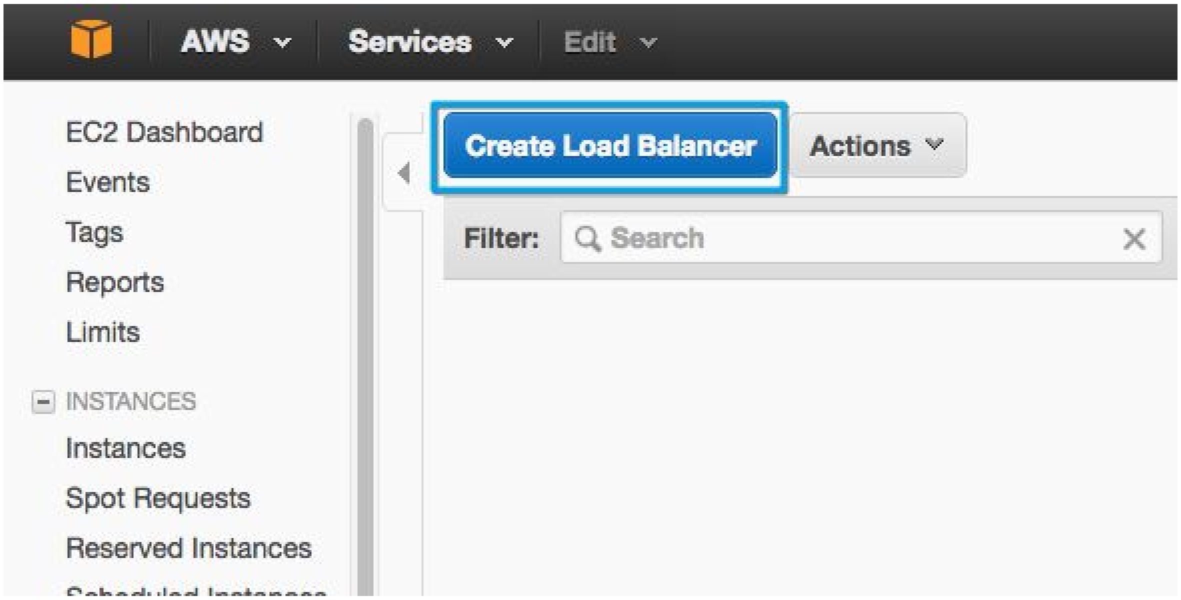
Service Status:

US West (Oregon):
This service is operating normally

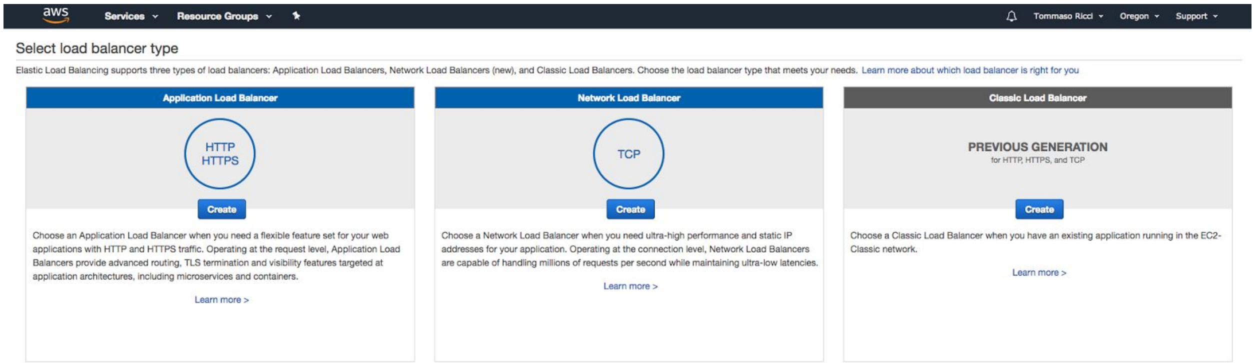
Availability Zone Status:

us-west-2a:
Availability zone is operating normally

Click on **Create Load Balancer**:



As stated before, you can choose from two flavors of ELBs: Application Load Balancer or Classic Load Balancer. In this lab, we will use the Classic Load Balancer, so simply choose the proper one in this step and click on **Create**.



Select load balancer type

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers (new), and Classic Load Balancers. Choose the load balancer type that meets your needs. Learn more about which load balancer is right for you

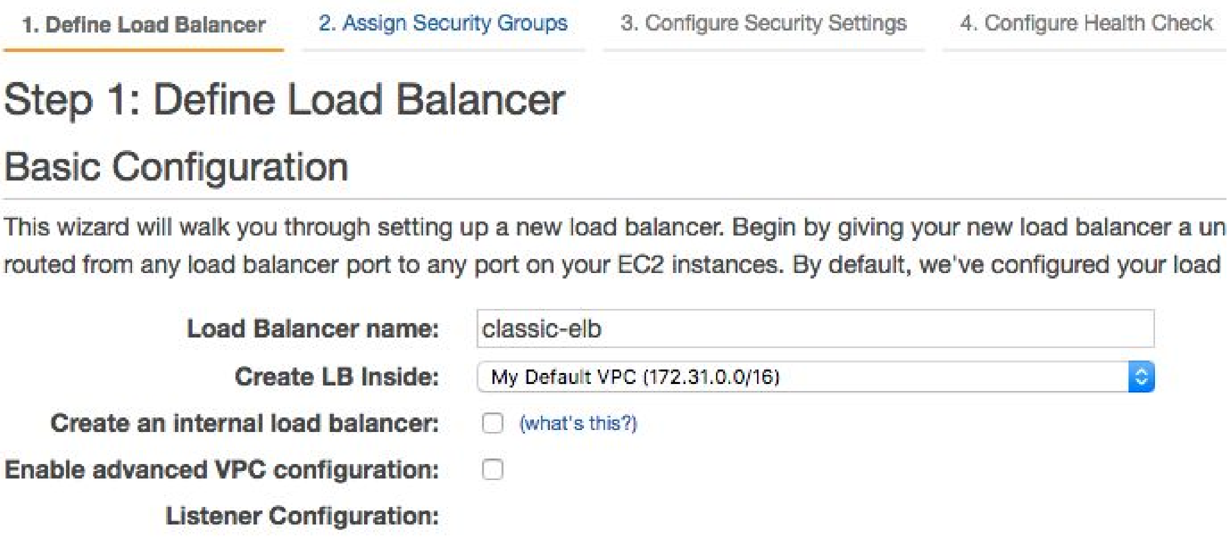
Choose an Application Load Balancer when you need a flexible feature set for your web applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing, TLS termination and visibility features targeted at application architectures, including microservices and containers.

Choose a Network Load Balancer when you need ultra-high performance and static IP addresses for your application. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second while maintaining ultra-low latencies.

Choose a Classic Load Balancer when you have an existing application running in the EC2-Classic network.

Now to start configuring the specifics of the Load Balancer (LB), you will need to follow a 7-step wizard.

In Step 1, you need to specify a name for the LB; this name can be anything that will make sense for you in the future. But be aware of the limitations (Only a-z, A-Z, 0-9 and hyphens are allowed). You need to select the VPC where the LB will live, this VPC should be the same VPC where the EC2 instances are running, you will probably have only the Default VPC in your account, choose this one. You now should have something like this:


1. Define Load Balancer
2. Assign Security Groups
3. Configure Security Settings
4. Configure Health Check

Step 1: Define Load Balancer

Basic Configuration

This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a un
routed from any load balancer port to any port on your EC2 instances. By default, we've configured your load

Load Balancer name: classic-elb

Create LB Inside: My Default VPC (172.31.0.0/16)

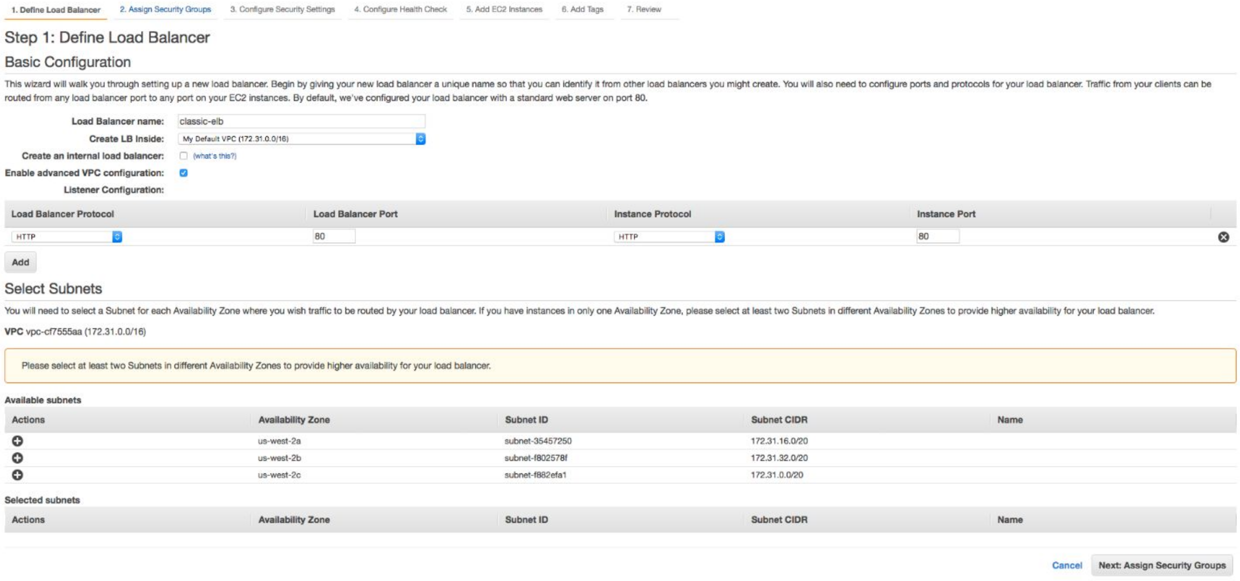
Create an internal load balancer: □ (what's this?)

Enable advanced VPC configuration: □

Listener Configuration:

Some info about the next config points. We will create an LB to receive traffic from the internet and forward to our instances, therefore we need a publicly accessible LB. If you select **Create an internal load balancer** you will be creating a load balancer that won't be publicly accessible - in this case, the LB will only be accessible inside the VPC, which is not the goal, so you should leave this box unchecked.

There are instances running in different availability zones, and we need to configure the LB to work in all subnets where we will be launching web instances. To configure this behavior we need to **Enable advanced VPC configuration** in order to select the subnets we want. After that you will be able to see more options, it will look like this:


1. Define Load Balancer
2. Assign Security Groups
3. Configure Security Settings
4. Configure Health Check
5. Add EC2 Instances
6. Add Tags
7. Review

Step 1: Define Load Balancer

Basic Configuration

This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a unique name so that you can identify it from other load balancers you might create. You will also need to configure ports and protocols for your load balancer. Traffic from your clients can be routed from any load balancer port to any port on your EC2 instances. By default, we've configured your load balancer with a standard web server on port 80.

Load Balancer name: classic-elb

Create LB Inside: My Default VPC (172.31.0.0/16)

Create an internal load balancer: □ (what's this?)

Enable advanced VPC configuration: ☑

Listener Configuration:

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port	
HTTP	80	HTTP	80	⊗

Select Subnets

You will need to select a Subnet for each Availability Zone where you wish traffic to be routed by your load balancer. If you have instances in only one Availability Zone, please select at least two Subnets in different Availability Zones to provide higher availability for your load balancer.

VPC vpc-cf7555aa (172.31.0.0/16)

Please select at least two Subnets in different Availability Zones to provide higher availability for your load balancer.

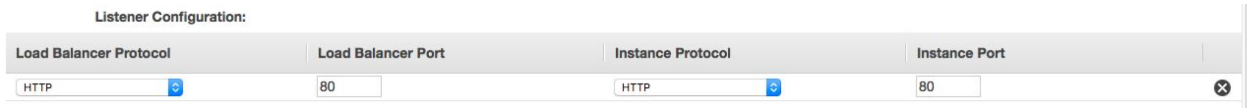
Available subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR	Name
⊕	us-west-2a	subnet-35457250	172.31.16.0/20	
⊕	us-west-2b	subnet-f802578f	172.31.32.0/20	
⊕	us-west-2c	subnet-f882efa1	172.31.0.0/20	

Selected subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR	Name

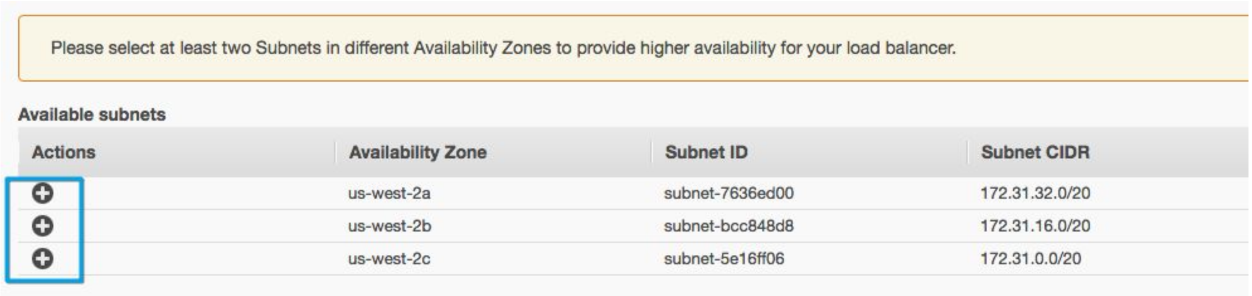
A LB listens to a specific port for requests coming from outside, in this case, the internet, and forwards the request to the instances running behind it on a specific port. In this lab, we will be using the port 80 for HTTP requests for both ELB and the EC2 instances, therefore, the default choice will work for us:



Listener Configuration:

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port	
HTTP	80	HTTP	80	⊗

All though there are only 2 instances running in the account, in different Availability Zones, we will want to select ALL the available subnets in this VPC, just in case we want to launch another instance later on in a different Availability Zone. Simply click on the plus button for all the subnets available in this VPC.

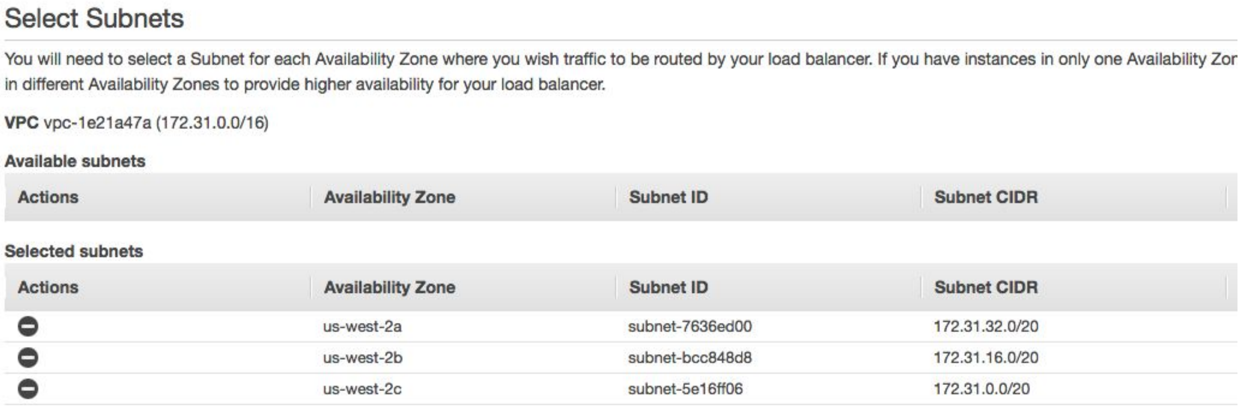


Please select at least two Subnets in different Availability Zones to provide higher availability for your load balancer.

