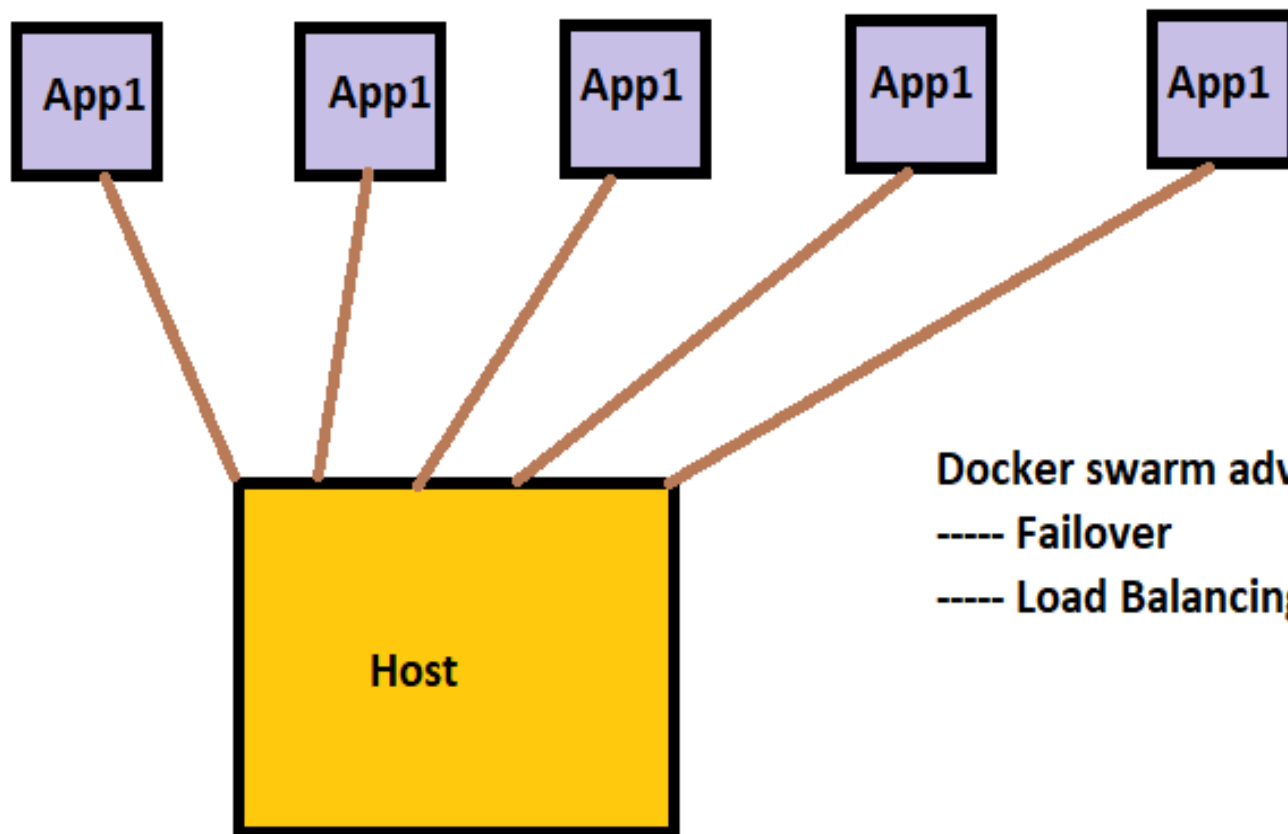


# Docker Swarm

# Docker swarm

- Docker Swarm is a clustering and scheduling tool for Docker containers.
- With Swarm, IT administrators and developers can establish and manage a cluster of Docker nodes as a single virtual system.
- Clustering is an important feature for container technology, because it creates a cooperative group of systems that can provide **redundancy** and **Load balancing**.
- A Docker Swarm cluster also provides administrators and developers with the ability to add or subtract container iterations as computing demands change.

