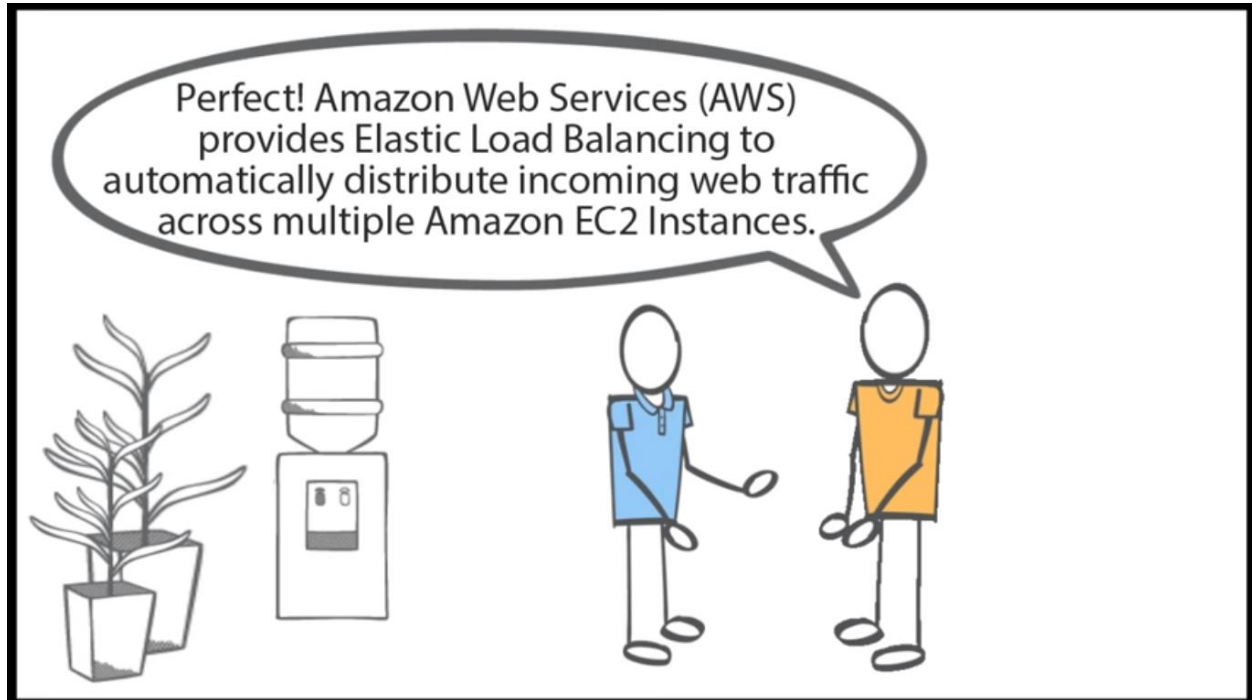


Elastic Load Balancing

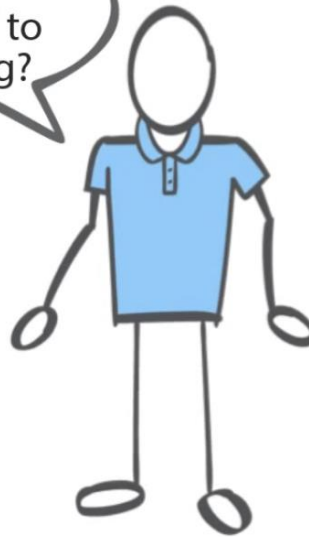
Elastic Load Balancing (ELB) automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables you to achieve greater fault tolerance in your applications and seamlessly provides the correct amount of load balancing capacity needed in response to incoming application traffic.

Elastic Load Balancing detects unhealthy instances within a pool and automatically reroutes traffic to healthy instances until the unhealthy instances have been restored to health. Customers can enable Elastic Load Balancing within a single Availability Zone or across multiple zones but not across multiple regions for greater consistent application performance.

Interactive tutorial:



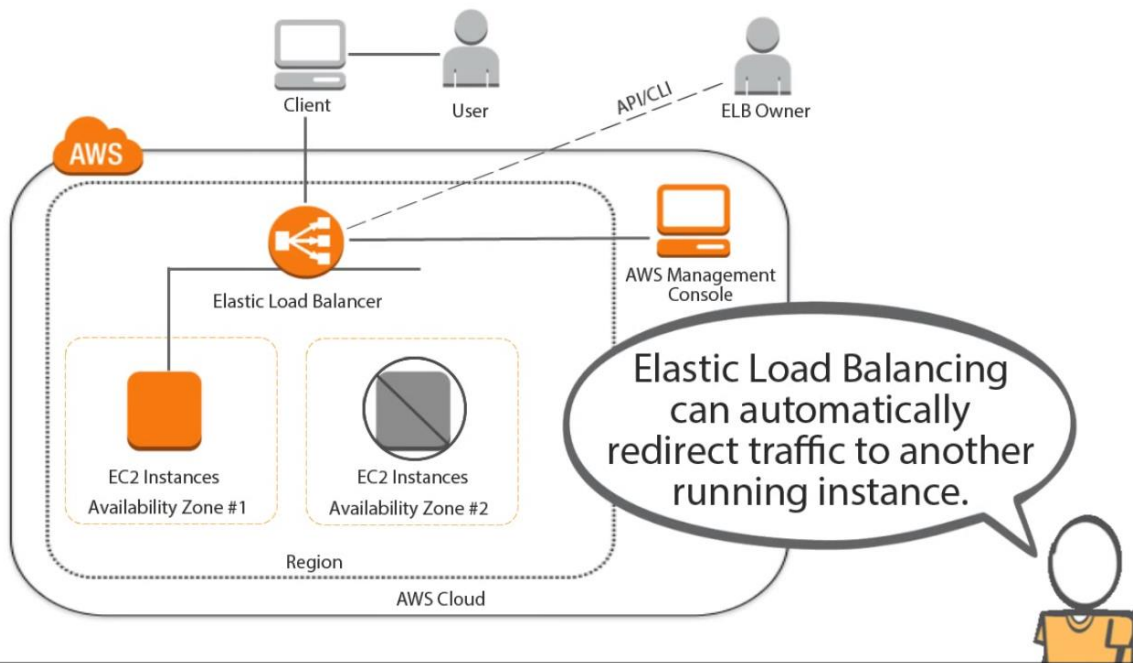
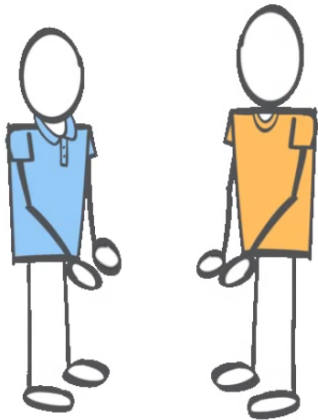
Does this mean that if one instance is getting too much traffic, some of it can be routed to another instance for processing?