Available subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR
⊕	us-west-2a	subnet-7636ed00	172.31.32.0/20
⊕	us-west-2b	subnet-bcc848d8	172.31.16.0/20
⊕	us-west-2c	subnet-5e16ff06	172.31.0.0/20

And in the end, you should see something like this:



Select Subnets

You will need to select a Subnet for each Availability Zone where you wish traffic to be routed by your load balancer. If you have instances in only one Availability Zor in different Availability Zones to provide higher availability for your load balancer.

VPC vpc-1e21a47a (172.31.0.0/16)

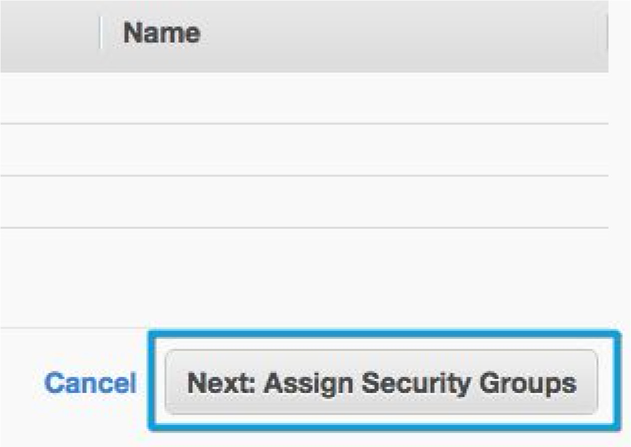
Available subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR

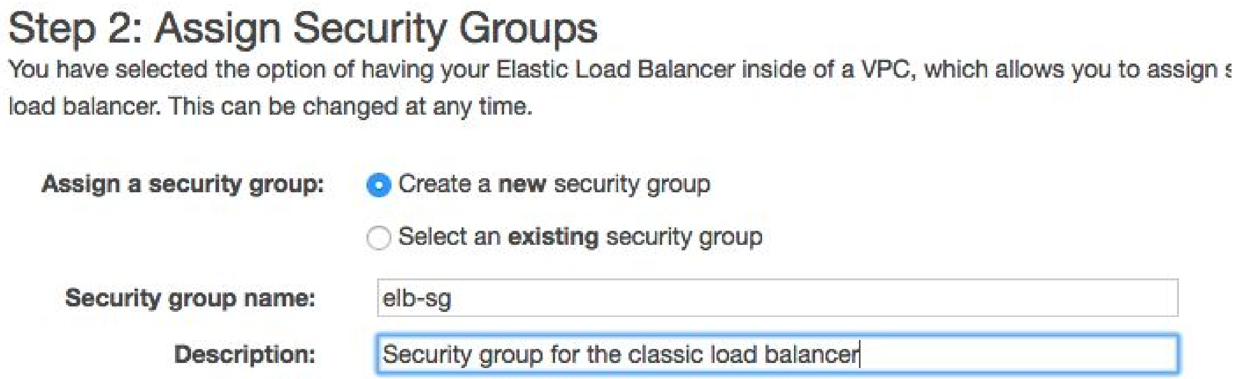
Selected subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR
⊖	us-west-2a	subnet-7636ed00	172.31.32.0/20
⊖	us-west-2b	subnet-bcc848d8	172.31.16.0/20
⊖	us-west-2c	subnet-5e16ff06	172.31.0.0/20

The first step of creating the LB is complete. You can click on **Next: Assign Security Groups**



On step 2, we need to configure a Security Group (SG) for the LB. This SG will be used to manage the security for the LB itself, therefore, since we only defined the port 80 (HTTP) in the listener section of the last step, we will want to create a new SG for the LB that will accept connections coming from anywhere to port 80 of the LB. To do that select **Create a new security group** and provide a name and a quick description for this SG.



Step 2: Assign Security Groups

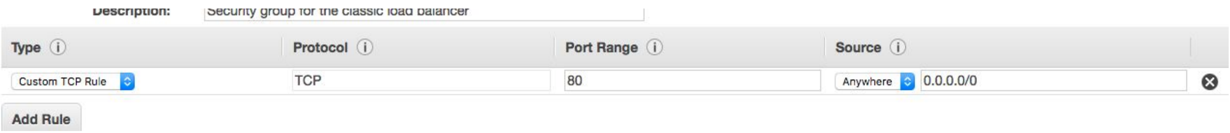
You have selected the option of having your Elastic Load Balancer inside of a VPC, which allows you to assign s
load balancer. This can be changed at any time.

Assign a security group: ● Create a **new** security group
○ Select an **existing** security group

Security group name: elb-sg

Description: Security group for the classic load balancer

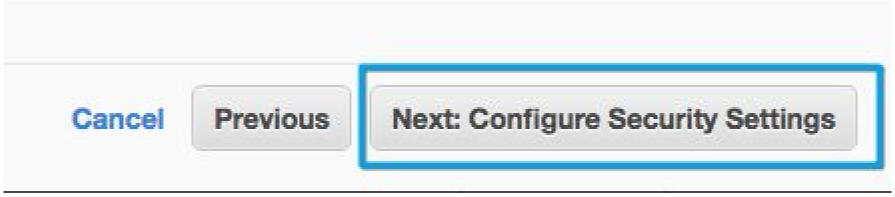
And allow connections coming from Anywhere (0.0.0.0/0) to the port 80 and nothing more



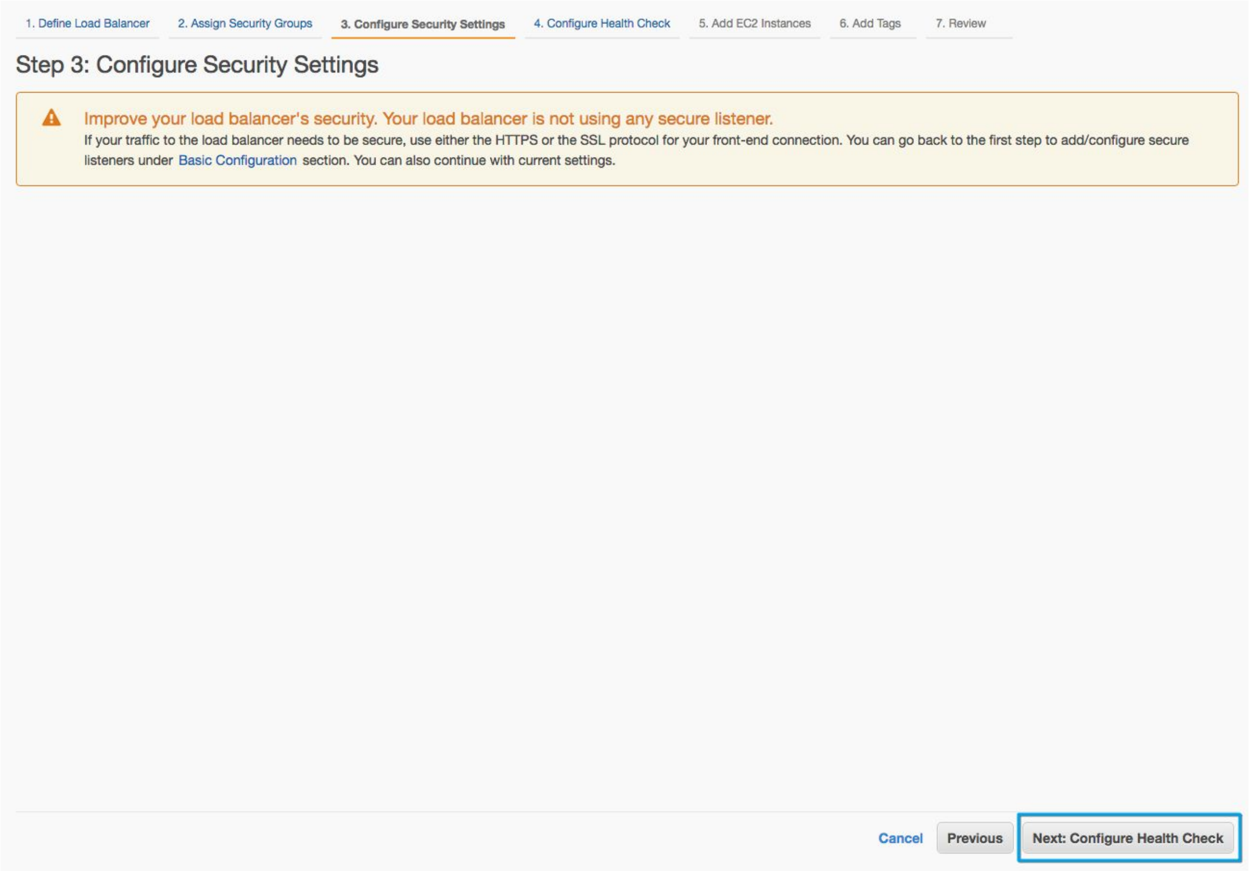
Description: Security group for the classic load balancer

Type	Protocol	Port Range	Source
Custom TCP Rule	TCP	80	Anywhere 0.0.0.0/0

After that, you can click on **Next: Configure Security Settings**



Step 3 consists of configuring the LB to use HTTPS or SSL for security purposes. Although it is highly recommended that you reinforce security in your applications, configuring it is beyond the scope of this lab, therefore, you can simply click on **Next: Configure Health Check**

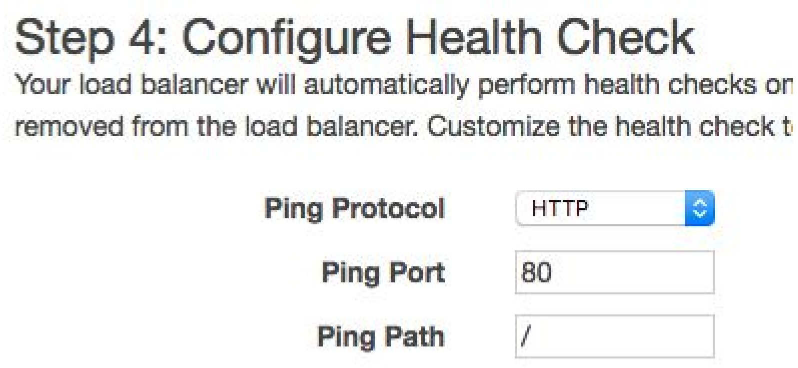


1. Define Load Balancer
2. Assign Security Groups
3. Configure Security Settings
4. Configure Health Check
5. Add EC2 Instances
6. Add Tags
7. Review

Step 3: Configure Security Settings

Improve your load balancer's security. Your load balancer is not using any secure listener.
If your traffic to the load balancer needs to be secure, use either the HTTPS or the SSL protocol for your front-end connection. You can go back to the first step to add/configure secure listeners under Basic Configuration section. You can also continue with current settings.

In step 4, you need to configure a health check. This is how the LB will evaluate the health of an EC2 instance and decide whether to send requests or avoid a particular instance. The first thing to configure in here is the protocol, port, and path that will be used for the health check. The instances running in the account are serving an application in **port 80**, using the **HTTP** protocol, and using the **root path (/). Y**ou should configure this in the first part of this step



Step 4: Configure Health Check

Your load balancer will automatically perform health checks on
removed from the load balancer. Customize the health check t

Ping Protocol HTTP

Ping Port 80

Ping Path /

There are some **Advanced details** in this step you can configure, but for the purposes of this lab we will stick with the default settings. However, for your reference, this is what they mean:

Response Timeout: The amount of time to wait when receiving a response from the health check, in seconds. Valid values: 2 to 60. Default: 5

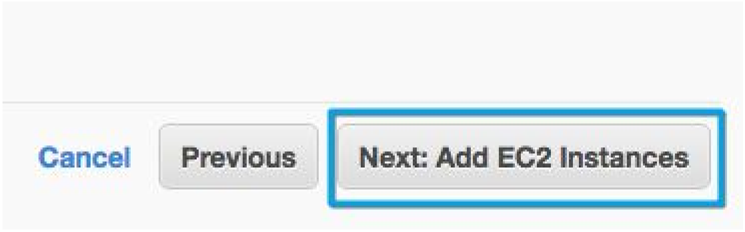
HealthCheck Interval: The amount of time between health checks of an individual instance, in seconds. Valid values: 5 to 300. Default: 30

Unhealthy Threshold: The number of consecutive failed health checks that must occur before declaring an EC2 instance unhealthy. Valid values: 2 to 10. Default: 2

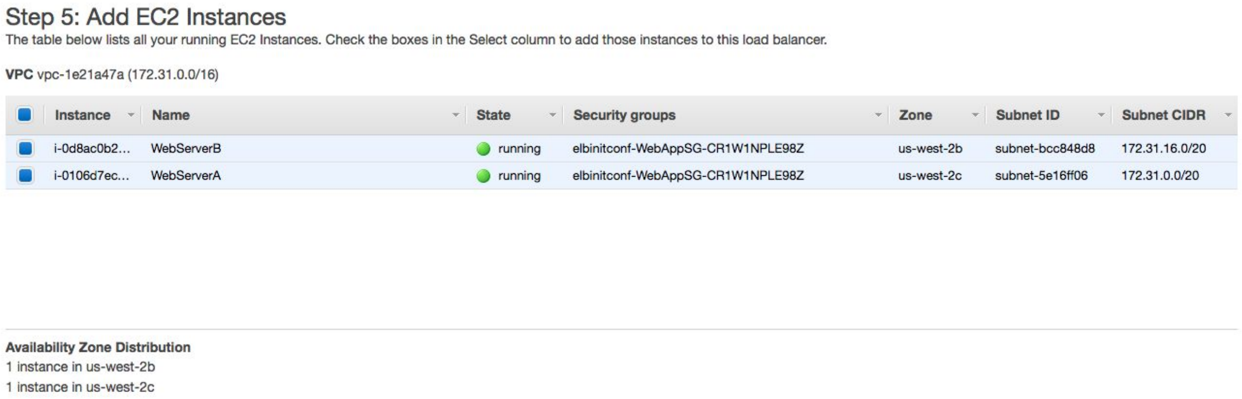
Healthy Threshold: The number of consecutive successful health checks that must occur before declaring an EC2 instance healthy. Valid values: 2 to 10. Default: 10

http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-healthchecks.html

Click on **Next: Add EC2 instances** to move on



In step 5, it is time to add EC2 instances to the LB. The first thing to do is to select the instances called **WebServerA** and **WebServerB**:



Step 5: Add EC2 Instances

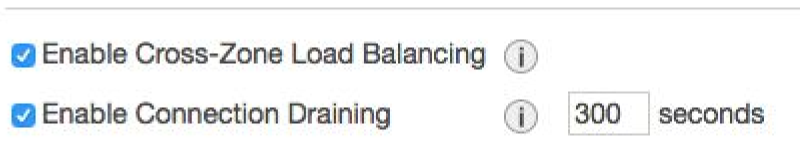
The table below lists all your running EC2 Instances. Check the boxes in the Select column to add those instances to this load balancer.

