



By  
Reyaz Shaik

# EXPECTATIONS

## PART - : AWS Services

- ❖ Introduction to Cloud Computing
- ❖ AWS Overview
- ❖ IAM
- ❖ Networking
- ❖ VPC
- ❖ Ec2
- ❖ RDS
- ❖ S3
- ❖ CloudWatch
- ❖ Auto Scaling



By  
Reyaz Shaik

# COURSE OUTLINE

1. **Cloud Computing**
2. **Client Server Architecture**
3. **Amazon Web Service Overview**
4. **High Availability Architecture**
5. **AWS Sign UP**
6. **MFA Configuration**
7. **AWS CLI**
8. **AWS VPC Theory & Practical**
9. **Cloud Trail**
10. **Simple Email Service (SES)**
21. **VPC Scenario Project**
11. **EC2 Theory & Practical**
12. **Security, Storage & Networking**
13. **S3 Theory & Practical**
14. **RDS Theory & Practical**
15. **IAM Theory & Practical**
16. **Route53 Theory & Practical**
17. **Cloud Watch Theory & Practical**
18. **ELB Theory & Practical**
19. **Auto Scaling Theory & Practical**
20. **Route53 DNS Failover Project**

# Client Server Architecture

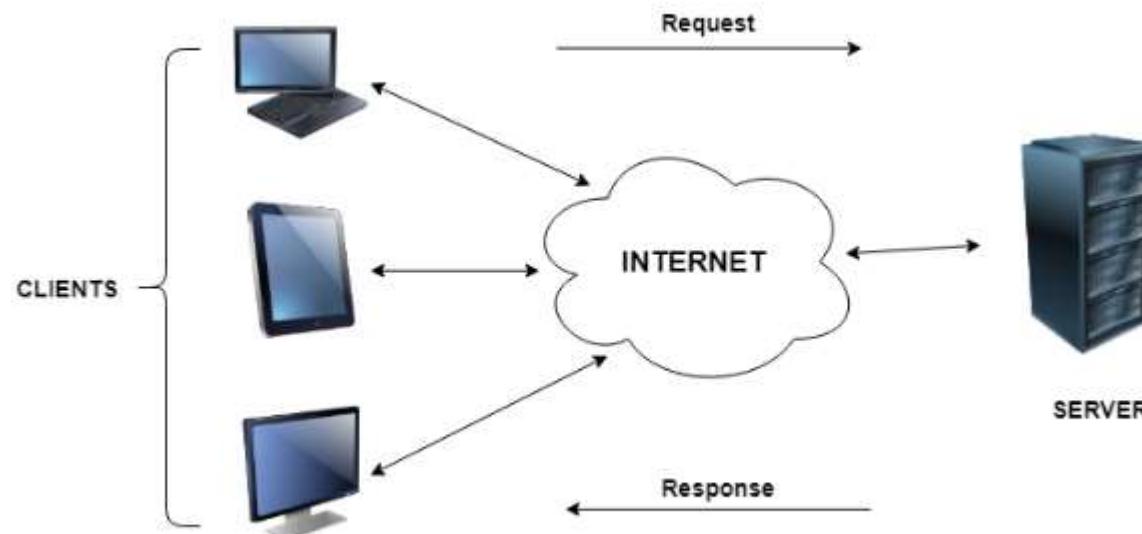
By  
Reyaz Shaik

# Client-Server

- The **clients requests** a resource and the **server respond** with that resource.
- A server may serve multiple clients at the same time while a client is in contact with only one server.
- There are two different structures :
  - Two - Tier Client/Server Structure
  - Three - Tier Client/Server Structure

# Two – Tier Client/Server Structure

- The two tier architecture primarily has two parts, a **client tier** and a **server tier**.
- The client tier sends a **request** to the server tier and the server tier **responds** with the desired information.
- An example of a two tier client/server structure is a **web server**. It returns the required web pages to the clients that requested them.



# Advantages of Two - Tier Client/Server Structure

Some of the advantages of the two-tier client/server structure are:

- This structure is quite easy to maintain and modify.
- The communication between the client and server in the form of request response messages is quite fast.

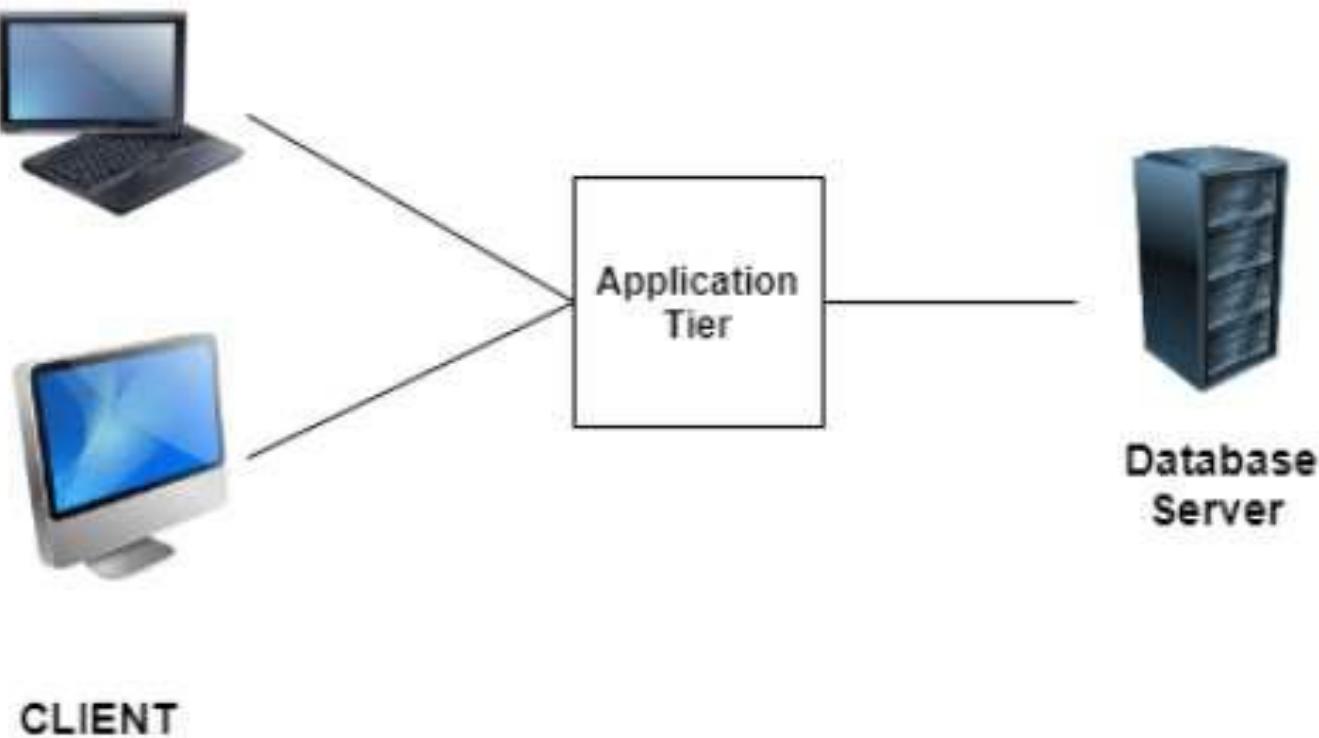
## Disadvantages of Two - Tier Client/Server Structure

- If the client nodes are increased beyond capacity in the structure, then the server is not able to handle the request overflow and performance of the system degrades.

# Three - Tier Client/Server Structure

- The three tier architecture has three layers namely **client**, **application** and **data layer**.
- **The client layer** is the one that requests the information. In this case it could be the GUI, web interface etc.
- **The application layer** acts as an interface between the client and data layer. It helps in communication and also provides security.
- **The data layer** is the one that actually contains the required data.

# Three – Tier Client/Server Structure



# Advantages of Three - Tier Client/Server Structure

Some of the advantages of the three-tier client/server structure are:

- The three tier structure provides much better service and fast performance.
- The structure can be scaled according to requirements without any problem.
- Data security is much improved in the three tier structure.

## Disadvantages of Three - Tier Client/Server Structure

- Three - tier client/server structure is quite complex due to advanced features.



Simple Isn't it?

Now i know

What happens when you type a URL in a browser and hit enter?



# What happens when you type a URL in a browser and hit enter

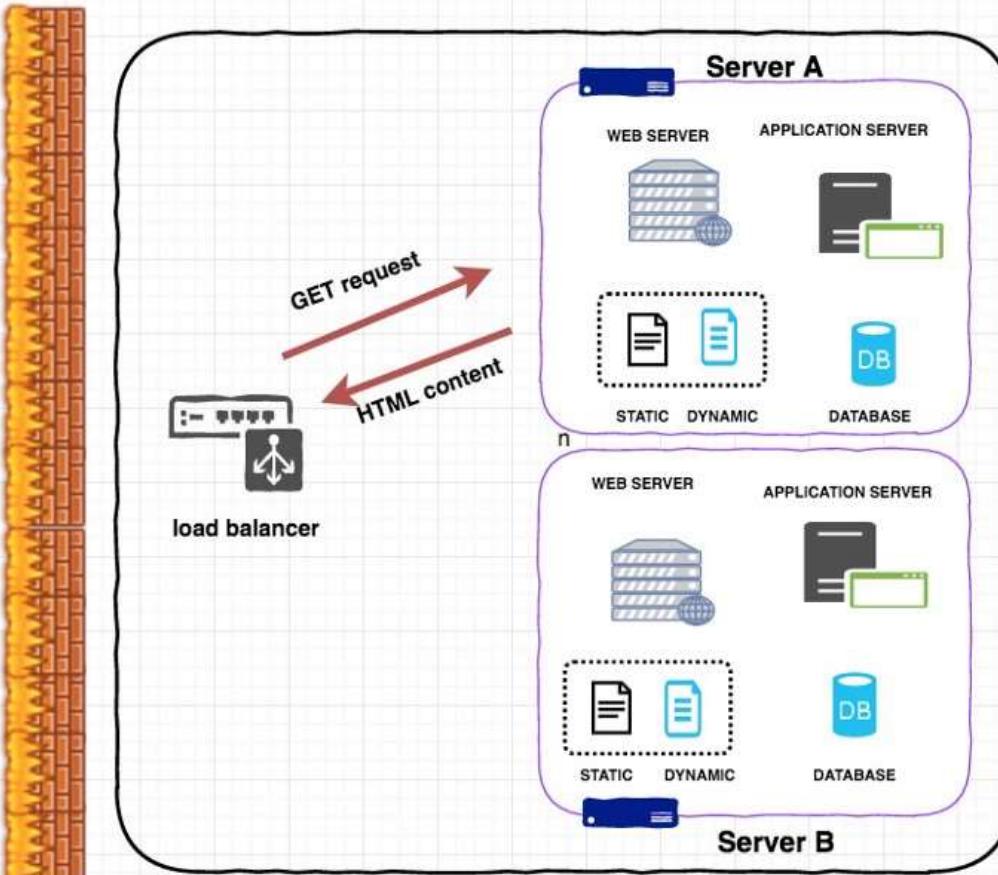
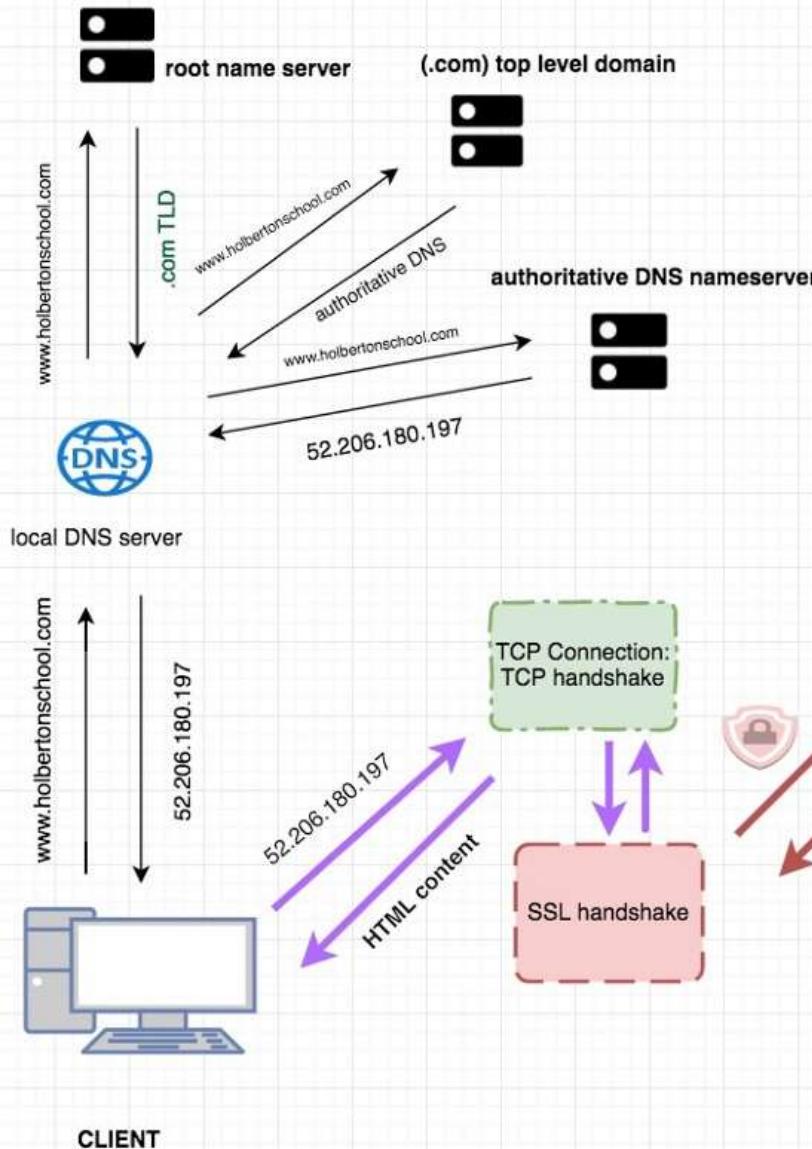
## Topics covered:

- DNS Lookup
- TCP/IP
- HTTPS/SSL/TLS
- **Server:** Firewall, Load Balancer, Web Server, Application Server, Database

www.holbertonschool.com

✓ A RECORD    DOMAIN NAME    TOP LEVEL DOMAIN

DNS A Record: points to server IP 8.8.8.8



- The **internet** is made up of a **network of computers** connected to each other
- In order to connect and **communicate between computers**, they must follow a set of rules, **Internet Protocol(IP)**, that govern how data is transmitted over a network.
- Every machine on the web has a unique identifier to distinguish from one another. It's similar to having a telephone number or a physical address.

- A typical IP address(IPv4) follows the format of 4 sets of numbers between 0–255. xxx.xxx.xxx.xxx.
- There are a total of 4,294,967,296 **IPv4** addresses available and 340,282,366,920,938,463,463,374,607,431,768,211,456 possible **IPv6** addresses.
- Having to look for the IP address when you want to visit a webpage will take a lifetime.
- So we just type in the domain name we're familiar with and let the magical system that is the DNS to take care of the rest :)

ONE DOES NOT SIMPLY

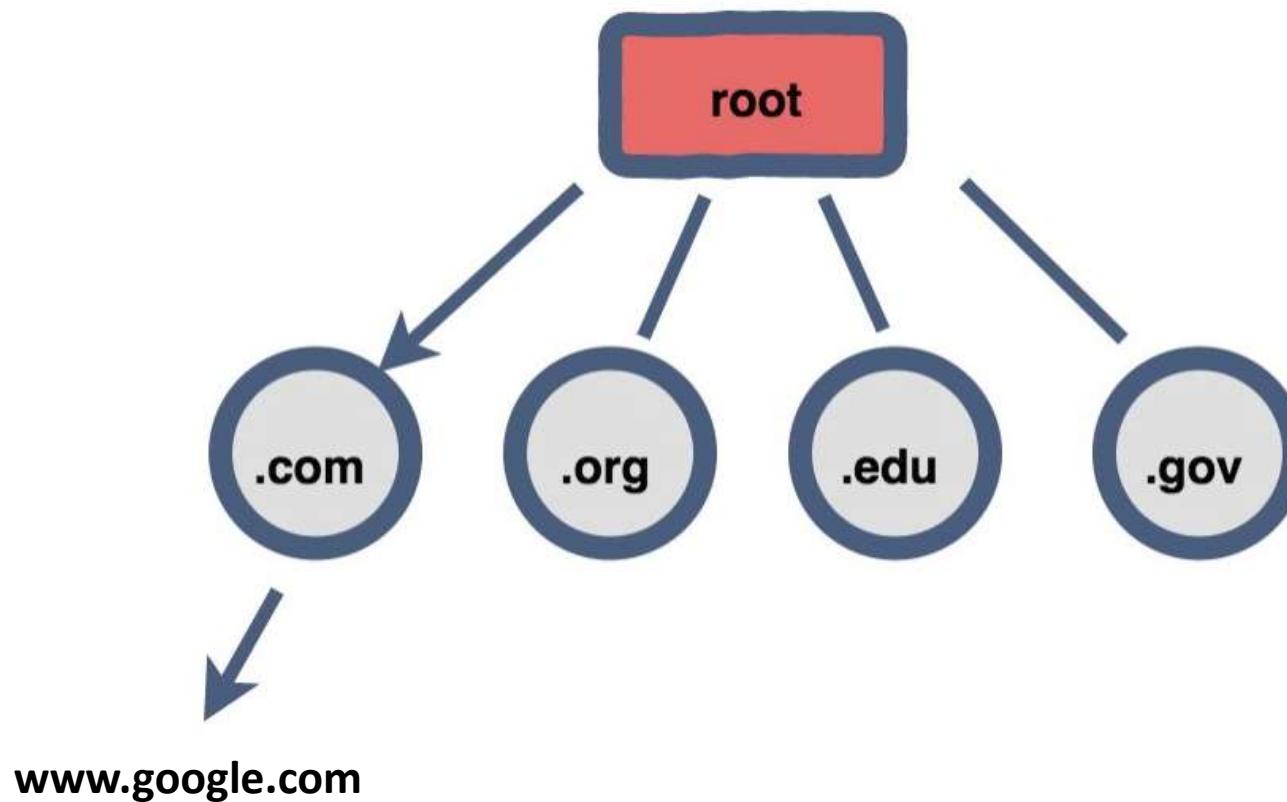


# DNS(Domain Name System)

- The Domain Name System is created to **keep track of IP addresses** for us so we can enter human-readable addresses in our browser's URL bar instead.
- To translate from **domain name to IP address**.
- When you type [www.google.com](http://www.google.com) in your web browser and hit enter, the request will be forwarded to a DNS server. DNS server will then perform a ***DNS lookup*** to locate the corresponding IP address
- Nslookup

- DNS uses a **client/server architecture** and the **DNS servers are organized in a hierarchical and distributed fashion**.
- In order to connect and communicate between computers, they must follow a set of rules, Internet Protocol(IP), that govern how data is transmitted over a network.

- And the route in which we take will look similar to a **upside down tree-like structure**



# Steps to resolving a domain name

1. Web browser and OS will first **check whether the domain is in their cache. If yes, done**
2. The web browser then will send a request to a **DNS resolver**. A DNS resolver is a local server with a central database of DNS nameservers. This DNS resolver will be hosted with your ISP
3. The resolver will first check its cache. If the IP address for google.com isn't in its cache it will forward the query **recursively up the to the root servers**,
4. Down to the Top Level Domain (TLD) of google.com( .com would be the TLD in this case), and then down to the authoritative name servers responsible for [www.google.com](http://www.google.com).

**DNS system is organized in an upside down tree-like structure right?**

We will go to the top and search downward. First stop is the root servers.

Root servers respond with address to the **.com** Top-Level Domain(TLD). Top-level domain just refers to the last chunk of a domain name after the dot symbol. Here we go to the .com TLD.

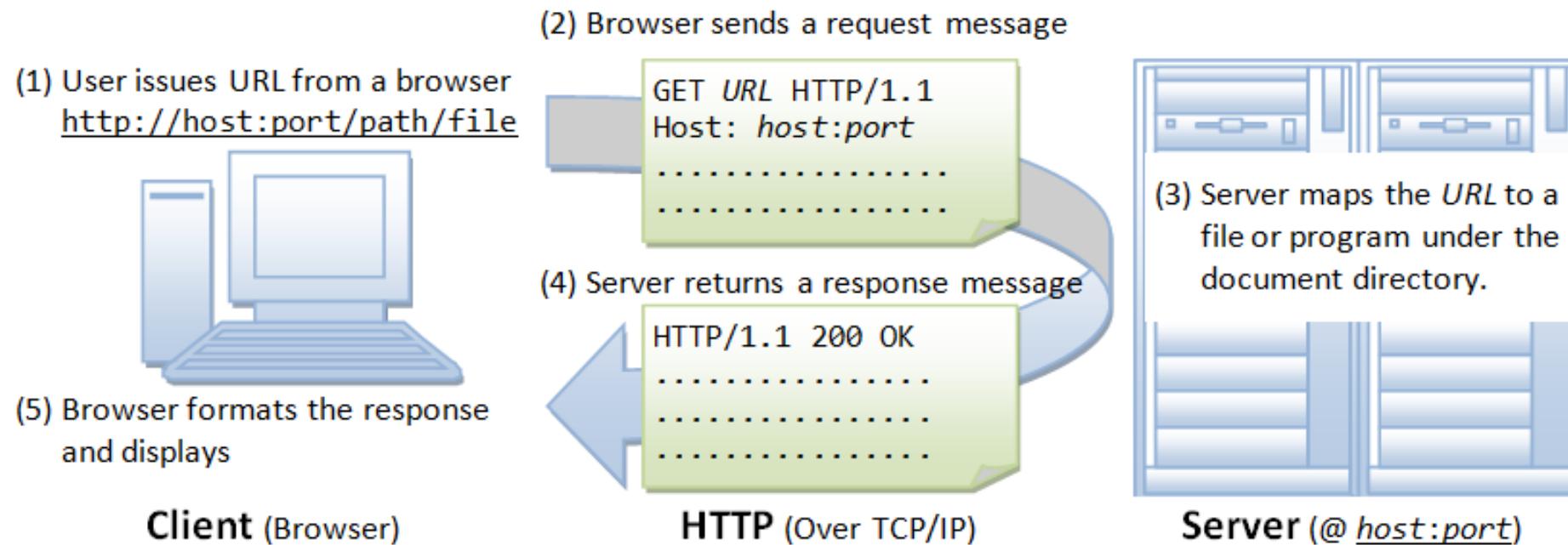
The resolver then queries .com servers for the **authoritative name servers** of our domain, google.com

The authoritative name servers are the ones with the **answer to our search**. They hold the actual DNS records and will provide the IP address of our query.

- Authoritative name servers will respond with the corresponding IP address of [www.google.com](http://www.google.com)
- But first, it will save this IP to its cache. **Caching every step of the way!**
- The user's operating system will also cache this IP address for reference in the future — in case you want to visit this website again ☺

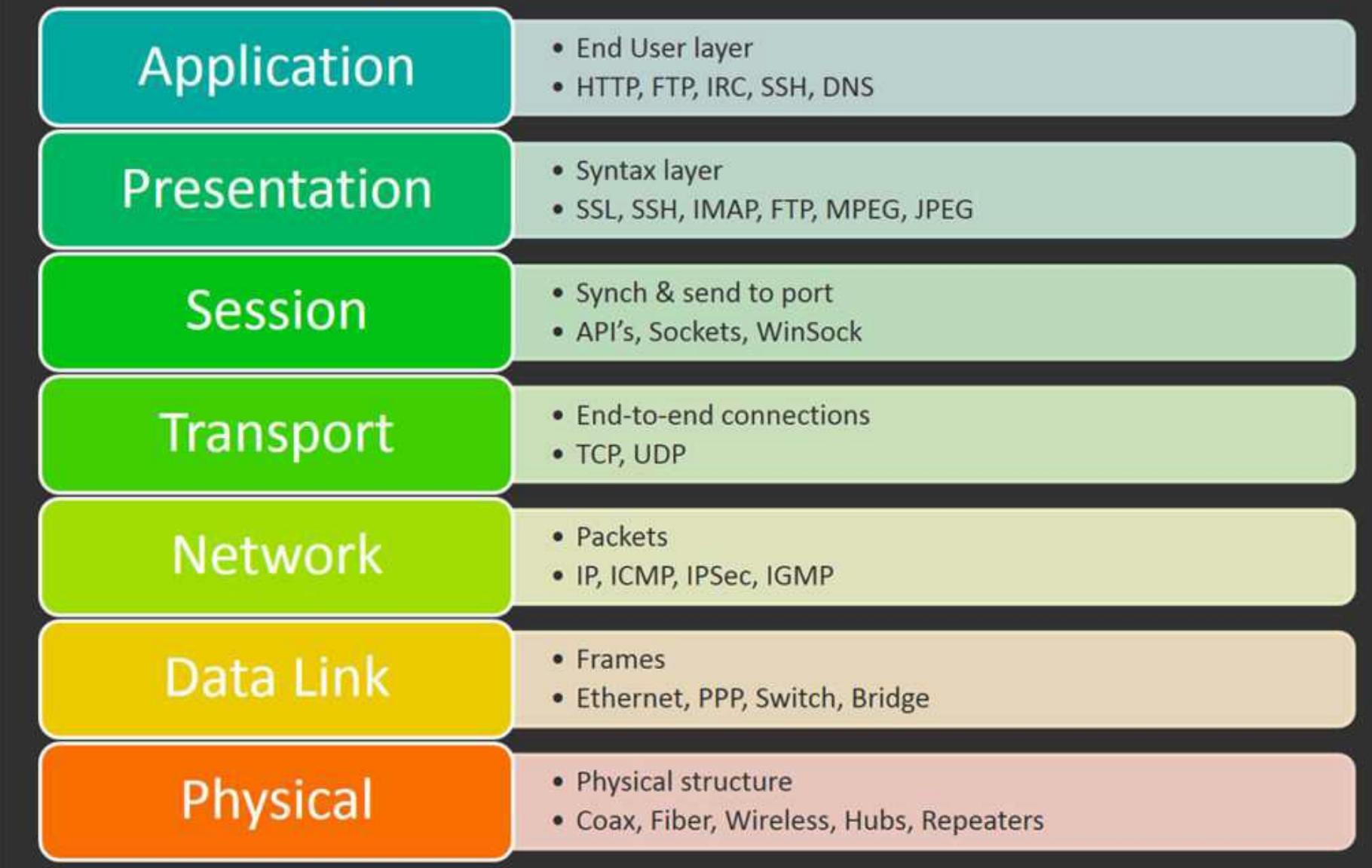
# DNS DONE !! NOW HTTP

- The user's web browser can now follow **HTTP(HyperText Transport Protocol)** and send a **GET request** to the server at google.com's corresponding IP address.



1. HTTP is the protocol used to transfer data to and from the website.
2. WWW is the identifier that indicates that it is a web site and it uses the HTTP protocol.
3. `HTTP://anything.com`, `WWW.anything.com`, `HTTP://WWW.anything.com` leads to the same site.

# 7 Layers of the OSI Model

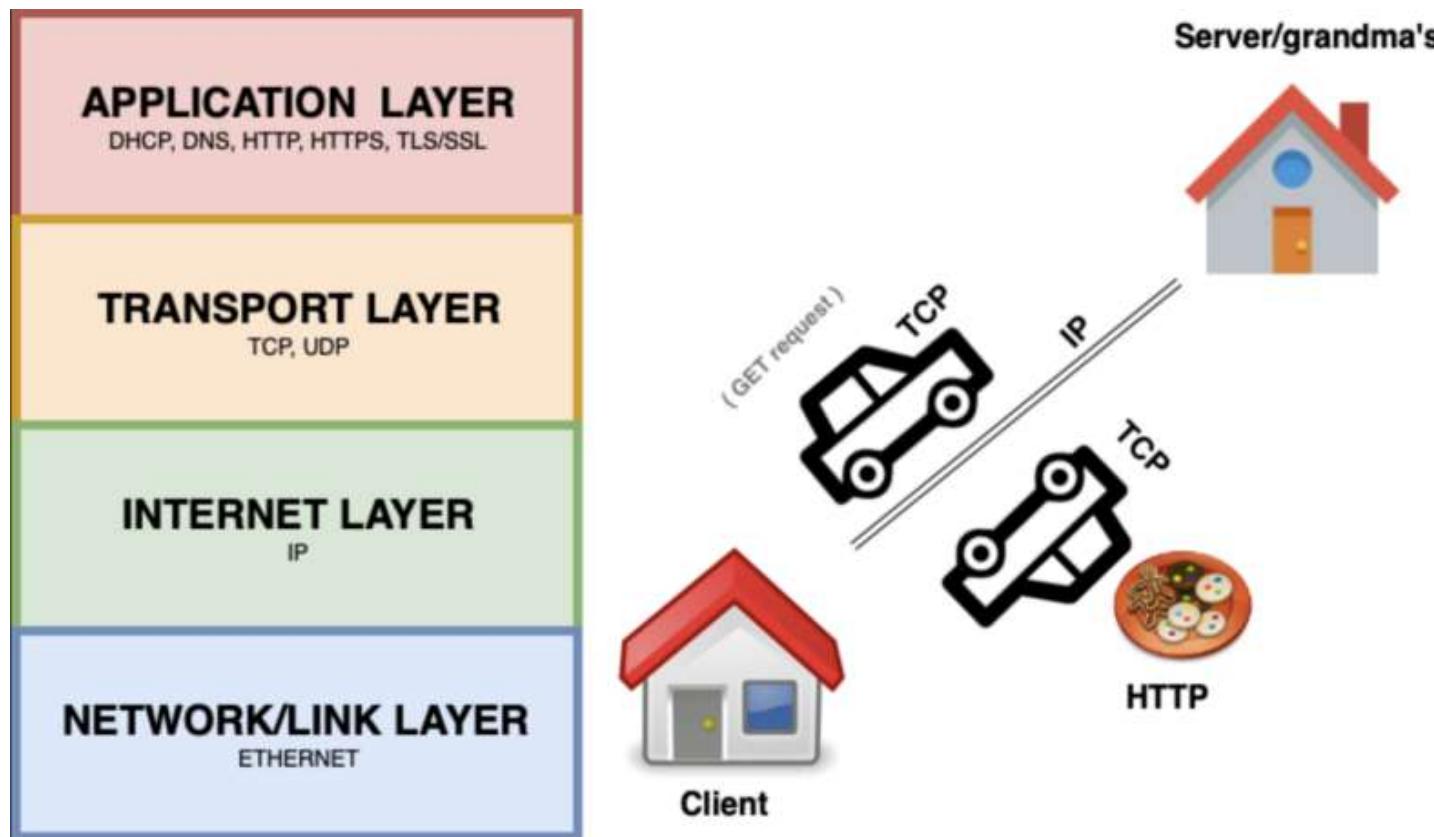


# HTTP & TCP

1. **Hypertext Transport Protocol (HTTP)** is an application layer protocol used for transmitting files/data across the web through **TCP/IP sockets**
2. **TCP(Transmission Control Protocol)** resides in the transport layer and is responsible for creating a reliable end-to-end connection between two hosts. It's similar to a messenger.
3. Allows data transfers of other protocols(like HTTP). TCP will break the data down into smaller packets and then reassemble them at the other end.