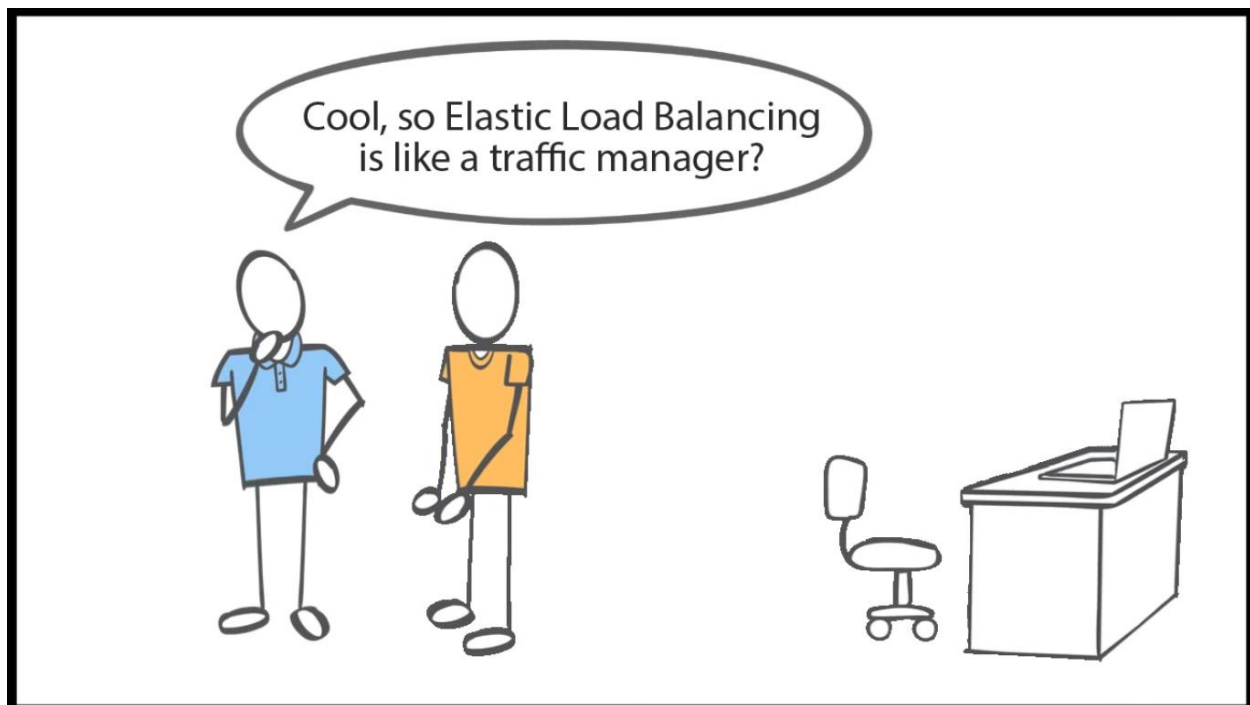
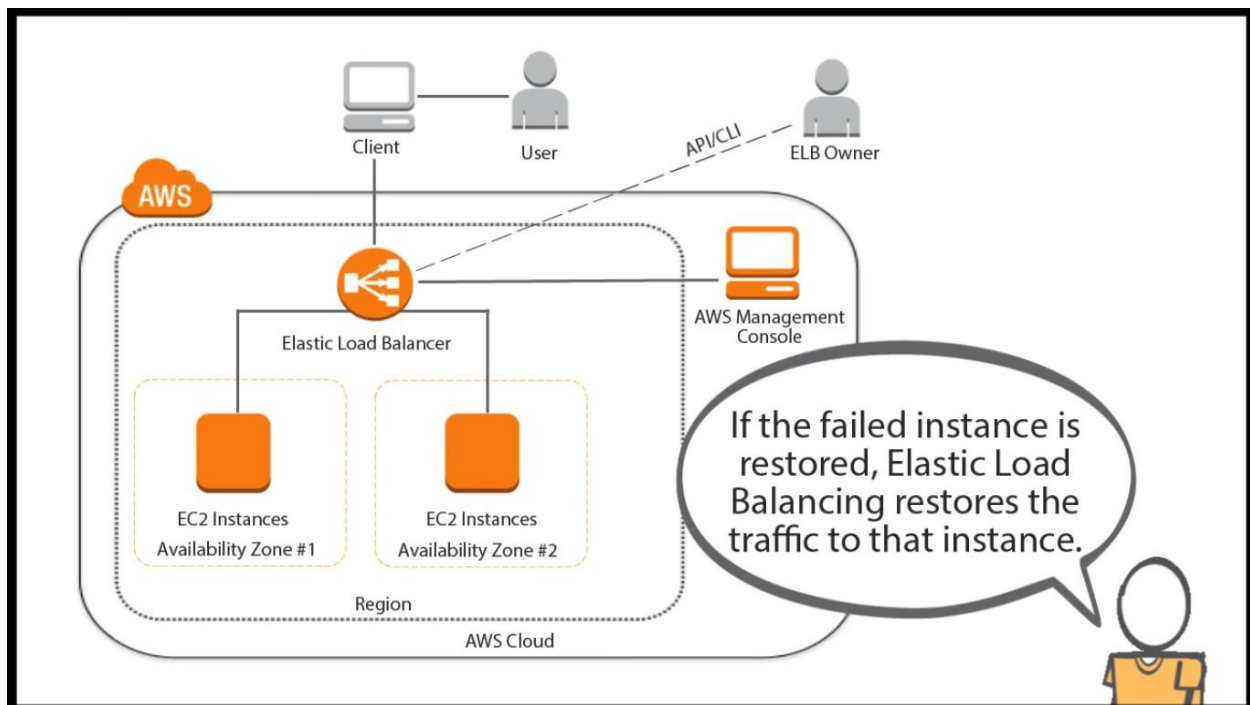


In addition, with Elastic Load Balancing you can add and remove Amazon EC2 instances as your needs change without disrupting the overall flow of information.