VPC vpc-1e21a47a (172.31.0.0/16)

■	Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
■	i-0d8ac0b2...	WebServerB	● running	elbinitconf-WebAppSG-CR1W1NPLE98Z	us-west-2b	subnet-bcc848d8	172.31.16.0/20
■	i-0106d7ec...	WebServerA	● running	elbinitconf-WebAppSG-CR1W1NPLE98Z	us-west-2c	subnet-5e16ff06	172.31.0.0/20

Availability Zone Distribution
1 instance in us-west-2b
1 instance in us-west-2c

There are 2 config points in here as well:



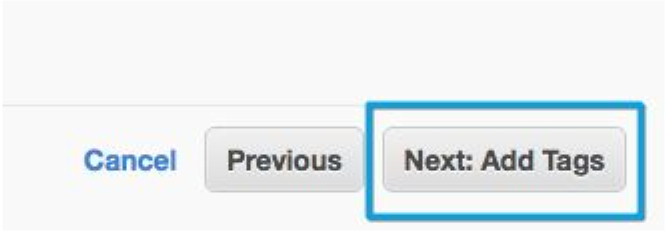
☑ Enable Cross-Zone Load Balancing

☑ Enable Connection Draining 300 seconds

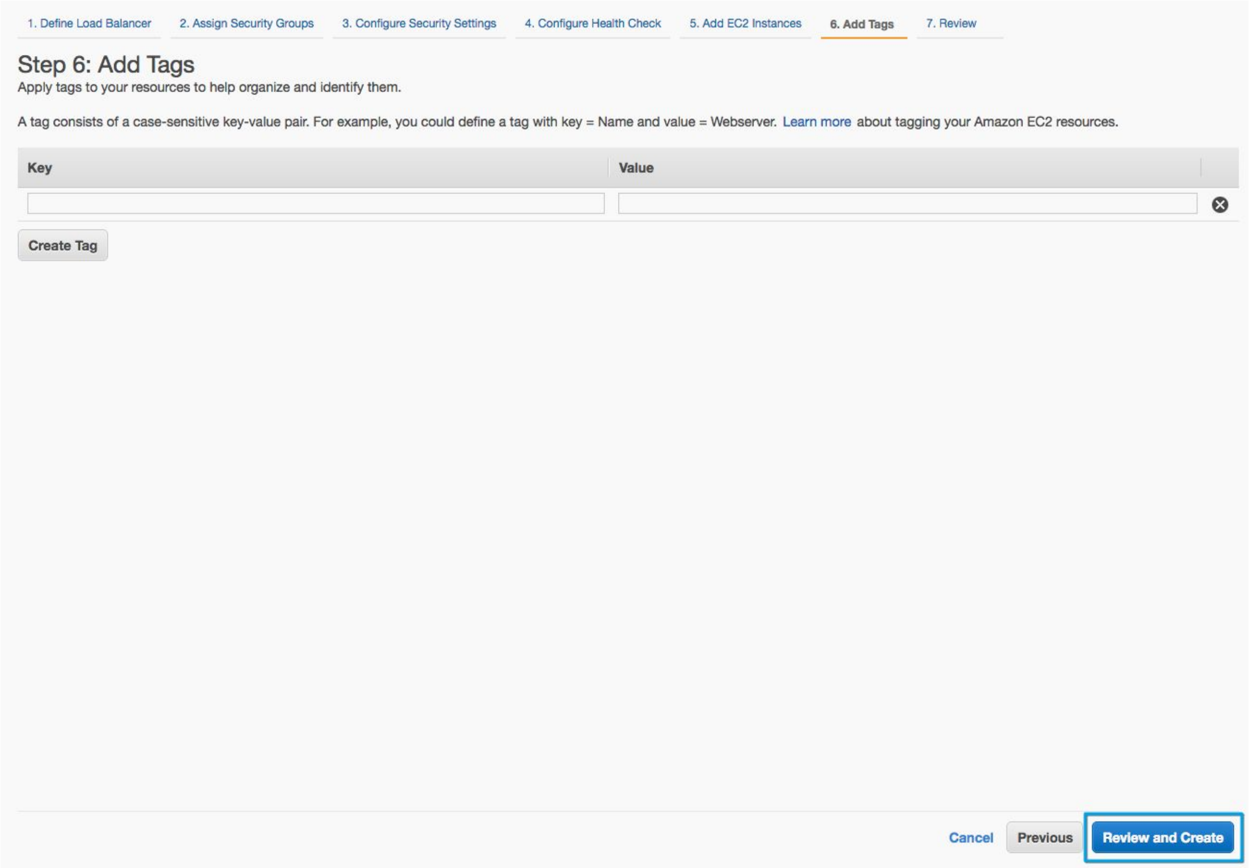
Cross-Zone Load Balancing is used to ensure that your LB distributes incoming requests evenly across all instances in its enabled Availability Zones. That means that the LB will ignore the default of round-robin and will also take into consideration the Availability Zone in which the instance is running. This reduces the need to maintain equivalent numbers of instances in each enabled Availability Zone, and improves your application's ability to handle the loss of one or more instances.

Connection Draining is used to ensure that a Classic Load Balancer stops sending requests to instances that are de-registering or unhealthy while keeping the existing connections open.

For the purposes of this lab, you can use the default settings and click on **Next: Add Tags** to move on.



In Step 6, you have the ability to add tags to the LB. You can either leave it in blank or add as many tags as you want and click on **Review and Create** to move on.



1. Define Load Balancer
2. Assign Security Groups
3. Configure Security Settings
4. Configure Health Check
5. Add EC2 Instances
6. Add Tags
7. Review

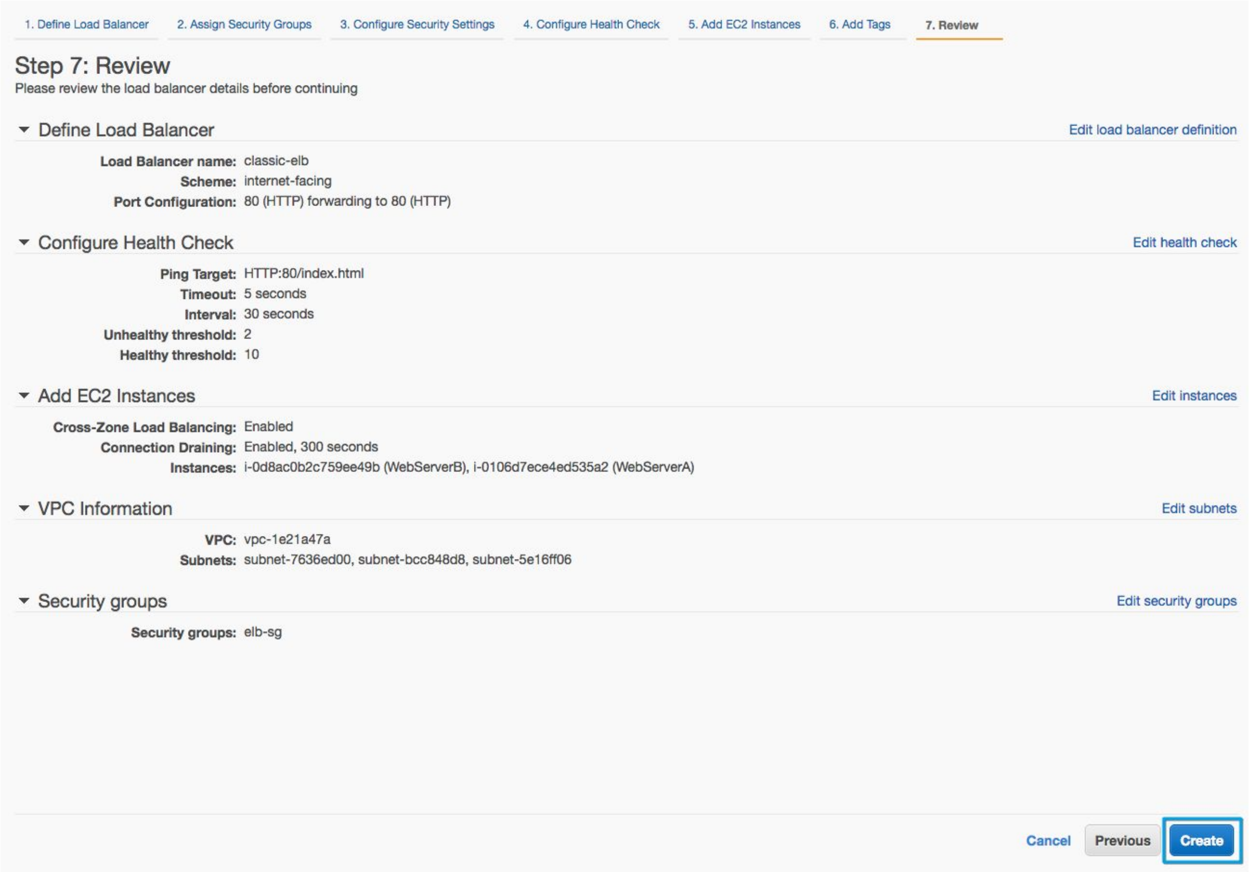
Step 6: Add Tags

Apply tags to your resources to help organize and identify them.

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. Learn more about tagging your Amazon EC2 resources.

Key	Value

In Step 7, it is time to review the LB's settings, double check the config points and click on **Create:**


1. Define Load Balancer
2. Assign Security Groups
3. Configure Security Settings
4. Configure Health Check
5. Add EC2 Instances
6. Add Tags
7. Review

Step 7: Review

Please review the load balancer details before continuing

Define Load Balancer — Edit load balancer definition

Load Balancer name: classic-elb
Scheme: internet-facing
Port Configuration: 80 (HTTP) forwarding to 80 (HTTP)

Configure Health Check — Edit health check

Ping Target: HTTP:80/index.html
Timeout: 5 seconds
Interval: 30 seconds
Unhealthy threshold: 2
Healthy threshold: 10

Add EC2 Instances — Edit instances

Cross-Zone Load Balancing: Enabled
Connection Draining: Enabled, 300 seconds
Instances: i-0d8ac0b2c759ee49b (WebServerB), i-0106d7ece4ed535a2 (WebServerA)

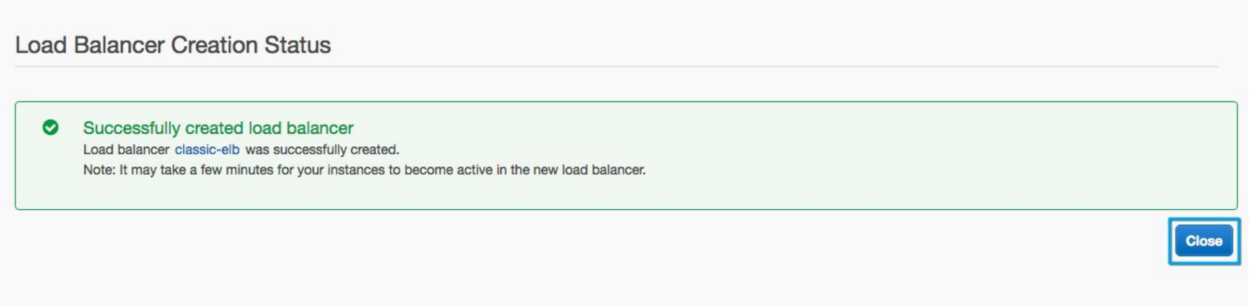
VPC Information — Edit subnets

VPC: vpc-1e21a47a
Subnets: subnet-7636ed00, subnet-bcc848d8, subnet-5e16ff06

Security groups — Edit security groups

Security groups: elb-sg

If you see a Success message it means that you LB is created, you can click on **Close** in the AWS console, and move on the the next lab step



Load Balancer Creation Status

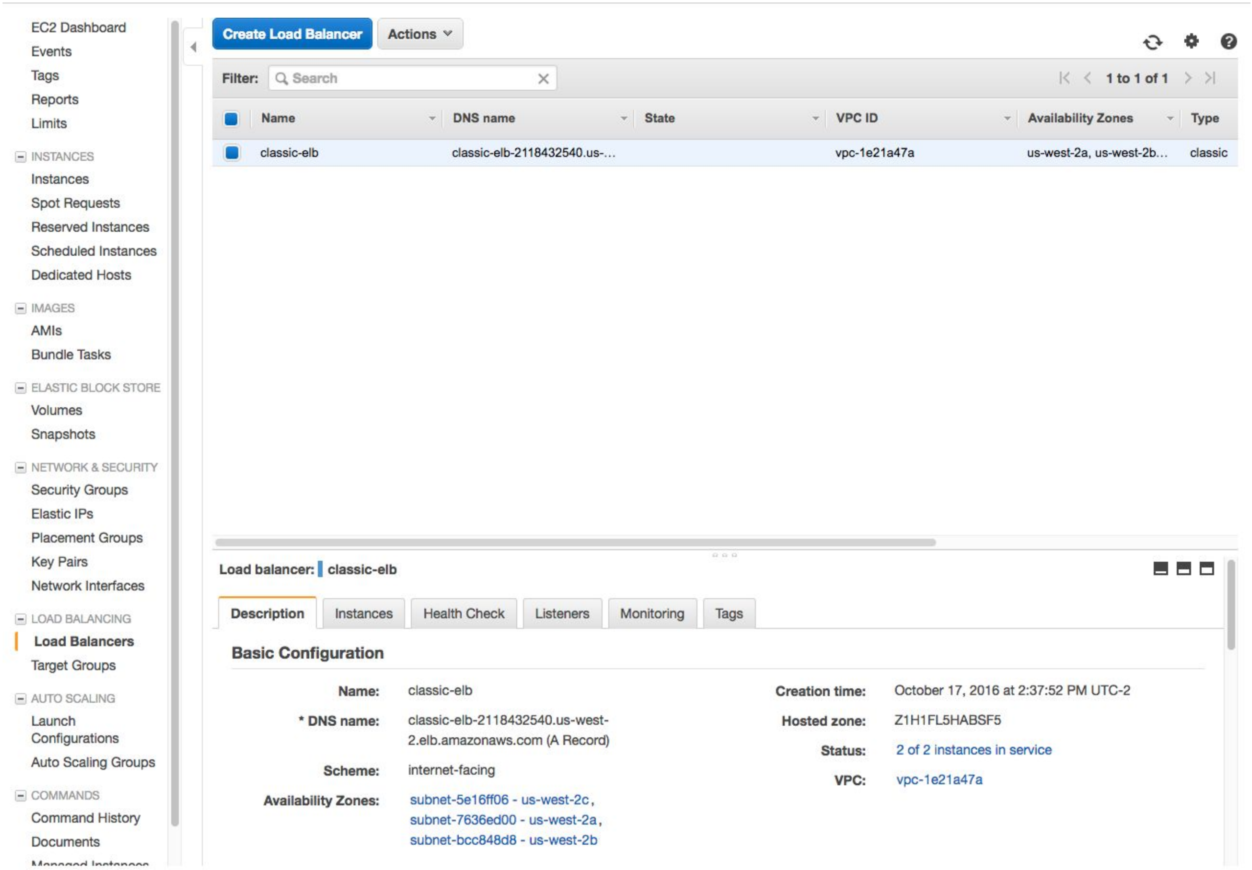
Successfully created load balancer

Load balancer classic-elb was successfully created.

Note: It may take a few minutes for your instances to become active in the new load balancer.

Step 4 Configuring security groups for ELB

Now that you completed the creation of you first Classic Load Balancer, you should be seeing this screen:



Name	DNS name	State	VPC ID	Availability Zones	Type
classic-elb	classic-elb-2118432540.us-…		vpc-1e21a47a	us-west-2a, us-west-2b…	classic

Load balancer: classic-elb

Basic Configuration

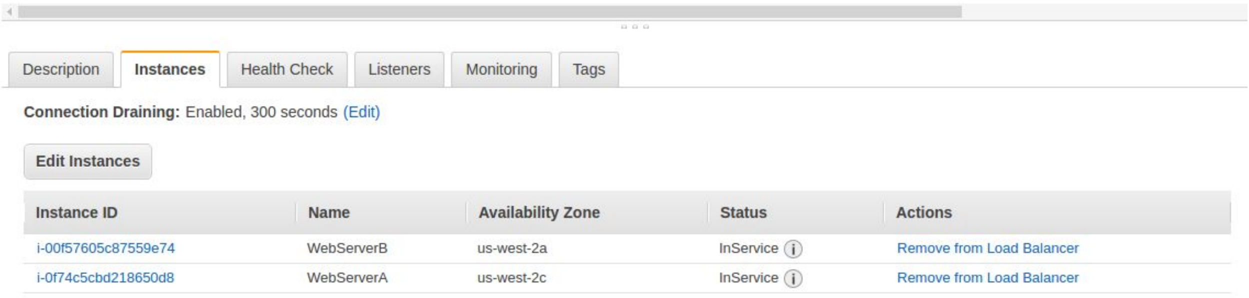
Name:	classic-elb	Creation time:	October 17, 2016 at 2:37:52 PM UTC-2
* DNS name:	classic-elb-2118432540.us-west-2.elb.amazonaws.com (A Record)	Hosted zone:	Z1H1FL5HABSF5
		Status:	2 of 2 instances in service
Scheme:	internet-facing	VPC:	vpc-1e21a47a
Availability Zones:	subnet-5e16ff06 - us-west-2c, subnet-7636ed00 - us-west-2a, subnet-bcc848d8 - us-west-2b		

In this step, we will configure the Security Group (SG) associated with the EC2 instances running in the account to accept only connections coming from the LB.