# Analogy of picking up cookies from grandma's house

1. IP would be the **road on which we drive**
2. TCP would be the **car**
3. And **HTTP** would be the box of **cookies** moving from one location to another



# TCP/IP

1. **HTTP** relies on **TCP** to establish a reliable connection between client and server. Four pieces of information are needed to establish a **TCP connection**:

1. **Client IP address**
2. **Client Port number**
3. **Source IP address**
4. **Source Port number**

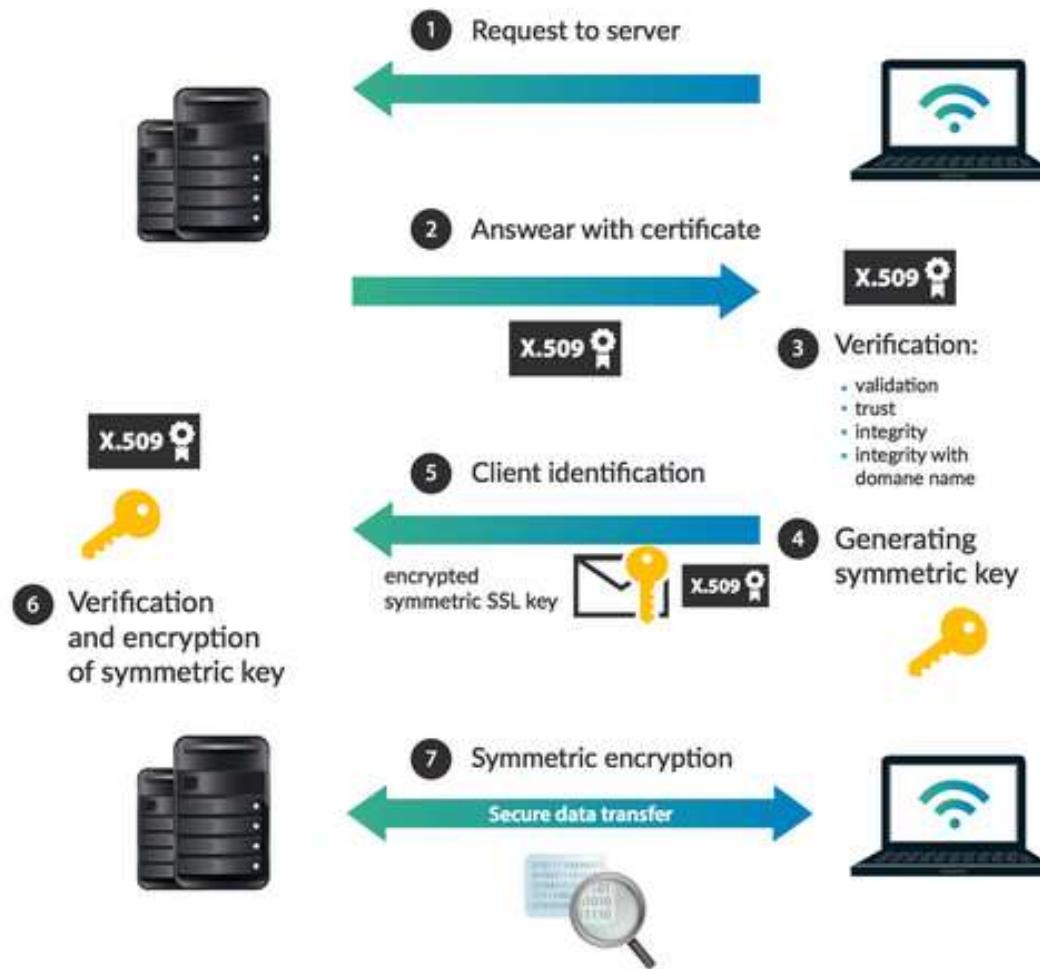
An IP address will identify the device, but a port number is also needed to identify the specific application/service.

It's similar to having an address to an apartment and a specific unit number.

**IP + port number = socket**

HTTP protocol uses port 80 as default

# SSL (HTTPS)



- Client say Hello
- Server says Hello (do you have certificate? I am SSL)
- Client generated symmetric key
- Server verify the key
- Once verification is successful → Secure data transfer

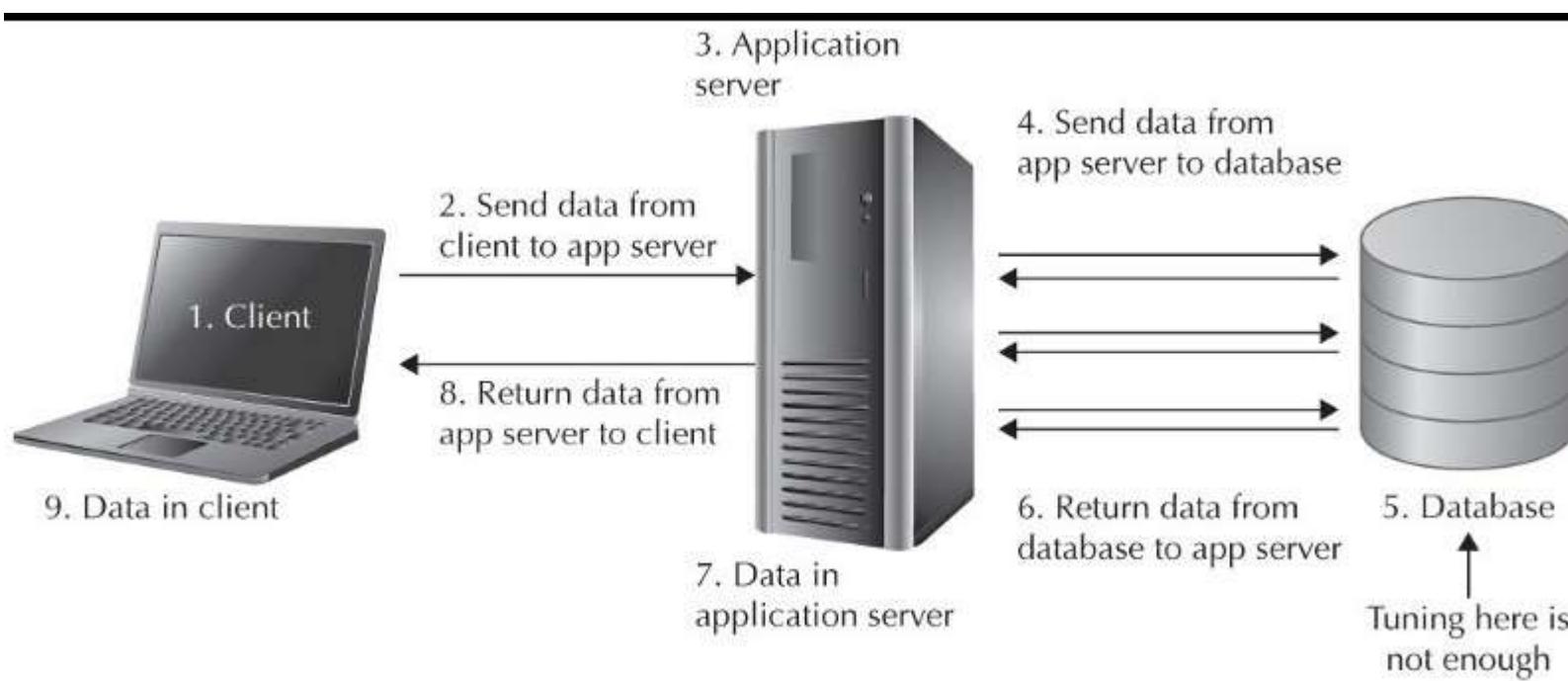
# Firewall

1. TCP network breaks data into chunks (packets). Along with data, a packet will have a header including control information such as **source address**, **destination address**, **connection state**, etc.
2. Protect (a network or system) from unauthorized access with a firewall



# Load Balancer, WebServer, Application Server and Database

1. **Load Balancer:** The job of a load balancer is to.... balance loads.
2. **WebServers:** Web servers supply the web content for web browsers; what the browser requests, the web server delivers through Internet network connections.
3. **Application servers** host and execute applications and can be used to communicate and extract data from a database
4. **Database:** A database is a data repository that stores information







# Common Use Cases for Infrastructure

- Web site / Application hosting
- Mobile and Social Applications
- Internal IT application hosting
- Content delivery and media distribution
- High performance computing, batch data processing, and large scale analytics
- Storage, backup, and disaster recovery
- Development and test environments



**VIRTUAL** GUEST **VIRTUALIZATION** **HOST**  
**MACHINE** DATA **COMPUTING** **SOFTWARE**  
**RESOURCES** **OS** **NETWORK** **SCALABILITY**  
**MANAGER** DESKTOP **HARDWARE** **PERFORMANCE**  
**DATA** **PLATFORM** **SNAPSHOT** **STORAGE**  
**FAILOVER** **CONCEPT**

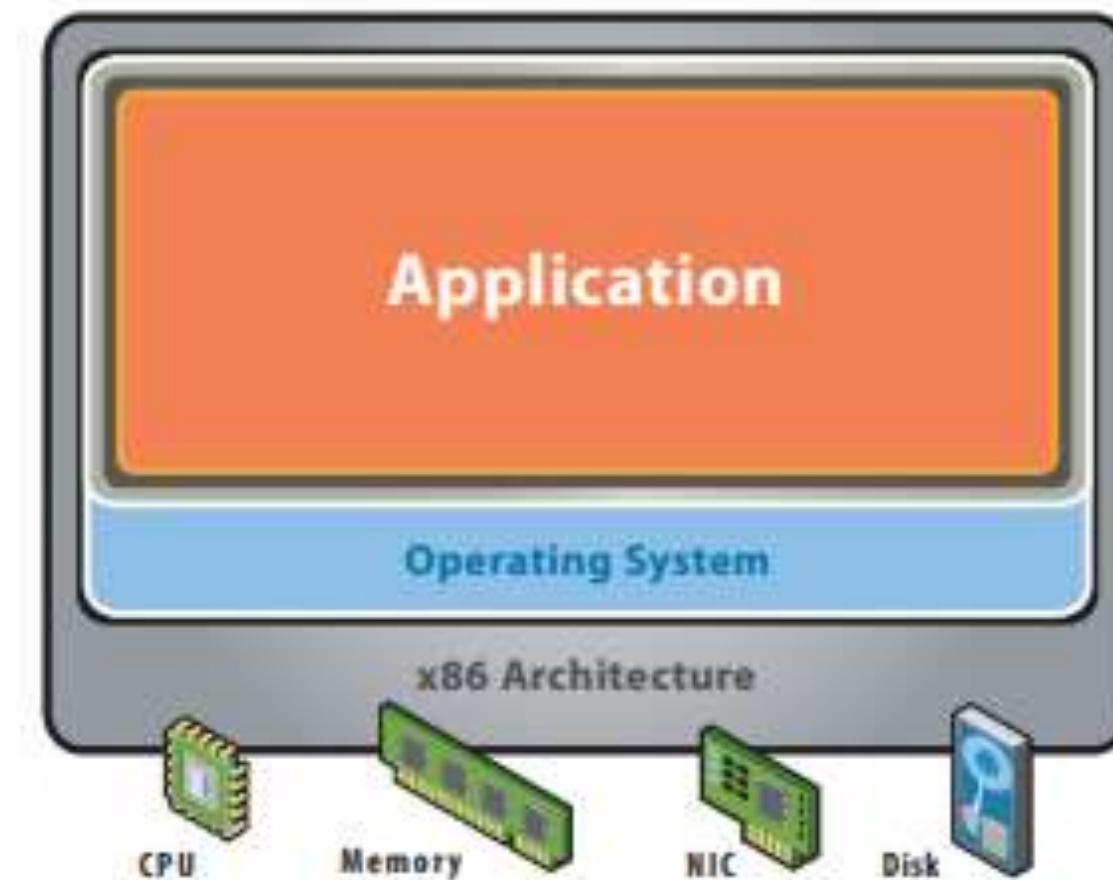
# **VIRTUALIZATION**

# Virtualization Concept

- Creating a virtual machine over existing operating system and hardware is referred as Virtualization.
- Virtual Machines provide an environment that is logically separated from the underlying hardware.
- The machine on which the virtual machine is created is known as **host machine** and **virtual machine** is referred as a **guest machine**.
- This virtual machine is managed by a software or firmware which is known as **hypervisor**.

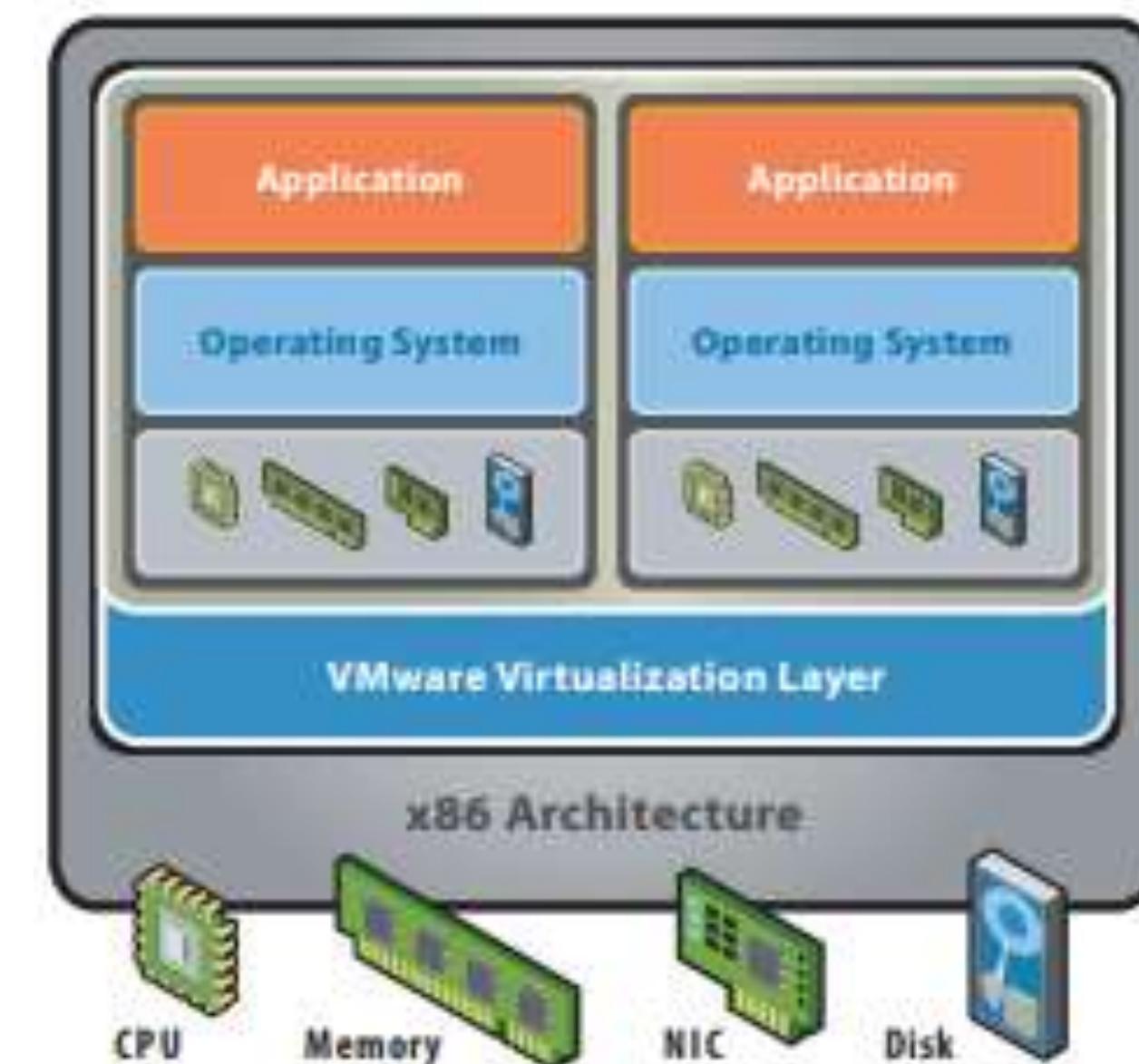
# Before Virtualization

- Single OS image per machine
- Software and hardware tightly coupled
- Running multiple applications on same machine often creates conflict
- Underutilized resources
- Inflexible and costly infrastructure



# After Virtualization

- Hardware-independence of operating system and applications
- Virtual machines can be provisioned to any system
- Can manage OS and application as a single unit by encapsulating them into virtual machines

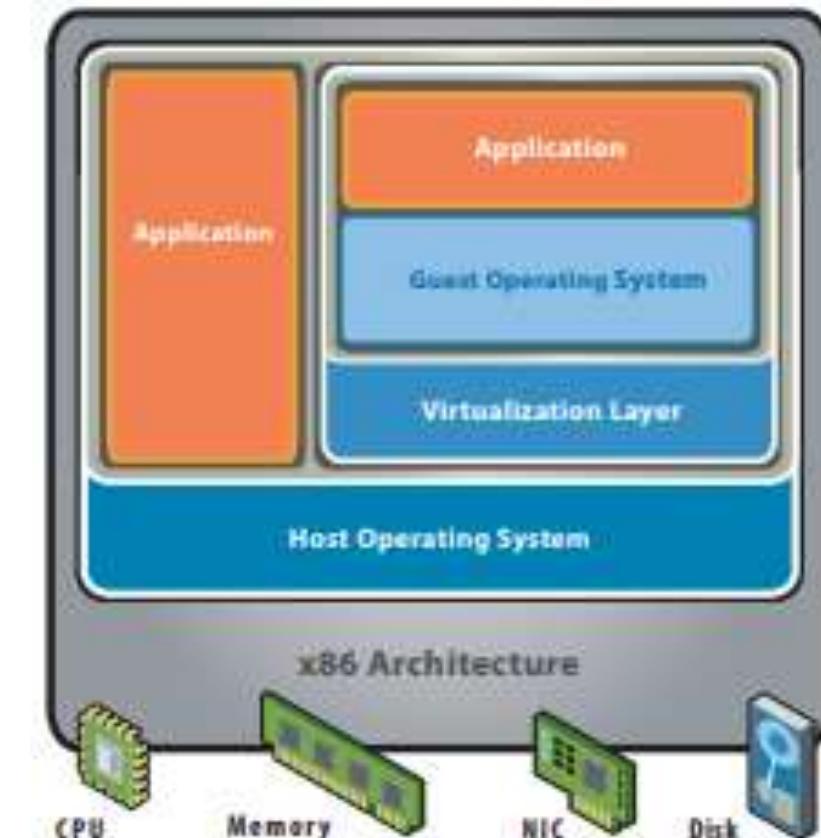


# Virtualization approaches

- **Hosted Architecture**

Installs and runs as an application

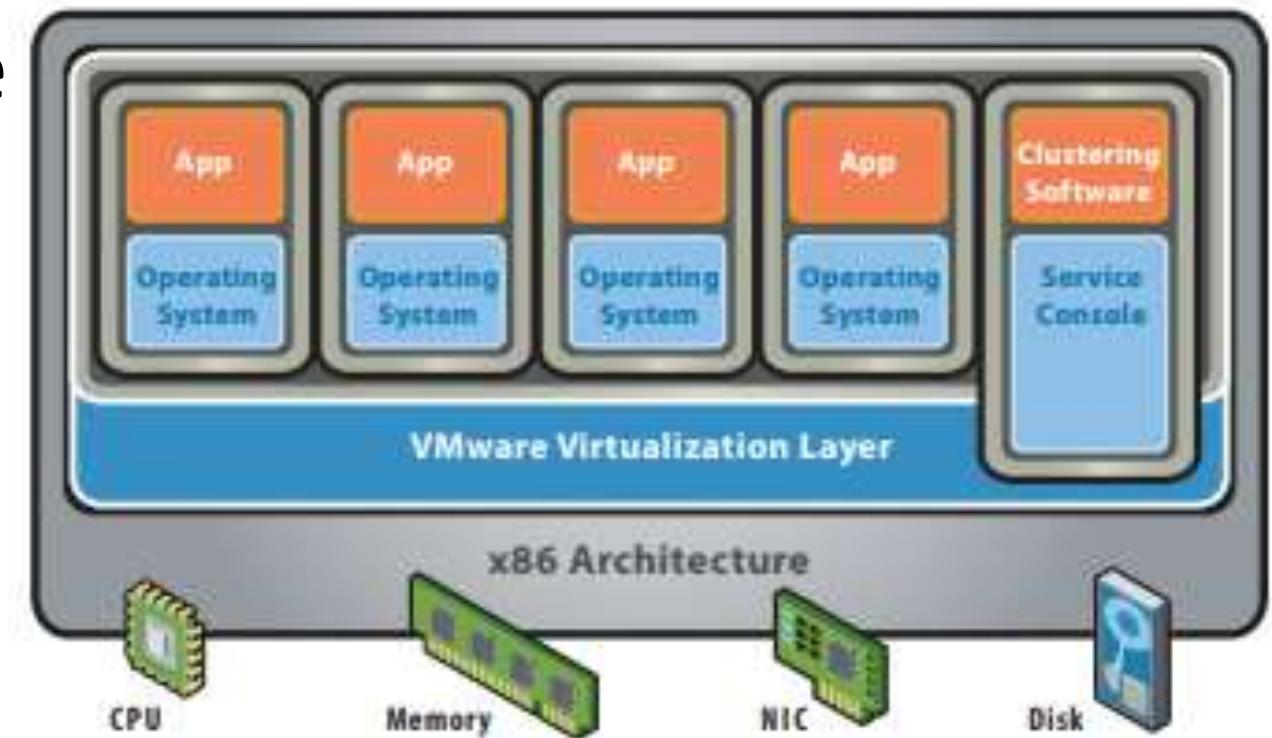
Relies on host OS for device support and physical resource management



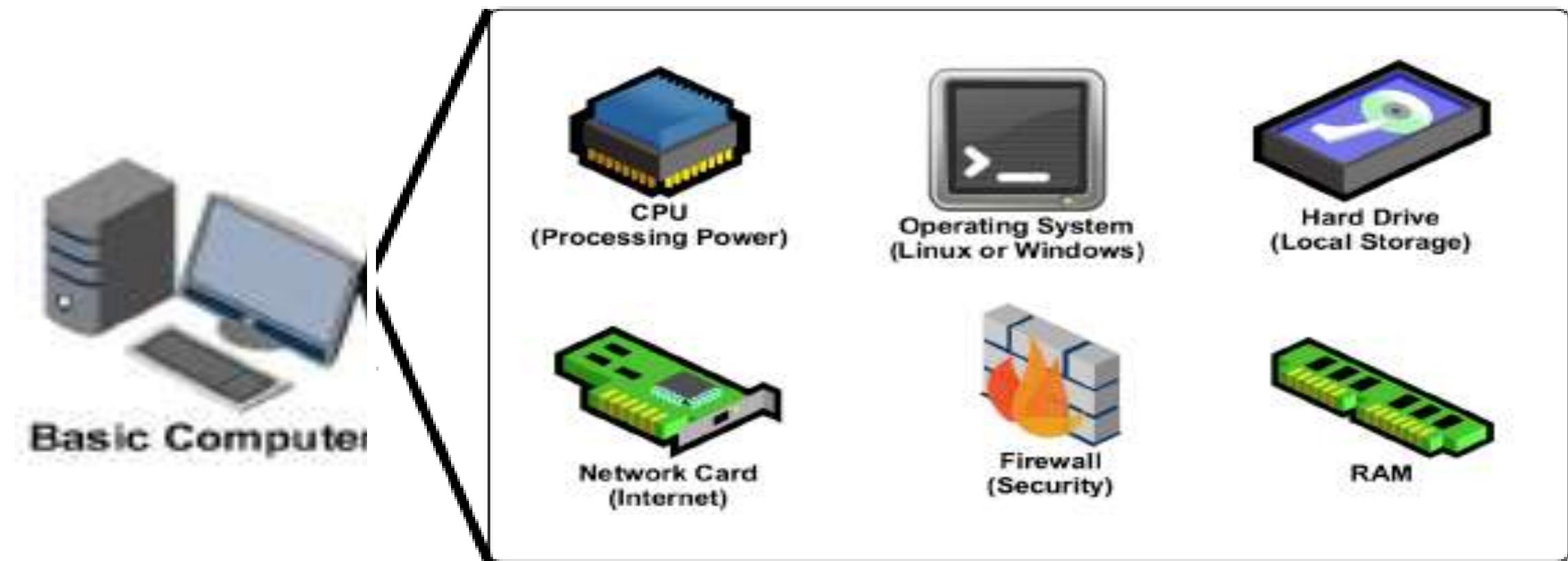
- **Bare Metal (Hypervisor) Architecture**

A bare-metal virtualization hypervisor does not require install a server operating system first.

Bare-metal virtualization means the hypervisor has direct access to hardware resources, which results in better performance, scalability and stability



- Basic Computer
- Networking
- Security
- Storage
- Data Base





# Where is Cloud”?

Cloud computing is currently the buzzword in IT industry, and many are curious to know what cloud computing is and how it works. More so because the term CLOUD is intriguing and some people even wonder how do clouds that rain can even remotely be used in Computing.

# Like this?



# What is Cloud”?

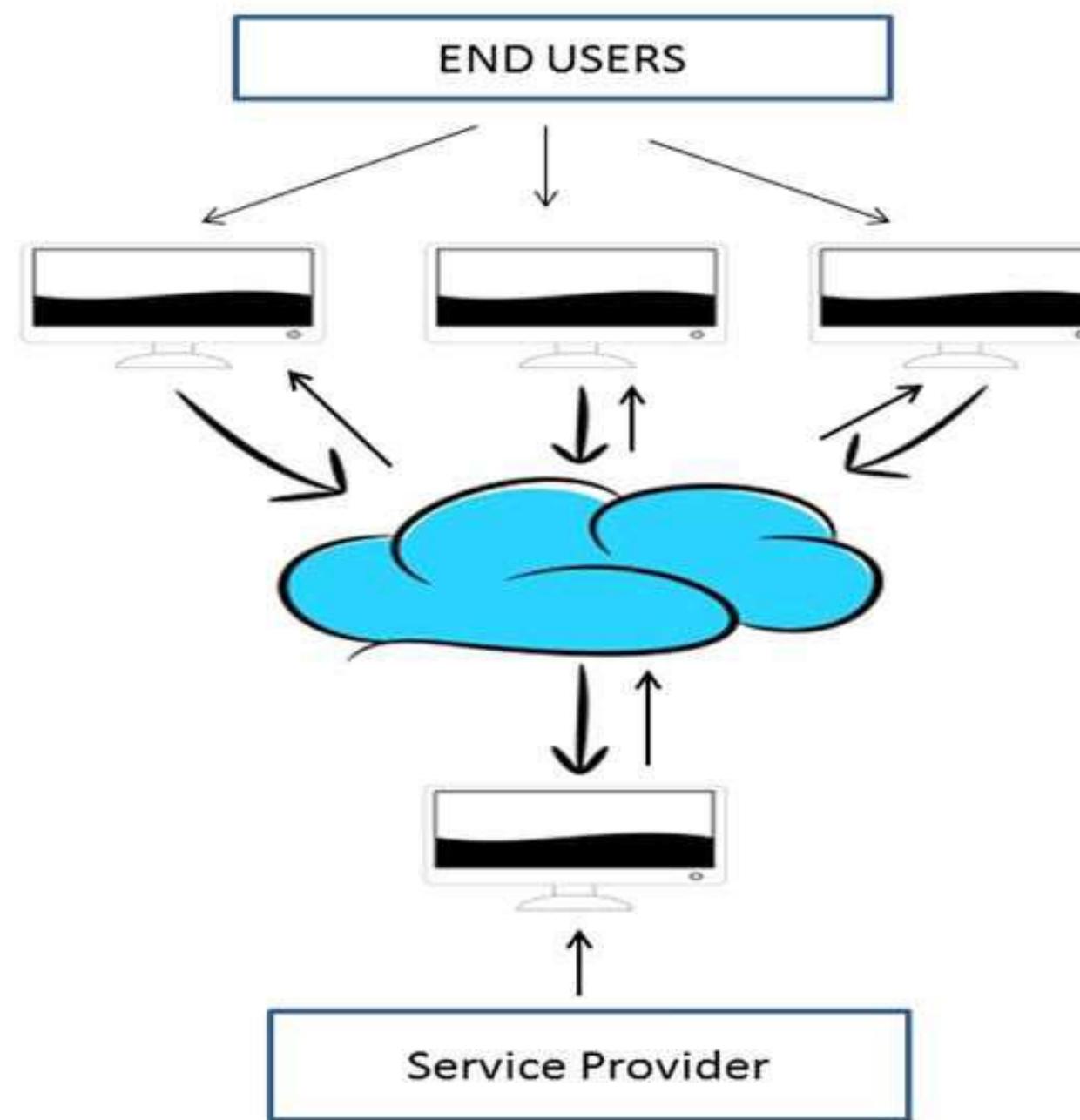
The term **Cloud** refers to a **Network** or **Internet**. In other words, we can say that Cloud is something, which is **present at remote location**.

Cloud can provide services over public and private networks

Cloud computing means storing and accessing data and programs over the Internet instead of your computer's hard drive.

# Why the Name “Cloud”?

- The term “Cloud” came from a network design that was used by network engineers to represent the location of various network devices and their inter-connection. The shape of this network design was like a cloud.



# What is cloud computing?

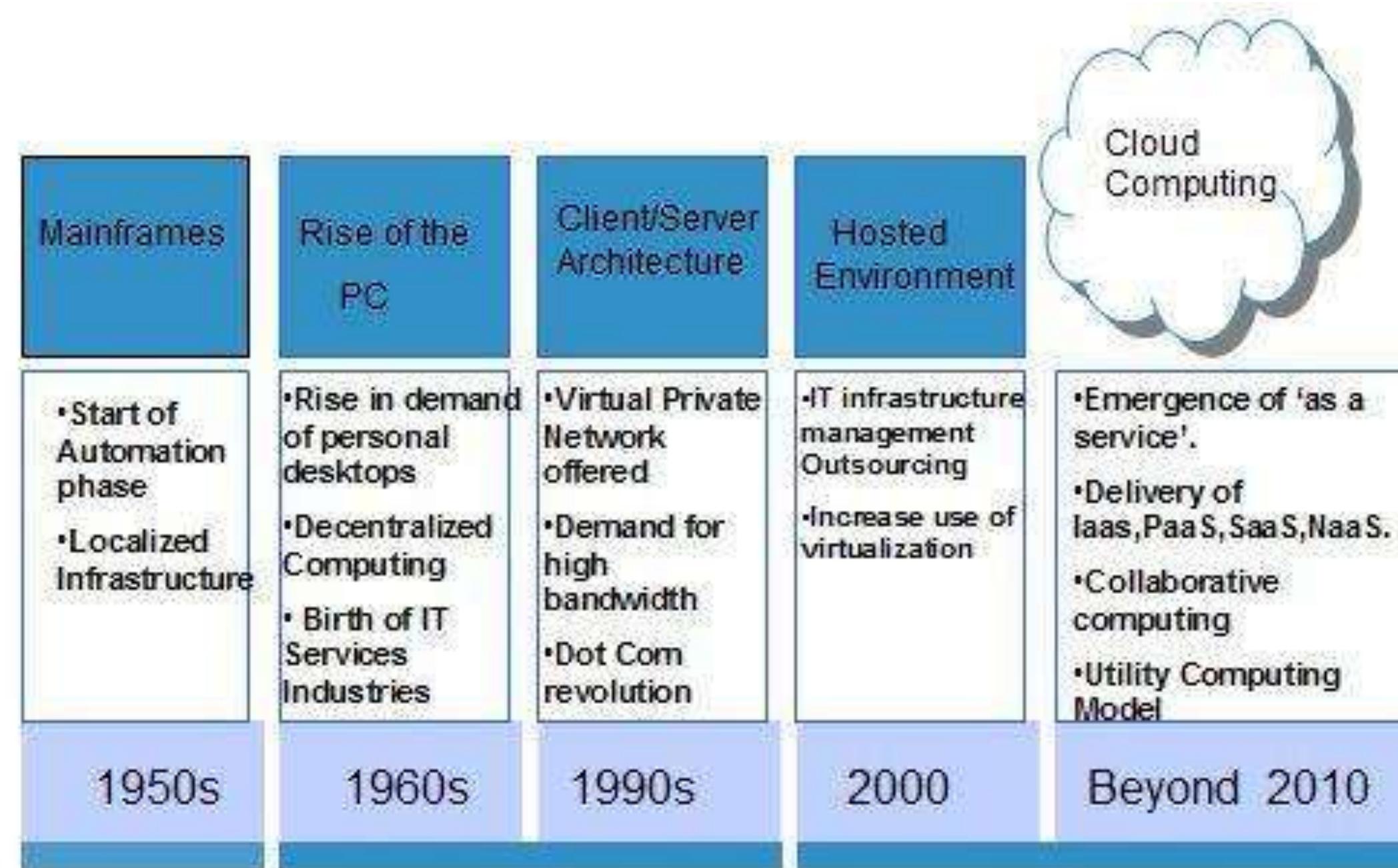
- Cloud Computing can be defined as delivering computing power( CPU, RAM, Network Speeds, Storage OS software) a service over the internet rather than physically having the computing resources at the customer location.