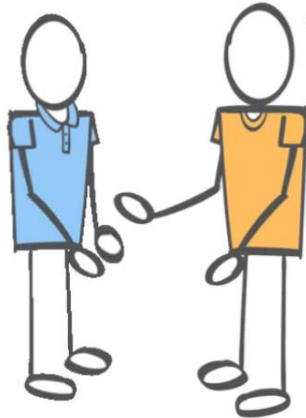


What if the instance happens to fail?

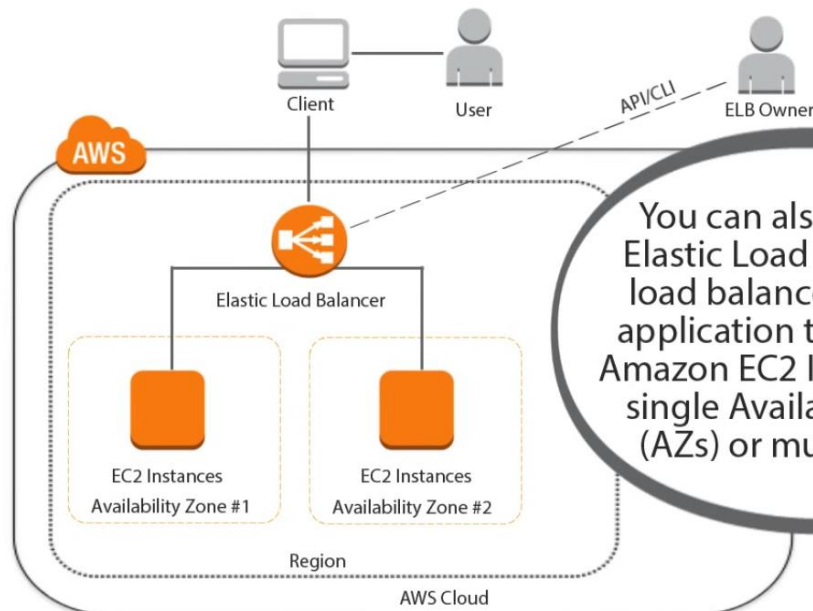




Exactly. Using Elastic Load Balancing to manage traffic to an application has a lot of benefits.



It minimizes the risk of overloading one single instance and provides continuous monitoring of the health of Amazon EC2 Instances.



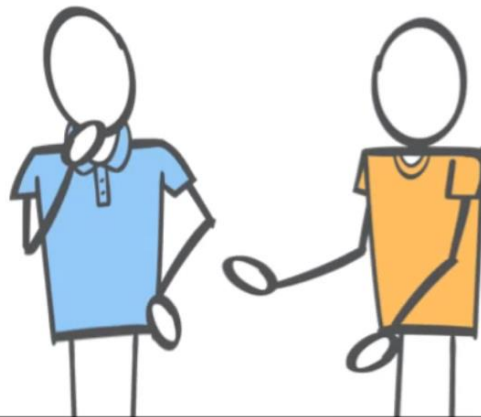
You can also setup an Elastic Load Balancer to load balance incoming application traffic across Amazon EC2 Instances in a single Availability Zone (AZs) or multiple AZs.



It minimizes the risk of overloading one single instance and provides continuous monitoring of the health of Amazon EC2 Instances.



How secure is Elastic Load Balancing?

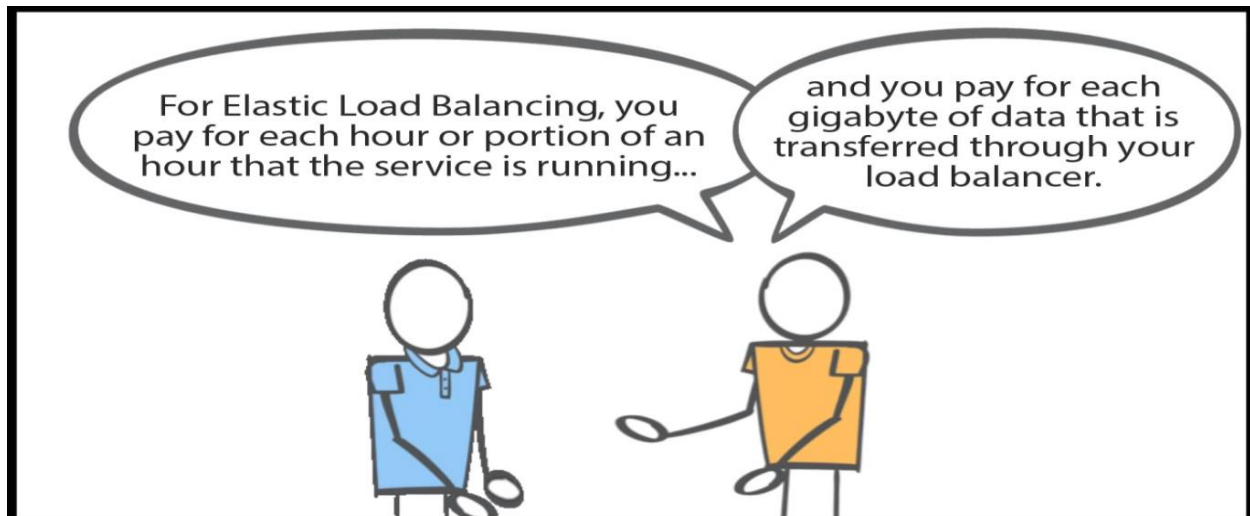


You can create and manage security groups associated with your Elastic Load Balancing to provide additional networking and security options.



Very cool. Is Elastic Load Balancing expensive?





Lab: How an ELB distributes traffic across multiple instances

Step 1. Click on EC2 Dashboard

Step 2. From the EC2 dashboard, click on launch instances with usual Amazon Linux AMI and select free tier general purpose of instance type. Click Next and select and in the **subnet select **us-west-2a** (assuming you have selected US west: Oregon region).**

1. Choose AMI 2. Choose Instance Type 3. Configure Instance **4. Add Storage** 5. Add Tags 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
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	vpc-d695eab3 (default)	Create new VPC
Subnet	No preference (default subnet in any Availability Zone)	Create new subnet
Auto-assign Public IP	No preference (default subnet in any Availability Zone)	
IAM role	subnet-21fc9156 Default in us-west-2a subnet-7bd49b1e Default in us-west-2b subnet-294ec870 Default in us-west-2c	Create new IAM role
Shutdown behavior		
Enable termination protection	<input type="checkbox"/> Protect against accidental termination	
Monitoring	<input type="checkbox"/> Enable CloudWatch detailed monitoring Additional charges apply.	