But before that, let's try to access the load balancer through its DNS address.

In order to do it, ensure that both instances are correctly attached to the Load Balancer and their status is *InService*, If not, just wait a few minutes.

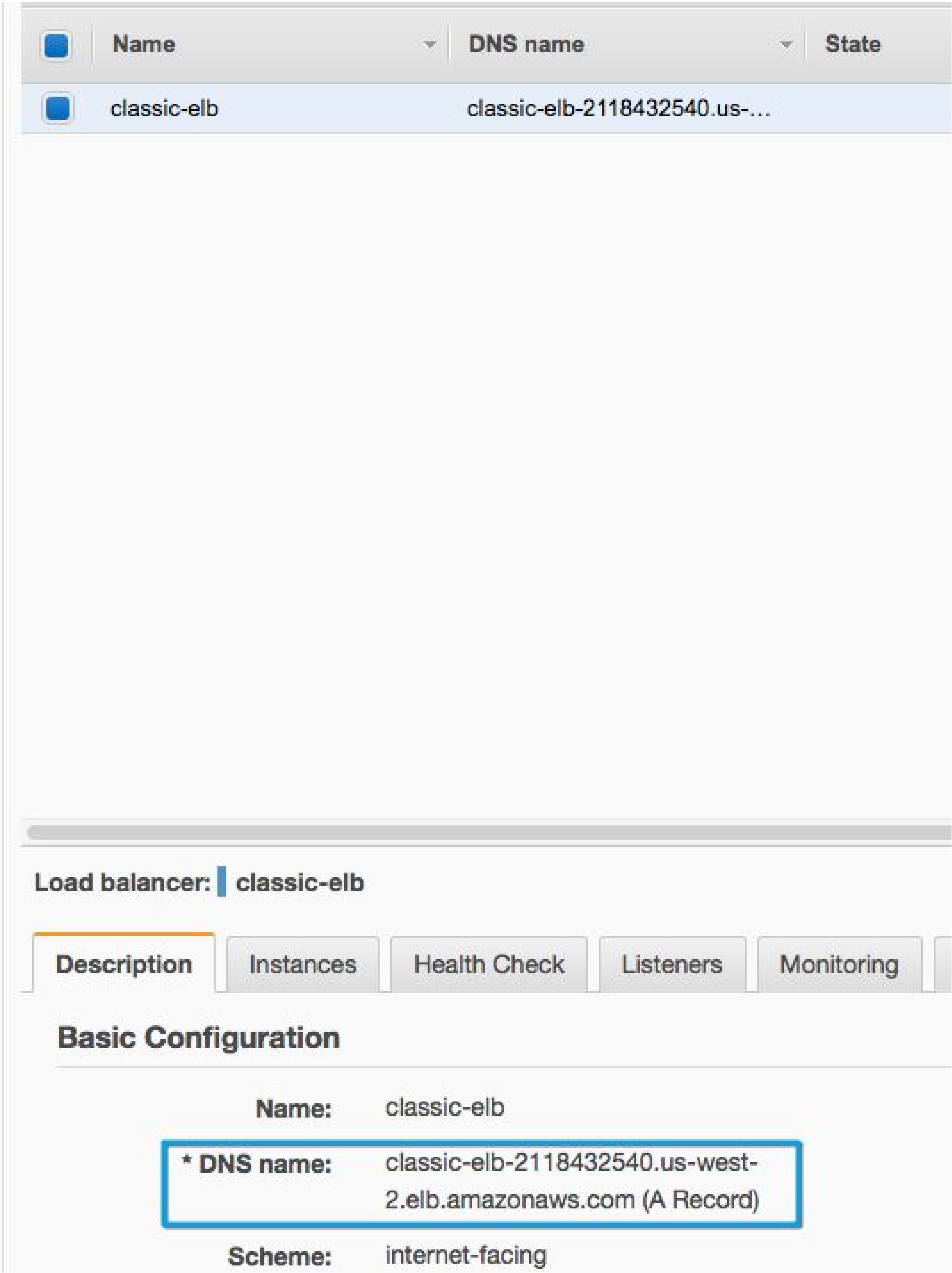
You can check the status of the instances selecting the Load Balancer and going in the *Instances* tab as you can see in the screenshot below.



Connection Draining: Enabled, 300 seconds (Edit)

Instance ID	Name	Availability Zone	Status	Actions
i-00f57605c87559e74	WebServerB	us-west-2a	InService	Remove from Load Balancer
i-0f74c5cbd218650d8	WebServerA	us-west-2c	InService	Remove from Load Balancer

After that both the instances are *InService* copy the Load Balancer **DNS Name.**



☐	Name	DNS name	State
■	classic-elb	classic-elb-2118432540.us-…	

Load balancer: classic-elb

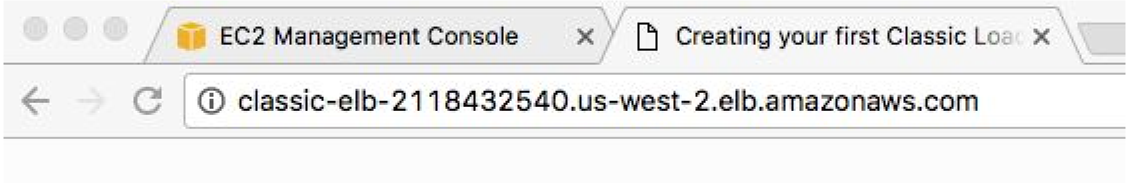
Basic Configuration

Name: classic-elb

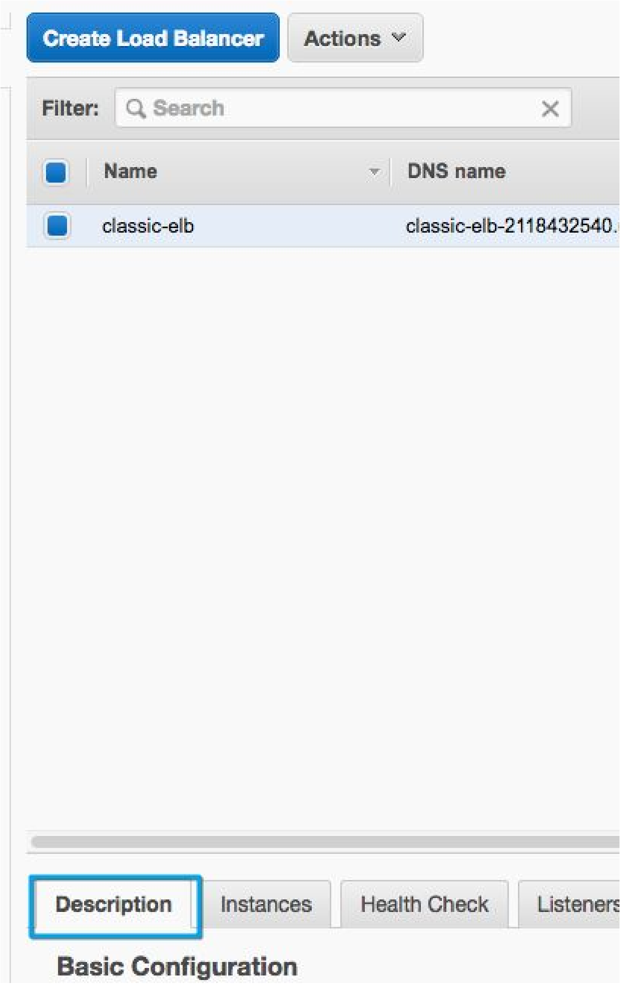
* DNS name: classic-elb-2118432540.us-west-2.elb.amazonaws.com (A Record)

Scheme: internet-facing

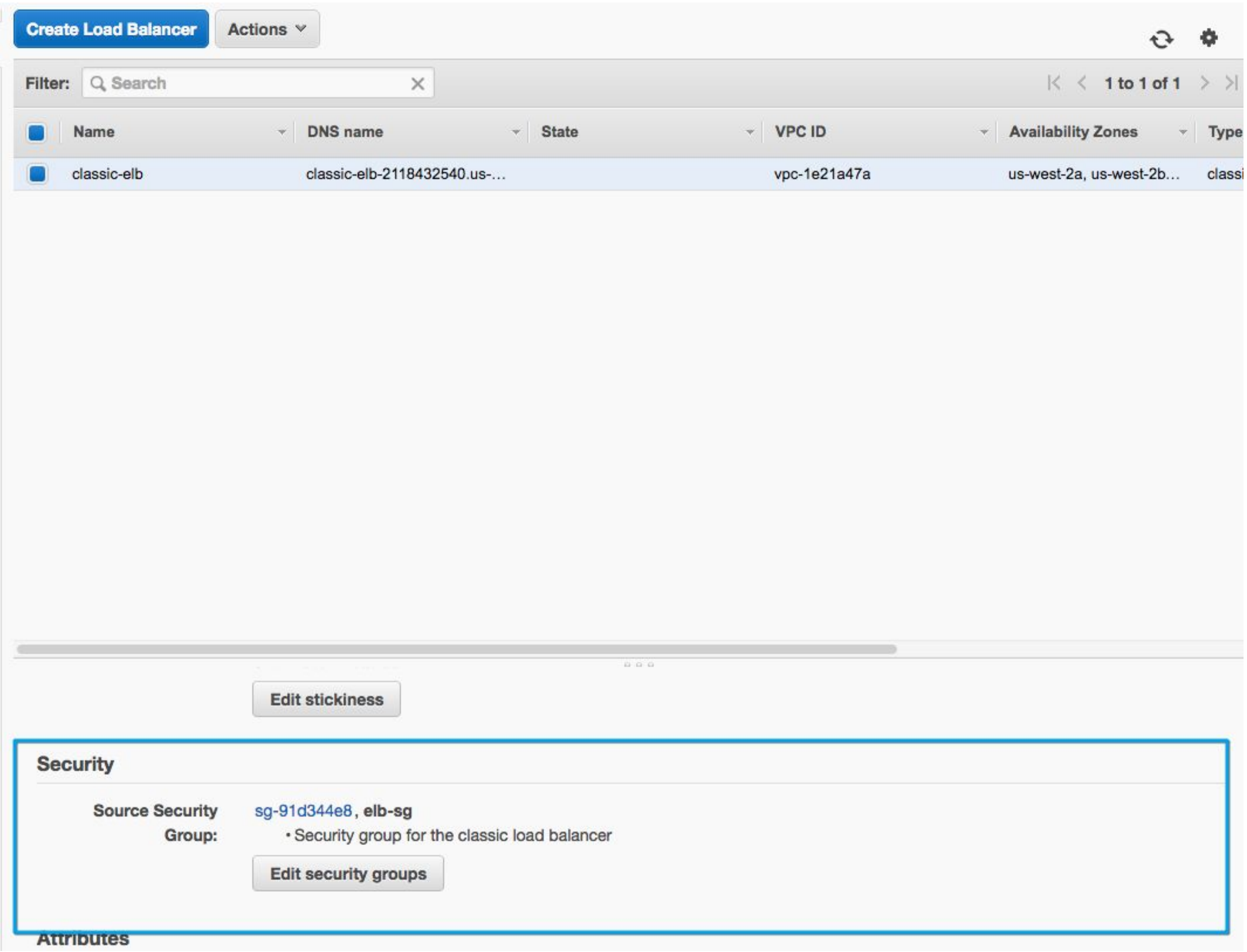
Paste this address into a new tab in your browser (don't include the "A Record" information in parentheses):


EC2 Management Console
Creating your first Classic Loa
classic-elb-2118432540.us-west-2.elb.amazonaws.com

Now you are accessing a particular instance directly, and we want to avoid that. To do so we need to change a few things. This is happening because the SG associated with the EC2 instances is allowing access from anywhere to the port 80; we want to change it in a way that the instances will only allow traffic coming from the LB we just created. To configure this go back to the **EC2 management console** on the **Load Balancers** page:



And scroll down until you see **Security**:

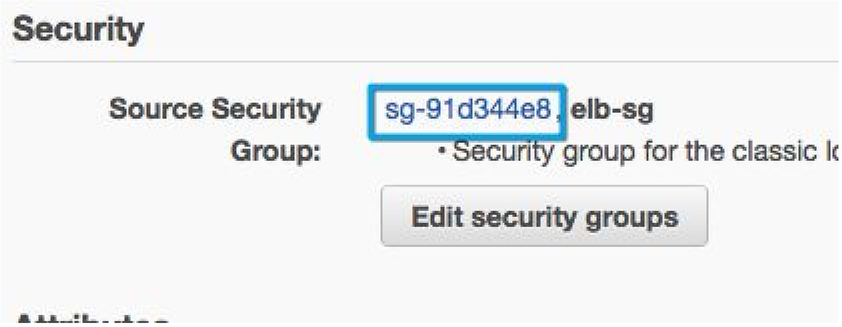


Name	DNS name	State	VPC ID	Availability Zones	Type
classic-elb	classic-elb-2118432540.us-…		vpc-1e21a47a	us-west-2a, us-west-2b…	classi

Security

Source Security Group: sg-91d344e8, elb-sg
• Security group for the classic load balancer

Copy the unique identifier of the LB's Security Group:

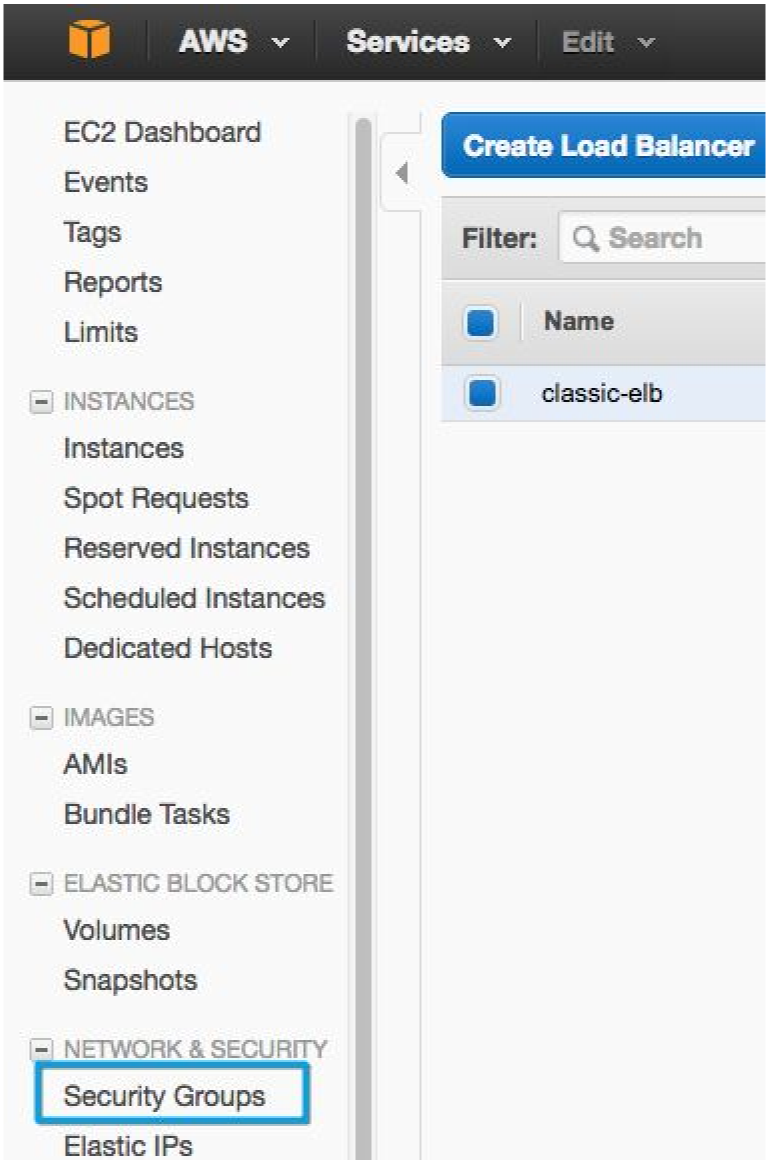


Security

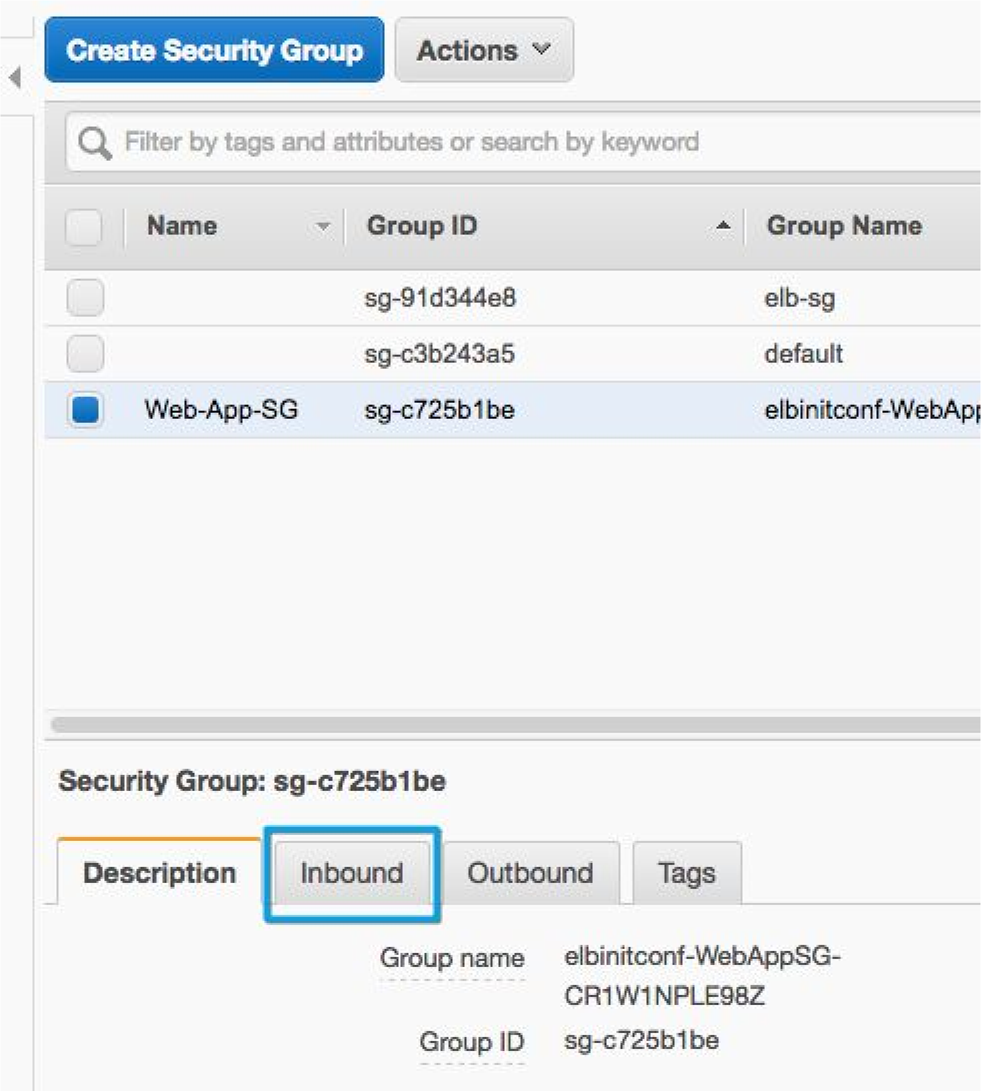
Source Security Group: sg-91d344e8, elb-sg
• Security group for the classic lo

We will use this information in just a moment.

Now click on **Security Groups:**



Select the SG called `Web-App-SG` and click on the **Inbound** tab:



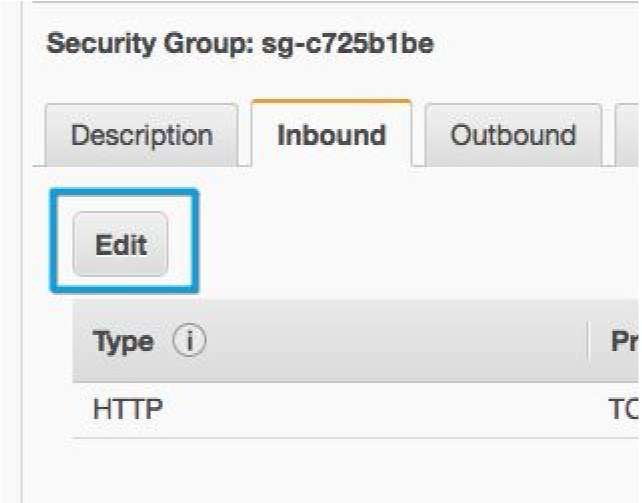
□	Name	Group ID	Group Name
□		sg-91d344e8	elb-sg
□		sg-c3b243a5	default
☑	Web-App-SG	sg-c725b1be	elbinitconf-WebApp

Security Group: sg-c725b1be

Group name elbinitconf-WebAppSG-CR1W1NPLE98Z

Group ID sg-c725b1be

Now, click on **Edit** to change the current rules associated with this SG:

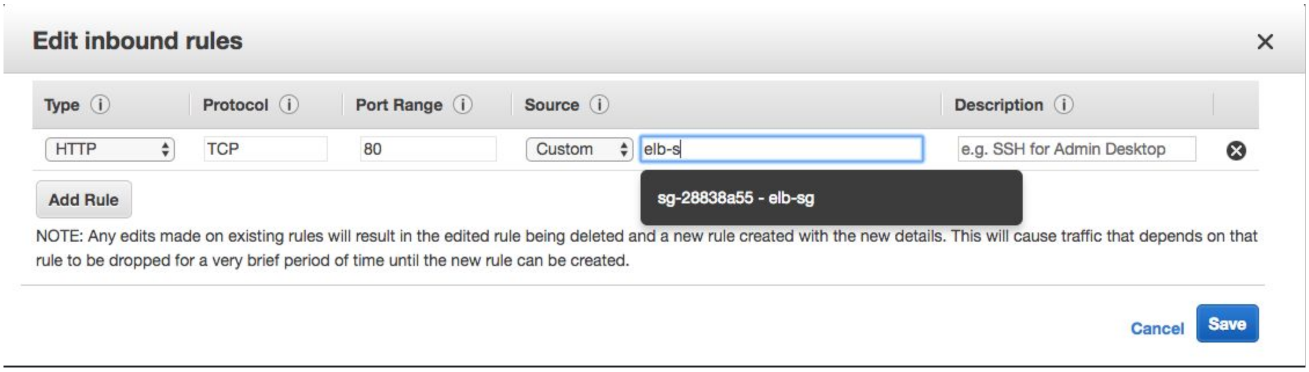


Security Group: sg-c725b1be

Type	Pr
HTTP	TC

We want to allow only connections coming from the LB to the instances, however, the **LB doesn't have a particular IP address** associated with it so we can't specify an IP address in here. Instead, we will restrict the access by using the SG we just created for the LB. We will change the current rule to deny access to anywhere and allow it only to members of the LB's security group. The process is very straight forward, in the source column, select custom and then simply replace the source in the HTTP rule that is already created with the LB's security group identifier that you just copied:

You can also start typing "sg-" and select the correct SG identifier in the list that will appear. Then click **Save**.



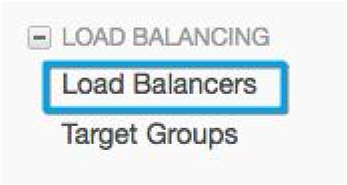
Edit inbound rules

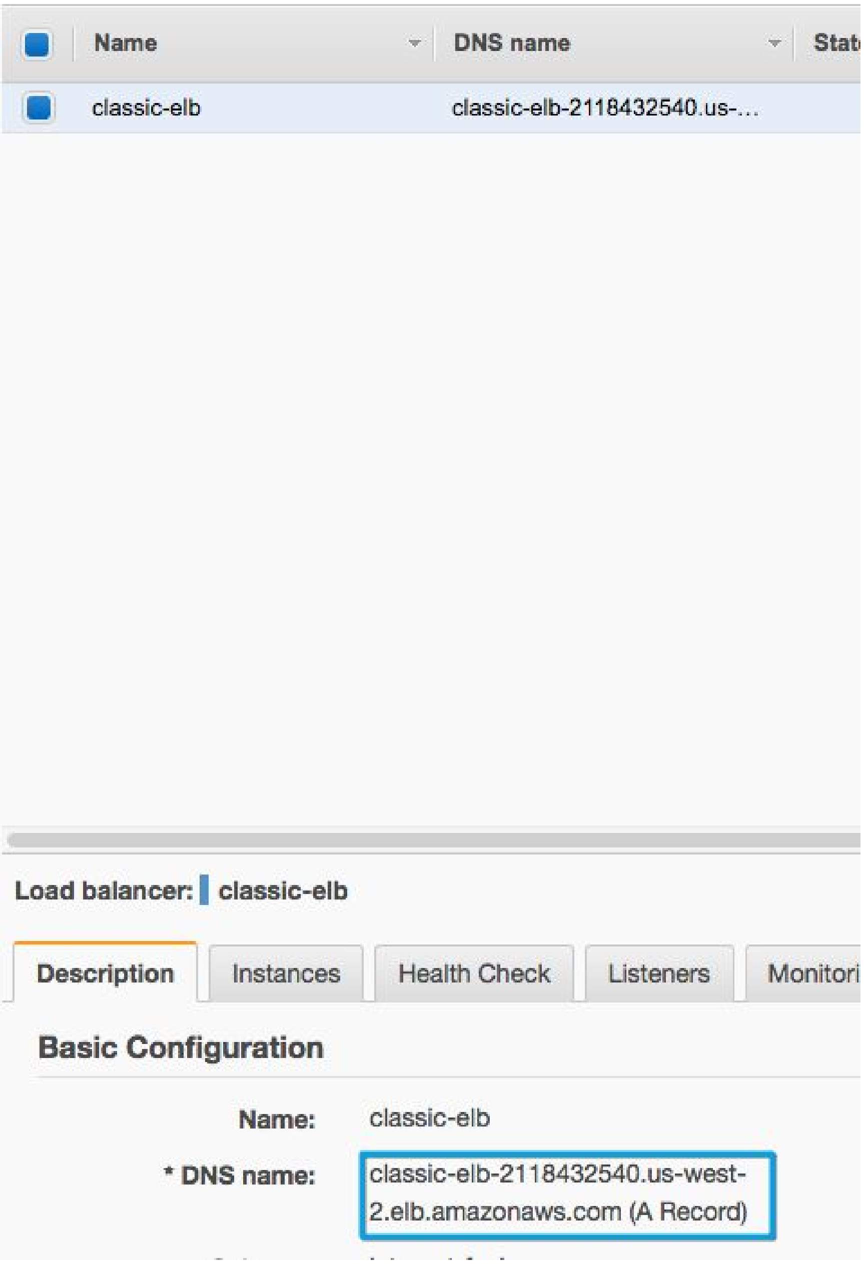
Type	Protocol	Port Range	Source		Description
HTTP	TCP	80	Custom	elb-s	e.g. SSH for Admin Desktop

sg-28838a55 - elb-sg

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

Now let's test the rule. Click on **Load Balancers** to be able to copy the URL for the LB again:





Name	DNS name	Stat
classic-elb	classic-elb-2118432540.us-…	

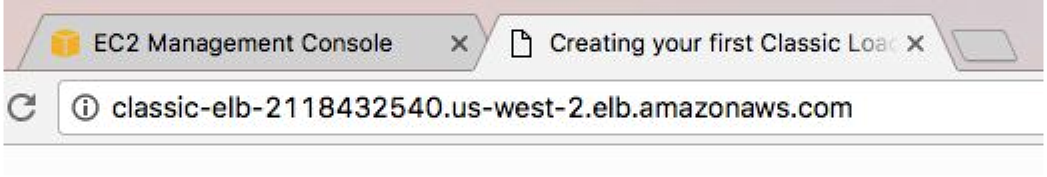
Load balancer: classic-elb

Basic Configuration

Name: classic-elb

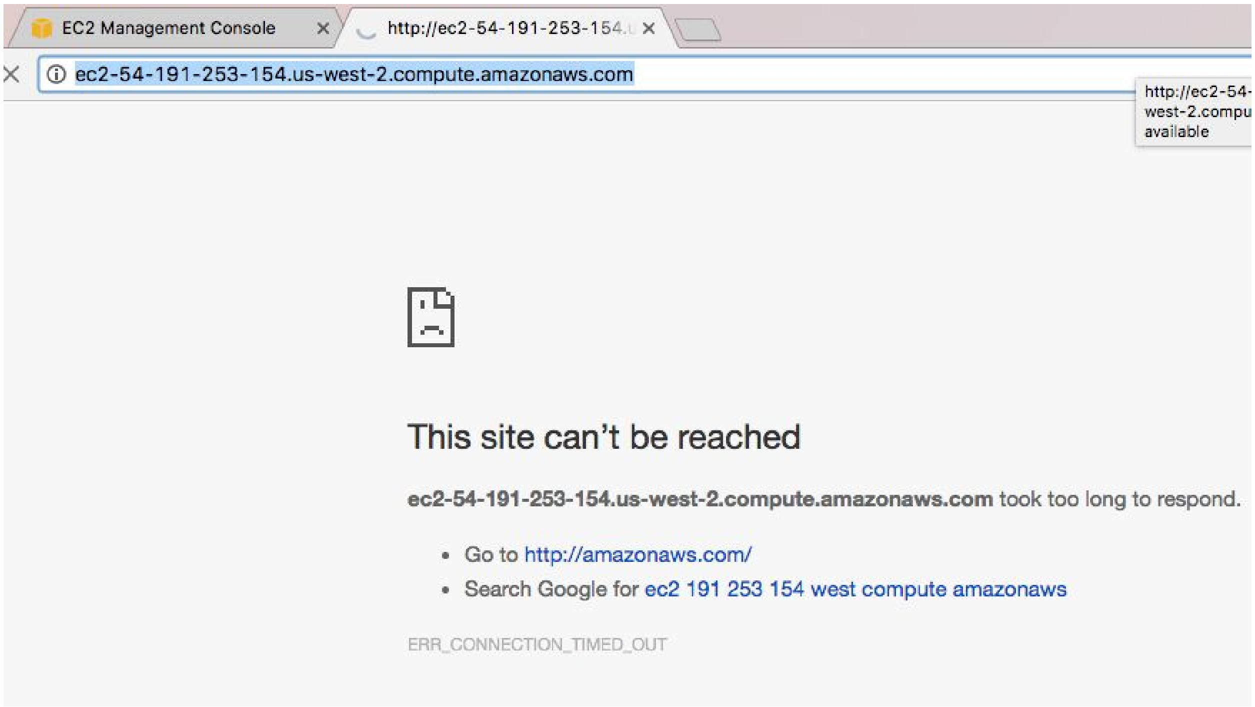
* DNS name: classic-elb-2118432540.us-west-2.elb.amazonaws.com (A Record)

Copy the **DNS Name** address and past it into a new tab in your browser:



classic-elb-2118432540.us-west-2.elb.amazonaws.com

Nothing new here. However, if you try to copy and past the public DNS address of the instance you are accessing, and paste it into your browser, you shouldn't be able to access the app - the connection will timeout.



EC2 Management Console
http://ec2-54-191-253-154.
ec2-54-191-253-154.us-west-2.compute.amazonaws.com
http://ec2-54-
west-2.compu
available

This site can’t be reached

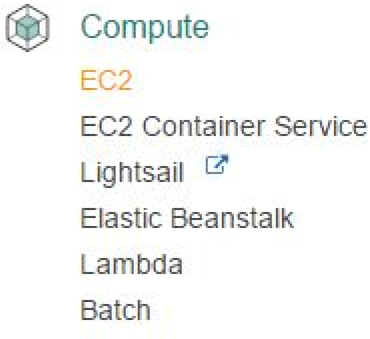
ec2-54-191-253-154.us-west-2.compute.amazonaws.com took too long to respond.