Or

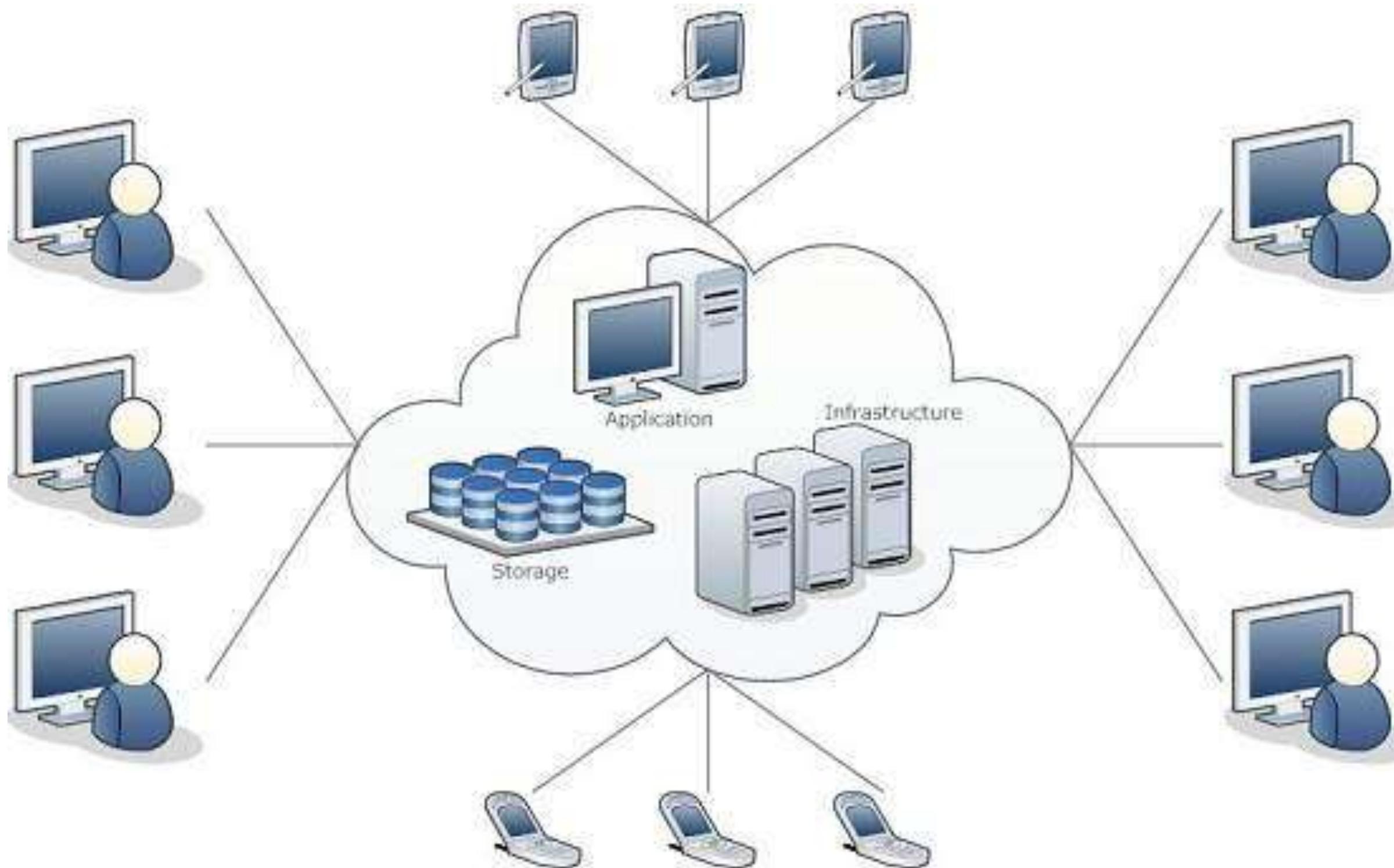
- Cloud Computing refers to manipulating, configuring, and accessing the hardware and software resources remotely. It offers online data storage, infrastructure, and application.
- Example: AWS, Azure, Google Cloud

# History of Cloud Computing

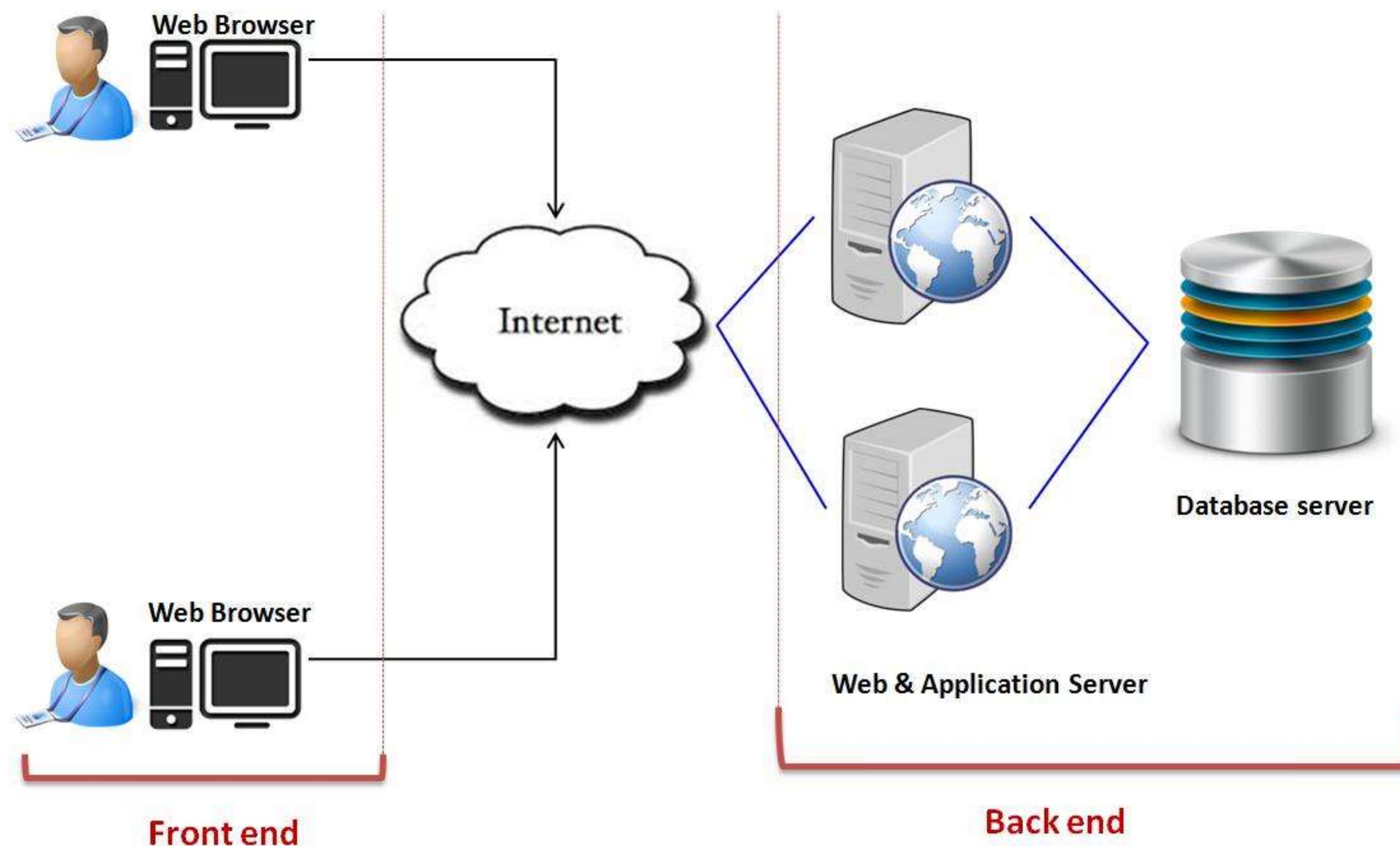
- The concept of **Cloud Computing** came into existence in the year 1950 with implementation of mainframe computers, accessible via **thin clients**. Since then, cloud computing has been evolved from thin clients to dynamic ones and from software to services. The following diagram explains the evolution of cloud computing:



# Conceptual view of cloud computing



# What is Cloud Computing Architecture?

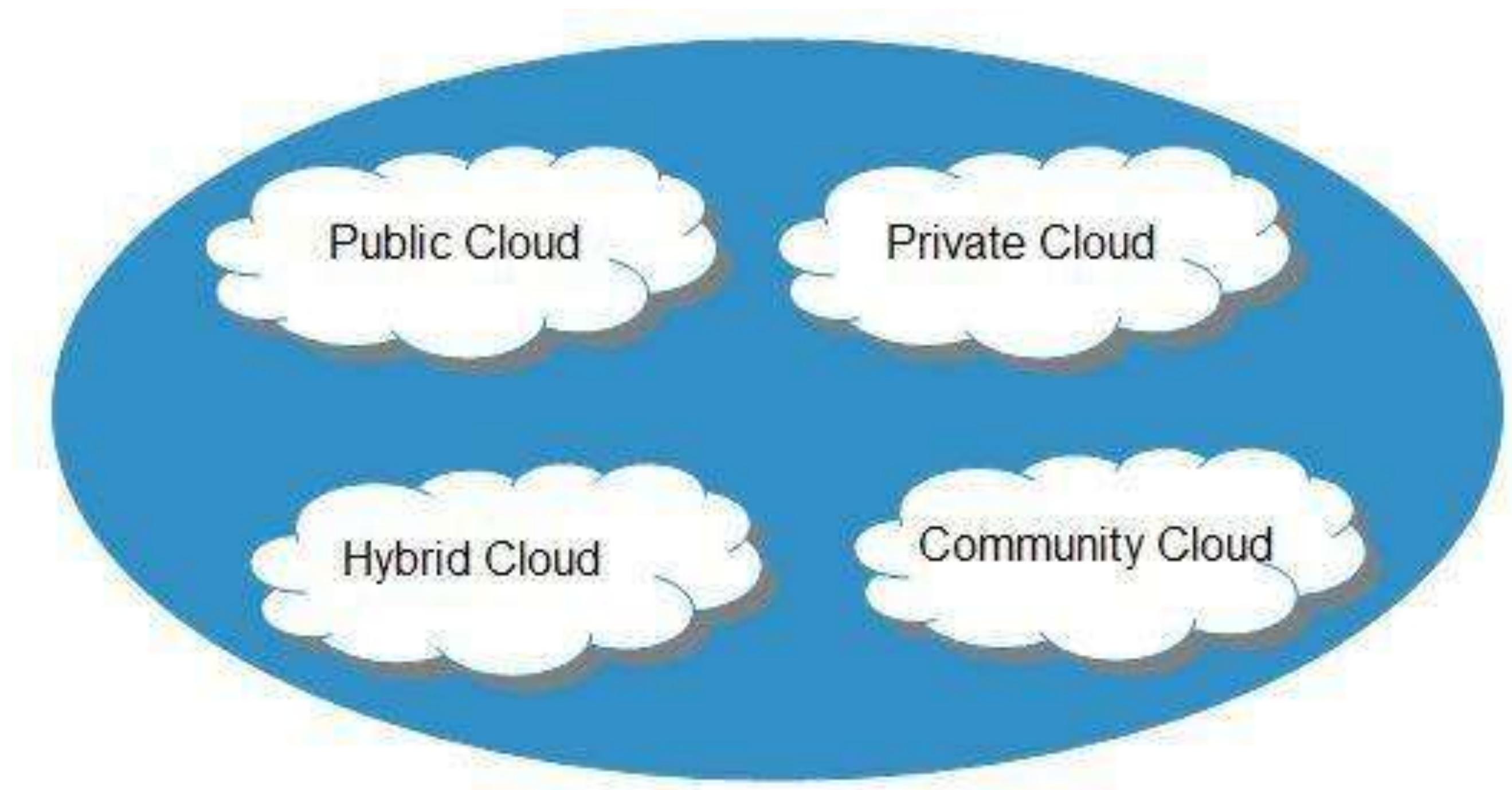


# Basic Concepts

There are certain **services** and **models** working behind the scene making the cloud computing feasible and accessible to end users. Following are the **working models** for cloud computing:

- **Deployment Models**
  - Public Cloud
  - Private Cloud
  - Hybrid Cloud
  - Community Cloud
- **Service Models**
  - IAAS
  - PAAS
  - SAAS
  - Anything-as-a-Service (XaaS) is yet another service model, which includes Network-as-a-Service, Business-as-a-Service, Identity-as-a-Service, Database-as-a-Service or Strategy-as-a-Service.

# Types of Clouds



# Types of Cloud Explained

- **PUBLIC CLOUD**

The **public cloud** allows systems and services to be easily accessible to the general public. Public cloud may be less secure because of its openness.

- **PRIVATE CLOUD**

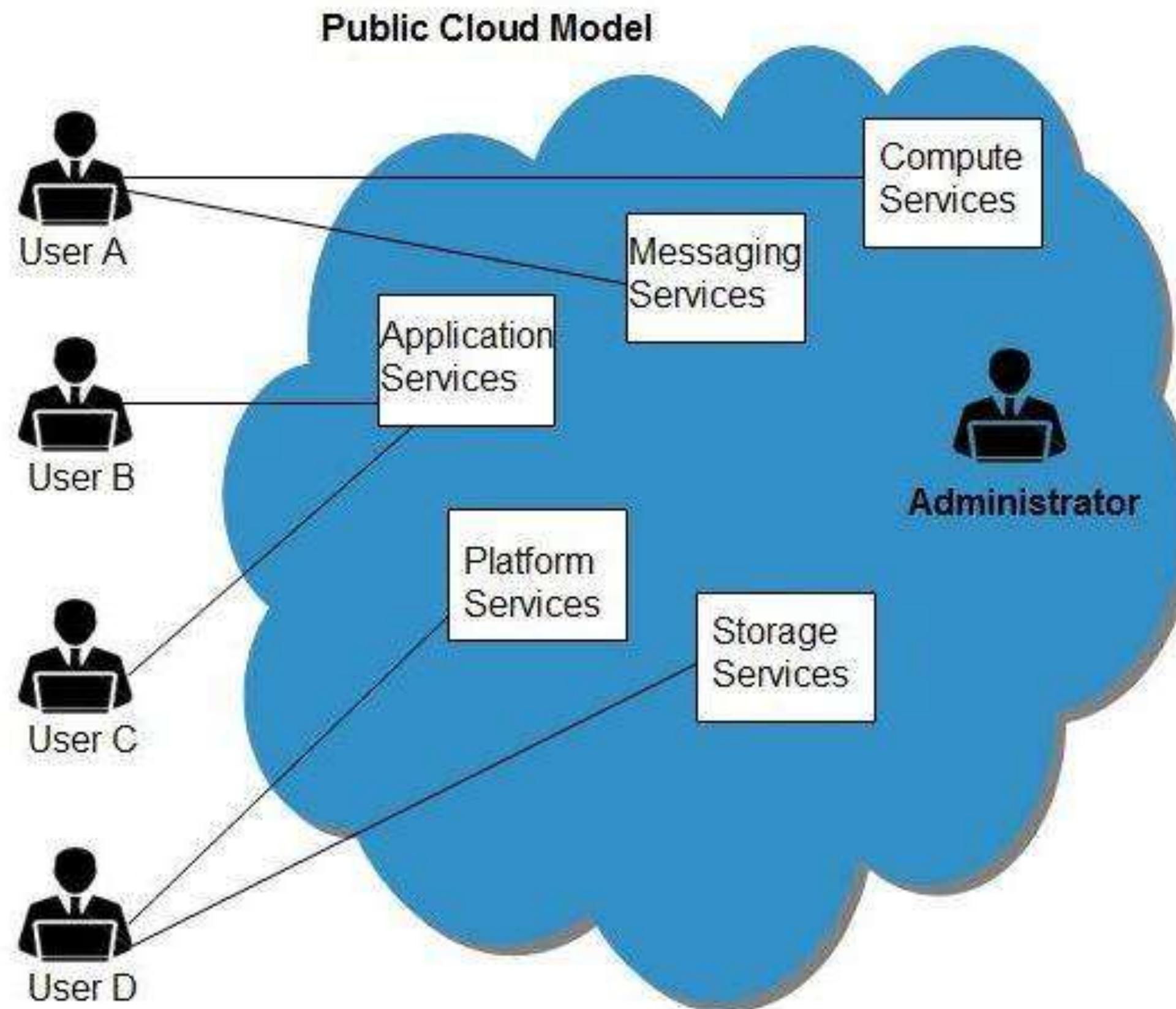
The **private cloud** allows systems and services to be accessible within an organization. It is more secured because of its private nature.

- **COMMUNITY CLOUD**

The **community cloud** allows systems and services to be accessible by a group of organizations.

- **HYBRID CLOUD**

The **hybrid cloud** is a mixture of public and private cloud, in which the critical activities are performed using private cloud while the non-critical activities are performed using public cloud.

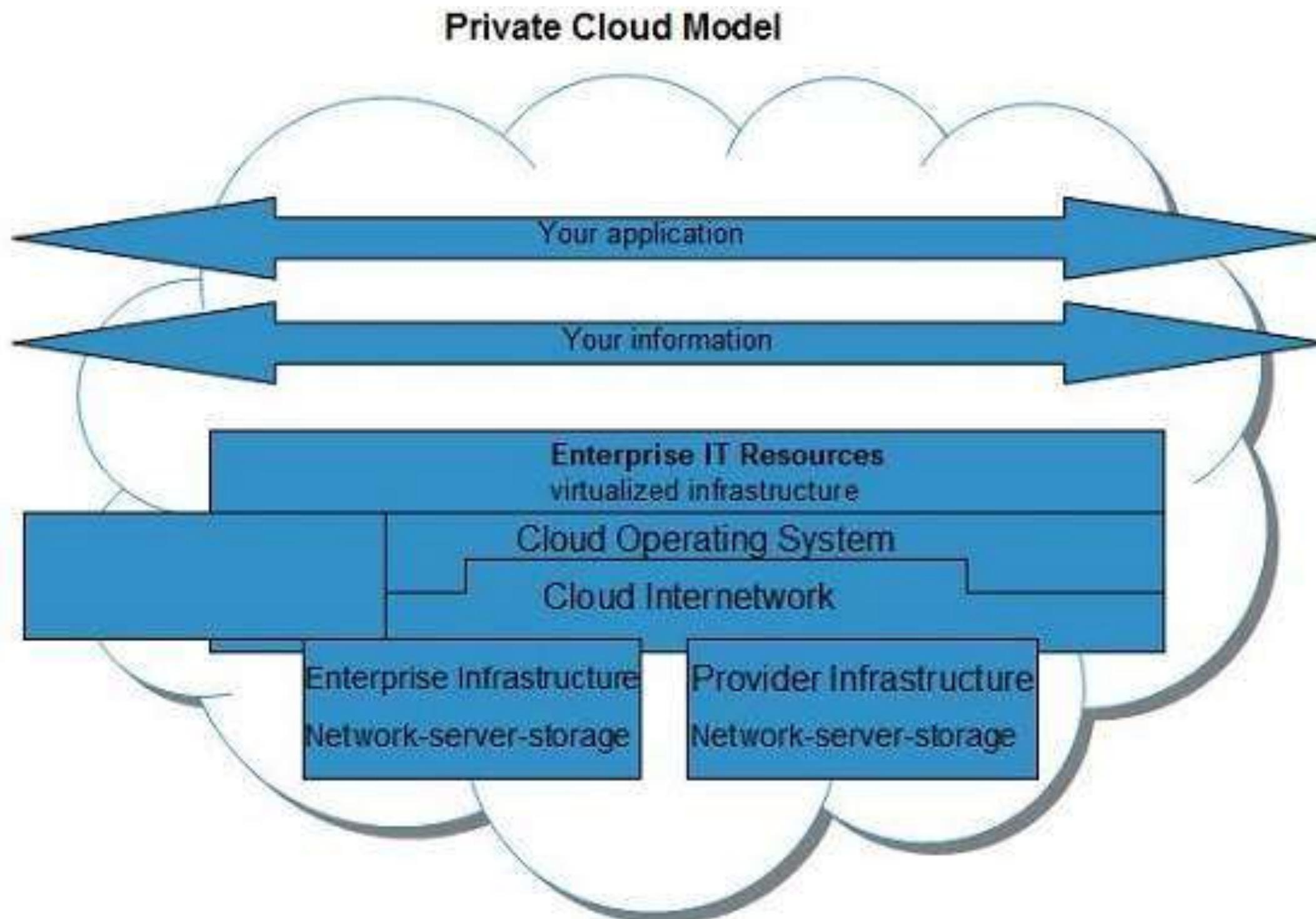


## **BENEFITS**

- Cost Effective
- Reliability
- Flexibility
- Location Independence
- Utility Style Costing
- High Scalability

## **DISADVANTAGES**

- Low Security
- Less customizable

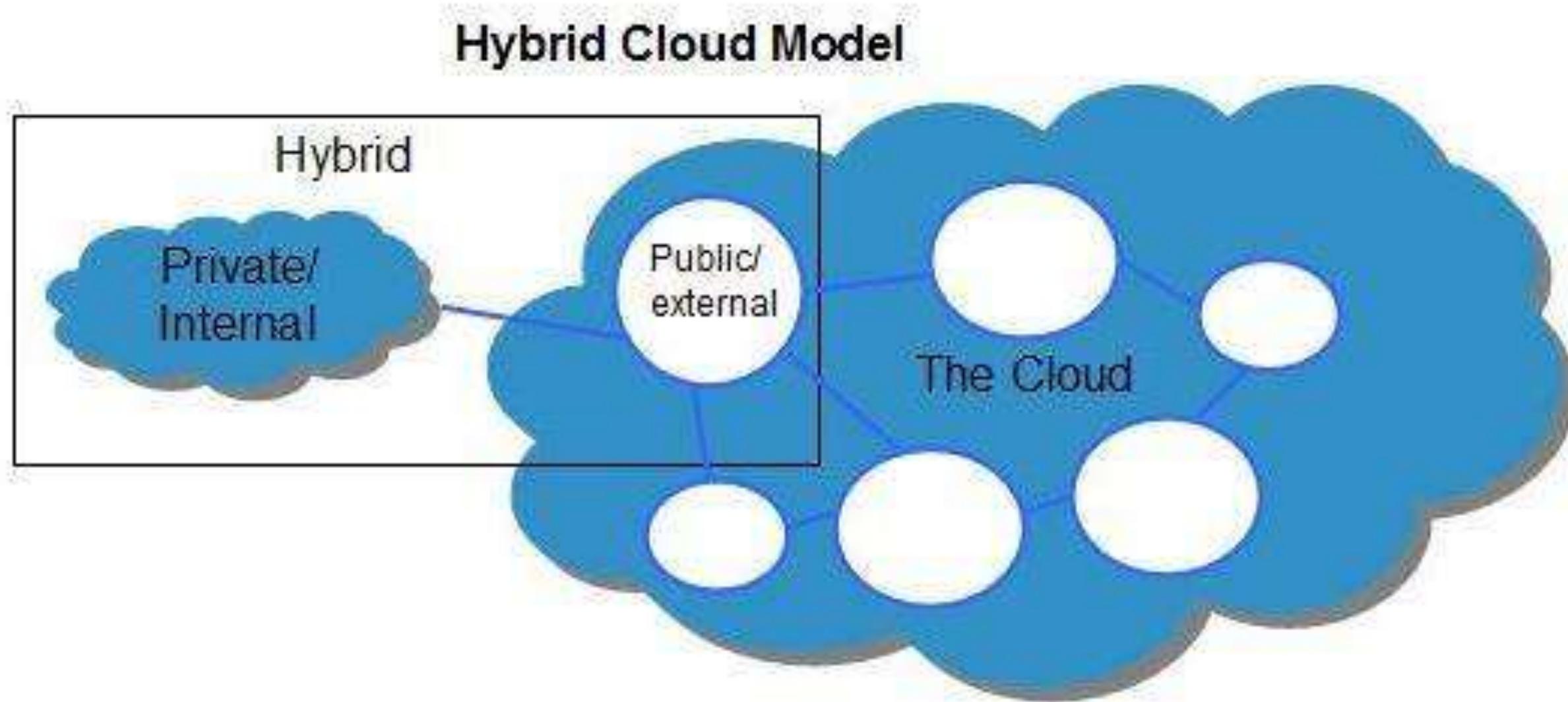


## **BENEFITS**

- Higher Security and Privacy
- More Control
- Cost and energy efficiency

## **DISADVANTAGES**

- Restricted Area
- Inflexible Pricing
- Limited Scalability
- Additional Skills

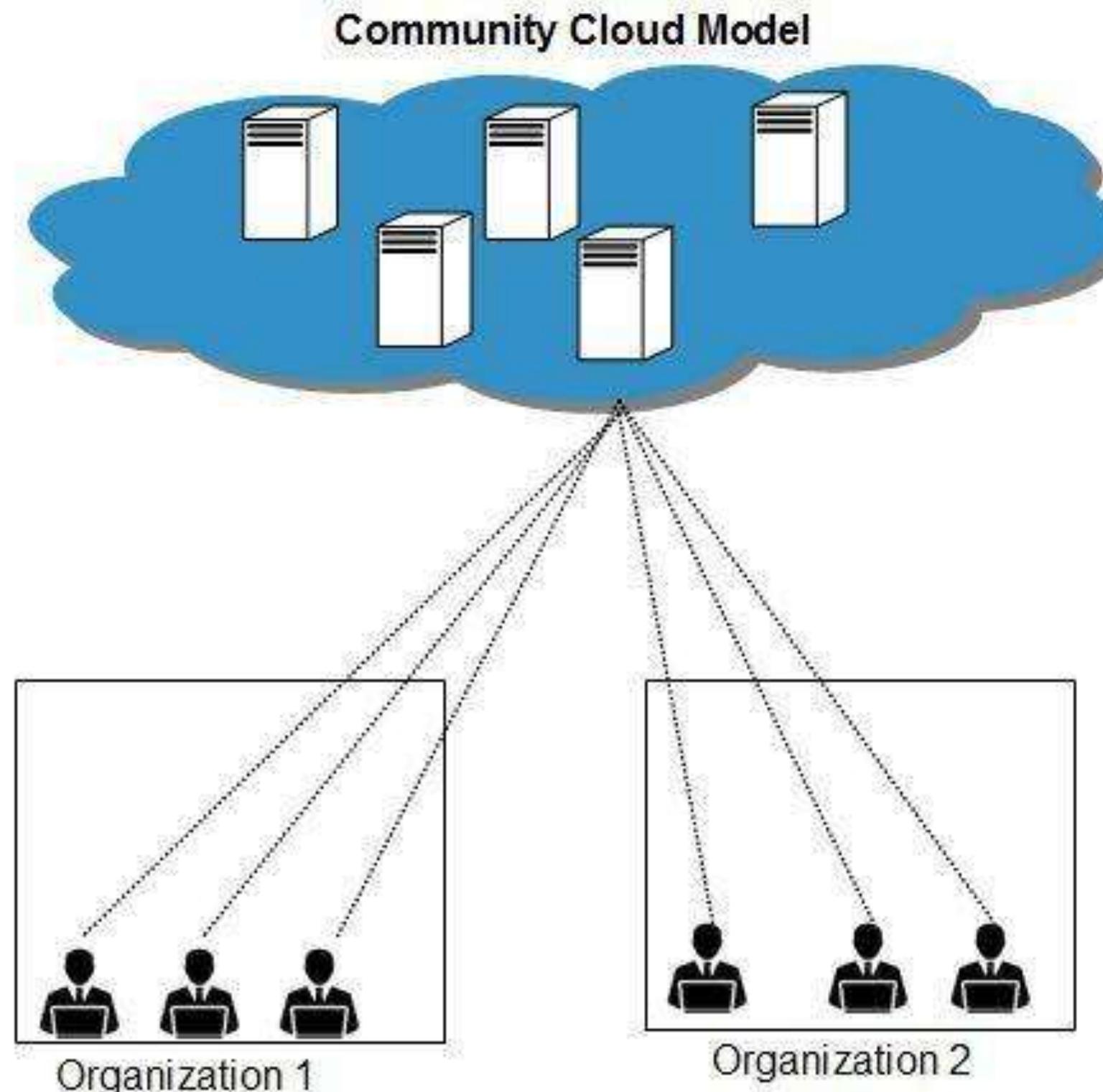


## BENEFITS

- Scalability
- Flexibility
- Cost Efficiencies

## DISADVANTAGES

- Networking Issues
- Security Compliance
- Infrastructural Dependency



## **BENEFITS**

- Cost effective
- Sharing Between Organizations
- Security

## **ISSUES**

- Since all data is housed at one location, therefore one must be careful in storing data in community cloud because it might be accessible by others.
- It is also challenging to allocate responsibilities of governance, security and cost.

# Service Models

- **INFRASTRUCTURE-AS-A-SERVICE (IAAS)**

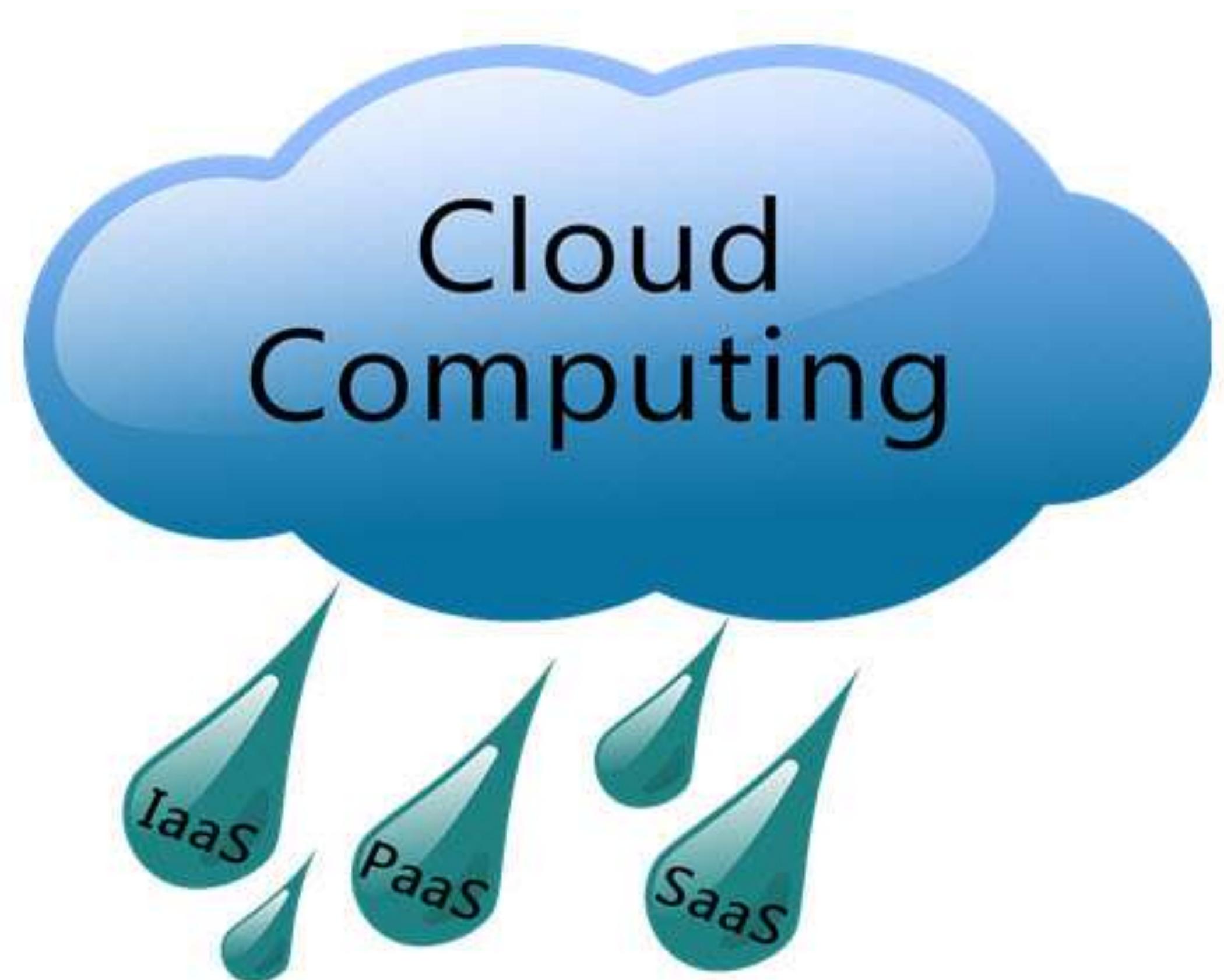
IaaS provides access to fundamental resources such as physical machines, virtual machines, virtual storage, etc.