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

Step 3: Then click next:Add storage and then click next:Add tags and then click next: configure security groups. Now configure security groups as shown below and give it a name 'mylab3'

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

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop

[Add Rule](#)

Warning

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Previous](#) [Review and Launch](#)

Step 4: Click review and launch and then launch and chose a key pair if you have already created one. Then click 'Launch instances'

Launch Status

✓ Your instances are now launching

The following instance launches have been initiated: i-042bf3ef8ffa8f745 [View launch log](#)

i Get notified of estimated charges

[Create billing alerts](#) to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).

How to connect to your instances

Your instances are launching, and it may take a few minutes until they are in the **running** state, when they will be ready for you to use. Usage hours on your new instances will start immediately and continue to accrue until you stop or terminate your instances.

Click **View instances** to monitor your instances' status. Once your instances are in the **running** state, you can **connect** to them from the Instances screen. [Find out](#) how to connect to your instances.

▼ Here are some helpful resources to get you started

- How to connect to your Linux instance
- Amazon EC2: User Guide
- Learn about AWS Free Usage Tier
- Amazon EC2: Discussion Forum

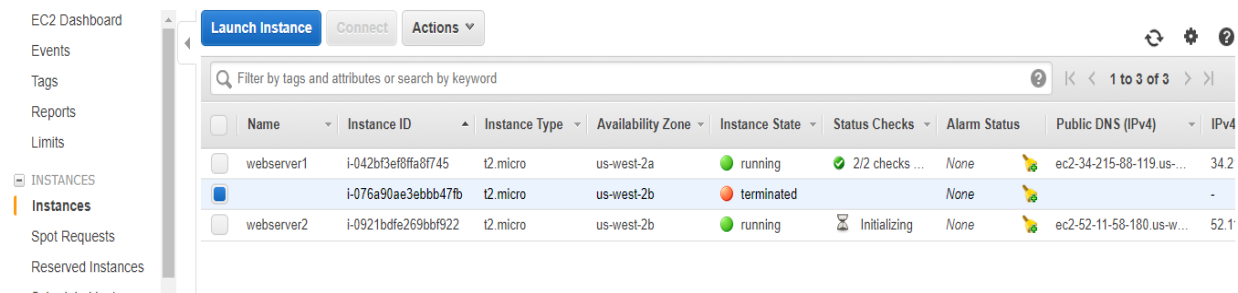
While your instances are launching you can also

As you can see you've created an instance.

Now repeat the same steps(1-4) and create another instance(you can use the same security group 'mylab3' but make sure that you create that instance in US west-2b or US west-2c but not in US west-2a as we've already created one.

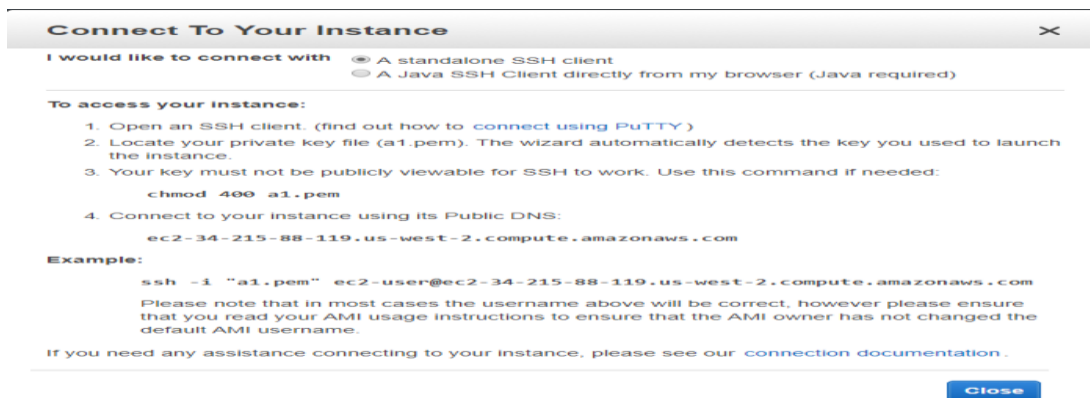
So, far we've created two instances in two availability zones. Now let's connect to each of them and install **httpd web server on it and do some stuff**

Step 5: I named my 1st instance as webserver 1 and 2nd instance and webserver2. Now I'm going to connect it webserver through putty.



	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4
<input type="checkbox"/>	webserver1	i-042bf3ef8ffa87745	t2.micro	us-west-2a	running	2/2 checks ...	None	ec2-34-215-88-119 us-...	34.2
<input checked="" type="checkbox"/>		i-076a90ae3ebbb47fb	t2.micro	us-west-2b	terminated		None		-
<input type="checkbox"/>	webserver2	i-0921bdfe269bbf922	t2.micro	us-west-2b	running	Initializing	None	ec2-52-11-58-180 us-w...	52.1

Select webserver 1 and click on 'connect' button to follow instructions:



Connect To Your Instance

I would like to connect with ☒ A standalone SSH client
☐ A Java SSH Client directly from my browser (Java required)

To access your instance:

1. Open an SSH client. (find out how to [connect using PuTTY](#))
2. Locate your private key file (a1.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:
`chmod 400 a1.pem`
4. Connect to your instance using its Public DNS:
`ec2-34-215-88-119.us-west-2.compute.amazonaws.com`

Example:

```
ssh -i "a1.pem" ec2-user@ec2-34-215-88-119.us-west-2.compute.amazonaws.com
```

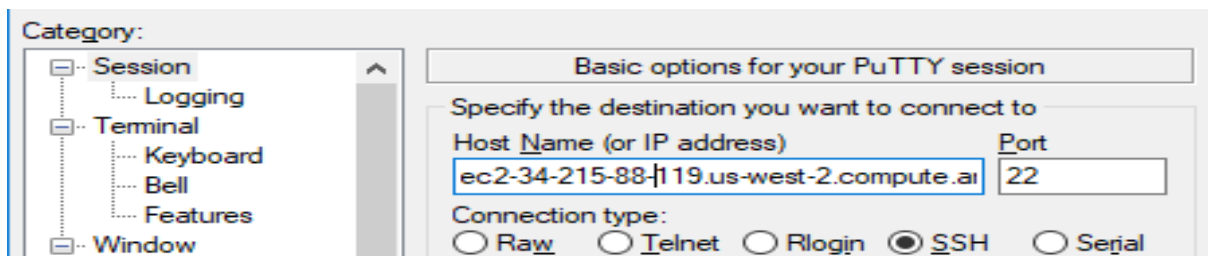
Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our [connection documentation](#).