- Go to http://amazonaws.com/
- Search Google for ec2 191 253 154 west compute amazonaws

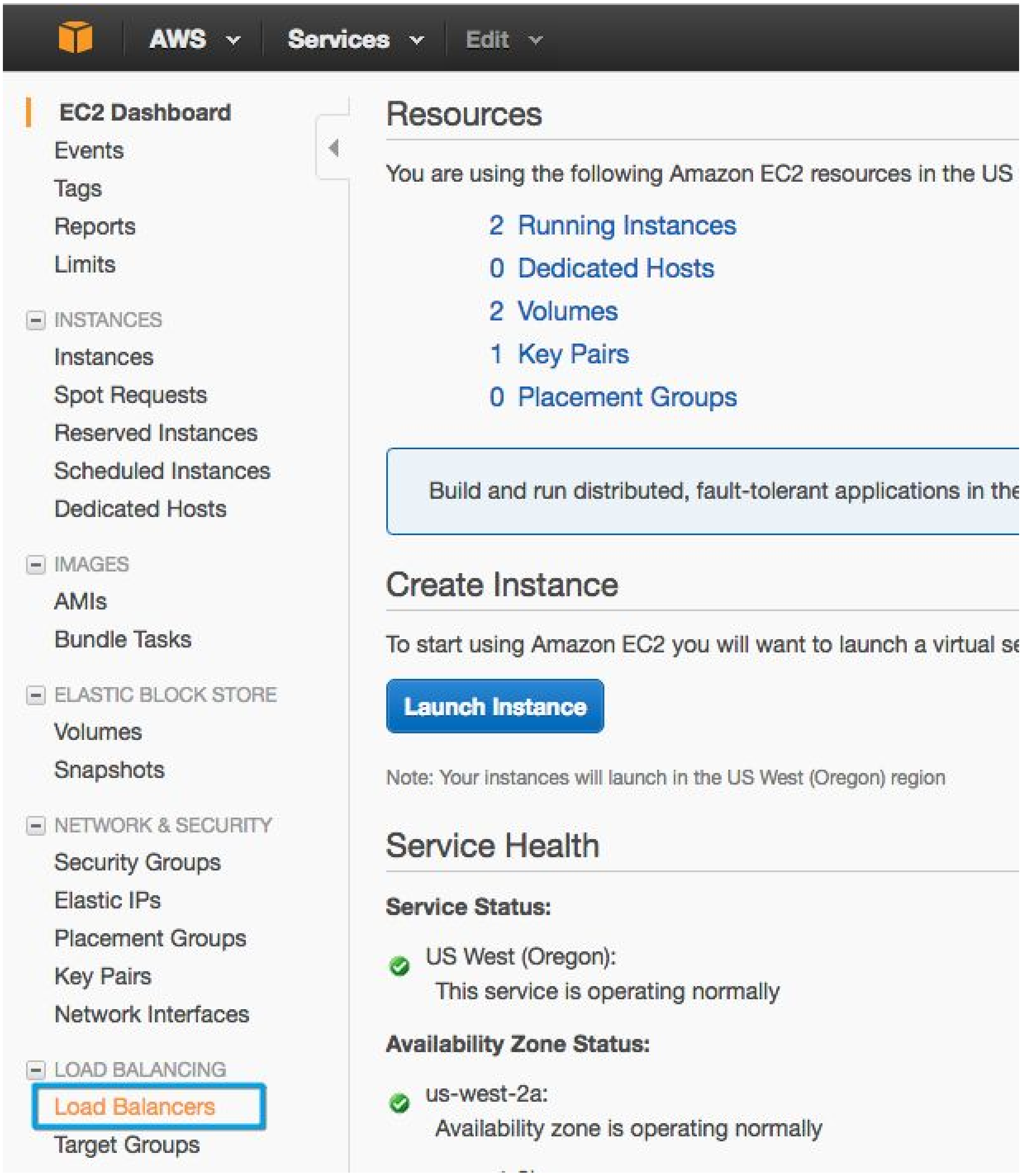
ERR_CONNECTION_TIMED_OUT

Step 5 Checking a load balancer's behavior during instance failures

In this step, we will make an EC2 instance fail and see how the ELB service responds to that. To do so, go to the EC2 console:



And click on Load Balancers:



Resources

You are using the following Amazon EC2 resources in the US

2 Running Instances
0 Dedicated Hosts
2 Volumes
1 Key Pairs
0 Placement Groups

Build and run distributed, fault-tolerant applications in the

Create Instance

To start using Amazon EC2 you will want to launch a virtual se

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

Service Health

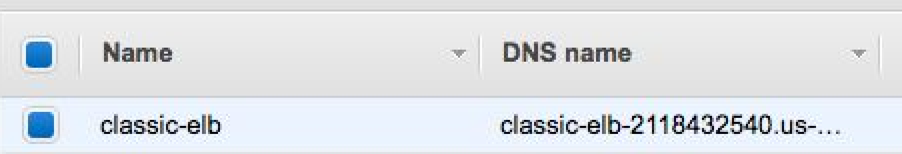
Service Status:

US West (Oregon):
This service is operating normally

Availability Zone Status:

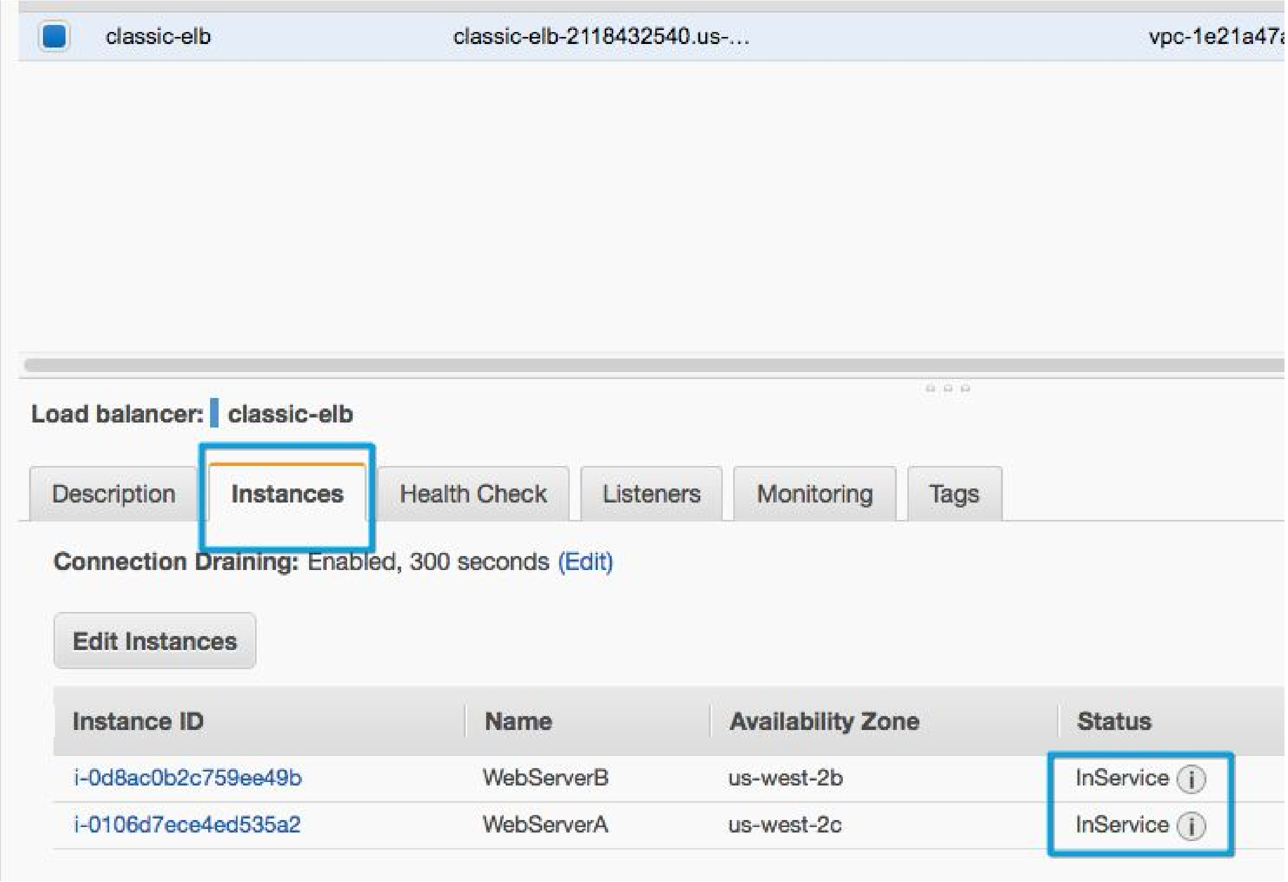
us-west-2a:
Availability zone is operating normally

Then select the LB you just created:



	Name	DNS name
■	classic-elb	classic-elb-2118432540.us-…

If you click on **Instances** you will be able to see that both instances are currently on service:



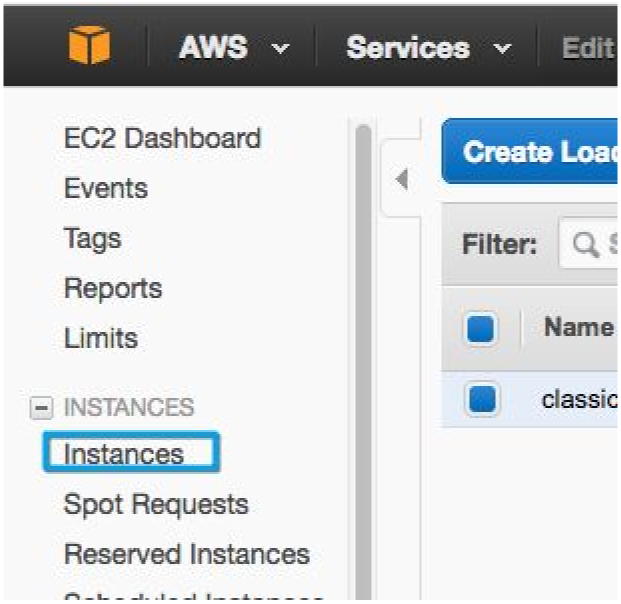
■ classic-elb classic-elb-2118432540.us-… vpc-1e21a47

Load balancer: classic-elb

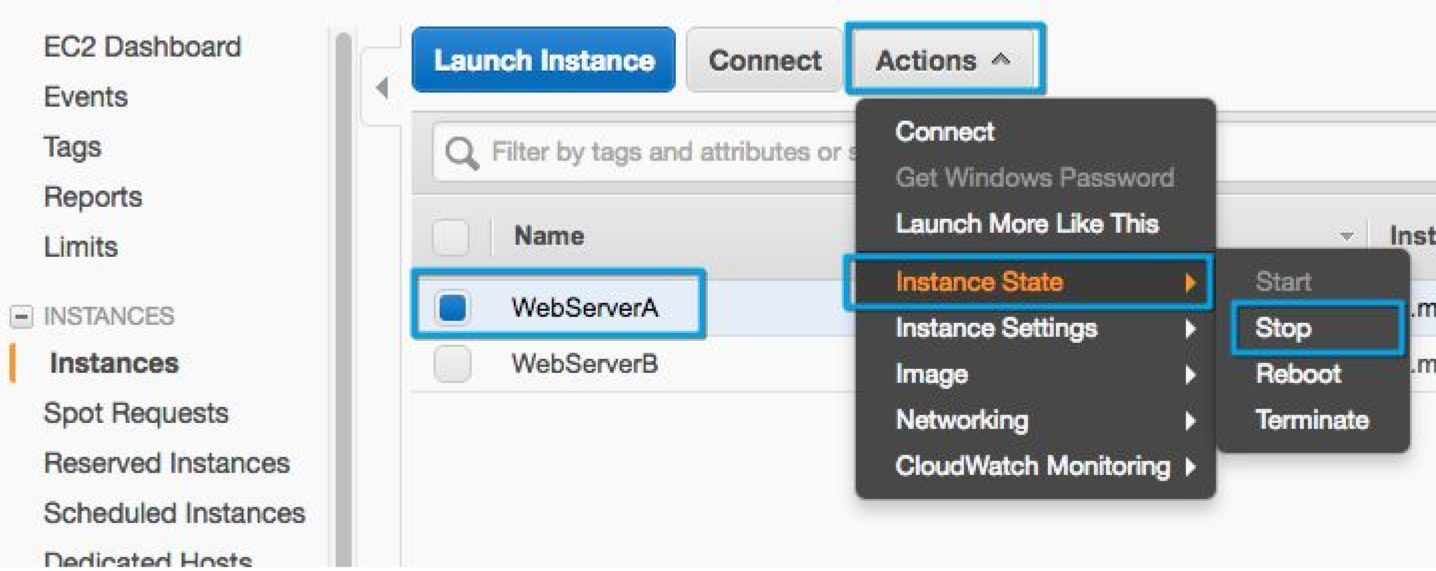
Connection Draining: Enabled, 300 seconds (Edit)

Instance ID	Name	Availability Zone	Status
i-0d8ac0b2c759ee49b	WebServerB	us-west-2b	InService ⓘ
i-0106d7ece4ed535a2	WebServerA	us-west-2c	InService ⓘ

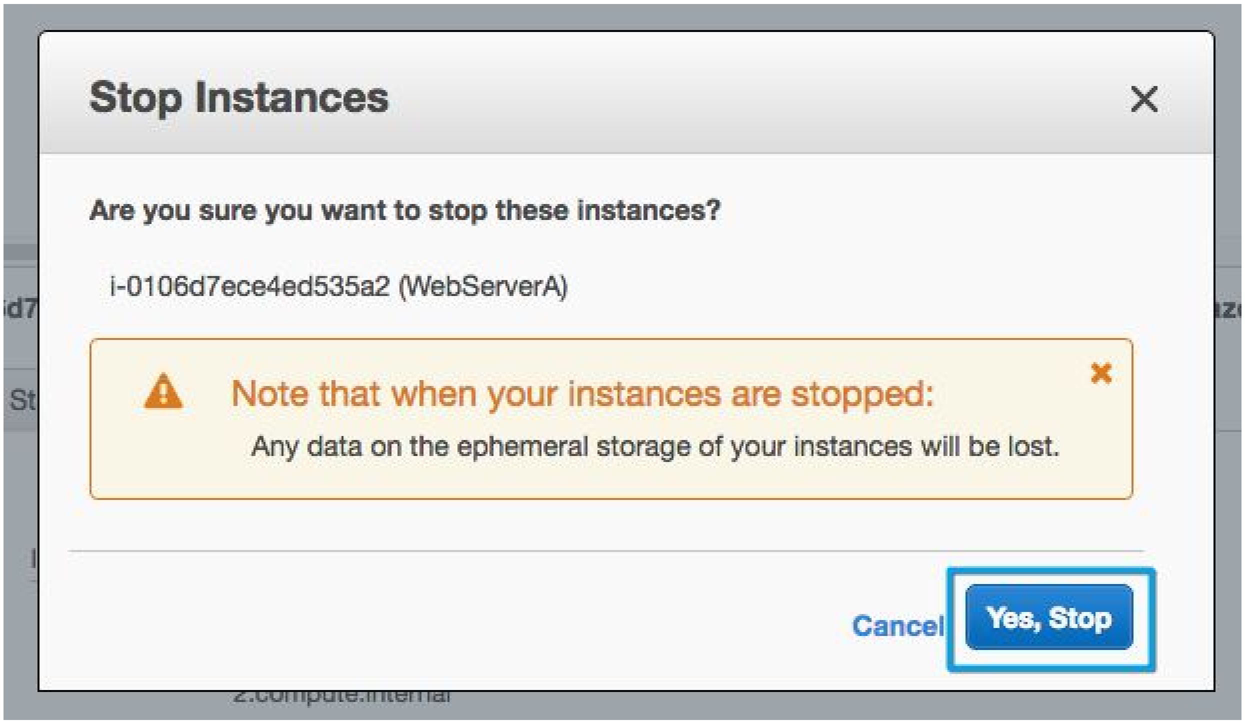
This means that the instances are answering the health checks we configured during the LB's creation, and thus the ELB service has decided to put them **InService.** To see what happens with the LB when there is a problem in a particular instance, let's simulate an instance failure. The easiest way to do this is to stop a running instance, so let's do that. Click on **Instances**:



And choose any of the running instances, click on **Actions, Instance State,** then on **Stop**:



Then click on **Yes, Stop** on the dialog box that pops up:



Stop Instances

Are you sure you want to stop these instances?