- **PLATFORM-AS-A-SERVICE (PAAS)**

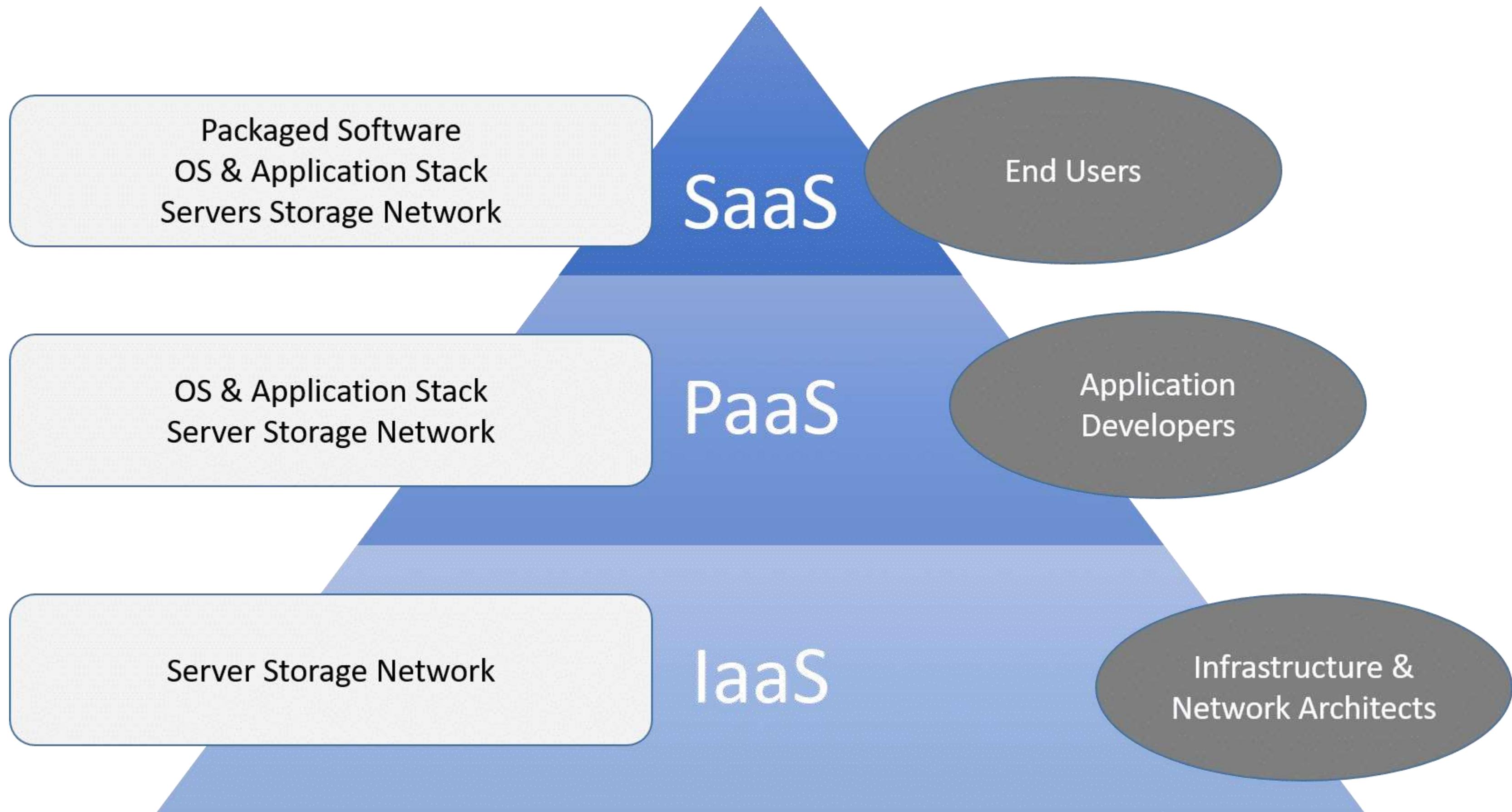
Deploy application without managing virtual servers (Google App Engine, , AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com)

- **SOFTWARE-AS-A-SERVICE (SAAS)**

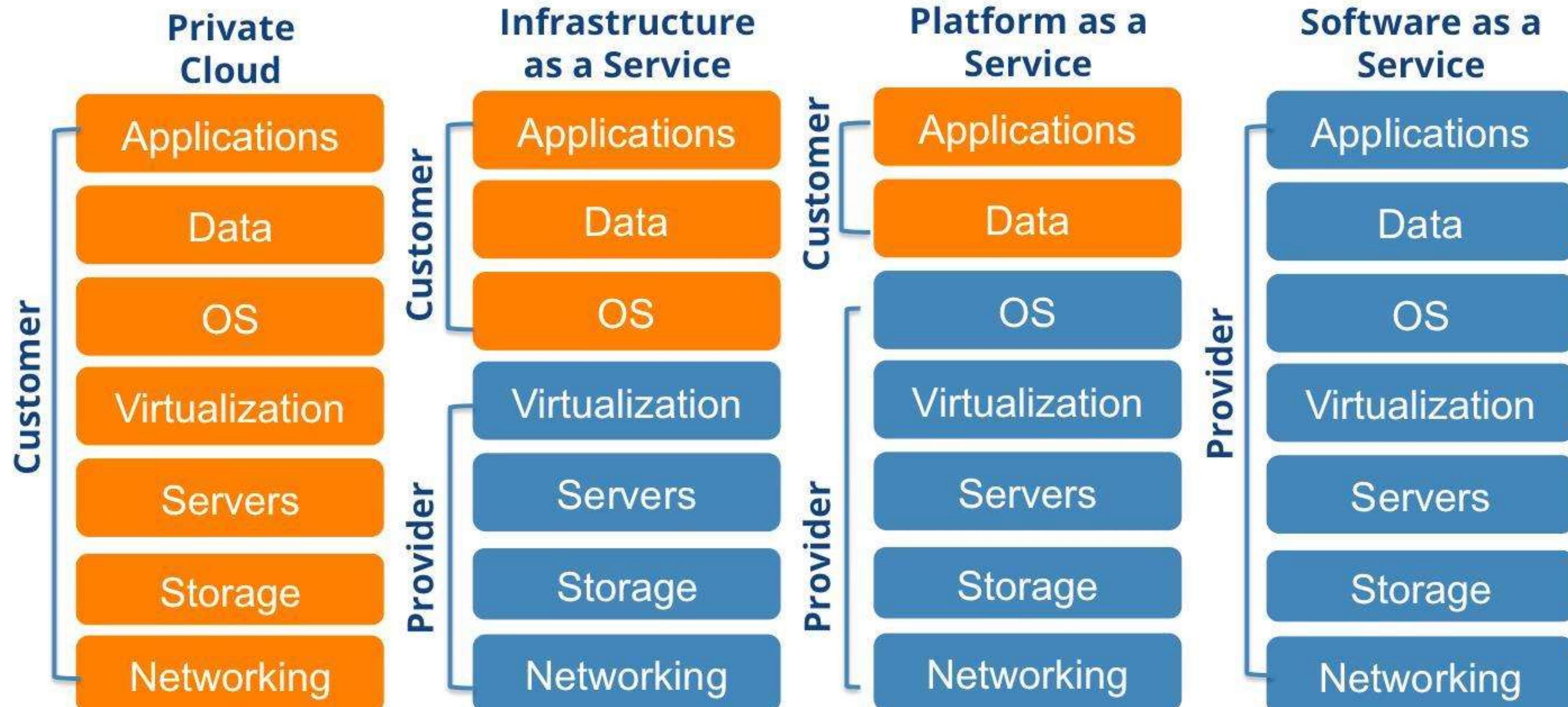
Ready to use software applications (Gmail, Office365, Google Apps, Dropbox, Salesforce, Cisco WebEx, Concur, GoToMeeting)



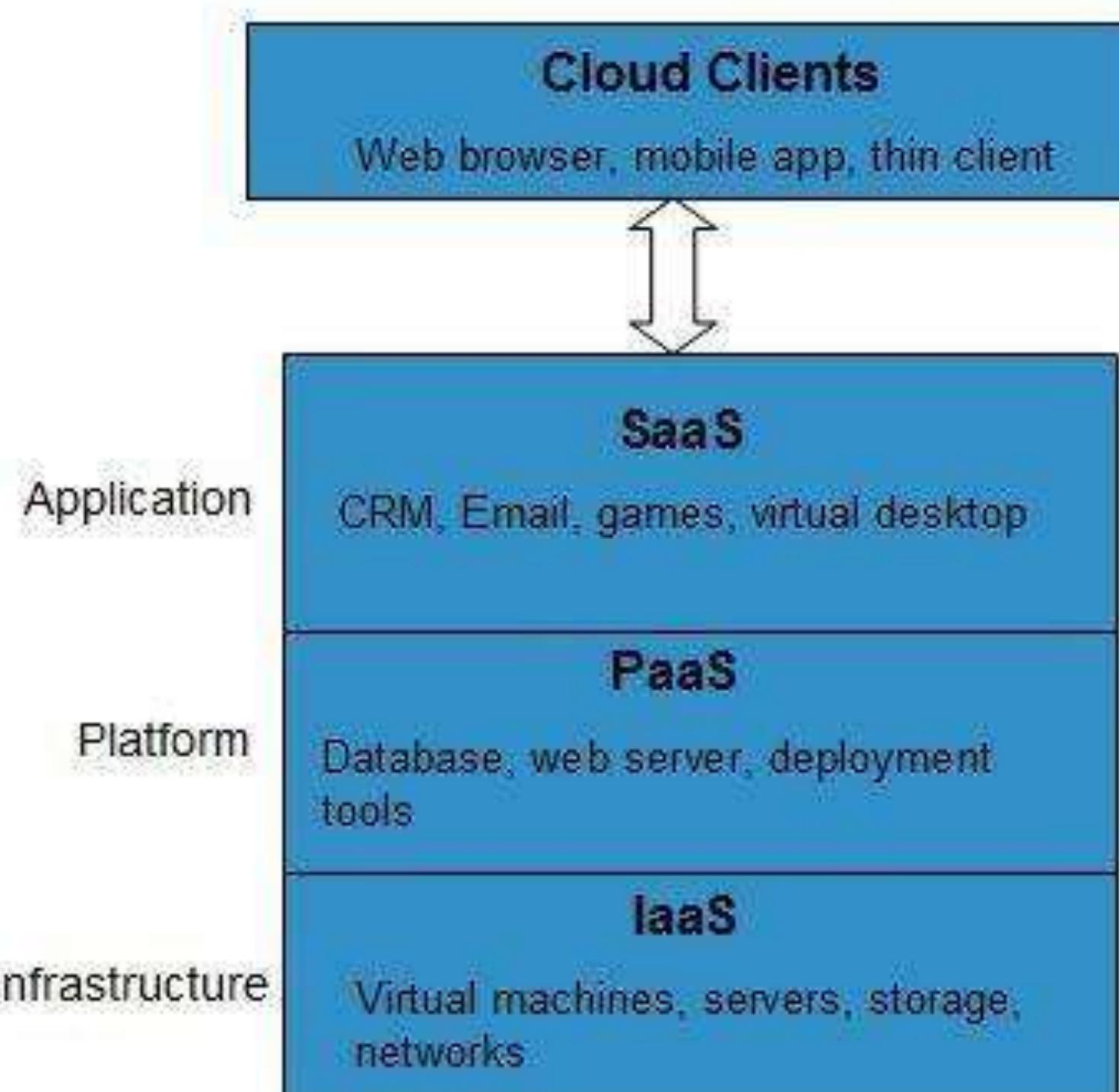
# Cloud Service Models



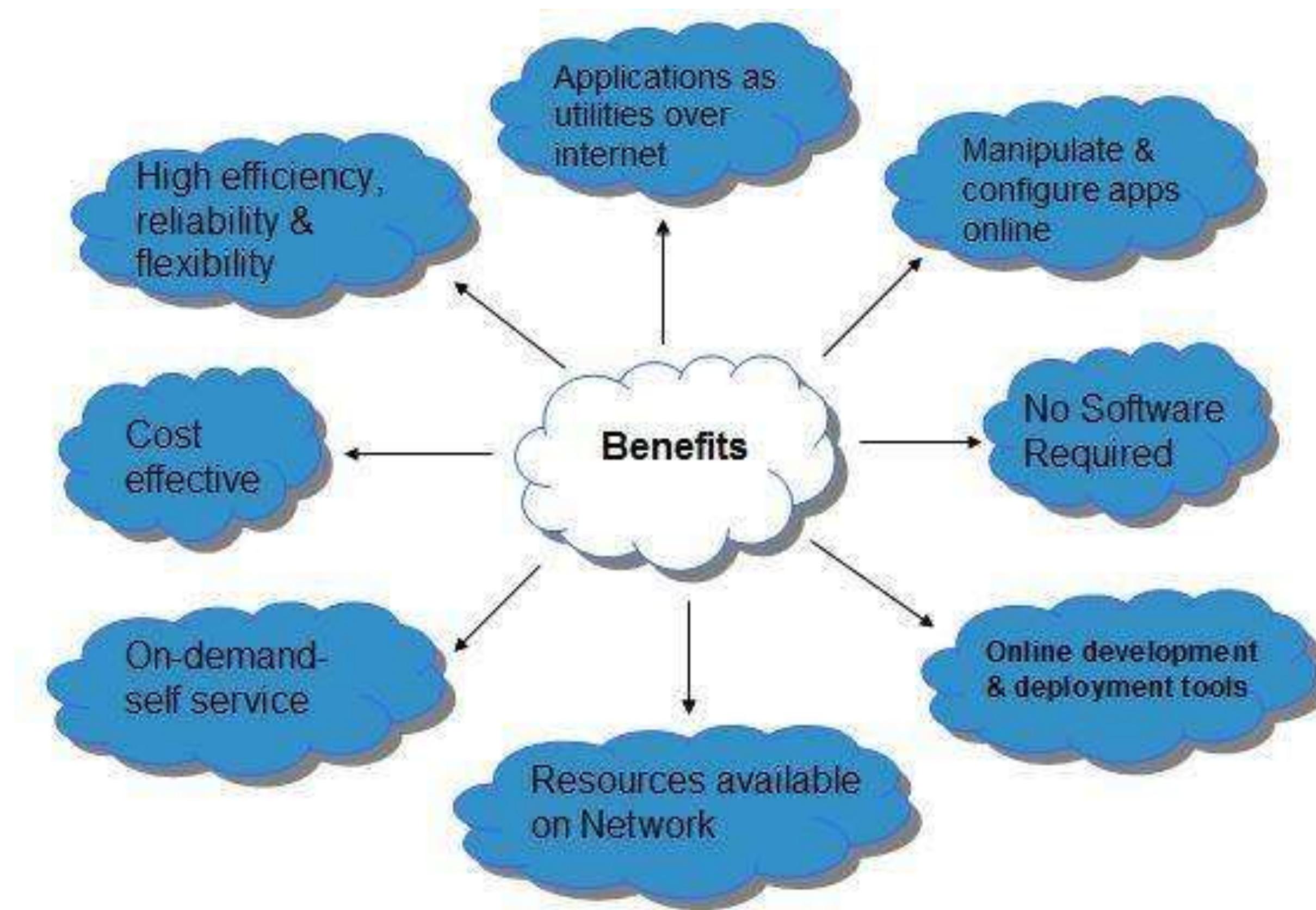
# Shared Model



# Service Models



# Benefits



# Benefits of Cloud Computing

Cloud Computing has numerous advantages. Some of them are listed below -

- One can access applications as utilities, over the Internet.
- One can manipulate and configure the applications online at any time.
- It does not require to install a software to access or manipulate cloud application.
- Cloud Computing offers online development and deployment tools, programming runtime environment through **PaaS model**.
- Cloud resources are available over the network in a manner that provide platform independent access to any type of clients.
- Cloud Computing offers **on-demand self-service**. The resources can be used without interaction with cloud service provider.
- Cloud Computing is highly cost effective because it operates at high efficiency with optimum utilization. It just requires an Internet connection
- Cloud Computing offers load balancing that makes it more reliable.

# Benefits of Cloud Computing

The potential for cost saving is the major reason of cloud services adoption by many organizations. Cloud computing gives the freedom to use services as per the requirement and pay only for what you use. Due to cloud computing it has become possible to run IT operations as a outsourced unit without much in-house resources.

- Lower IT infrastructure and computer costs for users
- Improved performance
- Fewer Maintenance issues
- Instant software updates
- Improved compatibility between Operating systems
- Backup and recovery
- Performance and Scalability
- Increased storage capacity
- Increase data safety

# Risks related to Cloud Computing

Although cloud Computing is a promising innovation with various benefits in the world of computing, it comes with risks. Some of them are discussed below:

- **Security and Privacy**

It is the biggest concern about cloud computing. Since data management and infrastructure management in cloud is provided by third-party, it is always a risk to handover the sensitive information to cloud service providers.

Although the cloud computing vendors ensure highly secured password protected accounts, any sign of security breach may result in loss of customers and businesses.

- **Lock In**

It is very difficult for the customers to switch from one **Cloud Service Provider (CSP)** to another. It results in dependency on a particular CSP for service.

- **Isolation Failure**

This risk involves the failure of isolation mechanism that separates storage, memory, and routing between the different tenants.

- **Management Interface Compromise**

In case of public cloud provider, the customer management interfaces are accessible through the Internet.

- **Insecure or Incomplete Data Deletion**

It is possible that the data requested for deletion may not get deleted. It happens because either of the following reasons

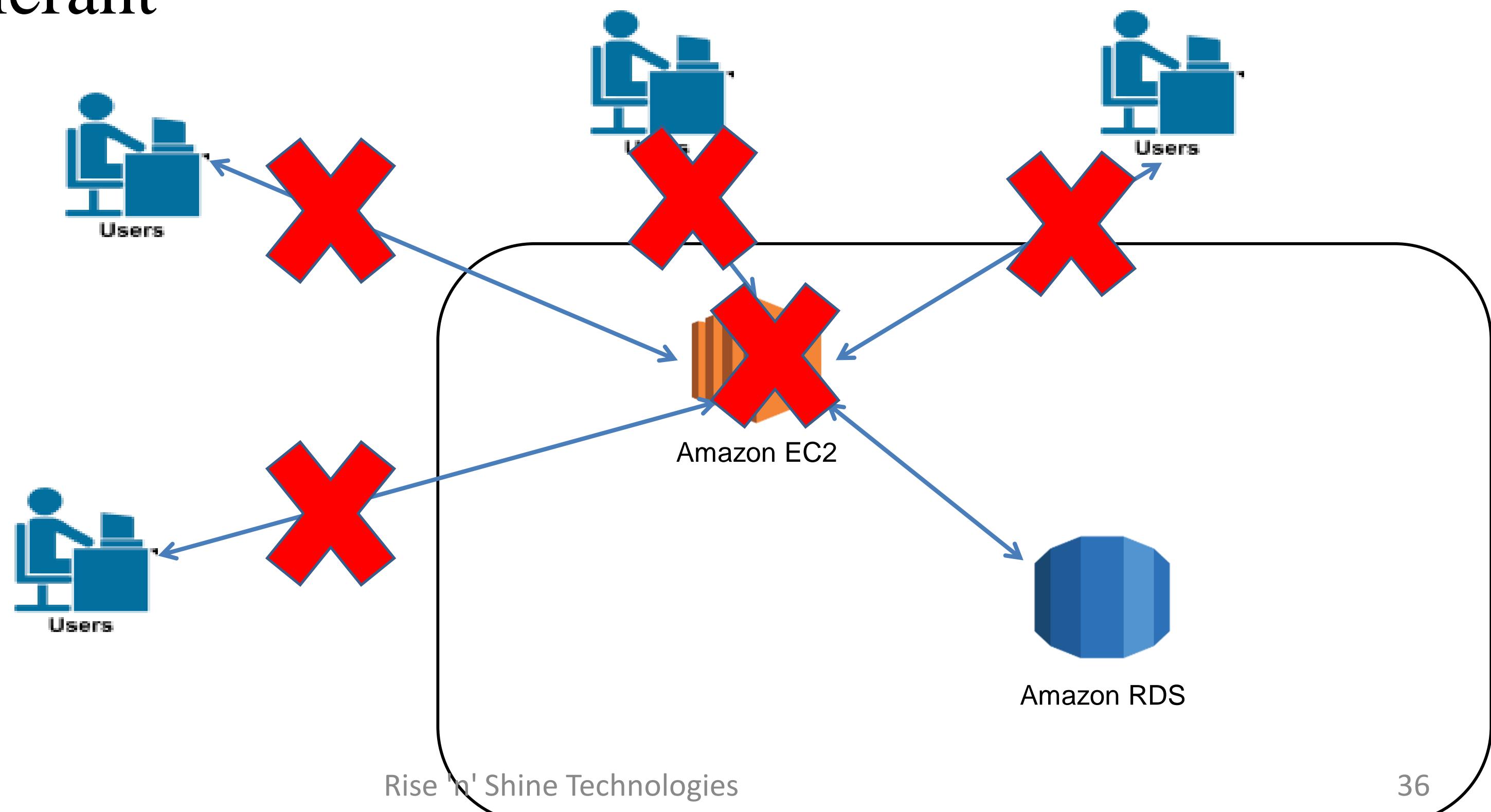
- Extra copies of data are stored but are not available at the time of deletion
- Disk that stores data of multiple tenants is destroyed.

# Cloud Terminology:

- **High Availability** In computing, the term availability is used to describe the period of time when a service is available
- **Fault Tolerant:** is the property that enables a system to continue operating properly in the event of the failure of some (one or more **faults** within) of its components.
- **Scalability:** "Increasing" the capacity to meet the "increasing" workload.
- **Elasticity:** "Increasing or reducing" the capacity to meet the "increasing or reducing" workload.

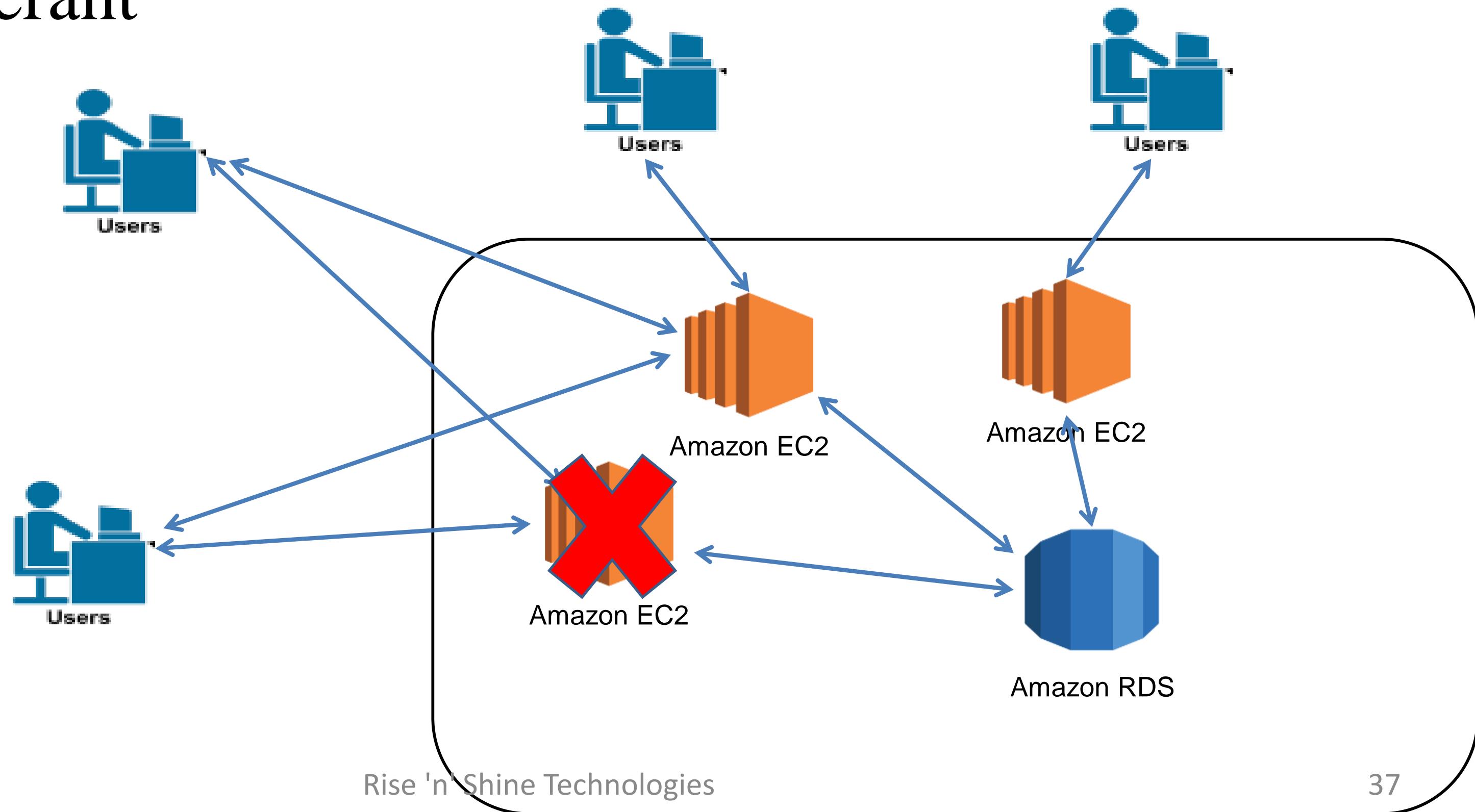
# Cloud Terminology:

- High Availability
- Fault Tolerant



# Cloud Terminology:

- High Availability
- Fault Tolerant



# The Cloud Scales: Customers in 190 Countries



# Summary

- **Common use cases of Infrastructure**
- **Virtualization**
- **Before & after Virtualization**
- **Virtualization approaches(Host & Bare metal Architecture)**
- **What is Cloud & Cloud Computing**
- **History of Cloud Computing**
- **Cloud Computing Architecture**
- **Deployment Models(Public, Private, Hybrid & Community Clouds)**
- **Service Models(IaaS, PaaS, SaaS & XaaS )**
- **Benefits of Cloud Computing**
- **Risks of Cloud Computing**
- **High Availability, Fault Tolerance, Scalability & Elasticity**

relax / it's done



# HA Architecture

# Types of Servers

1) Web Servers

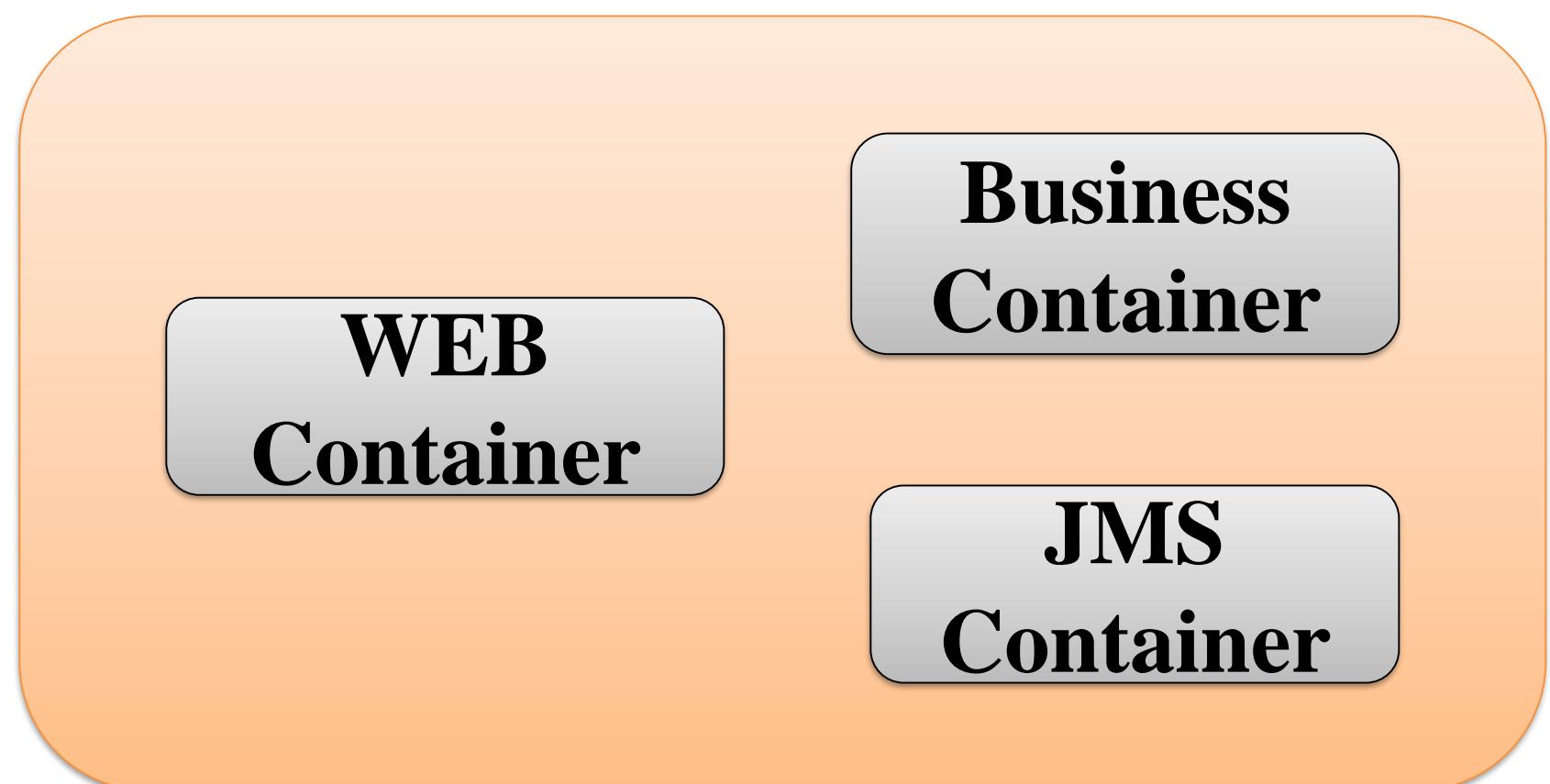
2) Application Servers



**Web Server**

Apache  
OHS  
IHS  
Nginx

RISE 'N' SHINE TECHNOLOGIES



**App Server**

Web Logic  
Web Sphere  
Jboss

66

<b>Web Server</b>	<b>Application Server</b>
<p>A Web server handles the <b>HTTP protocol</b>.</p> <p>Protocol Dependent (Http)</p>	<p>Application Server handles <b>any number of protocols (HTTP, TCP-IP, RMI (t3) ...etc)</b>.</p> <p>Protocol Independent.</p>
<p>Web Server is mostly designed to <b>serve static content</b>.</p>	<p>The application server is used to <b>run business logic</b> or dynamically generating presentation code.</p>
<p><b>It Serves Web Based Applications.</b></p>	<p><b>It serves Web Based Applications and Enterprise Based applications.</b></p>
<p>It provides <b>only Web container</b> (Servlet container + JSP container)</p>	<p><b>It provides Web Container + EJB Container + JMS Container</b></p>
<p>Apache, OHS, IHS, IIS, Sun one iplanet</p>	<p>Weblogic, Websphere, Oracle Application Server, JBOSS, GF</p>

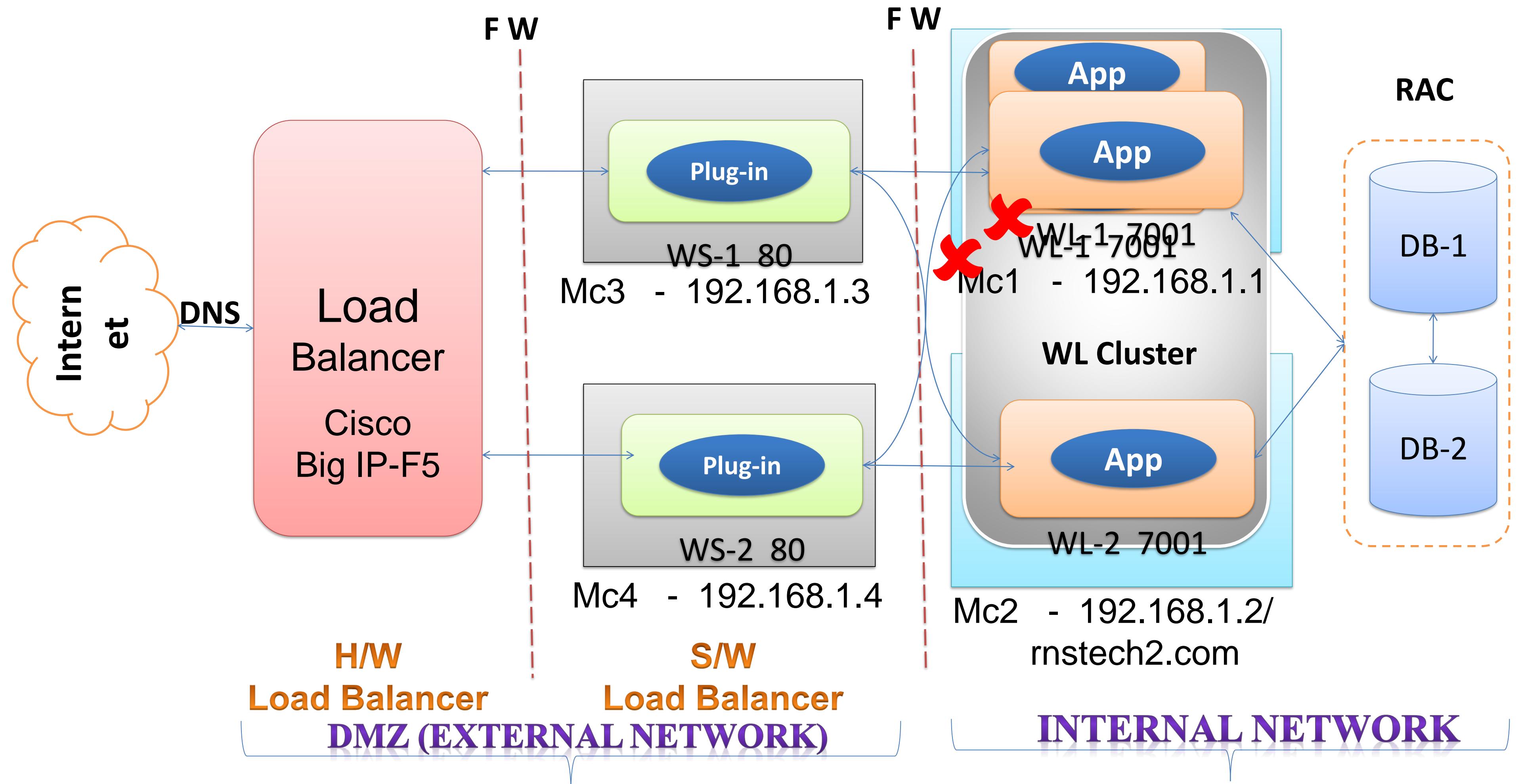
# HA Environment Features

1) High Availability

2) Fail Over

3) Load Balancing

4) Security



- <http://192.168.1.1:7001/App>
- <http://192.168.1.1:7002/App>

- <http://192.168.1.3:80/App>
- <http://192.168.1.4:80/App>
- <http://gmail.co.in>

# URLs:

- <http://192.168.1.1:7001/App>
- <http://192.168.1.2:7001/App>
- <http://192.168.1.3:80/App>
- <http://192.168.1.4:80/App>
- <http://gmail.co.in>

# High Availability Explained



By  
Reyaz Shaik

# High Availability is in the Eye of the beholder

- CEO: we don't loose sales
- Sales: we can extend our offer basing on HA level
- Accounts managers: we don't upset our customers (that often)
- Developers: we can be proud – our services are working ;)
- System engineers: we can sleep well (and fsck, we love to!)
- Technical support: no calls? Back to WoW then.. ;)

# So How many 9's?

**Monthly:** 1 hour of outage means  $100\% - 0.13888 \approx \mathbf{99.86112}$  of availability

**Yearly:** 1 hour of outage means  $100\% - 0.01142 \approx \mathbf{99.98858}$  of availability

Availability	Downtime (year)	Downtime (month)
90% ("one nine")	36.5 days	72 hours
95%	18.25 days	36 hours
97%	10.96 days	21.6 hours
98%	7.30 days	14.4 hours
99% ("two nines")	3.65 days	7.2 hours
99.5%	1.83 days	3.6 hours
99.8%	17.52 hours	86.23 minutes
99.9% ("three nines")	4.38 hours	21.56 minutes
99.99% ("four nines")	52.56 minutes	4.32 minutes
<b>99.999% ("five nines")</b>	<b>5.26 minutes</b>	<b>25.9 seconds</b>

## High Availability

- In computing, the term availability is used to describe the **period of time** when a **service is available**, as well as the time required by a system to respond to a request made by a user.
- In information technology, **high availability** refers to a system or component that is continuously operational for a desirably long length of time. **Availability** can be measured relative to "**100% operational**" or "**never failing.**"

## Fault Tolerance

- A good way to think of it is that you have **two separate machines** that are **mirrored**. In the event that the main system has a **hardware failure**, the secondary system takes over and there is **zero downtime**.