Close

Now open putty

In the hostname copy and paste the Public DNS in Hostname(or IP address) as shown in the pic just right above.



Category:

- Session
- Logging
- Terminal
- Keyboard
- Bell
- Features
- Window

Basic options for your PuTTY session

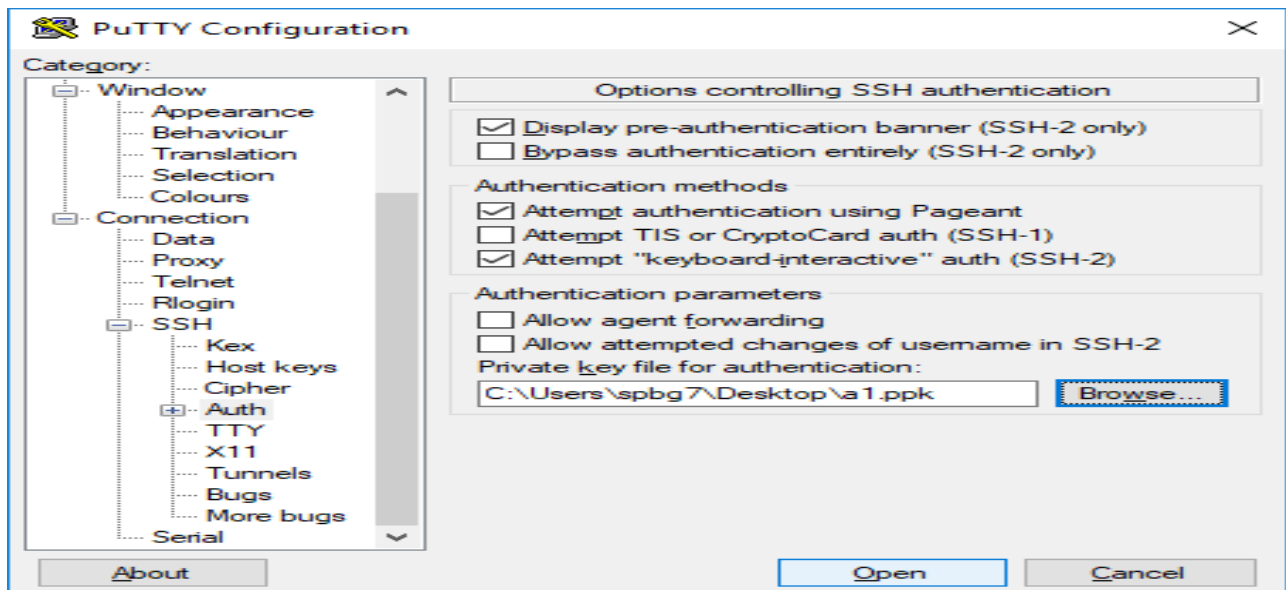
Specify the destination you want to connect to

Host Name (or IP address) Port

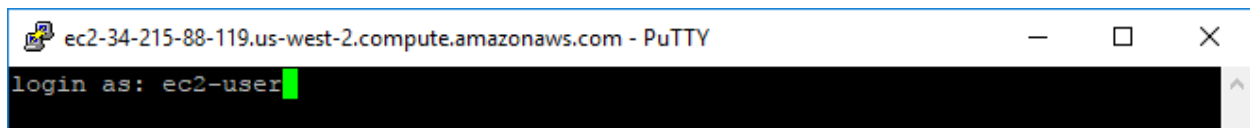
Connection type:

☐ Raw ☐ Telnet ☐ Rlogin ☒ SSH ☐ Serial

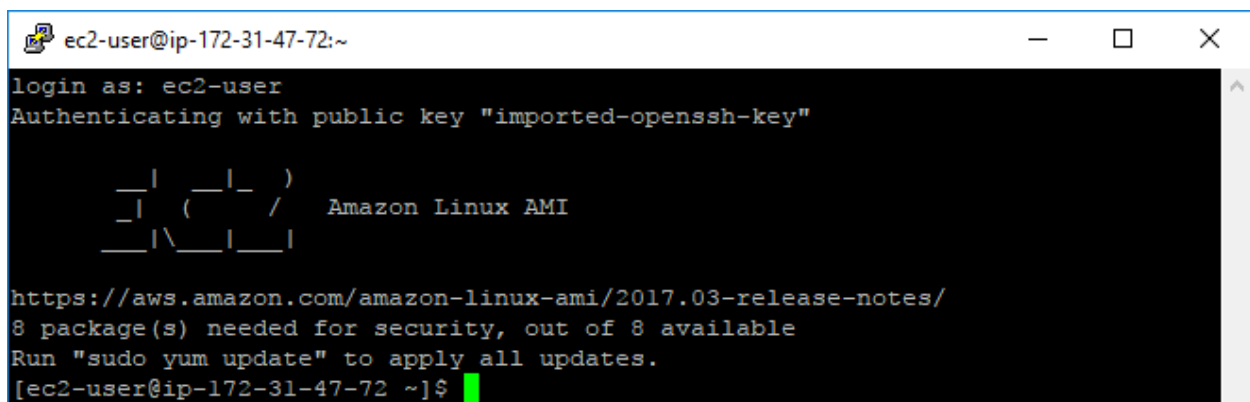
Then on the left column of 'category:' click on SSH ->Auth->browse private key(the one you already created and converted using Puttygen) and click OPEN



Click 'yes' for security alert and now a dark screen pops up and enter login as :ec2-user.Then click enter.