All containers hosting same application



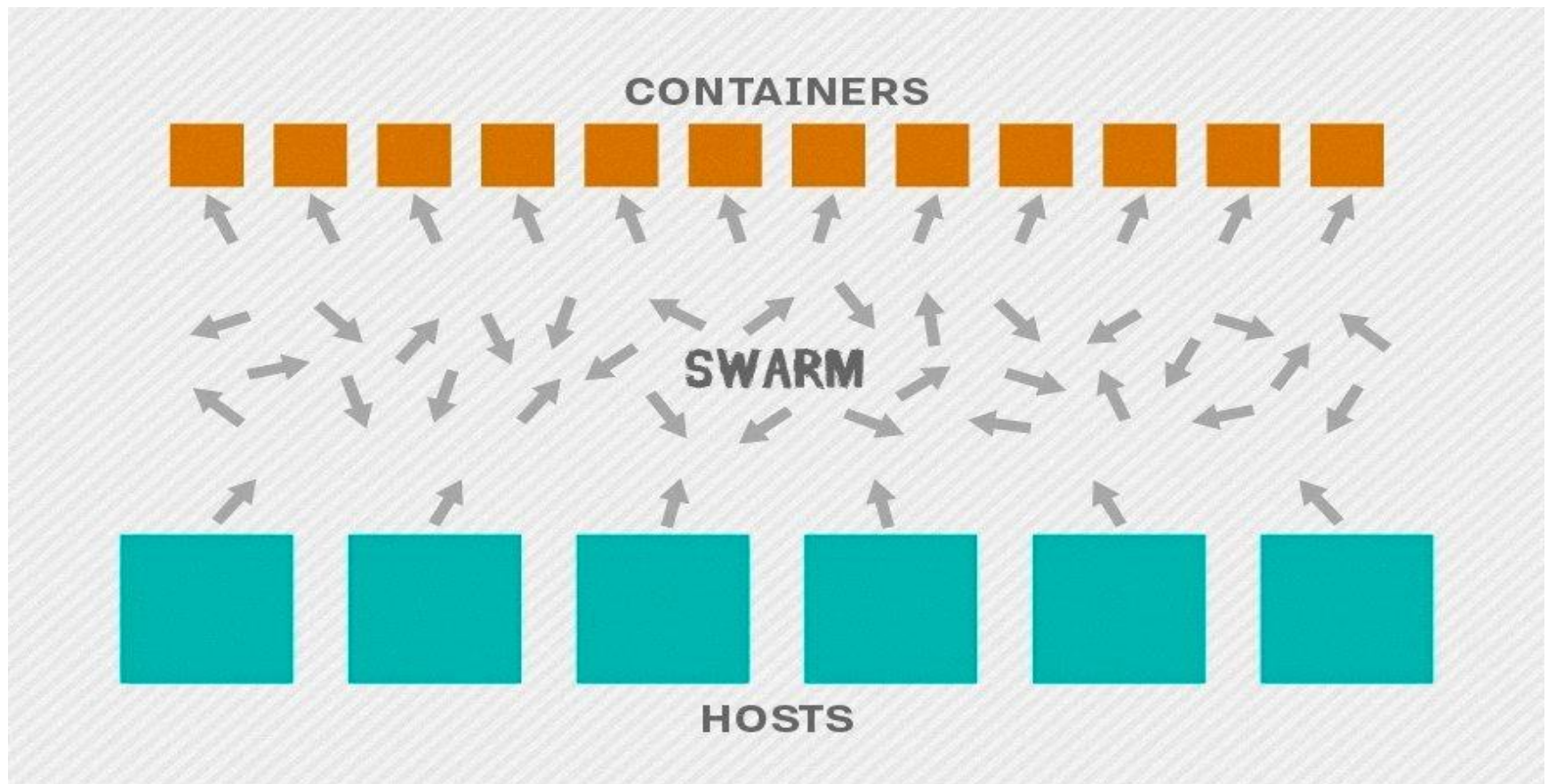
Docker swarm advantages

----- Failover

----- Load Balancing

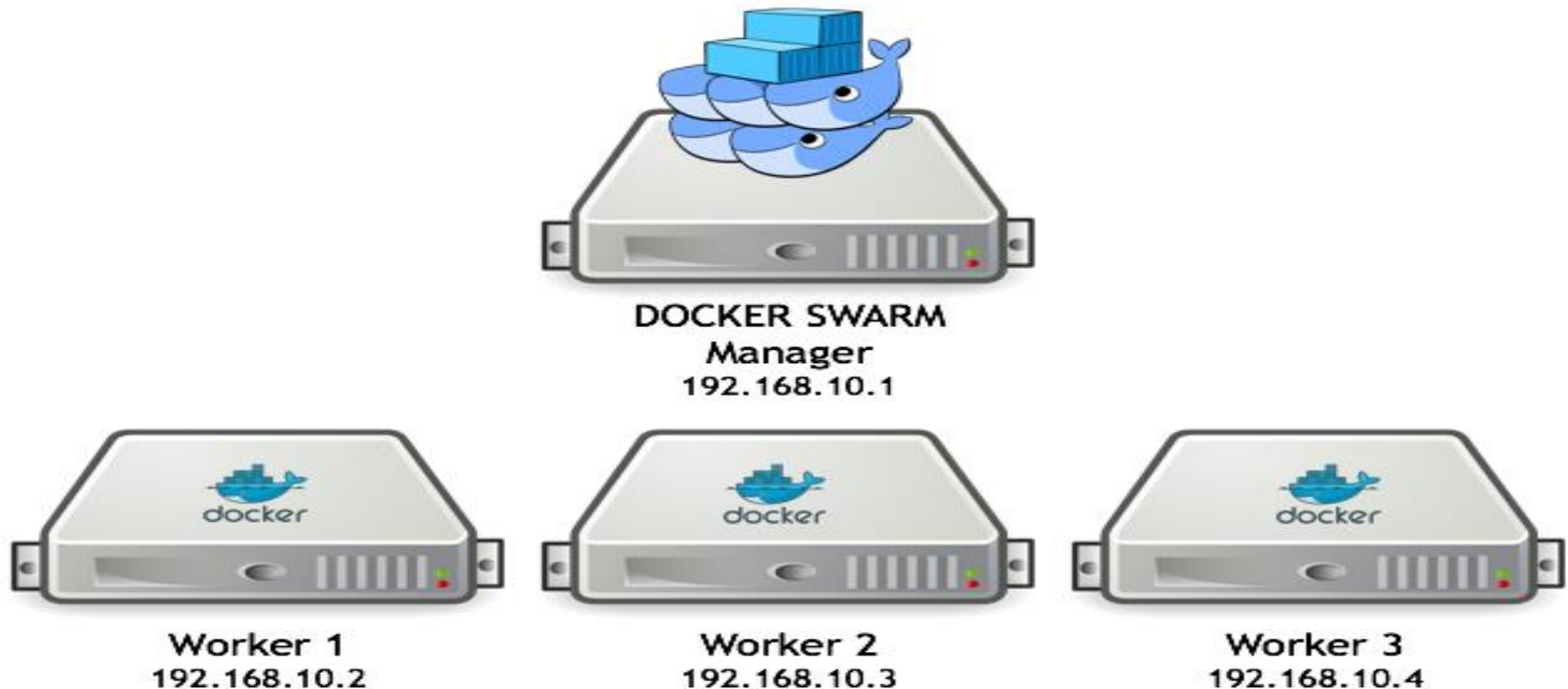
# Docker swarm

Swarm mode also exists natively for Docker Engine, the layer between the OS and container images. Multiple host for host level redundancy.



# Swarm management

- An IT administrator controls Swarm through a swarm **manager**, which orchestrates and schedules containers. The swarm manager allows a user to create a primary manager instance and multiple replica instances in case the primary instance fails. In Docker Engine's swarm mode, the user can deploy manager and **worker** nodes at runtime



## Hands on –1 (Creating Swarm Cluster)

Requirement: 2 Linux system (1 for manager and 1 for worker)

1) Install docker in both system

2) In manager

```
# docker swarm init --advertise-addr=172.16.32.65(manager private IP)
```

--- copy the output

3) In worker

paste the copied content

```
# docker swarm join --token .....172.16.32.65:2371
```

4) In manager

```
# docker node ls
```

both node status : ready

## Hands on –2 (Deploy app in docker swam)

How application works actually in swarm?

- Container on the cluster are deployed using services on docker swarm
- A service is a long running docker container that can be deployed to any node worker

1) Create a new image using docker file where web service is already configured.