# How High Availability Works?

To create a highly available system, three characteristics should be present:

- Redundancy
- Monitoring
- Failover

# How High Availability Works?

- **Redundancy**

In computing, *redundancy* means that there are **multiple components** that can perform the **same task**. This eliminates the single point of failure problem by allowing a second server to take over a task if the first one goes down or becomes disabled.

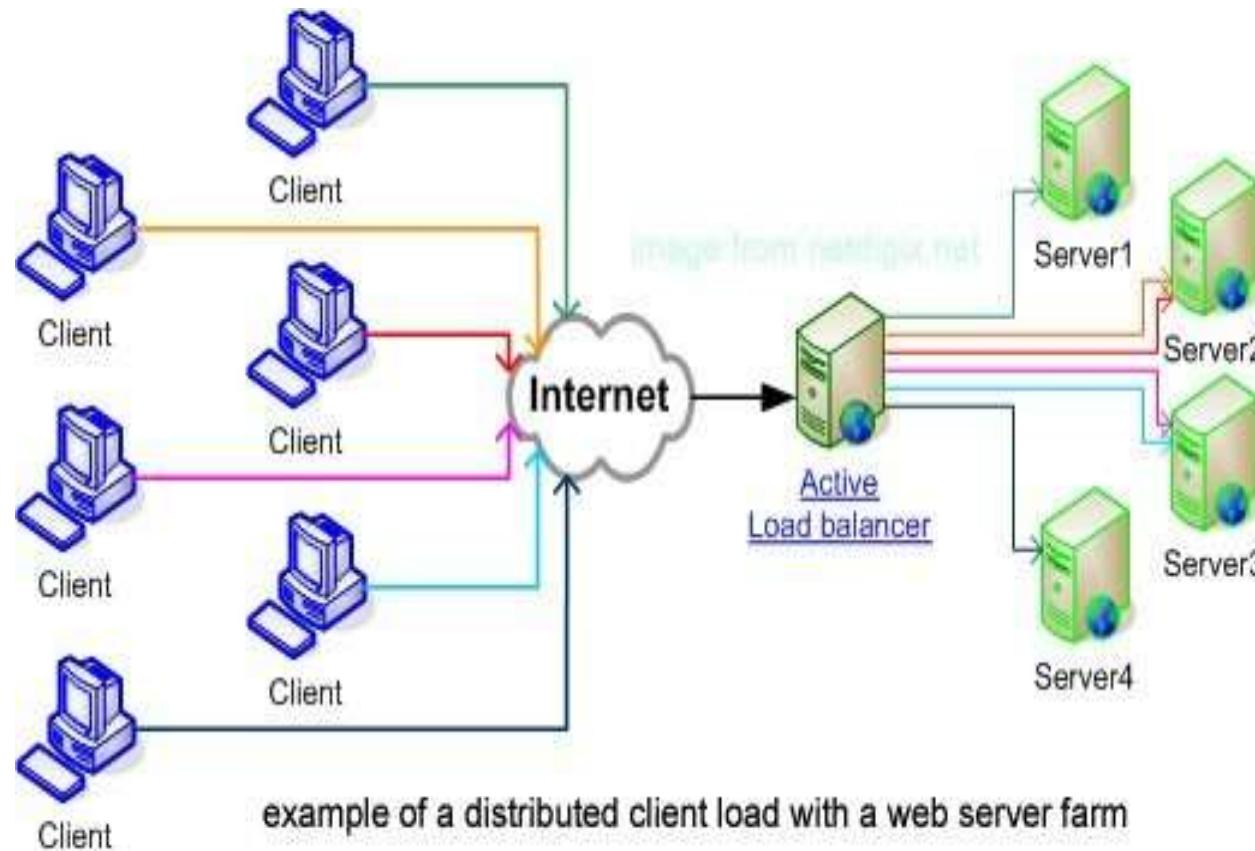
- **Monitoring**

In a highly available setup, the system needs to be able to monitor itself for failure. This means that there are regular checks to ensure that all components are working properly

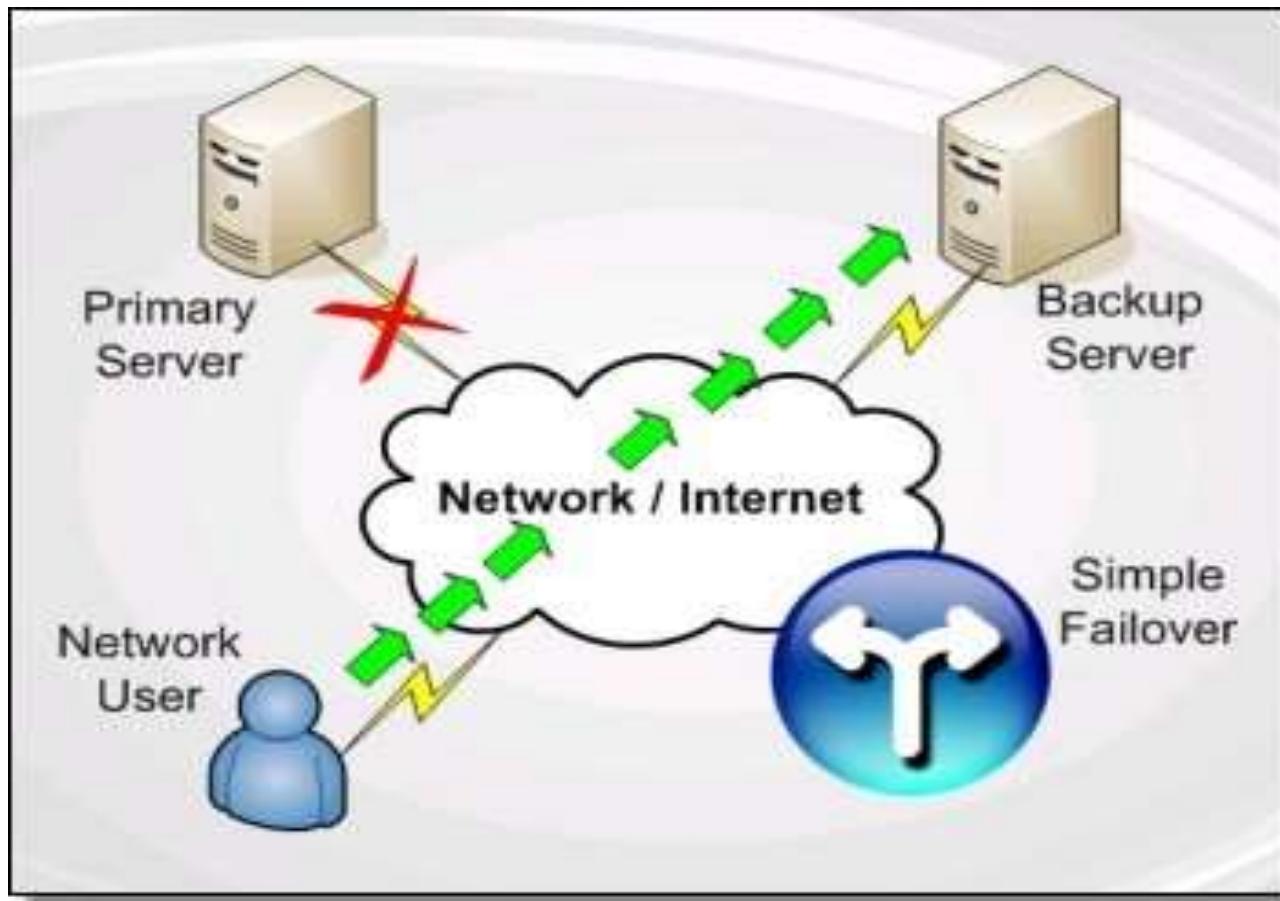
- **Failover**

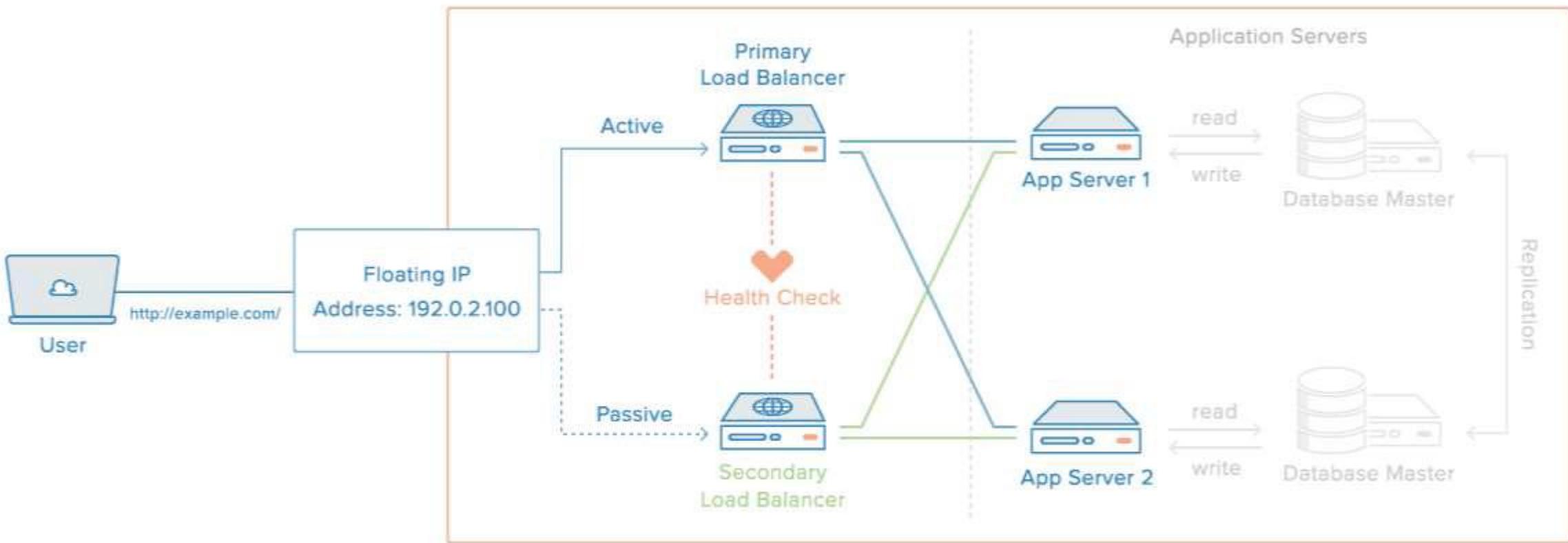
*Failover is the process by which one node takes over the job of another in the event that one becomes disabled. This comes as a result of monitoring for failures by the system..*

# Load-balancing



## Failover:





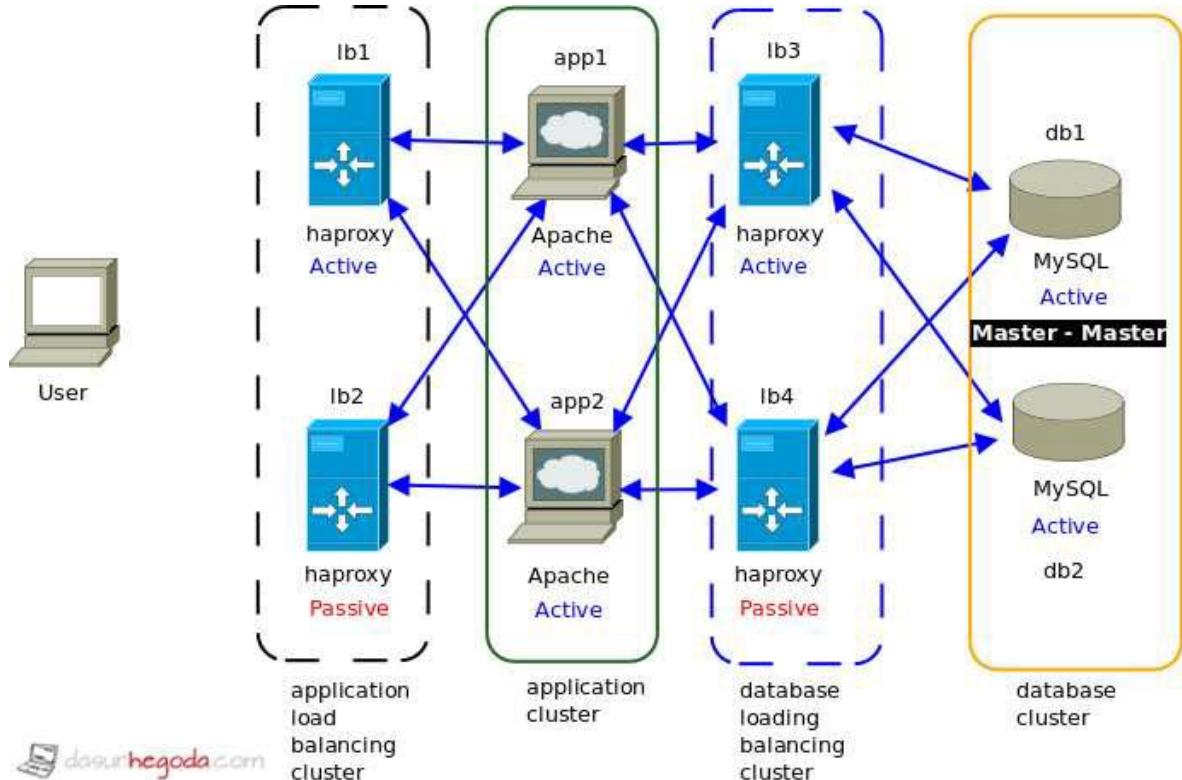
- 1 Active/Passive Cluster is healthy
- 2 Primary node fails
- 3 Floating IP is assigned to Secondary node

NYC3 DATACENTER

# HA vs FT

	Windows Server Failover Clustering(HA)	Fault Tolerant Solution
Hardware Failure	✓	✓
OS Level Failure	✓	
Application Failure	✓	

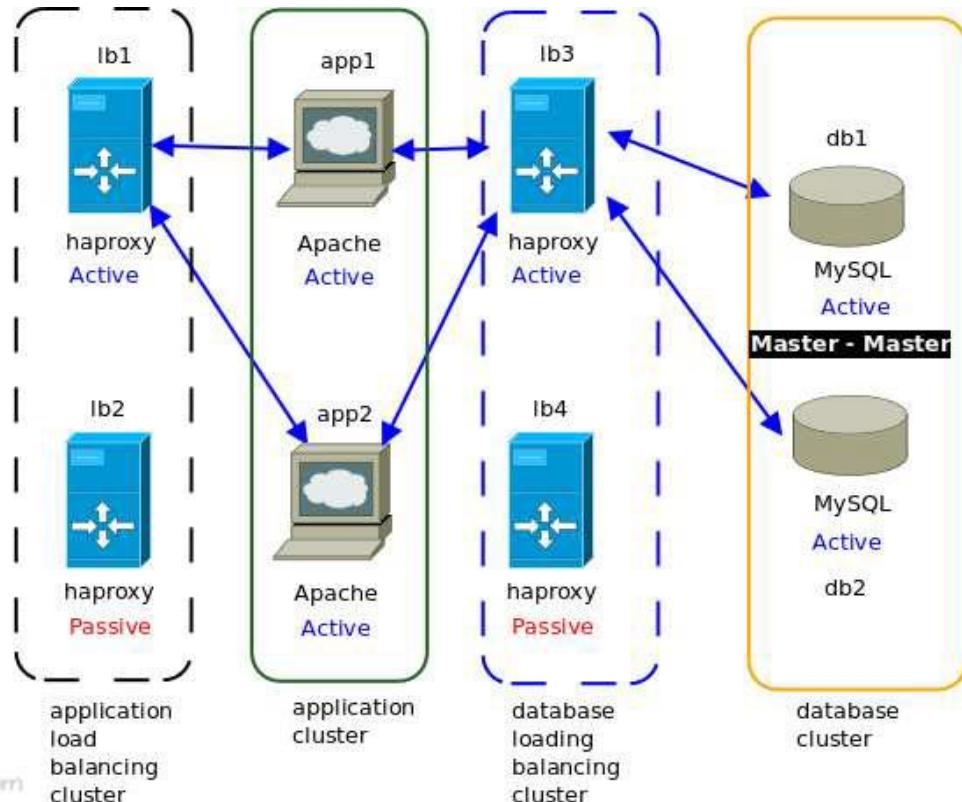
# HA Example



Without beating around the bush let me elaborate this more.

- **Ib1 & Ib2** are load balancers for the application servers. All together we call them application load balancing cluster.
- **app1 & app2** are application servers. All together we call them application cluster.
- **Ib3 & Ib4** are load balancers for the database servers. All together we call them database load balancing cluster.
- Then we have the database cluster at the end that are **db1** and **db2**. All together we call them database cluster.
- **Active** means particular component is accepting requests and **passive** means particular component is not accepting requests but when the active component is down passive component will take over and will start accepting requests.

# HA Example

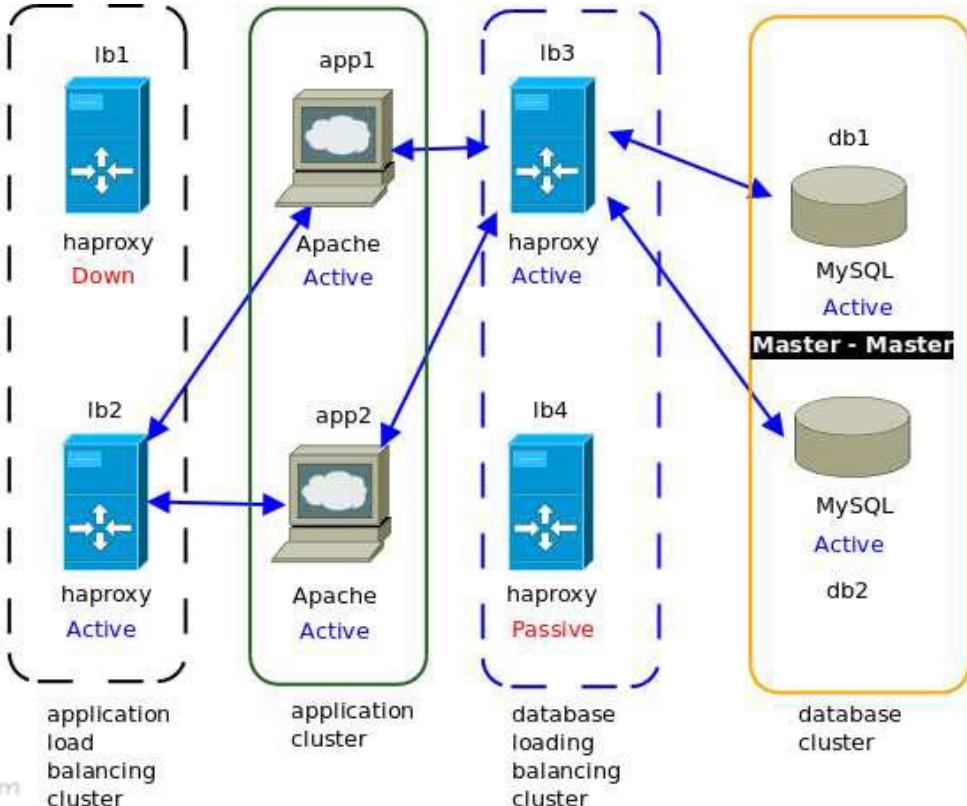


On a happy day user will make a request.

- The request will be accepted by lb1 and depending on the load balancing algorithm in lb1 request will be passed to the app1 or app2.
- From there the request will be handed over to the lb3 by app1 or app2. lb3 will communicate with the database.
- The response will follow the same path as the request's path.

Okay now let's get ready for some action because you can't expect a happy day everyday.

# What if the lb1 is down

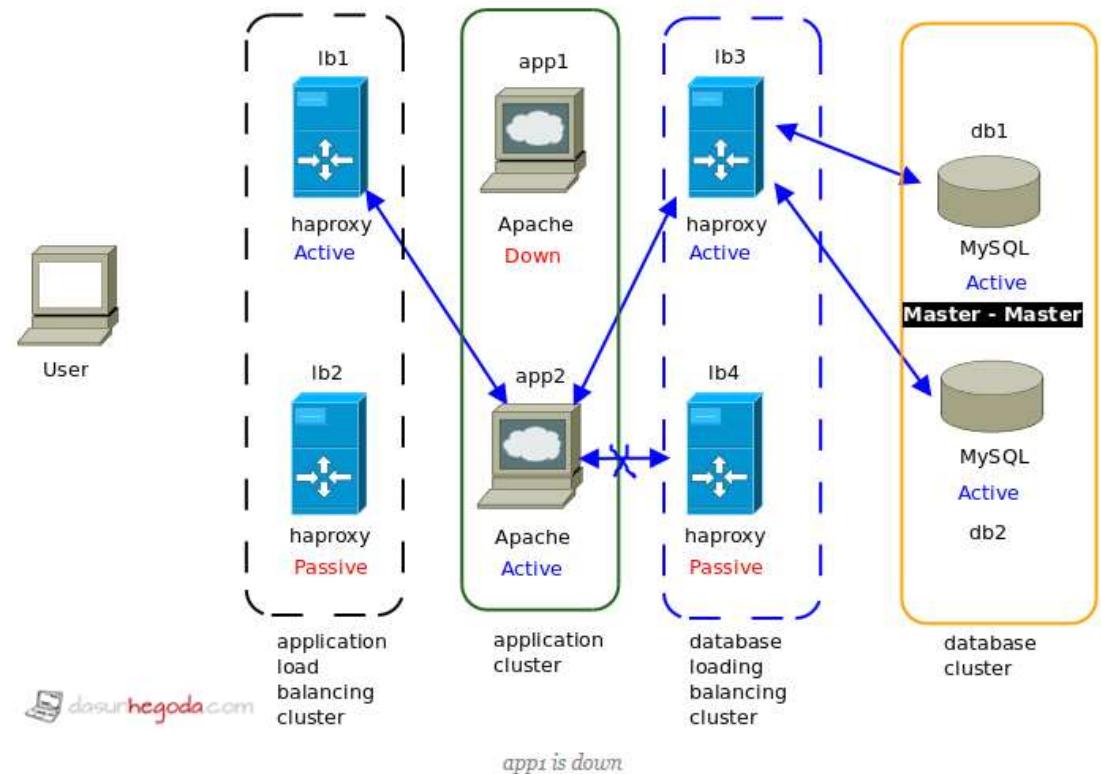


Here even though lb1 is down lb2 has taken over and now lb2 is in active status.

System is functional even though one load balancer has failed.

Best part is here users will not experience any downtime of the system due to the failure of lb1.

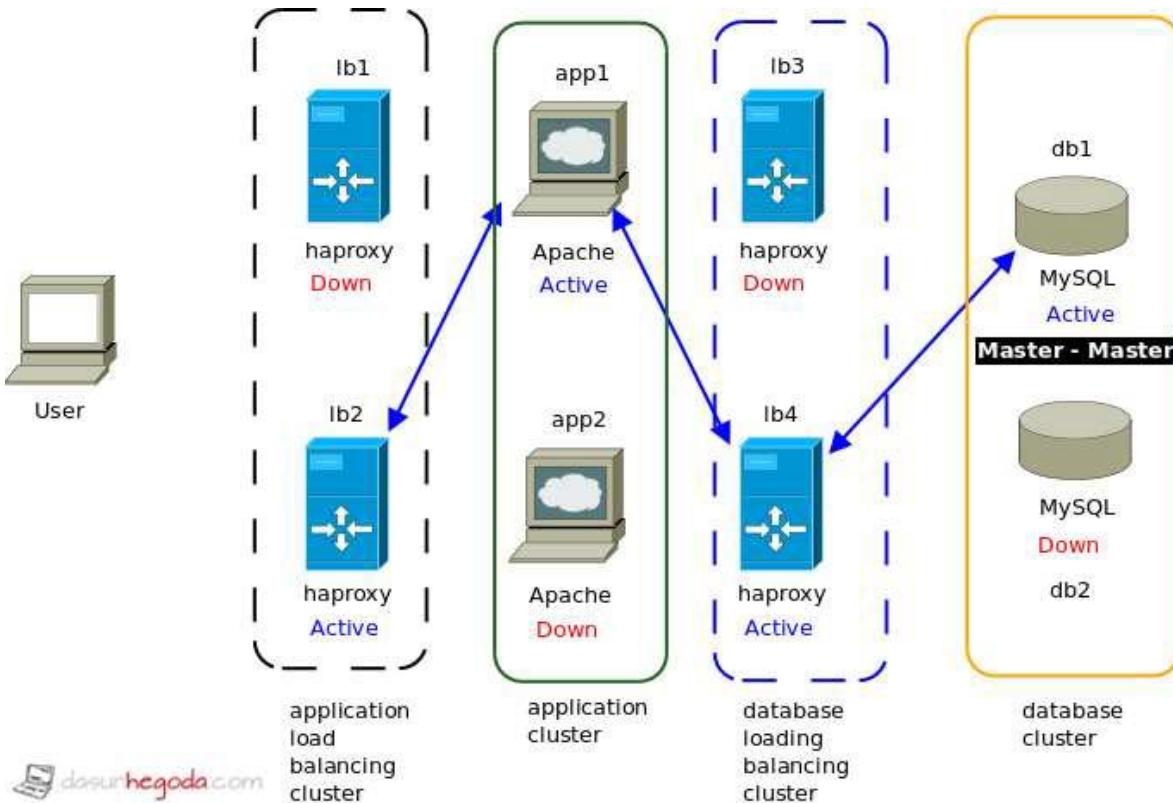
# Let's assume app1 is down



Now you can see even though one application sever went unavailable system is functional without an issue.

Just like the previous example.