Now you're connected to your instance(webserver1):



Step 6: Install httpd server and create a simple static web page

```
[ec2-user@ip-172-31-47-72 ~]$ sudo yum install httpd
```

```
Complete!
```

```
[ec2-user@ip-172-31-47-72 ~]$
```

Now start the httpd webserver and make sure it is running:

```
ec2-user@ip-172-31-47-72:~
```

```
[ec2-user@ip-172-31-47-72 ~]$ sudo service httpd start
```

```
Starting httpd:
```

```
[ OK ]
```

```
[ec2-user@ip-172-31-47-72 ~]$ sudo chkconfig httpd on
```

```
[ec2-user@ip-172-31-47-72 ~]$ sudo service httpd status
```

```
httpd (pid 2923) is running...
```

Now create a simple webpage index.html

```
ec2-user@ip-172-31-47-72:~
```

```
[ec2-user@ip-172-31-47-72 ~]$ sudo nano /var/www/html/index.html
```

Now an editor will be opened up .Copy and paste this simple HTML script there. Feel free to use if to use if want to create your own

```
<html>
```

```
<body bgcolor='#CC9900'>
```

```
<h1> Same Website</h1>
```

```
<h2> Quotes about Fear </h2>
```

```

```

```
<p> This is web server 1 </p>
```

```
</body>
```

```
</html>
```

```
ec2-user@ip-172-31-47-72:~
GNU nano 2.5.3 File: /var/www/html/index.html

<html>
<body bgcolor="#009900">
<h1> Same Website</h1>
<h2> Quotes about Fear </h2>

<p> This is web server 1 </p>
</body>
</html>

[ Wrote 9 lines ]
Get Help Write Out Where Is Cut Text Justify Cur Pos Prev Page First Line WhereIs Next Mark Text
Exit Read File Replace Uncut Text To Spell Go To Line Next Page Last Line To Bracket Copy Text
```

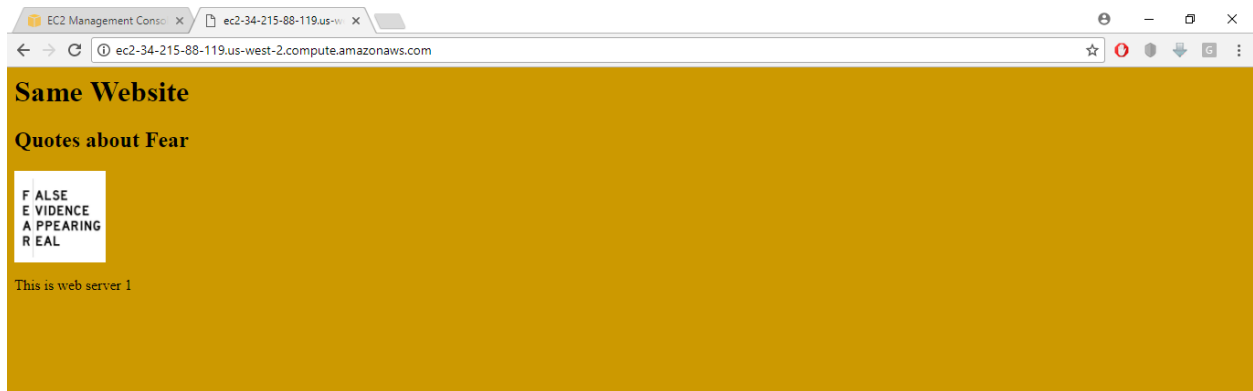
Now click Ctrl O to save and Ctrl X to exit.

Now restart the server

```
ec2-user@ip-172-31-47-72:~
[ec2-user@ip-172-31-47-72 ~]$ sudo nano /var/www/html/index.html
[ec2-user@ip-172-31-47-72 ~]$ sudo service httpd restart
Stopping httpd: [ OK ]
Starting httpd: [ OK ]
[ec2-user@ip-172-31-47-72 ~]$
```

Now copy and paste the public DNS of your instance in a separate browser.

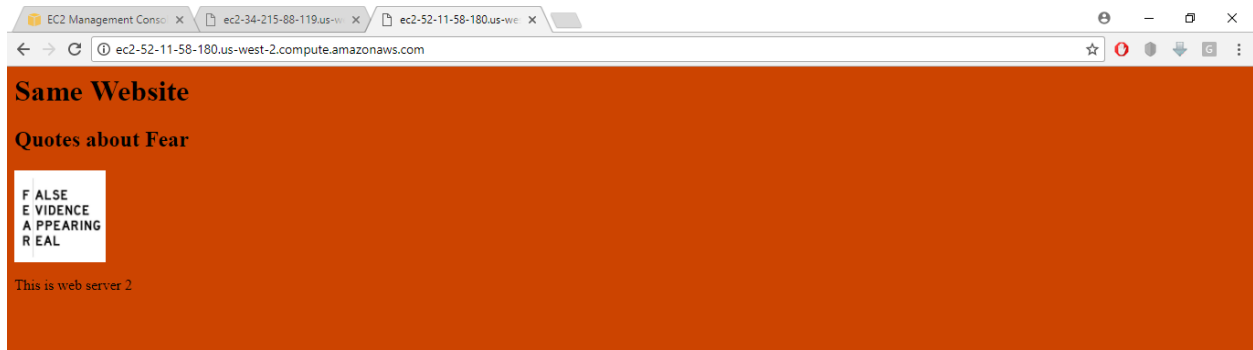
You'll see this



Step 7: Connecting instance 2(webserver 2)

Now repeat the steps 5-6 exactly for webserver 2 except for the HTML code.
Change the content slightly so that there is some difference in appearance
between webserver 1 and webserver2.

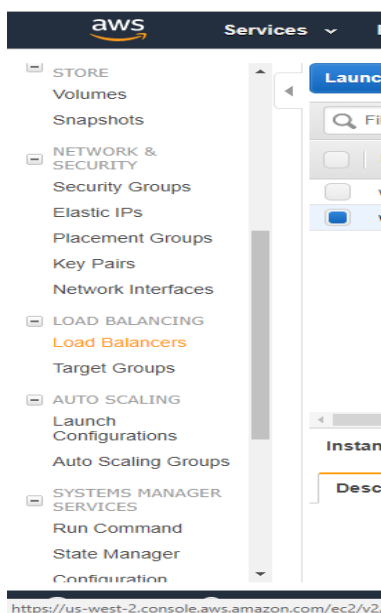
After that load the DNS for webserver2 on separate web browser you'll see this:



If you notice the 2 static web page images,the only change is the background
color and text (“This is web server 2”)

Step 7: Now launch an ELB which directs the traffic through it and distribute
across the instances

Go to EC2 dashboard and on the left-hand side you will find load balancer
column. Now Click on it.



Step 8: Select Classic Load Balancer and click Create

aws

Services

Resource Groups

SRIGANESH PERA

Oregon

Support

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](#)

Application Load Balancer

HTTP
HTTPS

Create

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.

[Learn more >](#)

Network Load Balancer

TCP

Create

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.