2) **In manager**

```
# docker service create --name apache --replicas 5 -p 83:80 newubuntu
```

```
# docker service ls
```

```
# docker ps
```

now open web browser

managerpublic-ip:83

workerpublic-ip:83

**In worker**

```
# docker ps
```

## Hands on –3 ( Check the availability)

### **In manager**

```
# docker rm -f $(docker ps -a -q)
```

now open web browser

mangerpublic-ip:83

### **In worker**

```
# docker rm -f $(docker ps -a -q)
```

now open web browser

workerpublic-ip:83

### **In manager**

```
# docker ps
```

some container launched automatically

### **In worker**

```
# docker ps
```

some container launched automatically



# Hands on –4 ( Scaling)

## Scale down

In manager

```
# docker service scale apache=2
```

```
# docker service ls
```

```
# docker ps
```

In worker

```
# docker ps
```

## Scale up

In manager

```
# docker service scale apache=10
```

```
# docker service ls
```

```
# docker ps
```

In worker

```
# docker ps
```

# Hands on -5 ( Leaving node)

## In worker

```
# docker swarm leave
```

wait for few moment

## In manager

```
# docker node ls
```

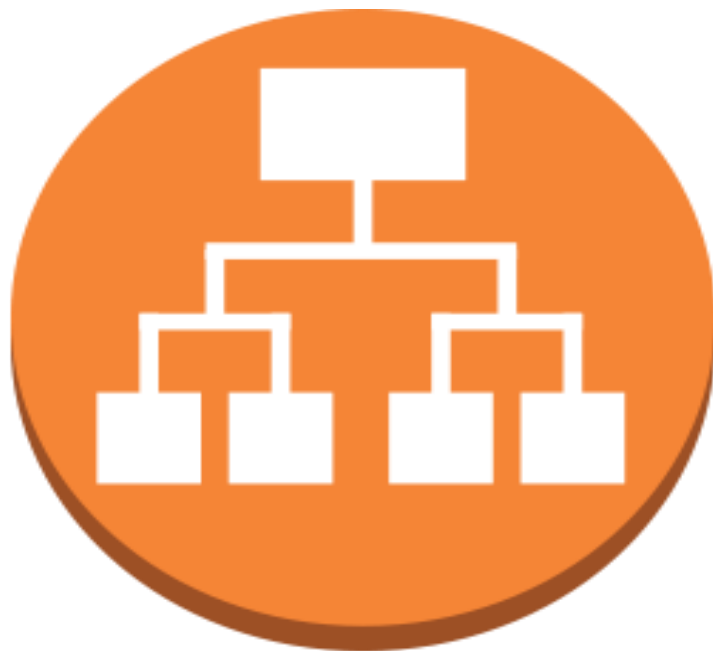
host 2 : down