# Now let's take the worst case scenario where 4 components are down

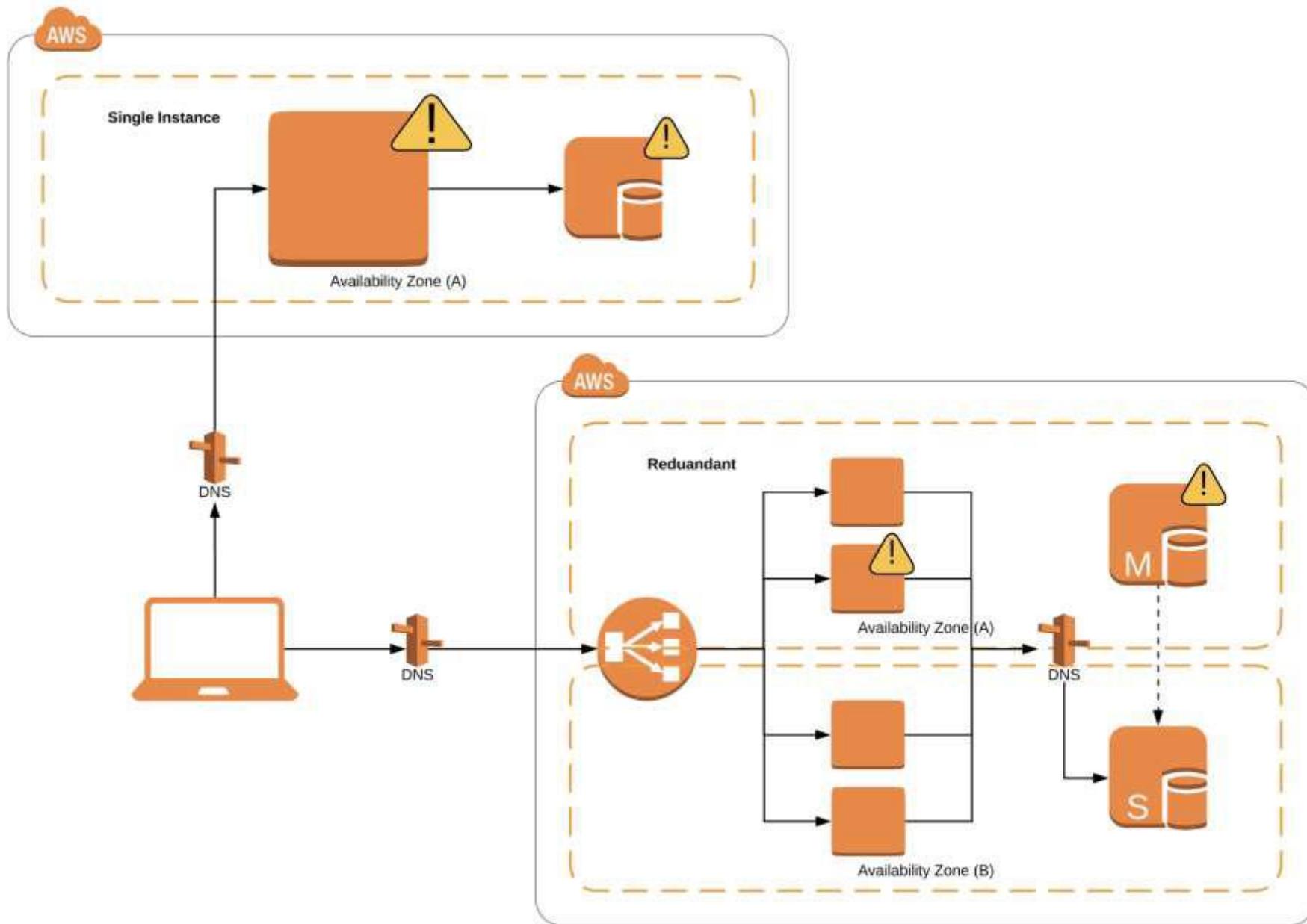


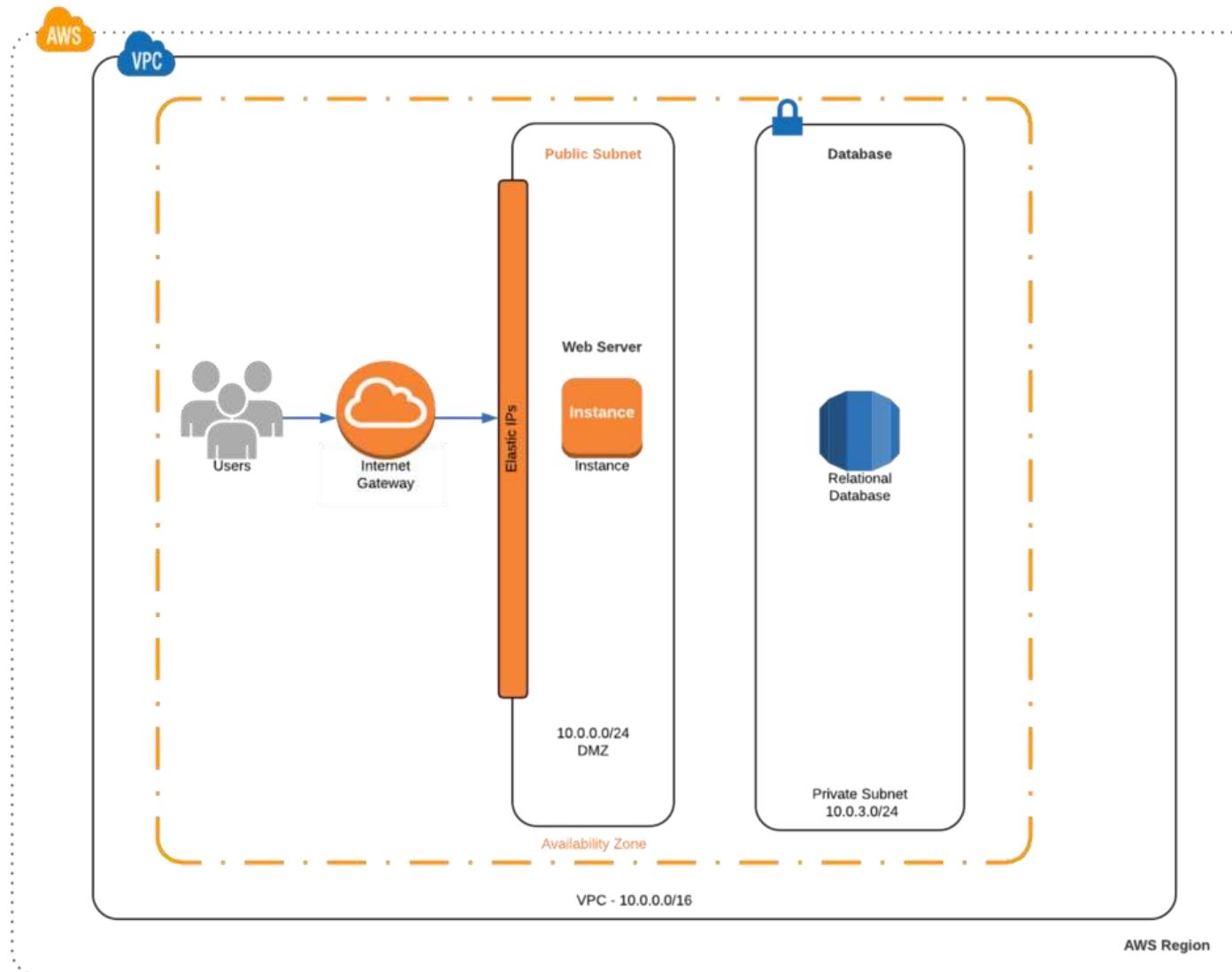
Oh! Nooohhh!!!

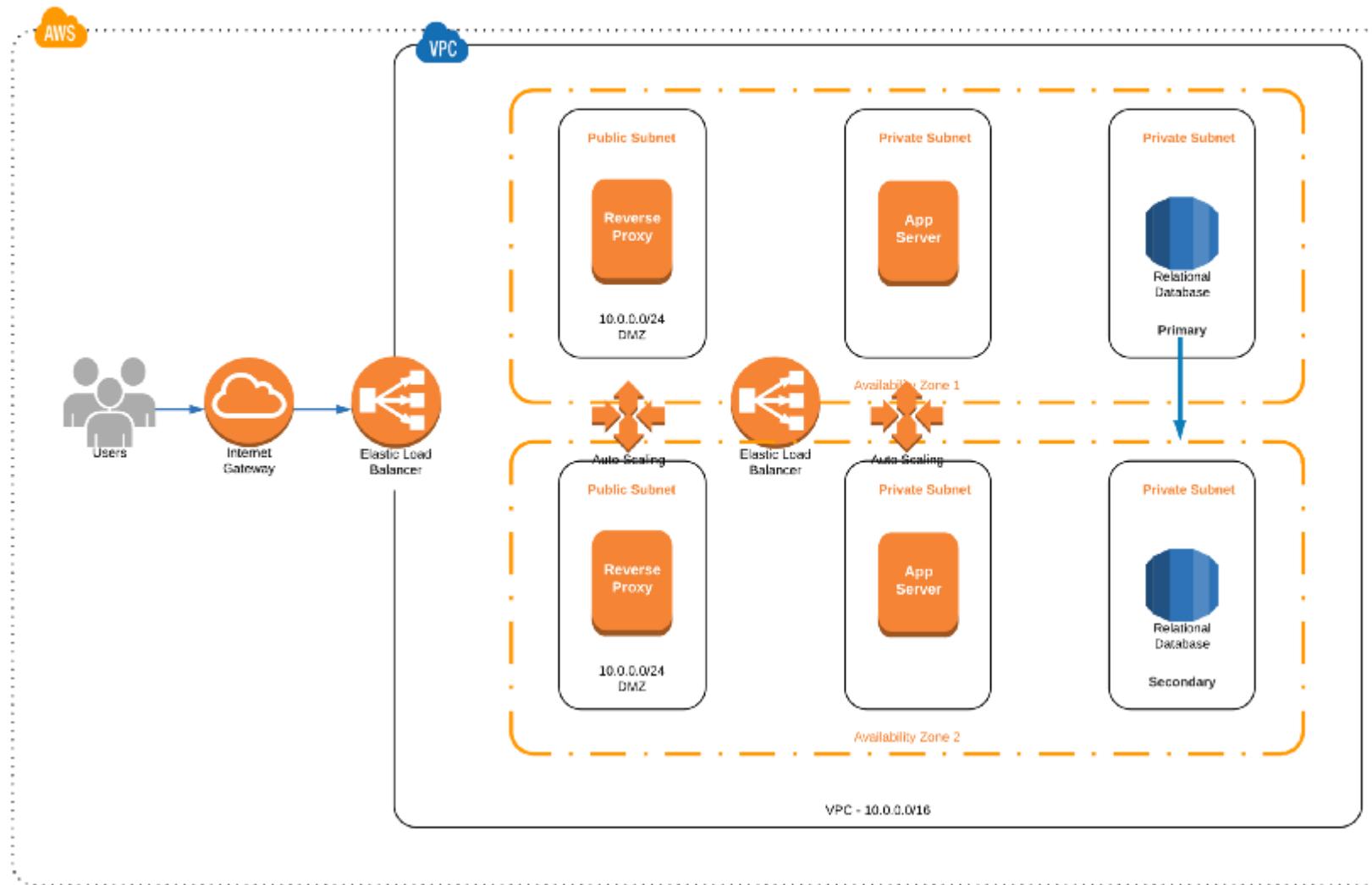
Here a single component is unavailable from each cluster that means four components are not unavailable altogether.

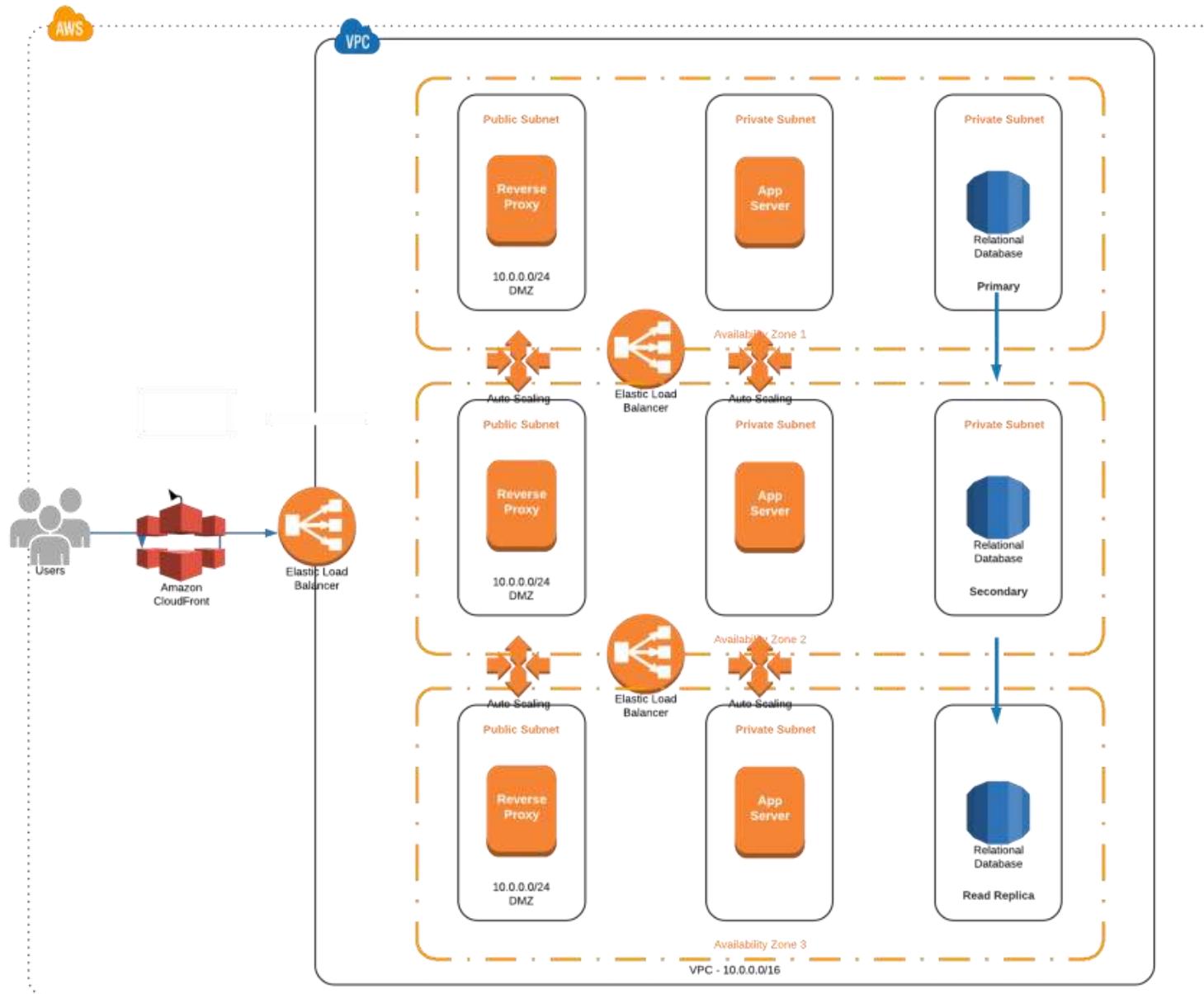
Will the application be able to function as on a happy day? Yes, a big yes.

You can image the power of having high availability in your application. Even though a single component or multiple components are unavailable your application will be available for it's intended parties without an issue. Perfect right!!!.













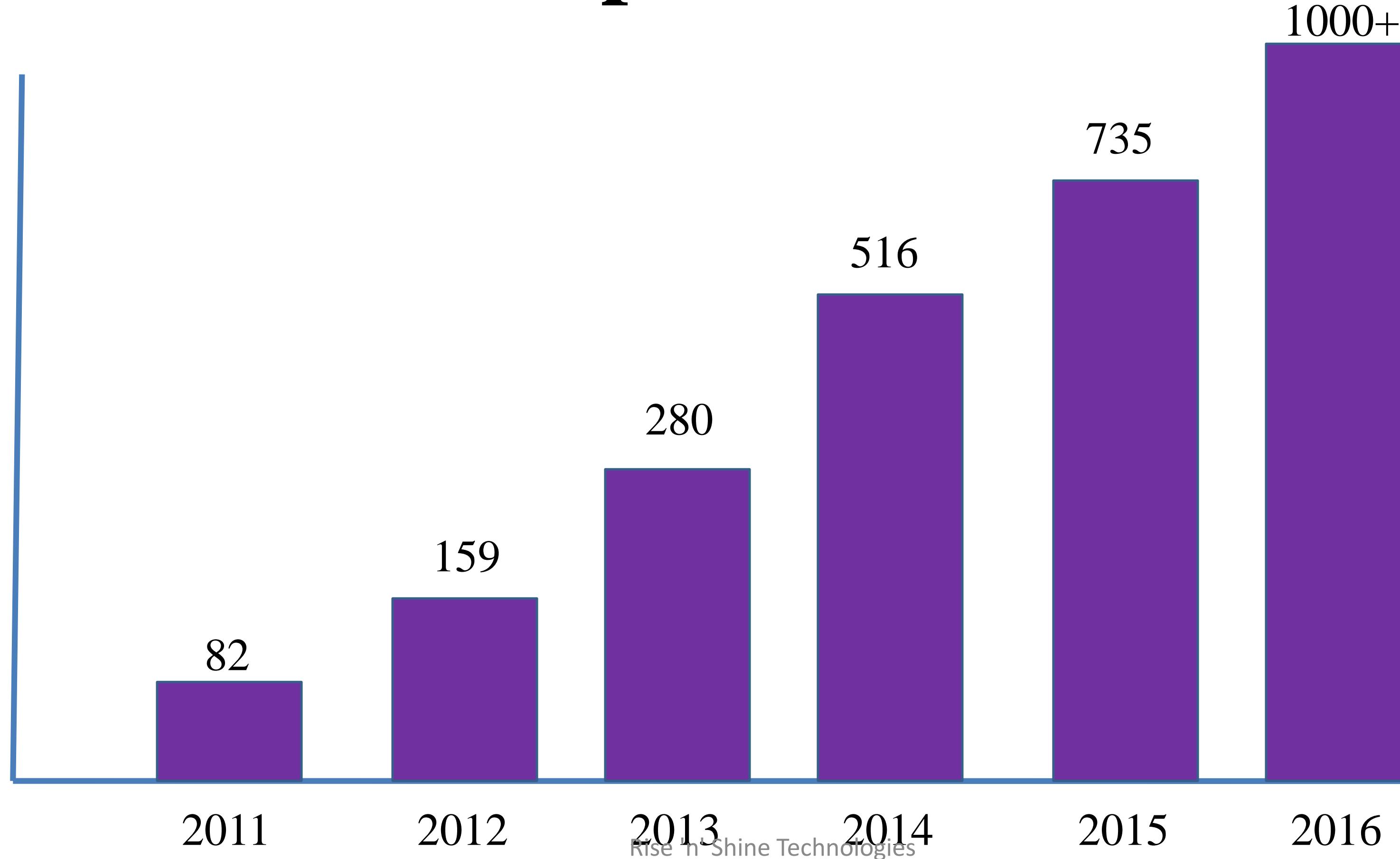
# Introducing Amazon Web Services:

- Amazon Web Services or AWS is a comprehensive public cloud computing platform.
- It offers a variety of web-based products and services on an on-demand and pay-per-use basis.

# Why Learn AWS?

- Fastest growing cloud computing platform on the Planet
- Largest public cloud computing platform on the planet
- More and more organizations are outsourcing their IT to AWS

# AWS new Service Announcements & Updates



# About AWS Certifications

Specialty

**Security**

**Advanced  
Networking**

**Big Data**

Professional  
Tier

**Certified Solutions  
Architect  
Professional**

**DevOps  
Professional**

Associate  
Tier

**Certified Solutions  
Architect Associate**

**Certified Developer  
Associate**

**Certified SysOps  
Administrator  
Associate**

# Gartner's Magic quadrant

- In Aug 2016, AWS was named as a leader in the IAAS Magic Quadrant for the 6th consecutive Year.
- 90% of the Cloud market by AWS
- 5% of the Cloud Market by Microsoft

# AWS Goal



# AWS Platform

Game Development

Artificial Intelligence

Messaging

Business  
Productivity

Internet Of Things

Desktop & App  
Streaming

Application Services

Developer Tools

Mobile Services

Analytics

Security & Identity

Mgmt Tools

Migration

Storage

Databases

Networking & Content Delivery

Compute

# AWS Global Infrastructure



Region & Number of Availability Zones	New Region (coming soon)
US East N. Virginia (6), Ohio (3)	China Beijing (2)
Europe	
US West N. California (3), Oregon (3)	Frankfurt (3), Ireland (3), London (2)
South America	
Asia Pacific Mumbai (2), Seoul (2), Singapore (2), Sydney (3), Tokyo (3)	São Paulo (3)
AWS GovCloud (US-West)	
Canada Central (2)	
	Bahrain
	China
	France
	Hong Kong
	Sweden
	AWS GovCloud (US-East)

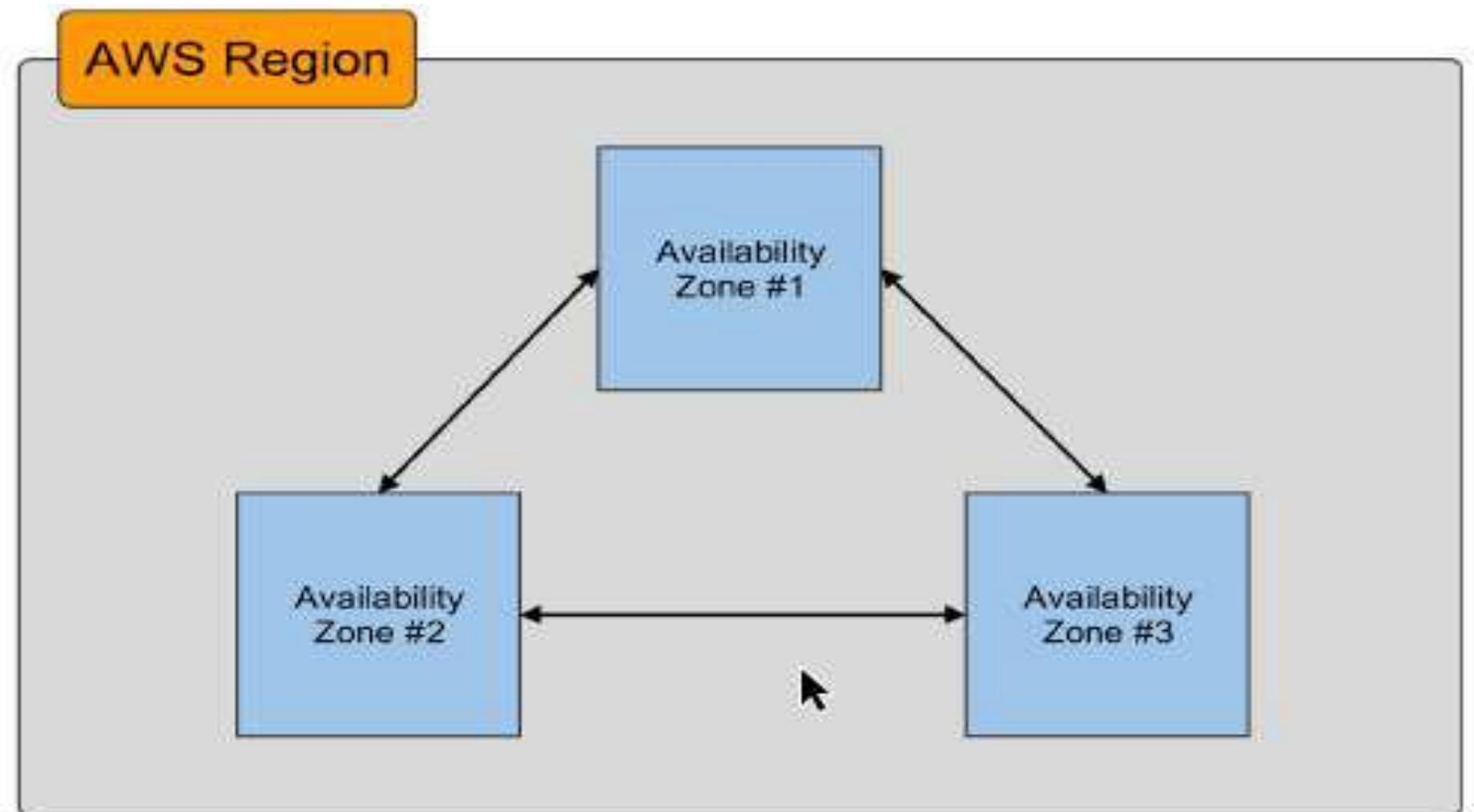
14 Regions & 38 Availability Zones – by Dec 2016  
4 More Regions & 11 More Availability Zones - 2017

# What is a Region & AZ

- A Region is a Geographical area. Each Region consists of 2 (or more) Availability Zones.
- An Availability zone(AZ) is a Simply a Data Center.
- A list of regions and their corresponding codes is provided here for your reference.
- The code is basically how AWS refers to its multiple regions:

Region	Name
US West (Oregon) Region	us-west-2
US West (N. California) Region	us-west-1
US East (Ohio) Region	us-east-2
US East (N. Virginia) Region	us-east-1
Asia Pacific (Mumbai) Region	ap-south-1
Asia Pacific (Seoul) Region	ap-northeast-2
Asia Pacific (Singapore) Region	ap-southeast-1
Asia Pacific (Sydney) Region	ap-southeast-2
Asia Pacific (Tokyo) Region	ap-northeast-1
Canada (Central) Region	ca-central-1
China (Beijing) Region	cn-north-1
EU (Frankfurt) Region	eu-central-1
EU (Ireland) Region	eu-west-1

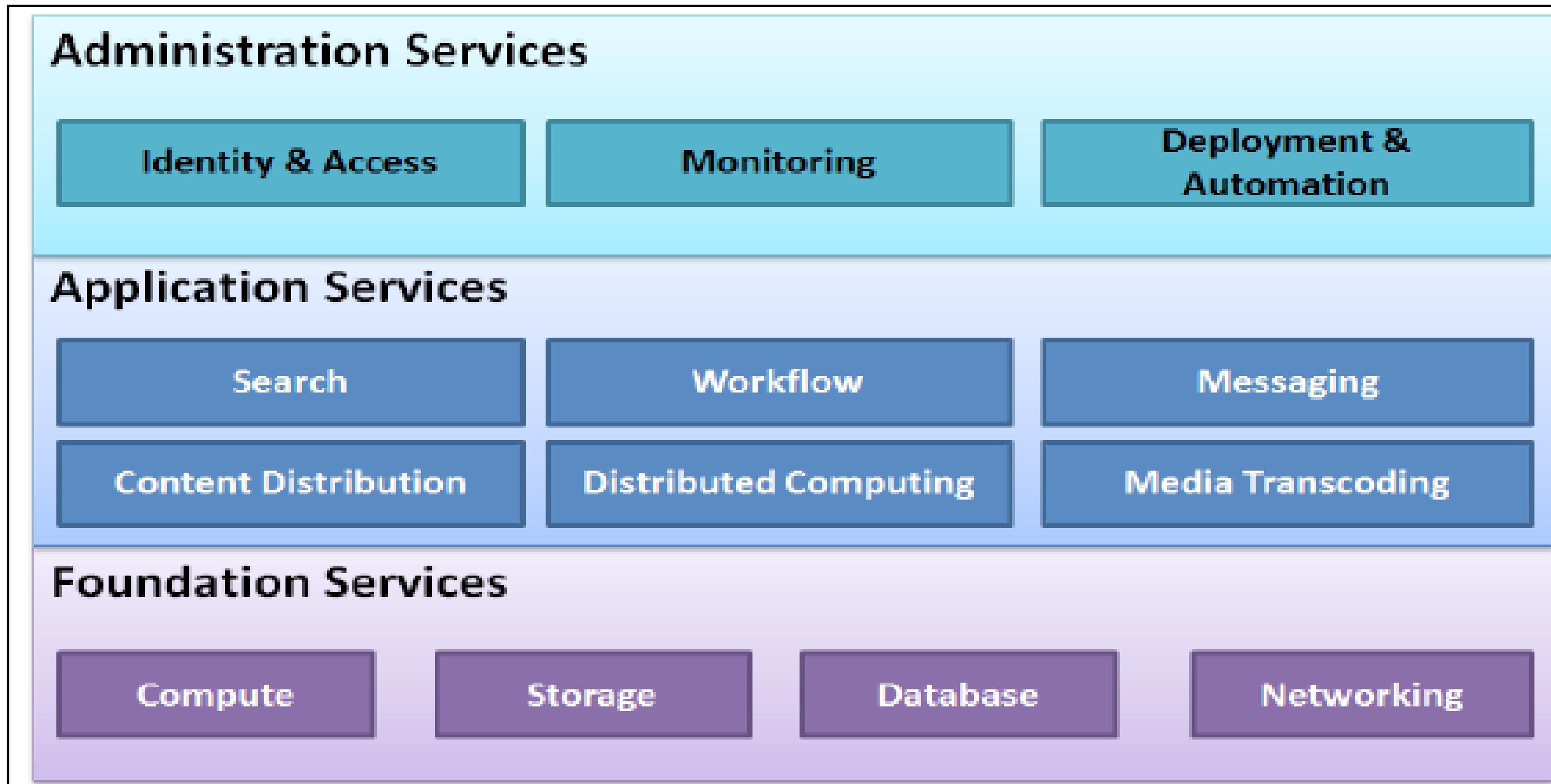
# What is a Region & AZ



# AWS Platform Overview

- The AWS platform consists of a variety of services that you can use either in isolation or in combination based on your organization's needs.
  - Foundation services
  - Application services
  - Administration services

# AWS Platform Services



# AWS Platform Services

- **Foundation services:** This is generally the pillars on which the entire AWS infrastructure commonly runs on, including the compute, storage, network, and databases.
- **Application services:** This class of services is usually more specific and generally used in conjunction with the foundation services to add functionality to your applications.
- **Administration services:** This class deals with all aspects of your AWS environment, primarily with IAM tools, monitoring your AWS services and resources, application deployments, and automation.

# The Foundation Services - Compute

- **Elastic Compute Cloud (EC2):** EC2 or Elastic Compute Cloud is a web service that provides flexible, resizable, and secure compute capacity on an on-demand basis.
- **EC2 Container Service:** A recently launched service, the EC2 Container Service, allows you to easily run and manage docker containers across a cluster of specially created EC2 instances.
- **Amazon Virtual Private Cloud (VPC):** VPC enables you to create secure, fully customizable, and isolated private clouds within AWS's premises.

# Storage Services:

- **Simple Storage Service (S3):** S3 is a highly reliable, fault tolerant, and fully redundant data storage infrastructure provided by AWS.
- **Elastic Block Storage (EBS):** EBS is a raw block device that can be attached to your compute EC2 instances to provide them with persistent storage capabilities.
- **Amazon Glacier:** It is a similar service of S3. It offers long-term data storage, archival, and backup services to customers.
- **Amazon Elastic File System:** (EFS) provides scalable and high-performance storage to EC2 compute instances in the form of an NFS filesystem.

# Databases services:

- **Amazon Relational Database Service (RDS):** RDS provides a scalable, high-performance relational database system such as MySQL, SQL Server, PostgreSQL, and Oracle in the cloud.
- **Amazon DynamoDB:** DynamoDB is a highly scalable NoSQL database as a service offering provided by AWS.
- **Amazon Redshift:** Amazon Redshift is a data warehouse service that is designed to handle and scale to petabytes of data. It is primarily used by organizations to perform real-time analytics and data mining.

# Networking services:

- **Elastic Load Balancer (ELB):** ELB is a dynamic load balancing service provided by AWS used to distribute traffic among EC2 instances.
- **Amazon Route 53:** Route 53 is a highly scalable and available DNS web service provided by AWS. Rather than configuring DNS names and settings for your domain provider, you can leverage Route 53 to do the heavy lifting work for you.

# Content Distribution and Delivery Service:

- **Amazon CloudFront:** It is basically a content delivery web service that can be used to distribute various types of content, such as media, files, and so on, with high data transfer speeds to end users globally.

# Workflow and Messaging Services:

- **Amazon Simple Notification Service (SNS):** SNS is a simple, fully managed push messaging service provided by AWS. You can use it to push your messages to mobile devices (SMS service) and even to other AWS services as API calls to trigger or notify certain activities.
- **Amazon Simple Email Service (SES):** As the name suggests, SES is used to send bulk e-mails to various recipients. These e-mails can be anything, from simple notifications to transactions messages, and so on.

# Administration Services

- **Monitoring:**

**Amazon CloudWatch** is a monitoring tool provided by AWS that you can use to monitor any and all aspects of your AWS environment, from EC2 instances to your RDS services to the load on your ELBs, and so on.

- **Identity and access management (IAM)**

**IAM:** AWS provides a rich set of tools and services to secure and control your infrastructure on the cloud.

# Support Plans provided by AWS:

- **Basic Support:**
  - This is the most basic level of support provided by AWS.
  - This support level provides you with access to the AWS community forums.
  - You can additionally contact customer services for any queries related to your account and bill generation.
- **Developer Support:**
  - This is a paid support service (\$49 per month).
  - You can create and raise tickets for your support case, which is generally answered within 12 working hours.

# Support Plans provided by AWS:

- **Business Support:**
  - This is a paid support service as well and is generally meant for enterprise-level customers running production workloads on AWS.
  - The SLAs for this support are much higher as a case has to be answered within an hour from its creation.
- **Enterprise Support:**
  - A paid support service with the highest SLA available (15 minutes);
  - these cases are generally handled by a separate team at AWS called the Technical Account Manager (TAM) who are subject matter experts in their own fields.

# AWS Service Overview

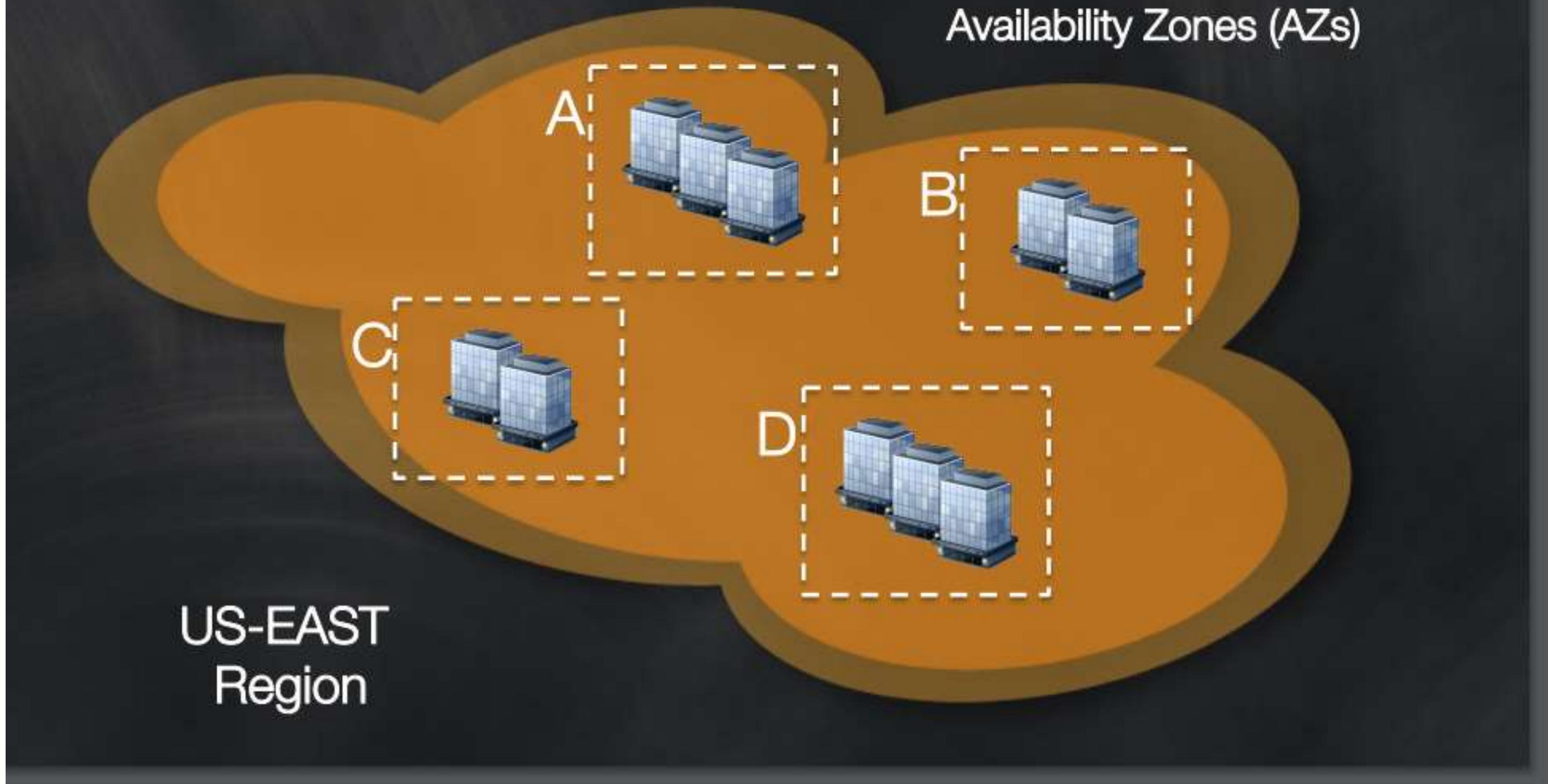
# AWS Global Infrastructure

11 regions  
28 availability zones  
52 edge locations



You choose where your apps and data go!