[Learn more >](#)

Classic Load Balancer

PREVIOUS GENERATION
for HTTP, HTTPS, and TCP

Create

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

[Learn more >](#)

Cancel

Feedback

English (US)

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Step 9: Name the load balancer and click Next:Assign Security groups

aws

Services

Resource Groups

SRIGANESH PERA

Oregon

Support

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:

My Default VPC (172.31.0.0/16)

Create an internal load balancer:

Enable advanced VPC configuration:

Listener Configuration:

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

Add

Cancel

Next: Assign Security Groups

Feedback

English (US)

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Step 10: Create your own security group or select one if you have one but the setting should be this:

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 security groups to your load balancer. Please select the security groups to assign to this 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:

Description:

Type	Protocol	Port Range	Source
SSH	TCP	22	My IP 202.3.72.230/32
HTTP	TCP	80	Custom 0.0.0.0/0

Step 11: Now click Next: Configure security settings and Next: Configure health checks

In the ping path just keep only the forward slash as show:

Step 4: Configure Health Check

Your load balancer will automatically perform health checks on your EC2 instances and only route traffic to instances that pass the health check. If an instance fails the health check, it is automatically removed from the load balancer. Customize the health check to meet your specific needs.

Ping Protocol:

Ping Port:

Ping Path:

Advanced Details

Response Timeout	<input type="text" value="5"/>	seconds
Interval	<input type="text" value="30"/>	seconds
Unhealthy threshold	<input type="text" value="2"/>	
Healthy threshold	<input type="text" value="10"/>	

Step 12: Now add EC2 instances

Select both the instances and click Next:Add Tags

The screenshot shows the AWS Management Console interface for configuring a Load Balancer. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a user profile 'SRIGANESH PERA' in 'Oregon'. The breadcrumb trail shows the steps: 1. Define Load Balancer, 2. Assign Security Groups, 3. Configure Security Settings, 4. Configure Health Check, 5. Add EC2 Instances (highlighted), 6. Add Tags, and 7. Review.

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-d695eab3 (172.31.0.0/16)

<input type="checkbox"/>	Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
<input checked="" type="checkbox"/>	i-0a3494f425ac6045f	webserver 1	running	launch-wizard-6	us-west-2a	subnet-21fc9156	172.31.32.0/20
<input checked="" type="checkbox"/>	i-03e207c1b469c82f5	webserver 2	running	launch-wizard-7	us-west-2b	subnet-7bd49b1e	172.31.16.0/20

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

☒ Enable Cross-Zone Load Balancing ⓘ
☒ Enable Connection Draining ⓘ 300 seconds


Buttons at the bottom: Cancel, Previous, Next: Add Tags

There are 2 config points in here as well:

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.

Step 13: Now click Review and Create and review all the settings you made and then click Create

 Services ▾ Resource Groups ▾ ★

🔔 SRIGANESH PERA ▾ Oregon ▾ Support ▾

Load Balancer Creation Status

**Your load balancer is being created**

Created security group

Authorized security groups

Created load balancer

Configured health check

Feedback

English (US)

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Step 14: Now click on your Load balancer which redirects to your ELB dashboard!

Load Balancer Creation Status

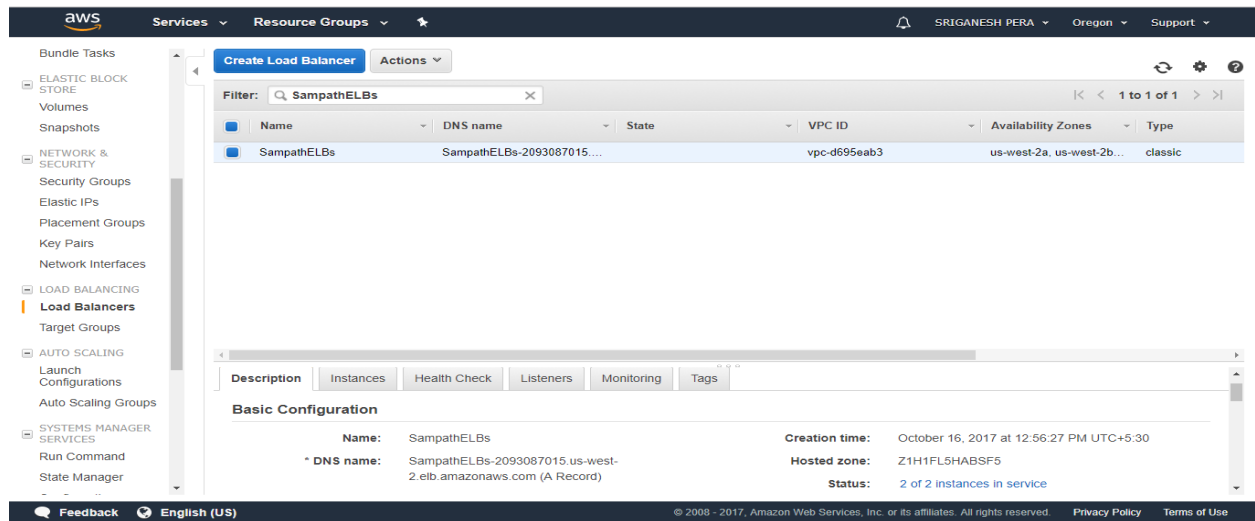


Successfully created load balancer

Load balancer [SampathELBs](#) was successfully created.

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

Close



Now click on Description tab and notice the status: it takes at least 5 mins for the load balancer to register those 2 instances. If load balancer is ready you can see the status 'in service'.

Now copy the DNS name without A record and paste it in your browser and refresh it multiples times to see the magic of ELB and how it routes internet traffic across two instances.



Step 15: Clean up!

Delete your instances and also your load balancer.

Questions to ponder:

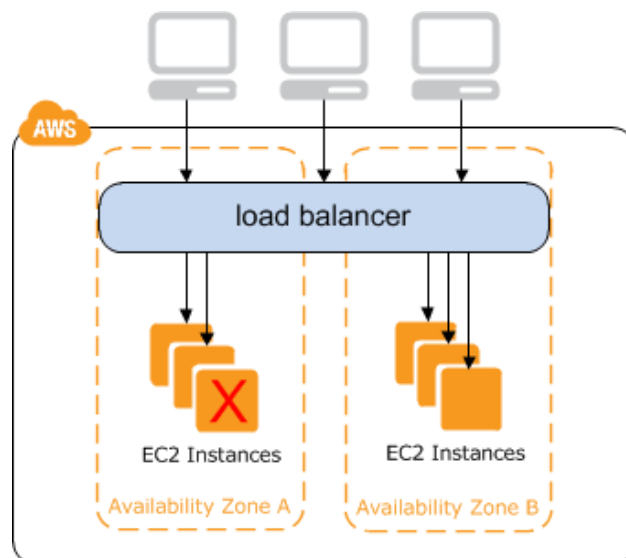
1. What does a load balancer do?
2. Can we set up a load balancer in multi AZ's or in multiple regions?
3. How does an ELB ensure health of instances ?