host 1: ready

```
# docker swarm leave --force
```

## Deleting swarm service

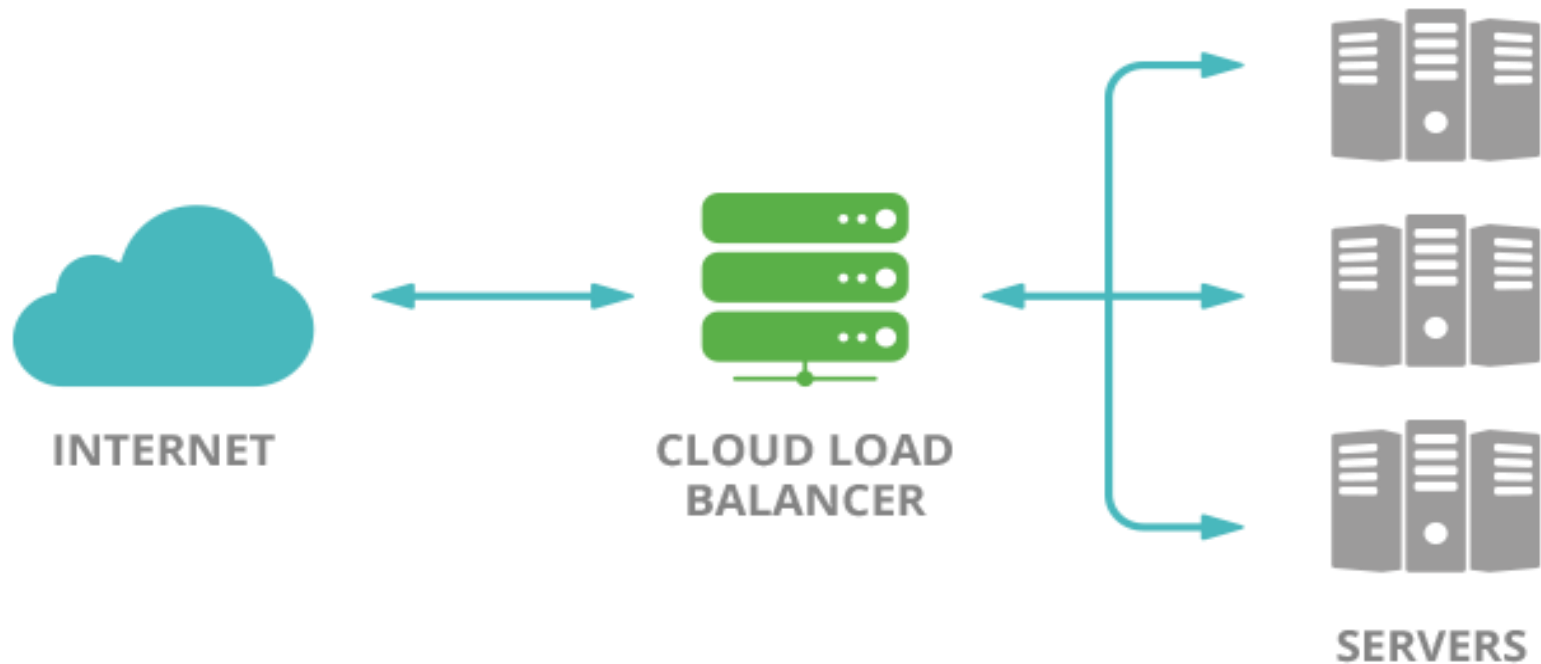
```
# docker service rm apache
```



**Elastic  
Load Balancer**

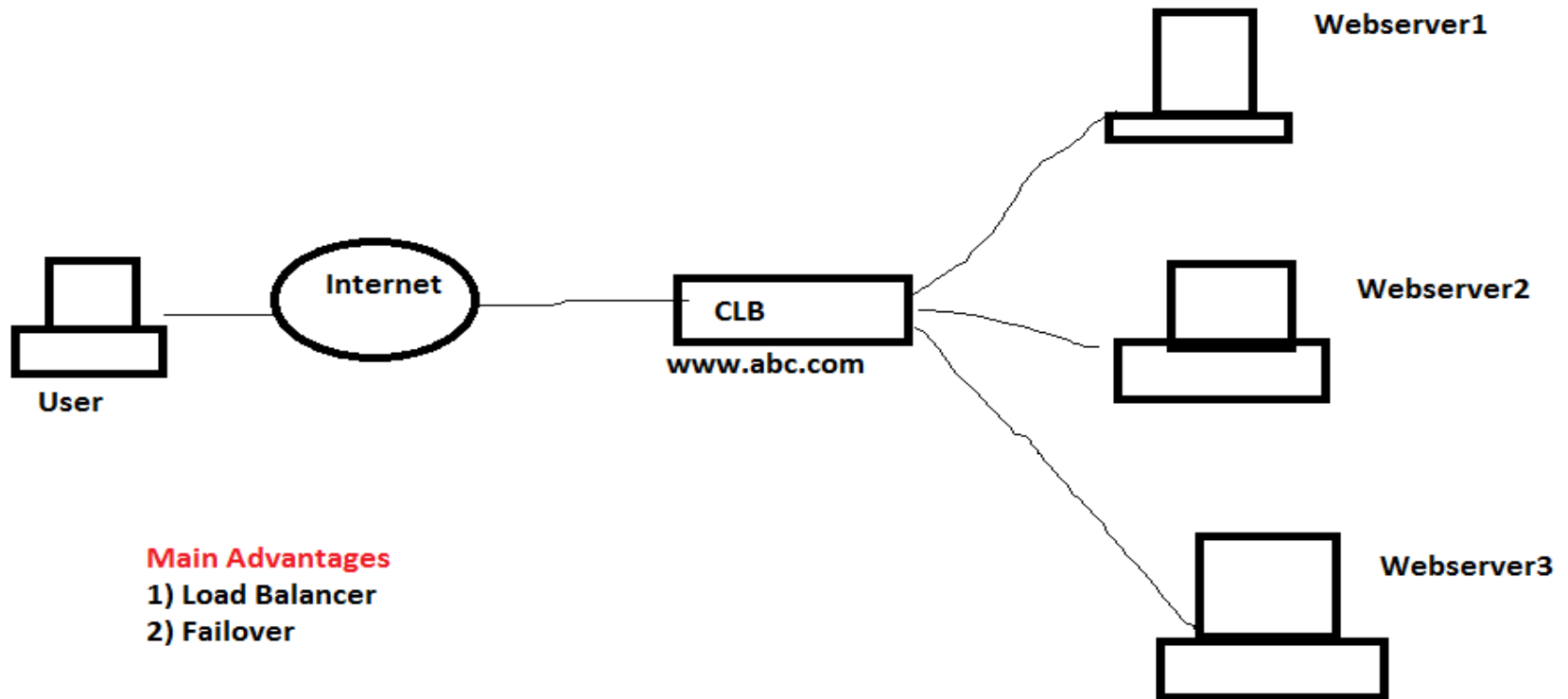
# ELB Advantages

- 1) Load balancer
- 2) Failover
- 3) Any time any number of instances can be added or removed



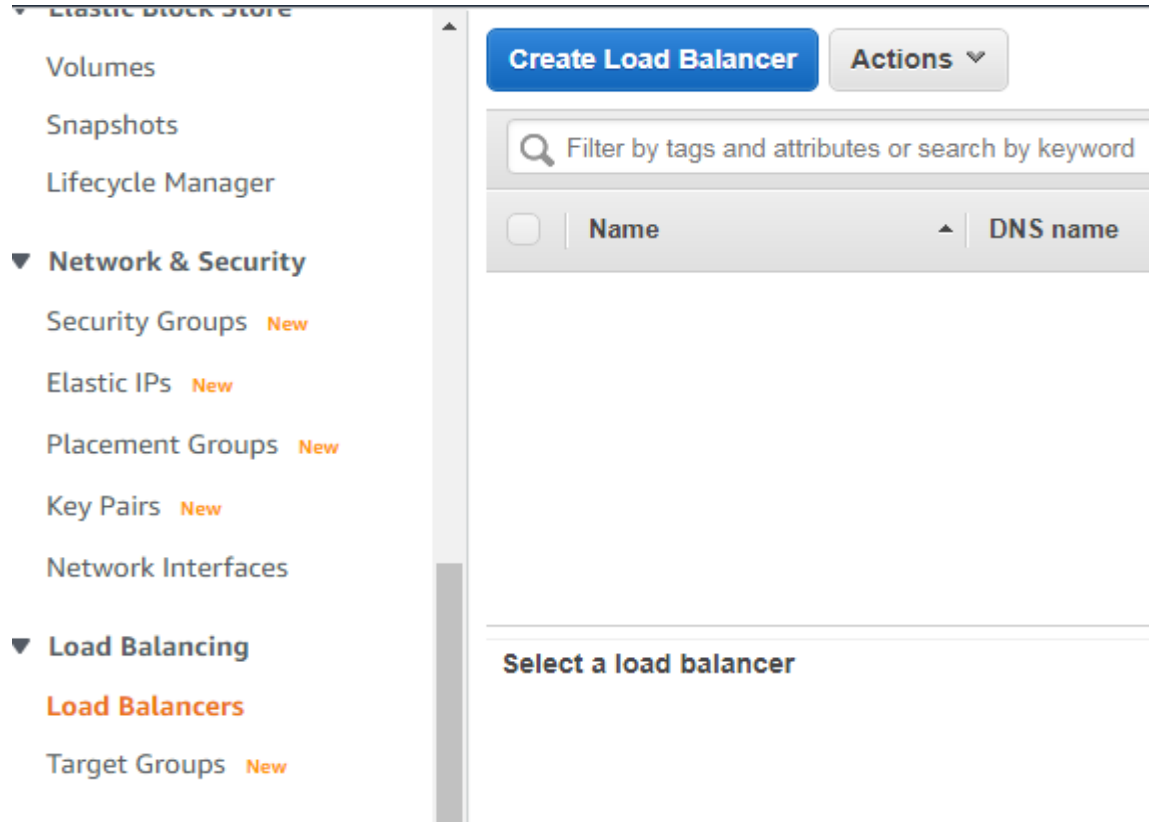
# Classic Load Balancers

ELB: Classic Load Balancer



# Classic Load Balancers Steps

- 1) Create 3 Instances and Configure different web page
- 2) Click on load balancer –Create load balancer --