# AWS Global Infrastructure



# Compute Services

## Amazon EC2

Elastic **Virtual servers**  
in the cloud



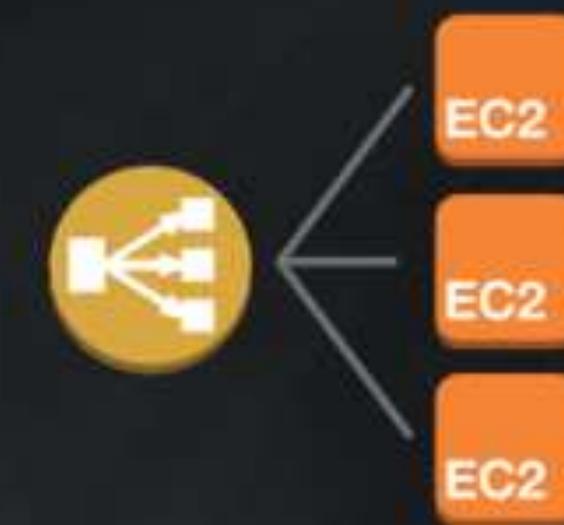
## Auto Scaling

**Automated scaling**  
of EC2 capacity



## Elastic Load Balancing

Dynamic **traffic distribution**



# Networking Services

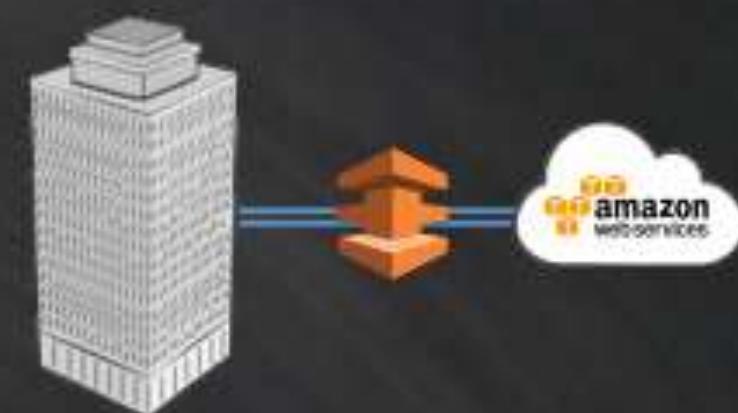
## Amazon VPC

Private, isolated  
section of the AWS  
Cloud



## AWS DirectConnect

Private connectivity  
between AWS and your  
datacenter



## Amazon Route 53

Domain Name System  
(DNS) web service.



# Storage Services

## Amazon EBS

Block storage for use with Amazon EC2



## Amazon S3

Internet scale storage via API



Images  
Videos  
Files  
Binaries  
Snapshots

## Amazon Glacier

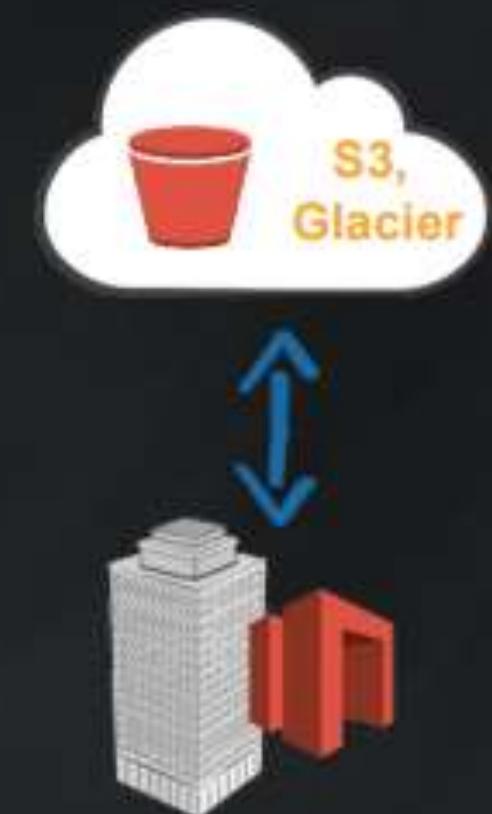
Storage for archiving and backup



Images  
Videos  
Files  
Binaries  
Snapshots

## AWS Storage Gateway

Integrates on-premises IT and AWS storage



# Database Services

## Amazon RDS

Managed **relational**  
database service



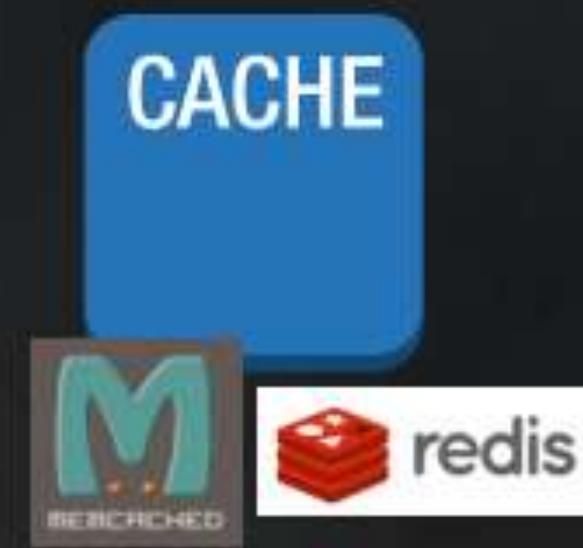
## Amazon DynamoDB

Managed **NoSQL**  
database service



## Amazon ElastiCache

In-Memory **Caching**  
Service



# Administration

## AWS IAM (Identity & Access Mgmt)

Manage [users](#),  
[groups](#) &  
[permissions](#)



## Amazon CloudWatch

[Monitor](#) resources



## AWS CloudTrail

AWS API call [logging](#) for  
governance &  
compliance



# Application Services

## Amazon CloudFront

Distribute content  
globally



## Amazon CloudSearch

Managed search  
service



## Amazon Elastic Transcoder

Video transcoding  
in the cloud



# Big Data Services

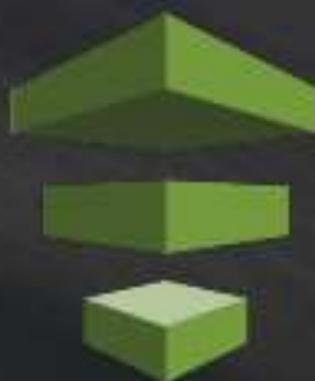
## Amazon EMR (Elastic Map Reduce)

Hosted **Hadoop** framework



## AWS Data Pipeline

**Move data** among AWS services and on-premises data sources



## Amazon Redshift

Petabyte-scale **data warehouse** service



## Amazon Kinesis

Real time processing of streaming data, at any scale



# Deployment

## AWS OpsWorks

Dev-Ops framework  
for application lifecycle  
management



## AWS CloudFormation

Templates to deploy  
& manage



## AWS Elastic Beanstalk

Automate resource  
management





# AWS Sign Up!!!



# Getting started with AWS

- A Free Tier
- No charge for a period of 12 months from the date of the actual signup
- For a complete insight into the free tier usage, check
  - <http://aws.amazon.com/free/>

# AWS Free Tier Account

AWS Product	What's free?
Amazon EC2	750 hours per month of Linux and Windows micro instance usage
Amazon S3	5 GB of standard storage 20,000 get requests 2,000 put requests
Amazon RDS	750 Hours of RDS Single-AZ micro instance 20 GB of DB Storage: any combination of general purpose (SSD) or magnetic 20 GB for backups 10,000,000 I/Os
Amazon ELB	750 hours per month 15 GB of data processing

# Sign Up for AWS

- <http://aws.amazon.com/>

The screenshot shows the AWS homepage with a dark blue header and an orange main content area. The header includes the AWS logo, navigation links (Contact Sales, Products, Solutions, Pricing, Getting Started, More, English, My Account), and a yellow 'Sign In to the Console' button. The main content features a large orange banner with the text 'Introducing EC2 P3 Instances' and a subtext 'Accelerate machine learning and HPC workloads with the most powerful GPU compute instances'. Below this is a 'Learn more »' link. To the right is a graphic of four server racks with GPUs, connected by a network of dashed lines. A callout box on the right says 'Manage Your Resources' with a 'Sign In to the Console' button, and 'AWS Console Mobile App' with a 'Download the Mobile App »' link. Navigation arrows are on the left and right sides of the banner.

nu

aws

Contact Sales Products Solutions Pricing Getting Started More English My Account Sign In to the Console

Introducing EC2 P3 Instances

Accelerate machine learning and HPC workloads with the most powerful GPU compute instances

Learn more »

Manage Your Resources

Sign In to the Console

AWS Console Mobile App

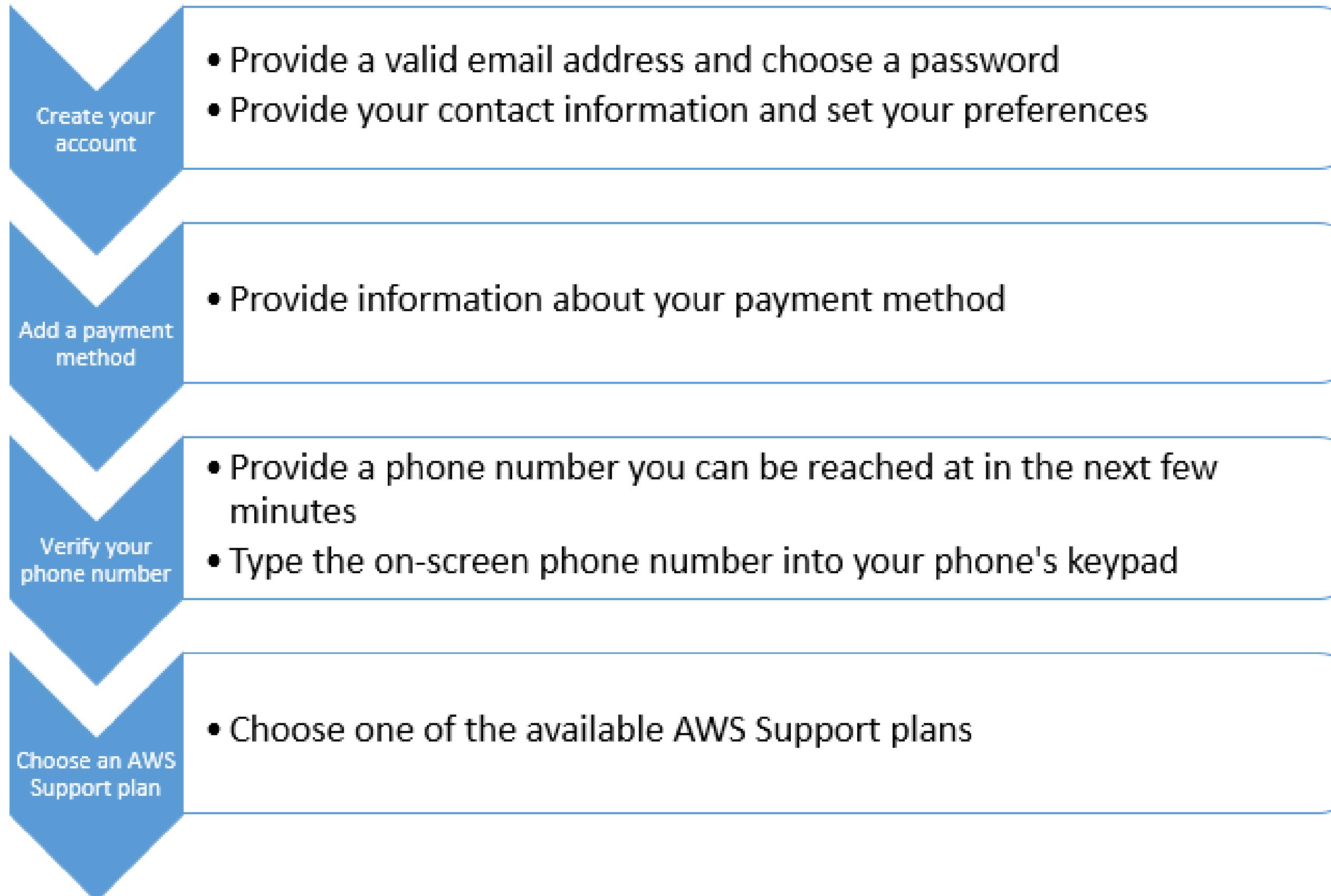
View your resources on iOS and Android devices

Download the Mobile App »

Rise 'n' Shine Technologies

77

# Sign Up for AWS



# Sign Up for AWS

In the Login page, Create a new AWS account

Name, E-mail address

Password, Confirm Password

Click on Create account when done.

The next screen is the Contact Information page. Provide your Full Name, Company Name, Country, Address, City, Postal Code, and Phone Number as requested. Check the Amazon Internet Service Pvt. Ltd. Customer Agreement checkbox and select the Create Account and continue options.



# Sign Up for AWS

- Enter a suitable Cardholder's Name and your Credit/Debit Card Number in the Payment Information page as shown:

## Payment Information

Please enter your payment information below. You will be able to try a broad set of AWS products for free via the Free Usage Tier. We will only bill your credit card for usage that is not covered by our Free Usage Tier.

AWS Free Usage Tier	Compute Amazon EC2	Storage Amazon S3	Database Amazon RDS
free for 1 year	750hrs/month*	5GB	750hrs/month*

[\\*View full offer details »](#)

Cardholder's Name

Credit/Debit Card Number

Expiration Date

01 ▾ 2015 ▾

# Sign Up for AWS

- The last part of the signup process is the **Identity Verification process**



# Introducing the AWS Management Console

AWS Management Console

**AWS services**

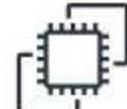
**Find services**  
You can enter names, keyword or acronyms.

Example: Relational Database Service, database, RDS

▶ All services

**Build a solution**  
Get started with simple wizards and automated workflows.

**Launch a virtual machine**  
With EC2  
~2-3 minutes



**Build a web app**  
With Elastic Beanstalk  
~6 minutes



**Build using virtual servers**  
With Lightsail  
~1-2 minutes



**Connect an IoT device**  
With AWS IoT  
~5 minutes



**Start a development project**  
With CodeStar  
~5 minutes



**Register a domain**  
With Route 53  
~3 minutes



**Deploy a serverless microservice**  
With Lambda, API Gateway  
~2 minutes



**Create a backend for your mobile app**  
With Mobile Hub  
~5 minutes



**Access resources on the go**

 Access the Management Console using the AWS Console Mobile App. [Learn more](#)

**Explore AWS**

**Amazon RDS**  
Set up, operate, and scale your relational database in the cloud. [Learn more](#)

**Run Serverless Containers with AWS Fargate**  
AWS Fargate runs and scales your containers without having to manage servers or clusters. [Learn more](#)

**Amazon Redshift**  
Fast, simple, cost-effective data warehouse that can extend queries to your data lake. [Learn more](#)

**Amazon SageMaker**  
Build, train, and deploy machine learning models. [Learn more](#)

# Introducing the AWS Management Console

- To the right-hand side
- The first your ‘**name**’ as an end user:
  - It will help you with configuring your account details, security credentials, and billing management.
- ‘**Region**’ from where you will currently be operating.
  - The US East (North Virginia) region is the cheapest region in AWS as it was one of the first regions to get set up and started.
- The final tab is the ‘**Support**’ tab:
  - Support Center, AWS Forums, and view the latest set of AWS Documentation

# Introducing the AWS Management Console

- To the left-hand side
- **Home Screen**
  - when clicked on will bring you to the AWS dashboard screen
- **Services**
  - which lists the AWS services according to their class.
  - It also has a history option to list recently used AWS services.
- **Resource Groups**
  - These are a collection of AWS resources that can be organized and viewed as per your requirements
- **Edit tab**
  - customize your toolbar by filling it with those AWS services that you use frequently

# Multi Factor Authentication



# What is MFA?

- **AWS** Multi-Factor Authentication (**MFA**) is the practice or requiring two or more forms of authentication to protect **AWS** resources.
- It is an added security feature available through Amazon Identity and Access Management (**IAM**) that strengthens username and password credentials.
- **Prerequisites** : Google Authenticator & BarCode Scanner installed on smart phone

Let's Setup MFA.... Shall We?

# Create IAM User



The screenshot shows the AWS IAM Users page. The left sidebar has a 'Users' link selected, indicated by an orange border. The main content area has a 'User name' dropdown set to 'reya'. The table shows one user entry:

User name	Groups	Access key age
reya	None	4 days

At the top, there are 'Add user' and 'Delete user' buttons. A search bar at the top is labeled 'Find users by username or access key'.

## Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

→Provide UserName

User name\*

[+ Add another user](#)

→Enable AWS Console access

## Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type\*  **Programmatic access**

Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

**AWS Management Console access**

Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password\*  Autogenerated password  
 Custom password

Require password reset

User must create a new password at next sign-in

Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

\* Required

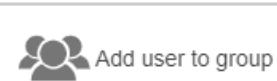
Cancel

Next: Permissions

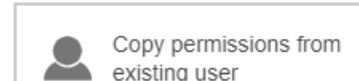
## Add user

1 2 3 4 5

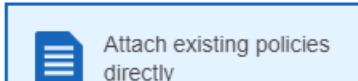
### Set permissions



Add user to group



Copy permissions from existing user



Attach existing policies directly

Create policy



Filter policies

Search

Showing 406 results

	Policy name	Type	Used as	Description
<input checked="" type="checkbox"/>	AdministratorAccess	Job function	Permissions policy (1)	Provides full access to AWS services and...
<input type="checkbox"/>	AlexaForBusinessD...	AWS managed	None	Provide device setup access to AlexaFor...
<input type="checkbox"/>	AlexaForBusinessF...	AWS managed	None	Grants full access to AlexaForBusiness r...
<input type="checkbox"/>	AlexaForBusinessG...	AWS managed	None	Provide gateway execution access to Ale...
<input type="checkbox"/>	AlexaForBusinessR...	AWS managed	None	Provide read only access to AlexaForBus...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	None	Provides full access to create/edit/delete ...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	None	Provides full access to invoke APIs in Am...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	None	Allows API Gateway to push logs to user'...

### Set permissions boundary

Cancel

Previous

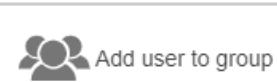
Next: Tags

- Select Attach Existing Policies
  - Select any policy name you required
- Next : Tags

## Add user

1 2 3 4 5

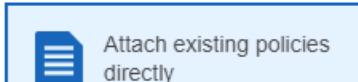
### Set permissions



Add user to group



Copy permissions from existing user



Attach existing policies directly

Create policy



Filter policies



Search

Showing 406 results

	Policy name	Type	Used as	Description
<input checked="" type="checkbox"/>	AdministratorAccess	Job function	Permissions policy (1)	Provides full access to AWS services and...
<input type="checkbox"/>	AlexaForBusinessD...	AWS managed	None	Provide device setup access to AlexaFor...
<input type="checkbox"/>	AlexaForBusinessF...	AWS managed	None	Grants full access to AlexaForBusiness r...
<input type="checkbox"/>	AlexaForBusinessG...	AWS managed	None	Provide gateway execution access to Ale...
<input type="checkbox"/>	AlexaForBusinessR...	AWS managed	None	Provide read only access to AlexaForBus...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	None	Provides full access to create/edit/delete ...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	None	Provides full access to invoke APIs in Am...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	None	Allows API Gateway to push logs to user'...

### Set permissions boundary

Cancel

Previous

Next: Tags

- Select Attach Existing Policies
  - Select any policy name you required
- Next : Tags → Name & Dev

## Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

## User details

<b>User name</b>	RNS
<b>AWS access type</b>	AWS Management Console access - with a password
<b>Console password type</b>	Autogenerated
<b>Require password reset</b>	No
<b>Permissions boundary</b>	Permissions boundary is not set

## Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	<a href="#">AdministratorAccess</a>

## Tags

The new user will receive the following tag

Key	Value
Name	Dev

→Review  
→ Create User

## Add user

- 1
- 2
- 3
- 4
- 5

### Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: [https://\[REDACTED\].signin.aws.amazon.com/console](https://[REDACTED].signin.aws.amazon.com/console)

 Download .csv

		User	Password	Email login instructions
--	--	------	----------	--------------------------

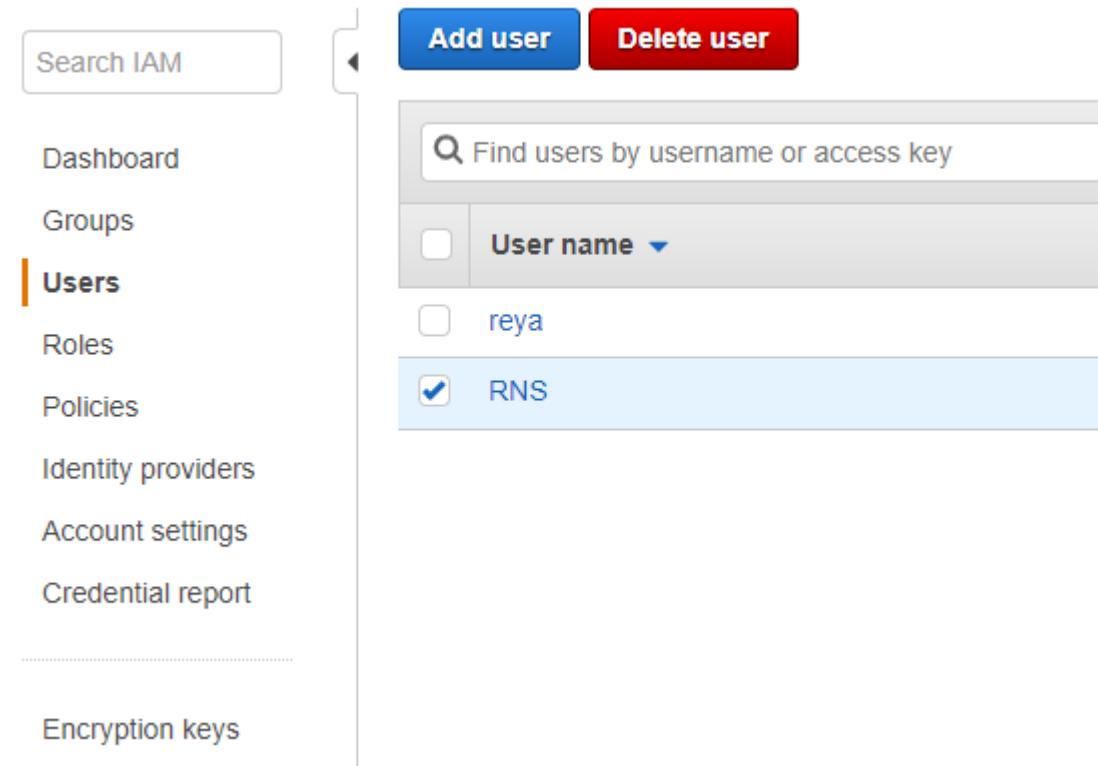
▼  RNS

\*\*\*\*\* Show

[Send email](#) 

- ✓ Created user RNS
- ✓ Attached policy AdministratorAccess to user RNS
- ✓ Created login profile for user RNS

→Select the created User and click on it



The screenshot shows the AWS IAM Users page. The left sidebar has a 'Users' section highlighted with an orange border. The main content area shows a search bar and a table with three rows. The first row has an empty checkbox and 'User name ▾'. The second row has an empty checkbox and 'reya'. The third row has a checked checkbox and 'RNS', which is highlighted with a light blue background. At the top right, there are 'Add user' and 'Delete user' buttons.

Find users by username or access key	
<input type="checkbox"/>	User name ▾
<input type="checkbox"/>	reya
<input checked="" type="checkbox"/>	RNS

## Summary

User ARN arn:aws:iam:.user/RNS 

Path /

Creation time 2018-12-22 15:52 UTC+0530

[Permissions](#)[Groups](#)[Tags \(1\)](#)[Security credentials](#)[Access Advisor](#)

## Sign-in credentials

## Summary

- Console sign-in link: [.signin.aws.amazon.com/console">https://!\[\]\(652ad6cf2d959a2d8715f02679b11303\_img.jpg\).signin.aws.amazon.com/console](https://<img alt=)

## Console password

Enabled (never signed in) | [Manage](#)

## Assigned MFA device

Not assigned | [Manage](#)

## Signing certificates

None 

## Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret k

[Create access key](#)

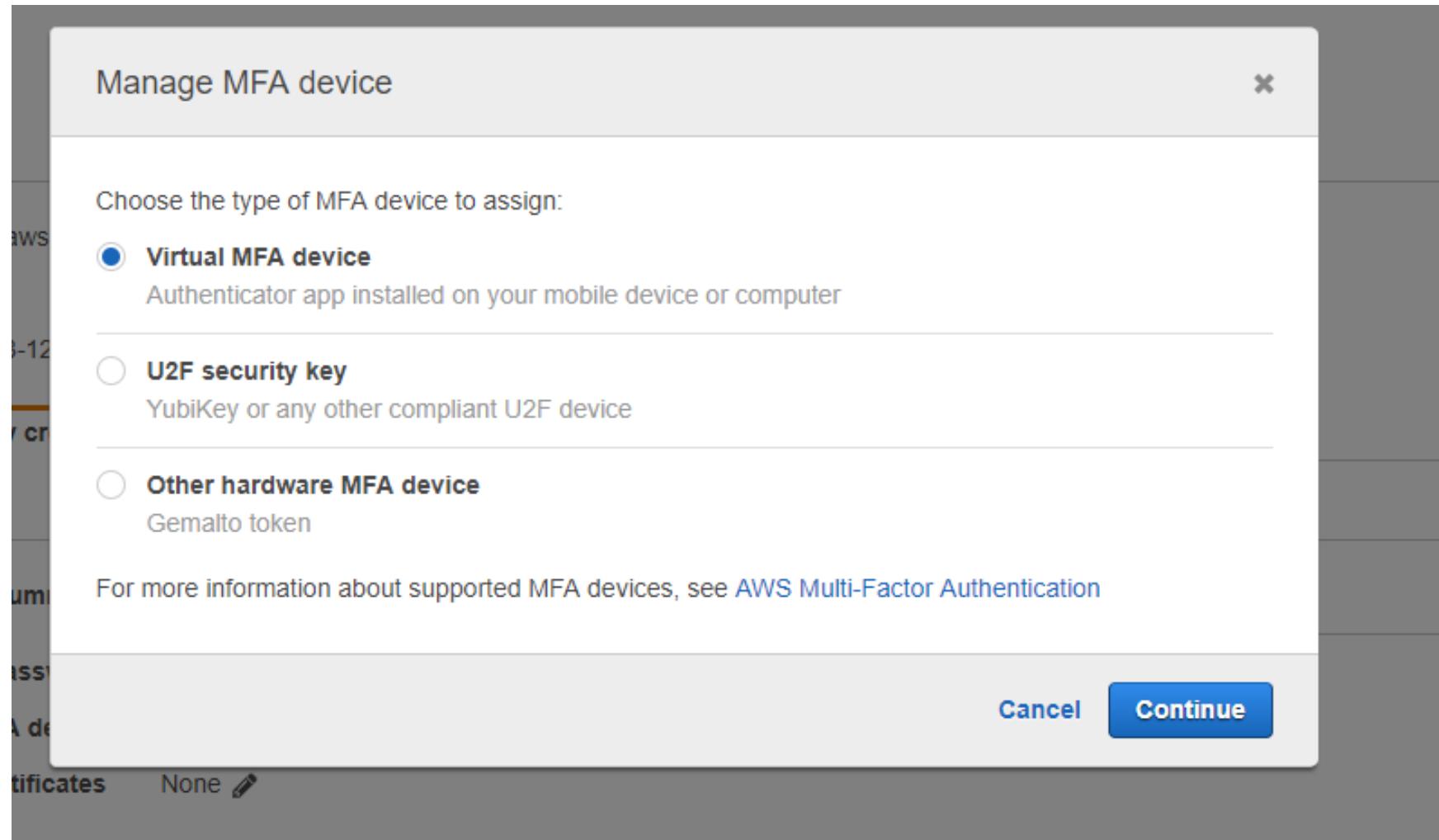
Access key ID

Created

Last used

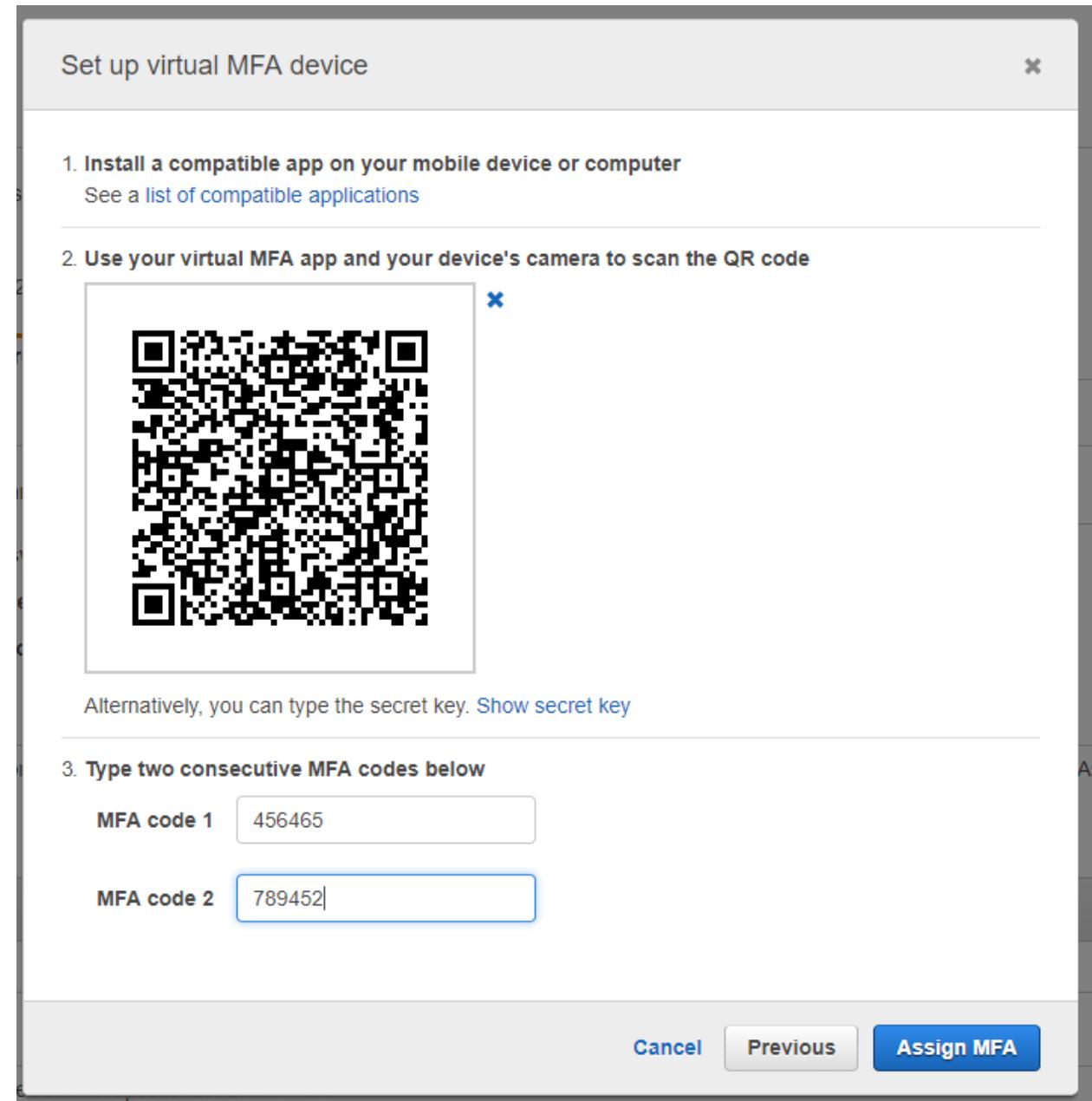
No results

→Virtual MFA Device



- You should have google authenticator and Barcode Scanner installed on your phone
- Scan using barcode scanner application from your phone or
- From google authenticator add account and scan the code
- Provide 2 MFA Codes below
- Click Assign MFA

Once you logout and login the console after entering your AWS console password it asks for MFA. You have to open the google authenticator from your phone and provide the code it has generated to the console



# NOW YOU ARE SECURE



**DEMONSTRAION?**

# AWS CLI

Rise 'n' Shine Technologies

By

Reyaz

# AWS CLI

- Using CLIs, you can automate the deployment and management of your AWS services using simple code and script, much like how you would use bash and shell scripting.
- Prerequisite:
  - The AWS CLI can be either installed on a Windows or a Linux machine.
- Windows, AWS provides an easy-to-use installer
  - The 64-bit AWS CLI installer for Windows can be downloaded from
  - <https://s3.amazonaws.com/aws-cli/AWSCLI64.msi>.