Elastic Load Balancer:

Elastic Load Balancing supports three types of load balancers:

1. Classic Load Balancers
2. Application Load Balancers.
3. Network load balancers

1. Classic Load balancer:



As you can notice from the figure EC2 instances in various availability zone can be handled by a single load balancer.

A classic load balancer serves as a single point of contact for clients. This increases the availability of your application.

You can **add and remove instances** from your load balancer, without disrupting the overall flow of requests to your application.

Elastic Load Balancing scales your load balancer as traffic to your application changes over time.

A listener checks for connection requests from clients, using the protocol and port that you configure, and forwards requests to one or more registered instances using the protocol and port number that you configure.

You add one or more listeners to your load balancer.

You can configure health checks, which are used to monitor the health of the of the registered instances so that the load balancer can send requests only to the healthy instances.

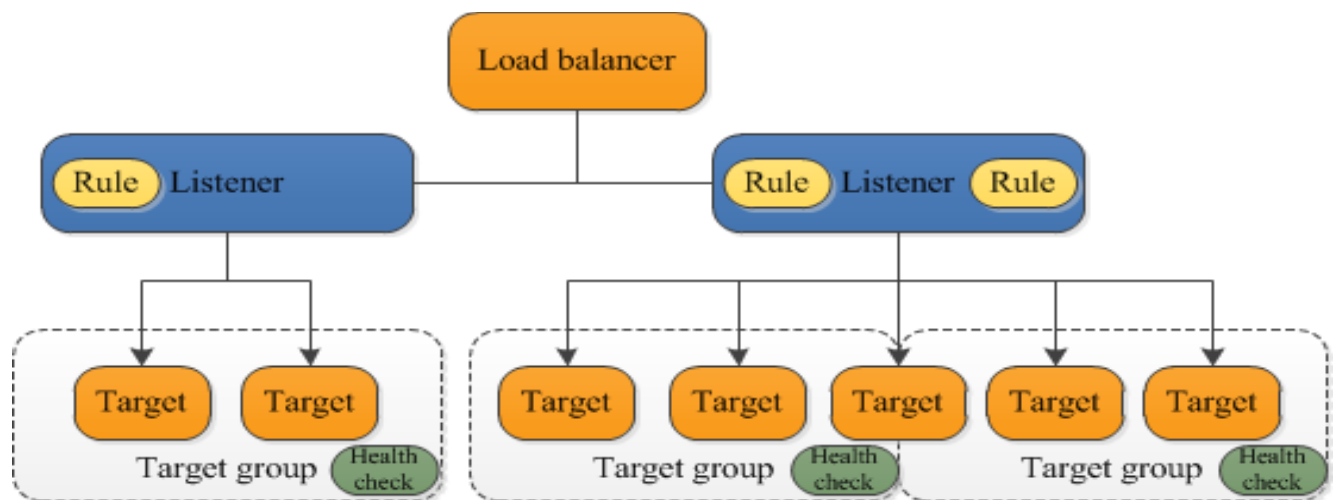
It is important to keep approximately the same number of instances in each Availability Zone registered with the load balancer.

For example, if you have ten instances in Availability Zone us-west-2a and two instances in us-west-2b, the requests are distributed evenly between the two Availability Zones. Thus, the two instances in us-west-2b serve the same amount of traffic as the ten instances in us-west-2a. Instead, you should have six instances in each Availability Zone.

2.Application load balancing:

An Application Load Balancer functions at the application layer of OSI model. The load balancer makes routing decisions based on the content of the application traffic in the HTTP messages.

The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. This increases the fault tolerance of your applications. Elastic Load Balancing detects unhealthy targets and routes traffic only to healthy targets.



Architecture of Application load balancer:

The load balancer serves as the single point of contact for clients. You add one or more listeners to your load balancer.

A listener checks for connection requests from clients, using the protocol and port that you configure, and forwards requests to one or more target groups.

Each rule specifies a target group, condition, and priority. When the condition is met, the traffic is forwarded to the target group. You must define a default rule for each listener, and you can add rules that specify different target groups based on the content of the request (also known as content-based routing).

Each target group routes requests to one or more registered targets, such as EC2 instances, using the protocol and port number that you specify.

You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group that is specified in a listener rule for your load balancer.

Benefits of using an application load balancer:

1. Support for path-based routing. You can configure rules for your listener that forward requests based on the URL in the request.

For eg: to the URL google.com/images and google.com/videos I can configure the listener to forward requests to those 2 URL's based on my requirements

2. Support for containerized applications. Amazon EC2 Container Service (Amazon ECS) can select an unused port when scheduling a task and register the task with a target group using this port. This enables you to make efficient use of your clusters.
3. Support for monitoring the health of each service independently, as health checks are defined at the target group level and many CloudWatch metrics are reported at the target group level. Attaching a target group to an Auto Scaling group enables you to scale each service dynamically based on demand.

4. Access logs contain additional information and are stored in compressed format.
5. Improved load balancer performance.

Network Load Balancer :

A Network Load Balancer functions at the fourth layer of the Open Systems Interconnection (OSI) model. It can handle millions of requests per second. After the load balancer receives a connection request, it selects a target from the target group for the default rule. It attempts to open a TCP connection to the selected target on the port specified in the listener configuration.

When you enable an Availability Zone for the load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone. Each load balancer node for your Network Load Balancer distributes traffic across the registered targets in its Availability Zone only. If you enable multiple Availability Zones for your load balancer, this increases the fault tolerance of your applications. For example, if all targets in one Availability Zone are unhealthy, we remove the IP address associated with the subnet from DNS, but the load balancer nodes in the other Availability Zones are still available to route traffic. If a client doesn't honor the time-to-live (TTL) and sends requests to the IP address after it is removed from DNS, the requests fail.

Bottomline:

Classic Load Balancer 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

You can create up to twenty (20) load balancers per region, which includes both Classic and Application Load Balancers and should you require more you can request Amazon Tech support.