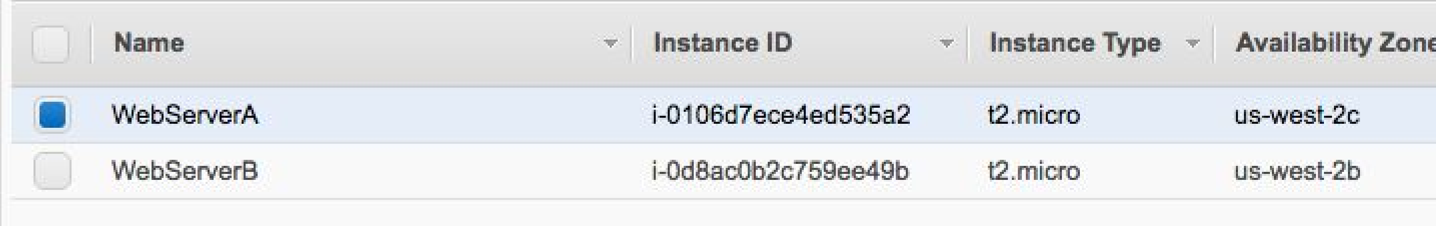
i-0106d7ece4ed535a2 (WebServerA)

Note that when your instances are stopped:

Any data on the ephemeral storage of your instances will be lost.

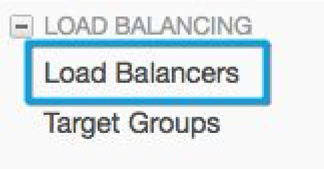
Doing this will stop a particular instance, which will make it fail the ELB's health checks.

Once the **Instance State** changes to **stopped** you can proceed:

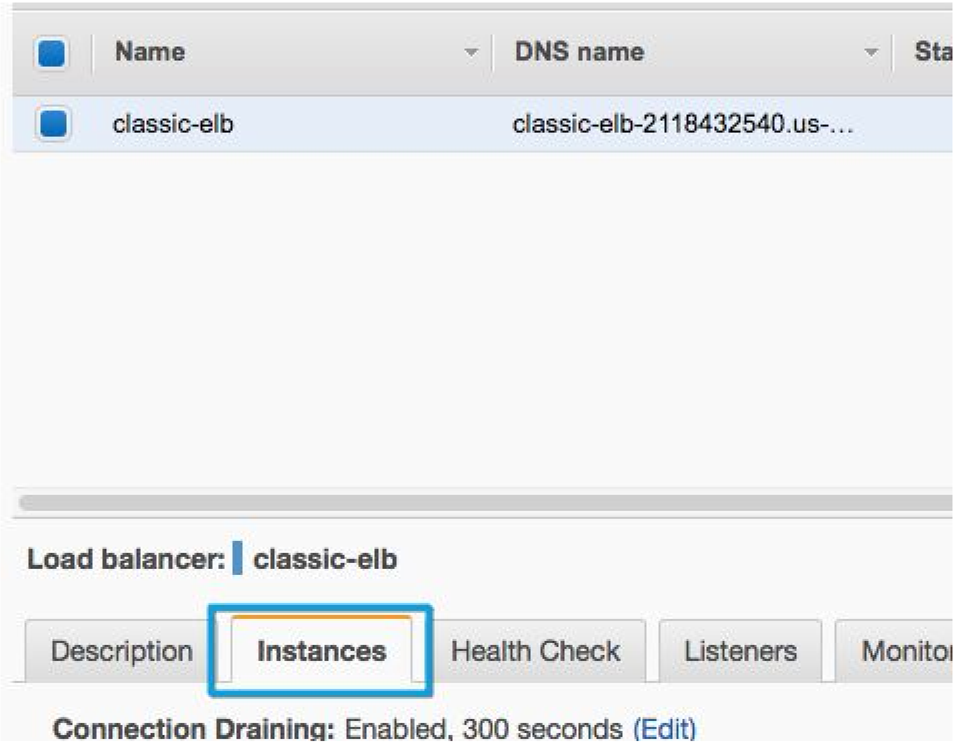


	Name	Instance ID	Instance Type	Availability Zone
☑	WebServerA	i-0106d7ece4ed535a2	t2.micro	us-west-2c
□	WebServerB	i-0d8ac0b2c759ee49b	t2.micro	us-west-2b

Now click on **Load Balancers**:



Select the LB and click on **Instances**:

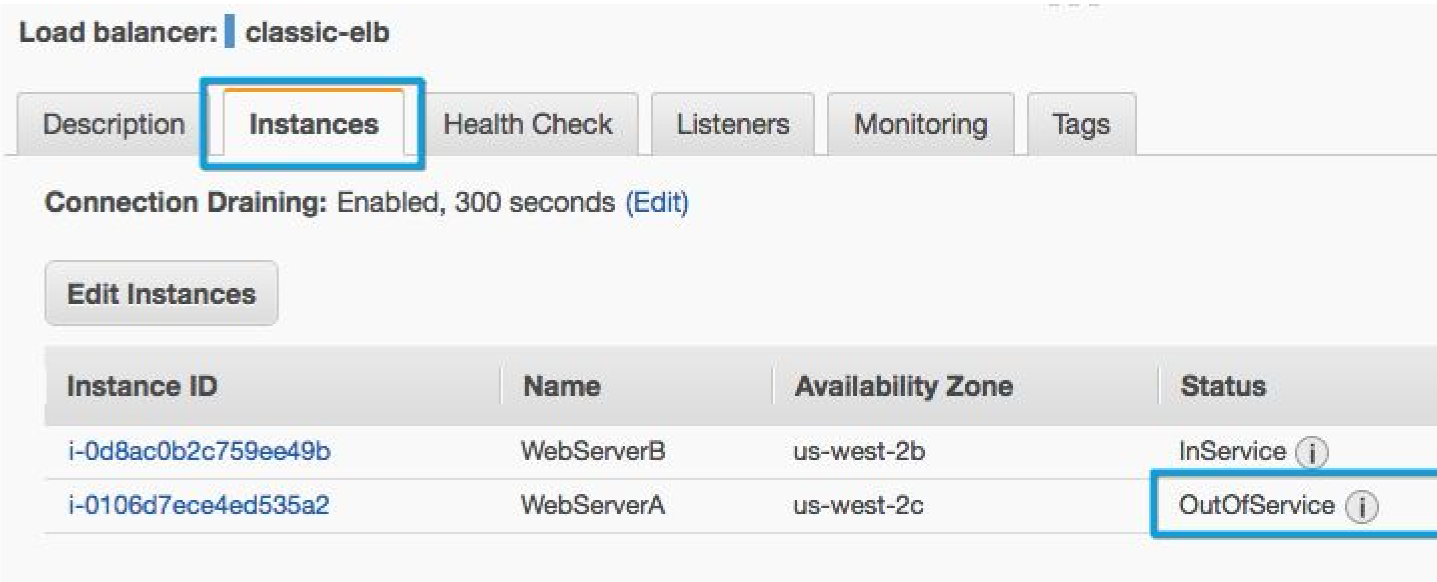


	Name	DNS name	Sta
■	classic-elb	classic-elb-2118432540.us-…	

Load balancer: classic-elb

Connection Draining: Enabled, 300 seconds (Edit)

Now you should be able to see an instance with the status **OutOfService**

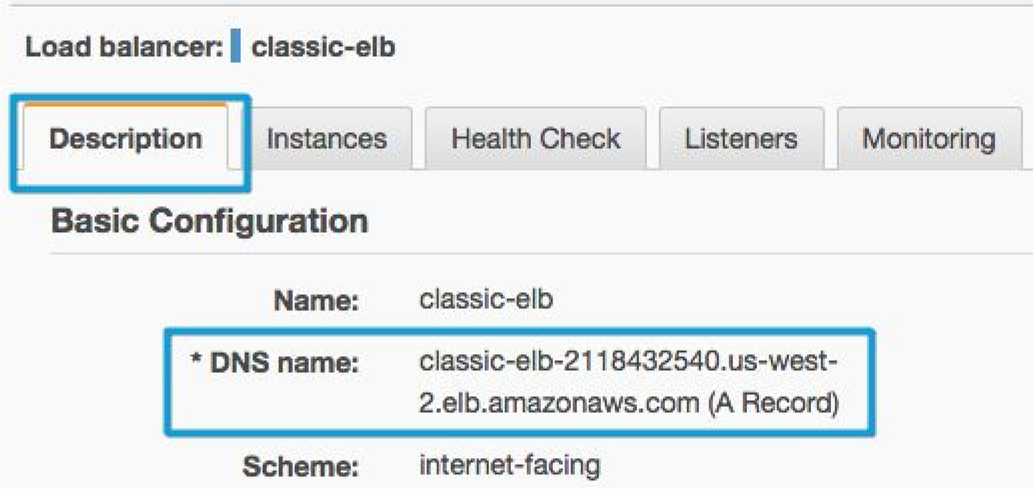


Load balancer: classic-elb

Connection Draining: Enabled, 300 seconds (Edit)

Instance ID	Name	Availability Zone	Status
i-0d8ac0b2c759ee49b	WebServerB	us-west-2b	InService
i-0106d7ece4ed535a2	WebServerA	us-west-2c	OutOfService

This means that there is only one instance serving the application, and therefore all the requests will be forwarded to the same instance. You can test this behavior by clicking on **Description** and accessing the **DNS name** of the LB:



Load balancer: classic-elb

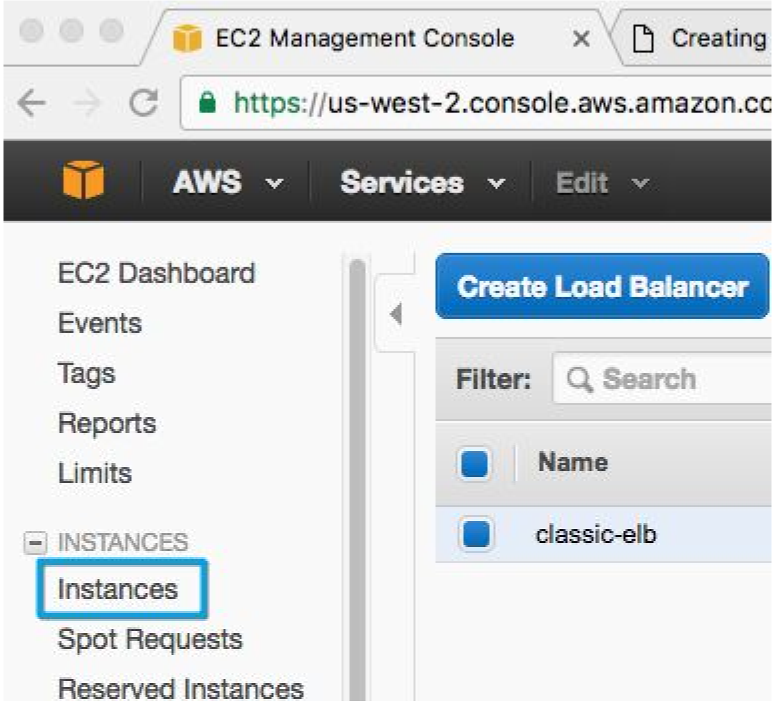
Basic Configuration

Name: classic-elb

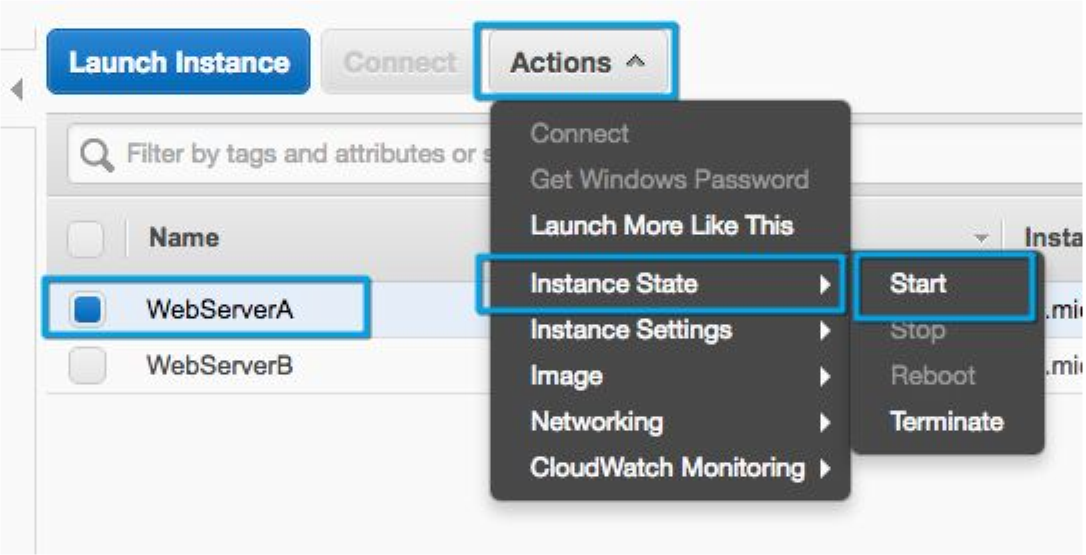
* DNS name: classic-elb-2118432540.us-west-2.elb.amazonaws.com (A Record)

Scheme: internet-facing

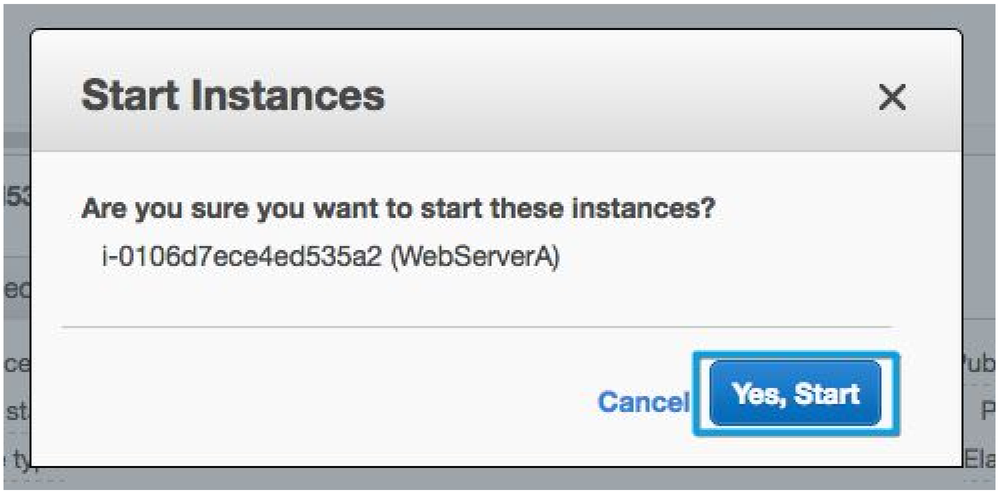
At this stage, no matter how many times you hit the refresh button, you will always be forwarded to the same instance. Hit refresh a few times, then go back to the **EC2 Management Console**, and click on **Instances**:



Start the stopped instance again by selecting it, clicking on **Actions, Instance State,** then on **Start**:



Click on **Yes, Start** in the dialogue box that pops up:

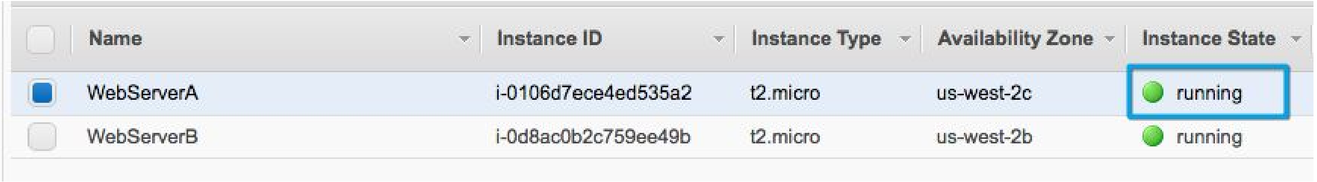


Start Instances

Are you sure you want to start these instances?