# Classic Load Balancers Steps

## Select Classic load balancer

### 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



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 and visibility features targeted at application architectures, including microservices.

#### Network Load Balancer



Create

Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your application. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per

#### 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-Classic network.

[Learn more >](#)

# Classic Load Balancers Steps

## Give load balancer name

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:

Create LB inside:

Create an internal load balancer: ☐ [\(what's this?\)](#)

Enable advanced VPC configuration: ☐

Listener Configuration:

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port	
<input type="text" value="HTTP"/>	<input type="text" value="80"/>	<input type="text" value="HTTP"/>	<input type="text" value="80"/>	<input type="button" value="x"/>
<input type="button" value="Add"/>				

[Cancel](#)

[Next: Assign Security Groups](#)



## Classic Load Balancers Steps

Select Security group(SSH or RDP and HTTP allowed)

### 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 balance. This can be changed at any time.

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

	Security Group ID	Name	Description
<input type="checkbox"/>	sg-08c1caa50b77360f1	all traffic	launch-wizard-2 created 2020-08-13T18:34:27.224+05:30
<input type="checkbox"/>	sg-9e8b8bfc	default	default VPC security group
<input type="checkbox"/>	sg-0b82a58710c602670	EFS-SG	abc
<input checked="" type="checkbox"/>	sg-06b9f816d6a387a04	Linux-SG	for project1
<input type="checkbox"/>	sg-0c597cc5f9b72976b	linux-Sg-aws	launch-wizard-2 created 2020-08-18T21:33:14.152+05:30
<input type="checkbox"/>	sg-0bd13efa05570edf1	windows -sg	launch-wizard-1 created 2020-08-06T08:48:12.876+05:30

# Classic Load Balancers Steps

Keep default value

1. Define Load Balancer

2. Assign Security Groups

3. Configure Security Settings

4. Con

## Step 4: Configure Health Check

Your load balancer will automatically perform health checks on your EC2 instances and on the load balancer. Customize the health check to meet your specific needs.

Ping Protocol

HTTP

Ping Port

80

Ping Path

/index.html

### Advanced Details

Response Timeout



5

seconds

Interval



30

seconds

Unhealthy threshold



2



Healthy threshold



10



## Classic Load Balancers Steps

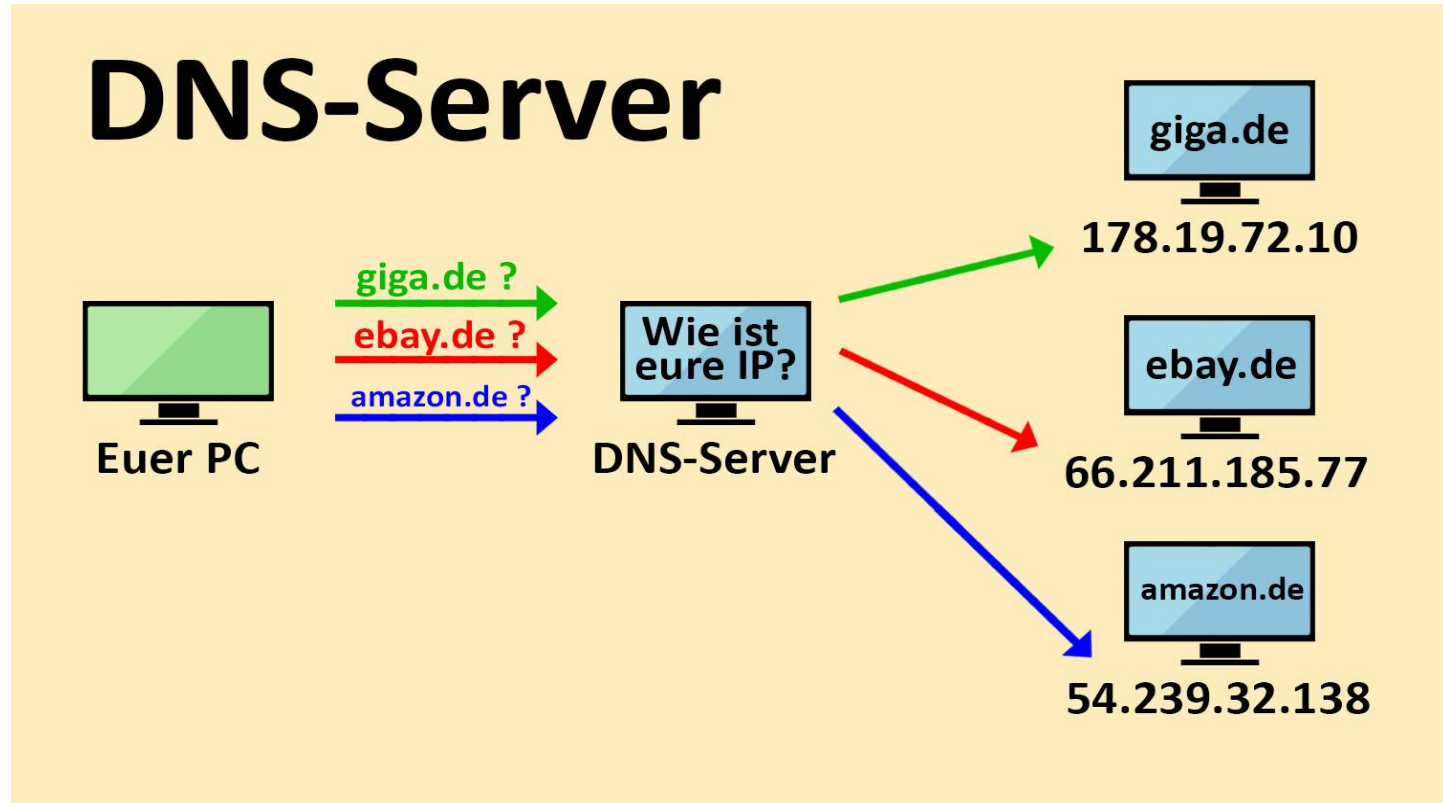
Next –add required number of instances –next-next--create

3) After Creating---scroll down –click on target—wait and refresh to check target status –changed from outservice to inservice

4)Description –copy dns name and paste in browser tab—keep refreshing



# DNS



*DNS is used to resolve host name to IP address and IP address to host name.*

# Route53 Lab –1 Register domain name

1) Register your domain name(deepak.tk) in freenom.com



Search input field: YourDomain2017

Check Availability button

Get one of these domains. They are free!

yourdomain2017 <b>.tk</b>	• FREE	GBP 0. <sup>00</sup>	Get it now!
yourdomain2017 <b>.ml</b>	• FREE	GBP 0. <sup>00</sup>	Get it now!
yourdomain2017 <b>.ga</b>	• FREE	GBP 0. <sup>00</sup>	Get it now!
yourdomain2017 <b>.cf</b>	• FREE	GBP 0. <sup>00</sup>	Get it now!
yourdomain2017 <b>.gq</b>	• FREE	GBP 0. <sup>00</sup>	Get it now!

# Map Freenom domain name with AWS

2) AWS console-services-Route53-DNS management-Hosted zone-create hosted zone-

Give domain name – deepak.tk ----- create

4) Now select NS value -- copy all NS value one by one and paste into –  
freenom.com –services—my domain– manage domain—management tool –  
name servers – use custom nameserver –paste here one by one ---change name  
servers

## Route53 Lab –2 Map IP with Domain name

- 1) Launch one Instance and configure web service there
- 2) Route53– Hosted Zone – open registered domain name – Create Record set – fill the detail – type – A – In IP address : Instance public IP –ok
- 3) Now copy domain name and paste in Browser

How to map multiple instance port with domain name

Eg: IP:82 , IP:83, IP:84 etc.

Configure multiple classic load balancer with these all port number and map load balancer dns with domain name.