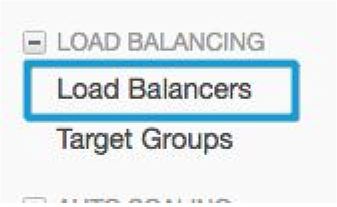
i-0106d7ece4ed535a2 (WebServerA)

Wait until the **Instance State** changes to **running**:

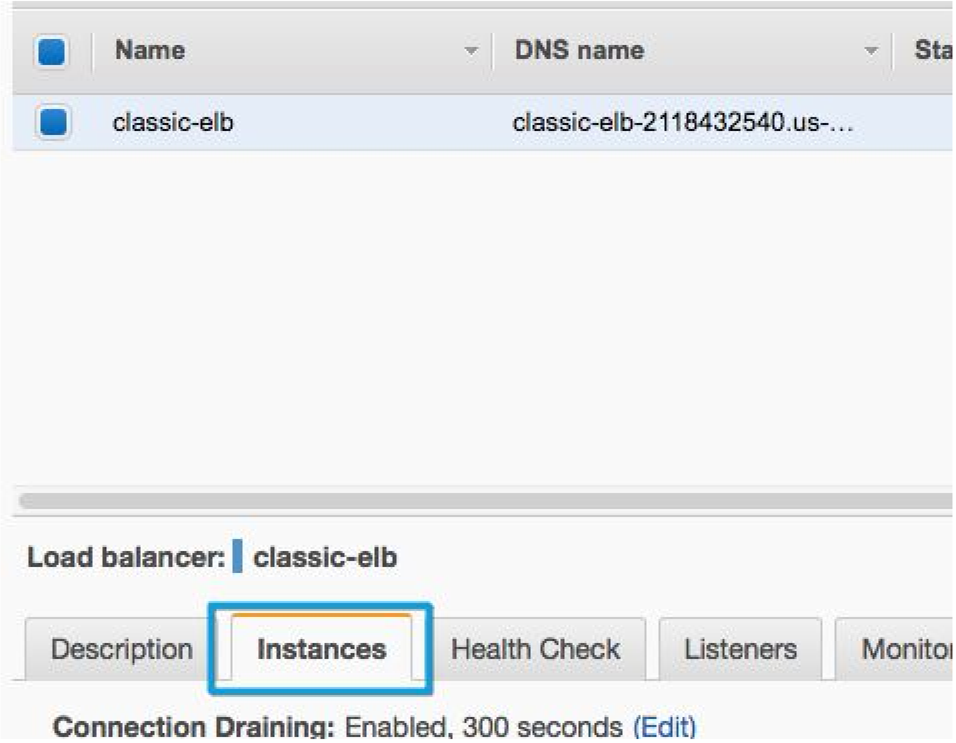


□	Name	Instance ID	Instance Type	Availability Zone	Instance State
■	WebServerA	i-0106d7ece4ed535a2	t2.micro	us-west-2c	● running
□	WebServerB	i-0d8ac0b2c759ee49b	t2.micro	us-west-2b	● running

Then click on **Load Balancers**:



Select the LB and click on **Instances**:

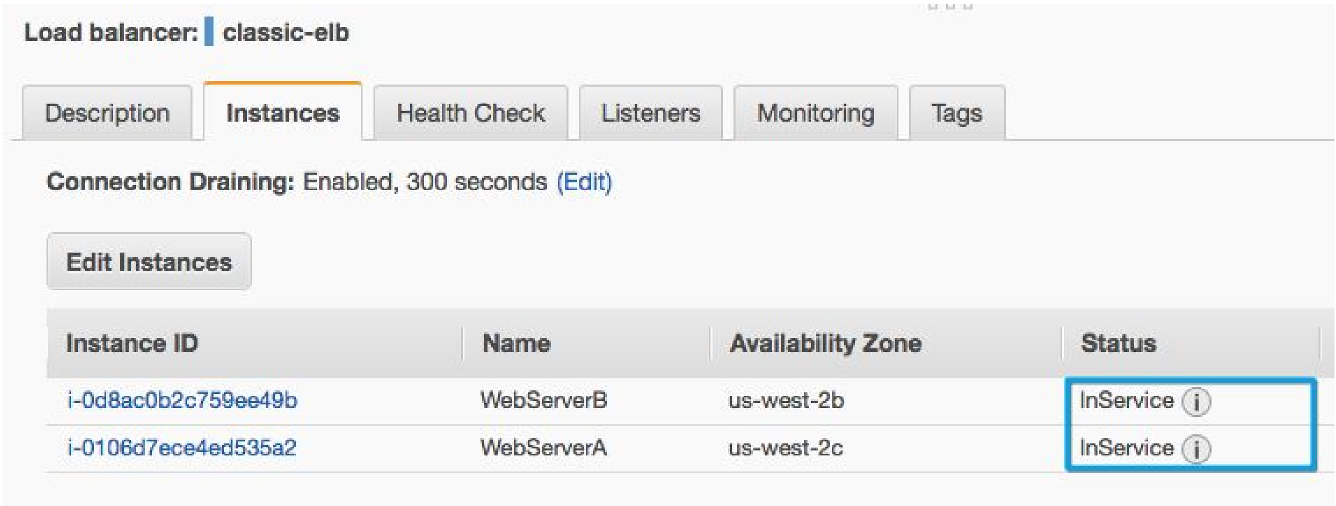


Name	DNS name	Sta
classic-elb	classic-elb-2118432540.us-…	

Load balancer: classic-elb

Connection Draining: Enabled, 300 seconds (Edit)

And you should be able to see that all the instances have an **InService** status again:



Load balancer: classic-elb

Connection Draining: Enabled, 300 seconds (Edit)

Instance ID	Name	Availability Zone	Status
i-0d8ac0b2c759ee49b	WebServerB	us-west-2b	InService
i-0106d7ece4ed535a2	WebServerA	us-west-2c	InService

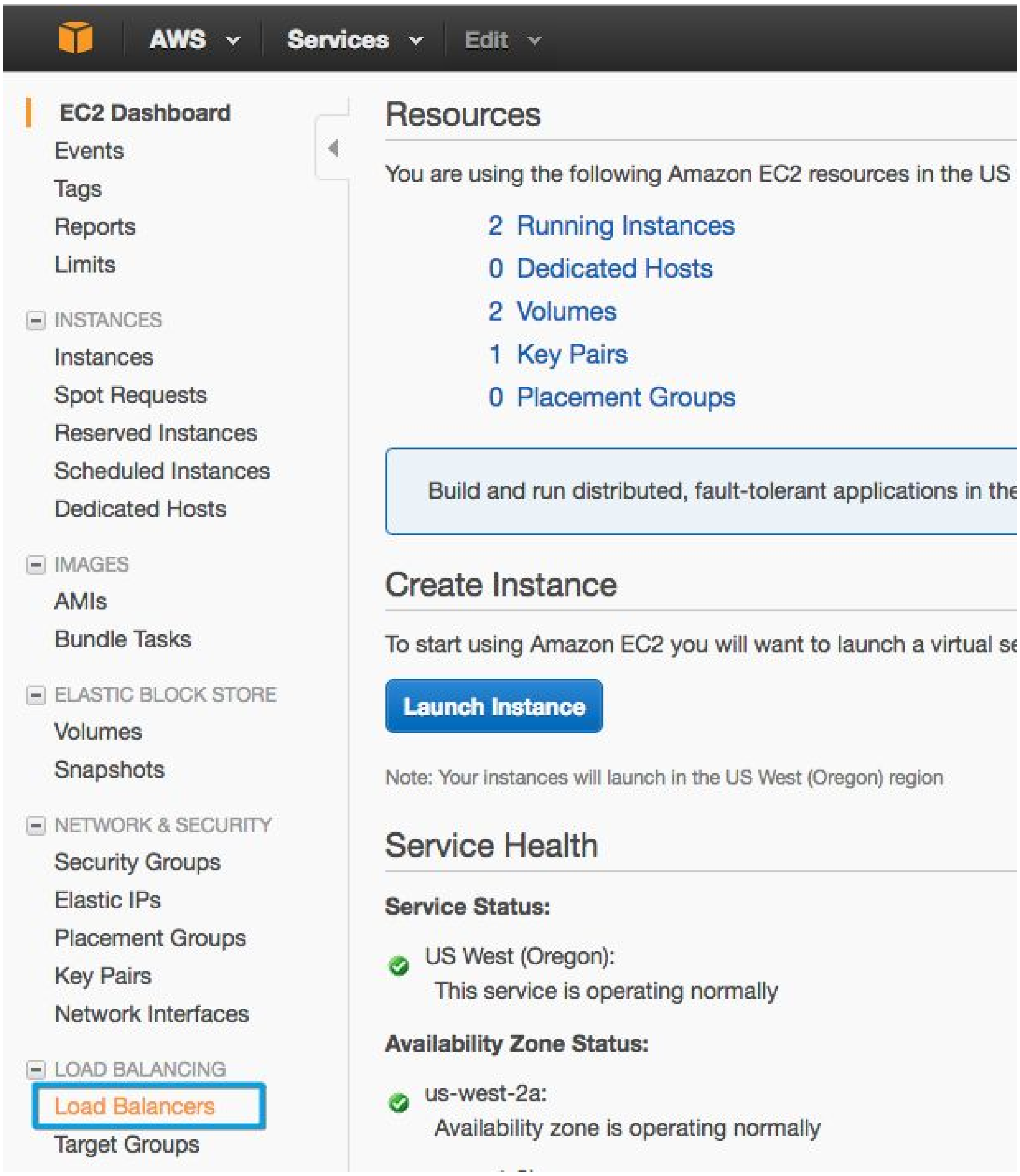
You can test the LB's behavior again by accessing its **DNS Name** in your browser. Once you're done testing move to the next lab step.

Step 6 Monitoring your Classic Load Balancer

In this step, we will take a quick look at the most common metrics for troubleshooting problems with your Classic Load Balancer and the EC2 instances running behind it.

There are two ways of doing that. One is on the CloudWatch console, which can be a bit frustrating for newcomers because it will hold metrics for ALL LBs that existed within the past 2 weeks in the AWS account you're using, and you might get lost with so many metrics. With that in mind, we will take a different approach, and we will take a look at the metrics related to our LB in the EC2 console. To do so, go to the EC2 console:

And click on Load Balancers:



Resources

You are using the following Amazon EC2 resources in the US

2 Running Instances
0 Dedicated Hosts
2 Volumes
1 Key Pairs
0 Placement Groups

Build and run distributed, fault-tolerant applications in the

Create Instance

To start using Amazon EC2 you will want to launch a virtual se

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

Service Health

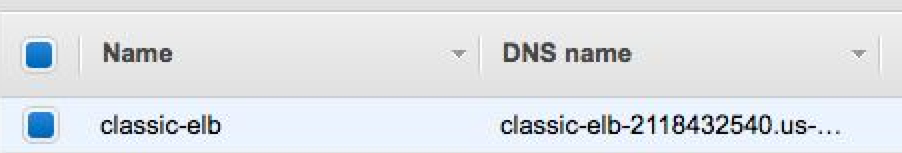
Service Status:

✓ US West (Oregon):
This service is operating normally

Availability Zone Status:

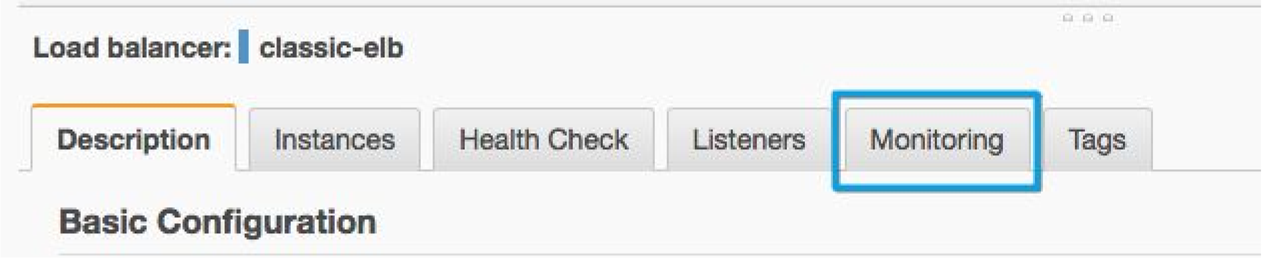
✓ us-west-2a:
Availability zone is operating normally

Then select the LB you just created:



■	Name	DNS name
■	classic-elb	classic-elb-2118432540.us-…

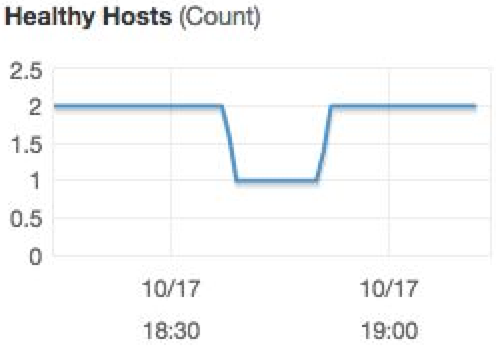
Click on the **Monitoring** tab to see the metrics of the LB you selected:



The ELB service reports metrics to CloudWatch only when requests are flowing through the LB. If there are requests flowing through the LB, ELB measures and sends its metrics in 60-second intervals. If there are no requests flowing through the load balancer, or no data for a metric, the metric is not reported. There are a few metrics related to a Classic Load Balancer, and in general they are self-explanatory. However, some of them may be unfamiliar to you, in which case you can take a look at the description for all metrics in here: http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-cloudwatch-metrics.html

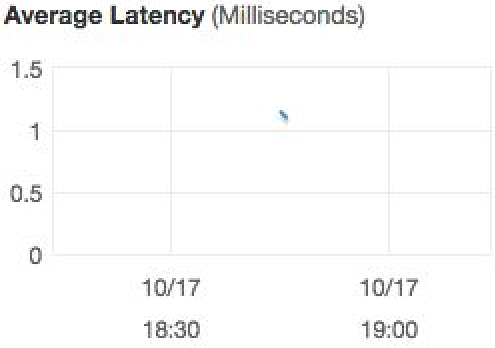
The metrics called **HealthyHostCount**, and **UnHealthyHostCount** will count the number of Healthy and Unhealthy instances respectively. These metrics can be useful for you to identify a major problem in your AWS account. For example, you could set up some CloudWatch Alarms to notify you when you have less than 2 instances running your application, though to be clear this is not a general rule: the number of instances that might identify a problem will vary depending on your environment.

Also notice that in these metrics, there is no way of seeing the Availability Zone to which the Healthy/Unhealthy instance belongs. In our lab, we stopped an instance for a few minutes, therefore you should be able to see something like this:



If the **Healthy Hosts** metric reaches 0, that means that people won't see anything when accessing your LB, and it is probable that you have a big problem in your infrastructure.

The **Average Latency** metric might be useful to identify potential issues in your setup. Maybe everything is working in your application, but you notice an increase in this metric. If you haven't changed anything in your application, that can be a potential issue - maybe you haven't provisioned enough EC2 instances, or you even have lots of instances but they don't have enough power to serve your increasing traffic.



The other metrics can be very useful for troubleshooting specific scenarios and will vary depending on your setup.

You have now covered all the Learning Objectives in this lab. You can take some time to play around with the metrics. You can do things such as:

- Stop ALL instances and make requests to the LB
- Make several requests to paths that don't exist, such as <dns name>/app, and <dns name>/users to generate some errors in the responses