# Setting up AWS CLI

- Python versions supported are Python 2 version 2.6.5 and above or Python 3 version 3.3 and above.
- Install the Python
  - **Yum install –y python**
- Verify the Python Installation
  - **python --version**

# Installation of AWSCLI

- Download the Python setup tools:
- `wget https://pypi.python.org/packages/source/s/setuptools/setuptools-7.0.tar.gz`
- `tar xvf setuptools-7.0.tar.gz`
- `cd setuptools-7.0`
- `python setup.py install`
- `wget https://bootstrap.pypa.io/get-pip.py`
- `python get-pip.py`
- `pip install awscli`
- `aws --version`

# Managing access and security using the AWS CLI

- Configuring the AWS CLI
  - `# aws configure`
- you will be prompted to enter the user's **Access Key ID and the Secret Access Key, along with the default region name and the default output format to use.**
- The default region name is a mandatory field and can be any of the regions from which your users will be operating, for example, us-east-1, us-west-2, and so on
- The output format accepts any of these three values as the preferred method to display the output of the commands: table, text, or json.
- Note: Any of these values can be changed at any time by rerunning the `aws configure` command.

# Accessing CLI Commands

- AWS will store these credentials and configuration details in two separate files named `~/.aws/credentials` and `~/.aws/config`, respectively.
- Let's try out the CLI by executing some commands. To start off, let's try listing the users present in our account.
  - `# aws iam list-users --profile admin`

# Managing Users using AWSCLI

- Configuring the AWS CLI:
  - >> aws configure
  - >> aws configure --profile admin
  - >> aws iam list-users --profile admin
  - >> aws iam create-user --user-name YoYo --profile admin
  - >> aws iam create-login-profile --user-name YoYo --password P@\$\$w0rD --profile admin (--password-reset-required)
  - >> aws iam create-access-key --user-name YoYo --profile admin
  - >> aws iam create-group --group-name SuperUsersGroup --profile admin
  - >> aws iam add-user-to-group --user-name YoYo --group-name SuperUsersGroup --profile admin
  - 
  -

# Managing Users using AWSCLI

- `# vi /tmp/MyPolicy.json`
- Add the following contents to your policy file as shown:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "*",  
      "Resource": "*"  
    }  
  ]  
}
```
- Next, run the following command to attach this policy document to your newly created group:
- `# aws iam put-group-policy --group-name SuperUsersGroup --policy-name Admin-Access-All --policy-document file:///vagrant/myPolicy.json --profile admin`

# Now I Know AWS CLI !!!!!



# Amazon Elastic Compute Cloud



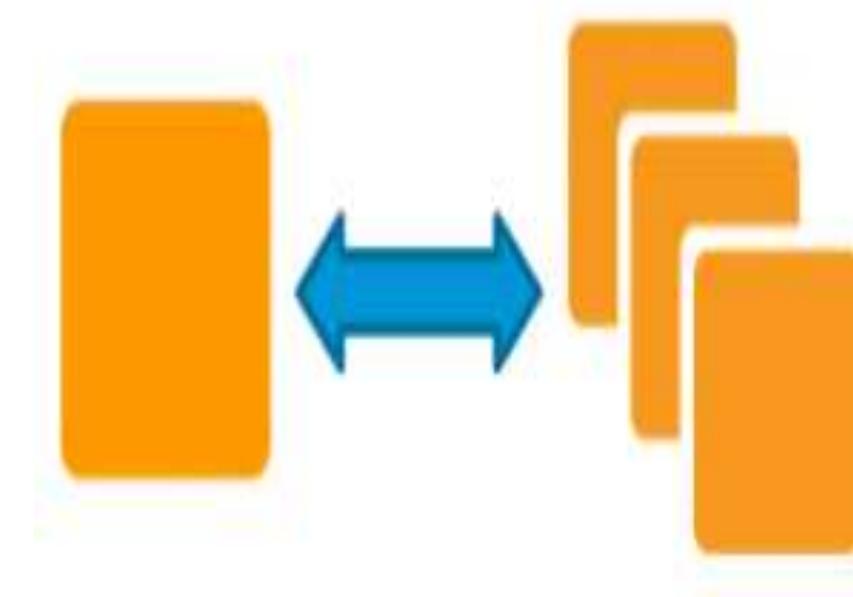
# What is AWS EC2

Amazon Elastic Compute Cloud, EC2 is a web service from Amazon that provides **resizable** compute services in the cloud.



# How are they re-sizable?

They are re-sizable because you can quickly scale up or scale down the number of server instances you are using if your computing requirements change.



# Instances

An instance is a virtual server for running applications on Amazon's EC2. It can also be understood like a tiny part of a larger computer, a tiny part which has its own Hard drive, network connection, OS etc.

But it is actually all virtual. You can have multiple “tiny” computers on a single physical machine, and all these tiny machines are called Instances.

# Let's understand the types of EC2 Computing Instances:

Computing is a very broad term, the nature of your task decides what kind of computing you need

Therefore, AWS EC2 offers many types of instances which are few as follows:

## ***General Instances***

For applications that require a balance of performance and cost.

E.g email responding systems, where you need a prompt response as well as it should be cost effective, since it doesn't require much processing.

## ***Compute Instances***

For applications that require a lot of processing from the CPU.

E.g analysis of data from a stream of data, like Twitter stream

## ***Memory Instances***

For applications that are heavy in nature, therefore, require a lot of RAM.

E.g when your system needs a lot of applications running in the background i.e multitasking.

## ***Storage Instances***

For applications that are huge in size or have a data set that occupies a lot of space.

E.g When your application is of huge size.

## ***GPU Instances***

For applications that require some heavy graphics rendering.

E.g 3D modelling etc.

# Instance Types

**Now, every instance type has a set of instances which are optimized for different workloads:**

General Instances

- t2
- m4
- m3

Compute Instances

- c4
- c3

Memory Instances

- r3
- x1

Storage Instances

- i2
- d2

GPU Instances

- g2

# Burstable Performance Instances

*T2 instances* are burstable instances, meaning the CPU performs at a baseline, say 20% of its capability.

When your application needs more than 20% of the performance of the CPU, the CPU enters into a burst mode giving higher performance for a limited amount of time, therefore work happens faster.

# EBS-optimized Instances

*C4, M4, and D2 instances*, are EBS optimized by default, EBS means Elastic Block Storage, which is a storage option provided by AWS in which the IOPS\* rate is quite high.

Therefore, when an EBS volume is attached to an optimized instance, single digit millisecond latencies can be achieved.

\*IOPS (Input/Output Operations Per Second, pronounced eye-ops) is a performance measurement used to characterize computer storage devices.

# Cluster Networking Instances

*X1, M4, C4, C3, I2, G2 and D2 instances* support cluster networking. Instances launched into a common placement group are put in a logical group that provides high-bandwidth, low latency between all the instances in the group.

A placement group is basically a logical cluster where some select EC2 instances which are a part of that group can utilize up to 10Gbps for single flow and 20Gbps for multi flow traffic in each direction.

Instances which are not a part of that group are limited to 5 Gbps speed in multi flow traffic. Cluster Networking is ideal for high performance analytics system.

# Dedicated Instances

They are the instances that run on single-tenant hardware dedicated to a single customer.

They are perfect for workloads where a corporate policy or industry regulation requires that your instance should be isolated from any other customer's instance, therefore they go for their own separate machines, and their instances are isolated at the hardware level.

# Before we jump

## Few things to know

### **Amazon Machine Image (AMI)**

AMIs are templates of OS and they provide the information needed to launch an instance.

For preconfigured AMIs you have to select it from AWS marketplace.

For setting up your own, go to quick-start and select one.

### **Elastic Block Storage (EBS)**

Is a persistent block level storage volumes which are used with EC2. Here each block acts as a hard drive.

**Provisioned IOPS:** This category is for workloads which are mission critical, it provides high IOPS rates.

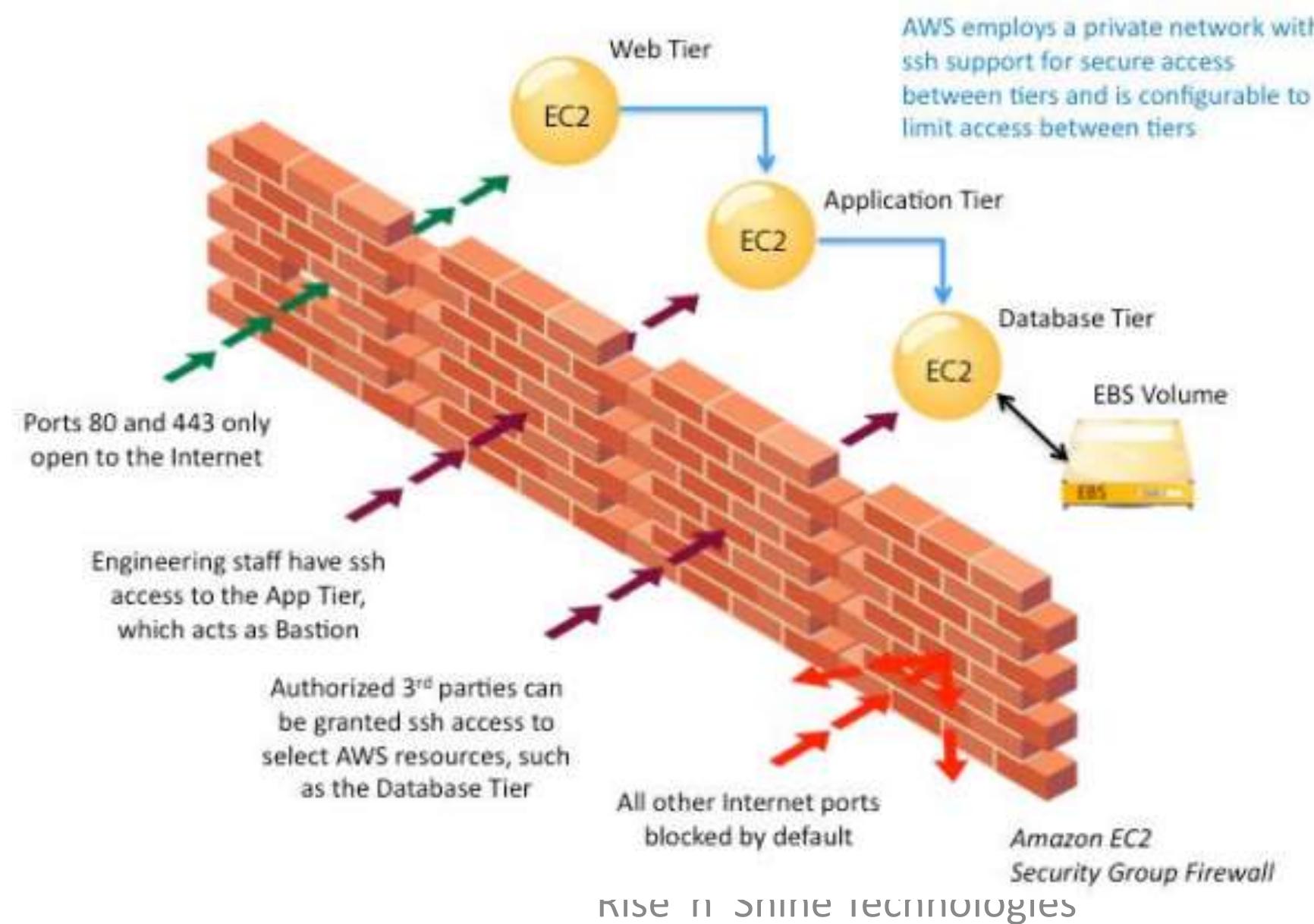
**General Purpose:** It is for workloads which need a performance and cost balance.

**Magnetic:** It is for data which is accessed less frequently, and also retrieval time is more.

# Security Groups

## Security Groups:

A security group acts as a firewall to control inbound and outbound traffic. Each security group has rules according to which the traffic is governed.



# Key Pair

## Key Pair:

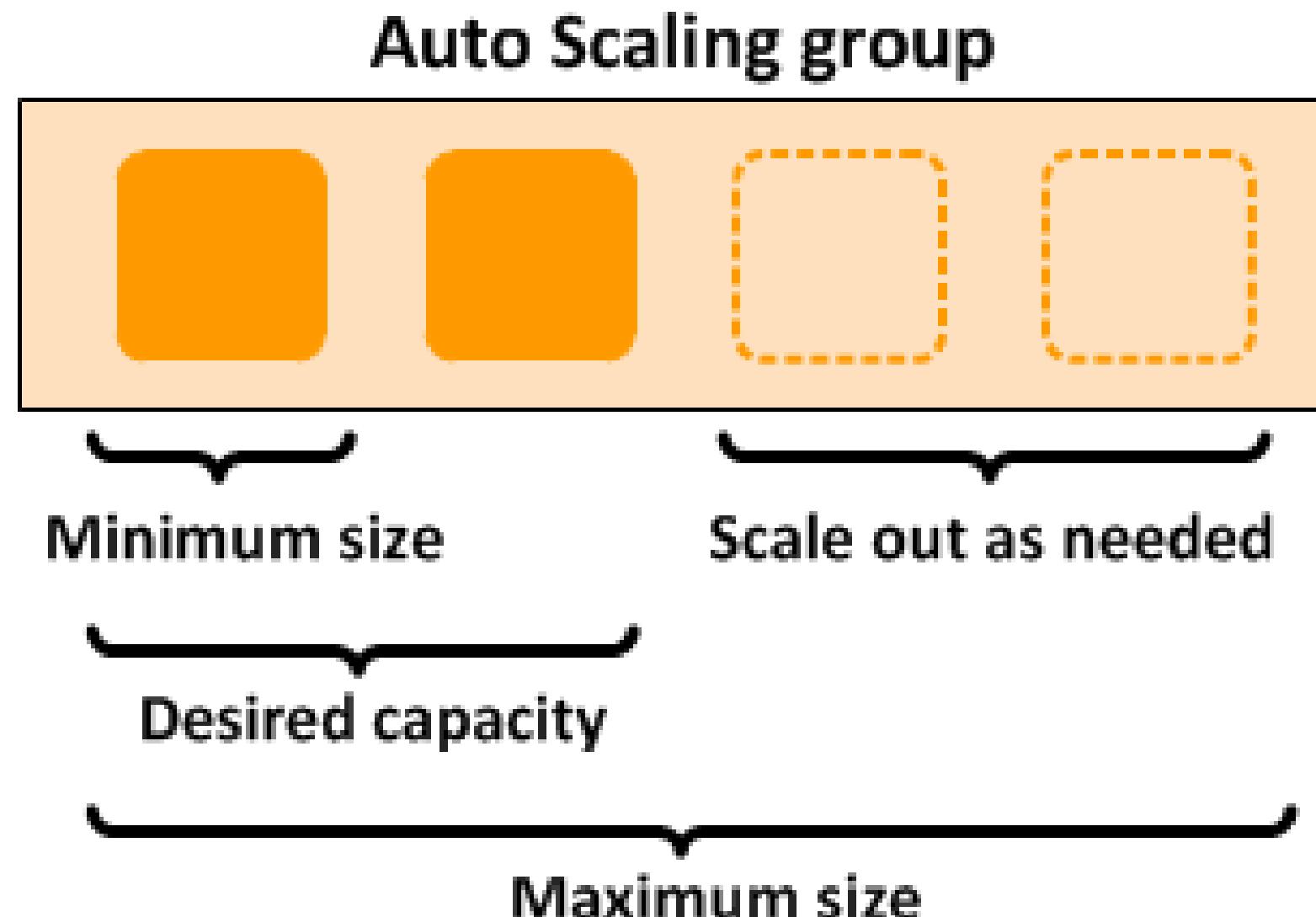
Amazon EC2 uses public–key cryptography to encrypt and decrypt login information. Public–key cryptography uses a public key to encrypt a piece of data, such as a password, then the recipient uses the private key to decrypt the data. The public and private keys are known as a *key pair*.



# AutoScaling

## AutoScaling

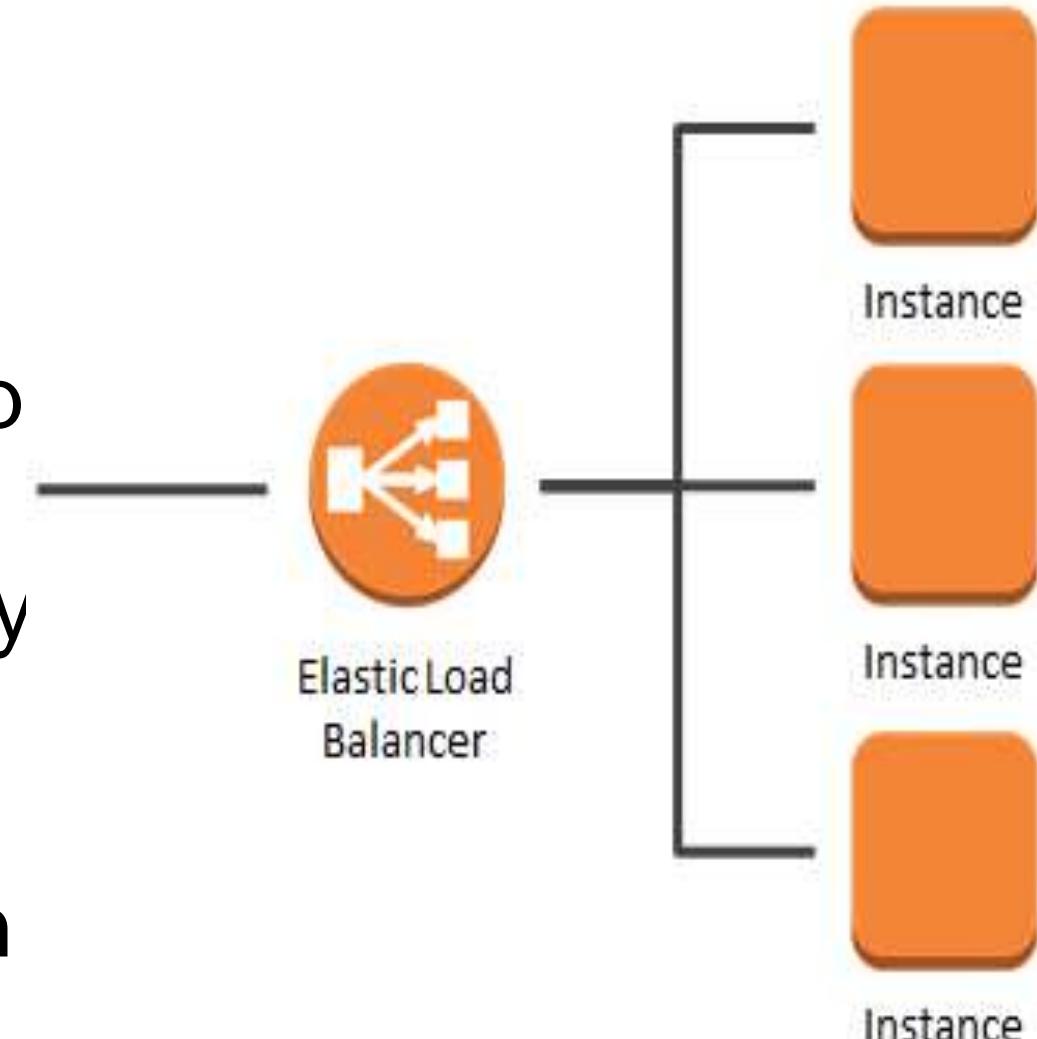
*Auto Scaling* is a service designed by AWS EC2, which automatically launch or terminate EC2's instances based on user defined policies, schedules and health checks.



# Elastic Load Balancer

## ***Elastic Load Balancer (ELB)***

- Automatically distributes incoming application traffic across multiple EC2 instances, in multiple Availability Zones.
- Availability zones are basically places where amazon has set up their servers. Since they have customers from the whole globe, they have set up multiple Availability zones to reduce the latency



*Elastic IP Addresses* are static IP addresses which are associated with your AWS account, they can be used to mask the failure of an instance by automatically remapping your address to another working instance in your account

# Getting started with AWS

- A Free Tier
- No charge for a period of 12 months from the date of the actual signup
- For a complete insight into the free tier usage, check
  - <http://aws.amazon.com/free/>

# AWS Free Tier Account

AWS Product	What's free?
Amazon EC2	750 hours per month of Linux and Windows micro instance usage
Amazon S3	5 GB of standard storage 20,000 get requests 2,000 put requests
Amazon RDS	750 Hours of RDS Single-AZ micro instance 20 GB of DB Storage: any combination of general purpose (SSD) or magnetic 20 GB for backups 10,000,000 I/Os
Amazon ELB	750 hours per month 15 GB of data processing

# AWS Pricing

There are basically 3 pricing options in EC2:

- On Demand Instances
- Reserved Instances
- Spot Instances

***On Demand Instances:*** are used when you want to pay for the hour, with no long term commitments and upfront payments. They are useful for applications that may have unpredictable workloads or for test applications that are being deployed for the first time.

***Reserved Instances:*** provide you with significant discounts as compared to On Demand Instances. With Reserved Instances you reserve instances for a specific period of time.

***Spot Instances:*** is a pricing option which enables you to bid on unused EC2 instances. The hourly price for a Spot Instance is set by AWS EC2, and it fluctuates according to the availability of the instances in a specific Availability zone

Let us **Fire** the EC2 instance now

!!!!Demo!!!!



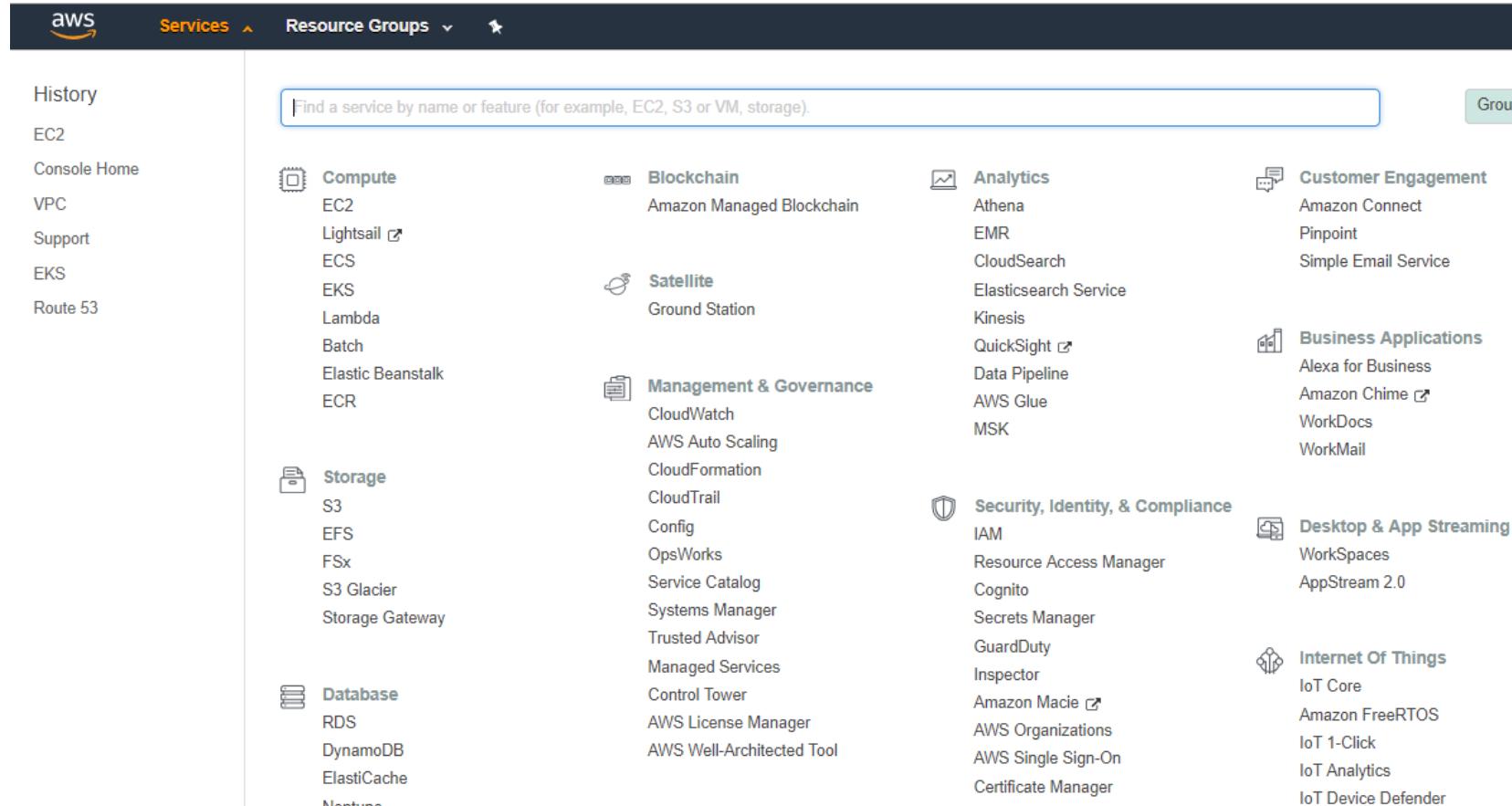


# Login and access to AWS services

**Step 1)** In this step,

Login to your AWS account and go to the AWS Services tab at the top left corner.

Here, you will see all of the AWS Services categorized as per their area viz. Compute, Storage, Database, etc. For creating an EC2 instance, we have to choose Compute EC2 as in the next step.



Here is the EC2 dashboard. Here you will get all the information about the AWS EC2 resources running..

The screenshot shows the AWS EC2 Dashboard. The top navigation bar includes the AWS logo, Services dropdown, and Edit dropdown. The left sidebar has a 'EC2 Dashboard' button (which is highlighted with a red box and a red arrow pointing to it), followed by a list of navigation items: Events, Tags, Reports, Limits, INSTANCES (with sub-options: Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts), IMAGES (with sub-options: AMIs, Bundle Tasks), and ELASTIC BLOCK STORE (with sub-options: Volumes, Snapshots). The main content area is titled 'Resources' and displays the following statistics: 3 Running Instances, 0 Dedicated Hosts, 12 Volumes, 22 Key Pairs, 0 Placement Groups, 4 Elastic IPs, 17 Snapshots, 0 Load Balancers, and 28 Security Groups. A callout box highlights the resource statistics. Below this, a message encourages using Amazon Simple Queue Service. The 'Create Instance' section contains a 'Launch Instance' button and a note stating instances will launch in the US East (N. Virginia) region. The bottom of the page includes a footer with the AWS logo, a 'Contact Us' link, and the text 'Rise 'n' Shine Technologies'.

EC2 Dashboard

Events

Tags

Reports

Limits

INSTANCES

Instances

Spot Requests

Reserved Instances

Scheduled Instances

Commands

Dedicated Hosts

IMAGES

AMIs

Bundle Tasks

ELASTIC BLOCK STORE

Volumes

Snapshots

Resources

You are using the following Amazon EC2 resources in the US East (N. Virginia) region:

3 Running Instances	4 Elastic IPs
0 Dedicated Hosts	17 Snapshots
12 Volumes	0 Load Balancers
22 Key Pairs	28 Security Groups
0 Placement Groups	

Need fast, reliable, scalable, fully-managed message queuing? Try Amazon Simple Queue Service. X

Create Instance

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

**Launch Instance**

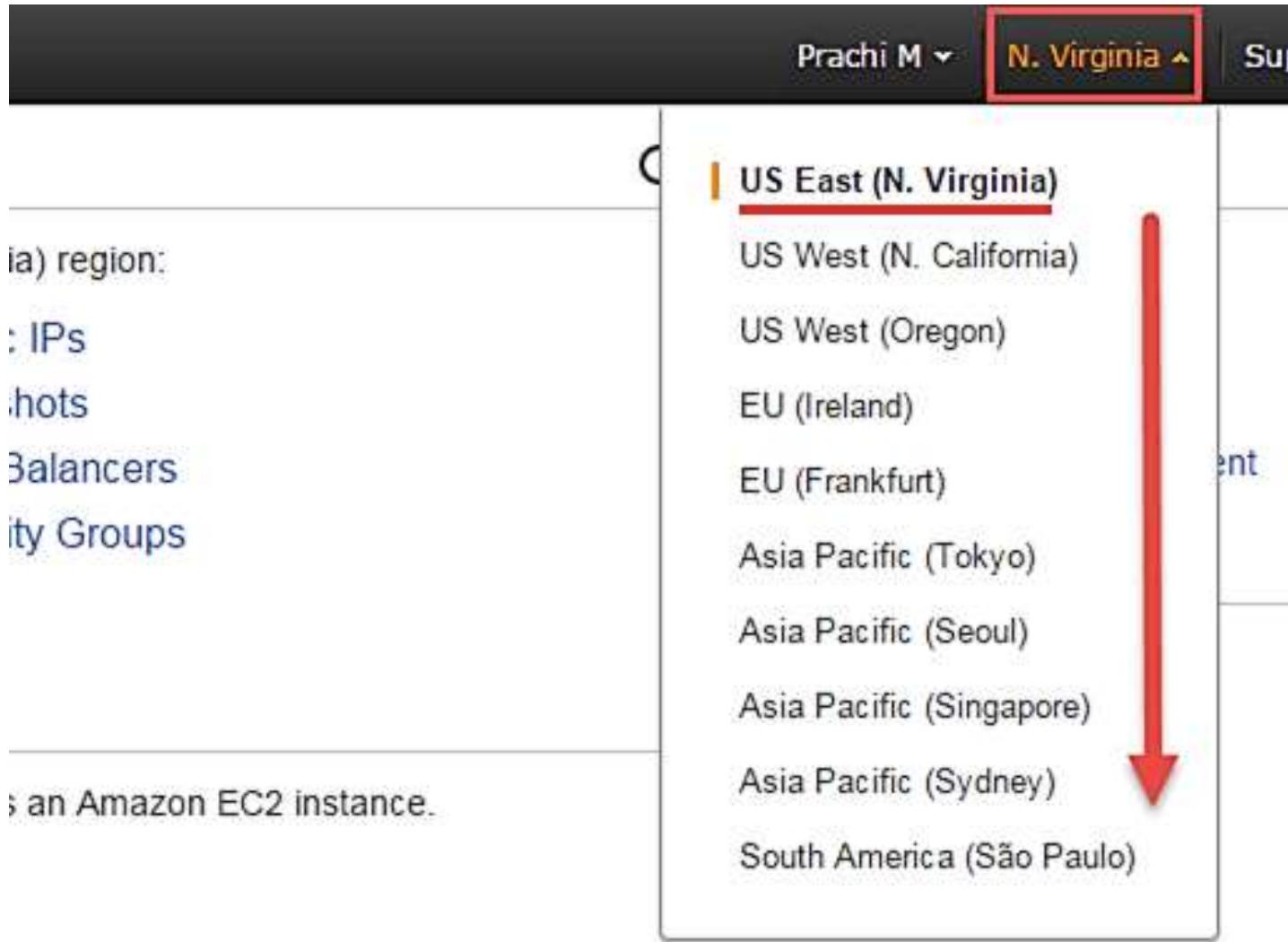
Note: Your instances will launch in the US East (N. Virginia) region

Contact Us

Rise 'n' Shine Technologies

**Step 2)** On the top right corner of the EC2 dashboard, choose the AWS Region in which you want to provision the EC2 server.

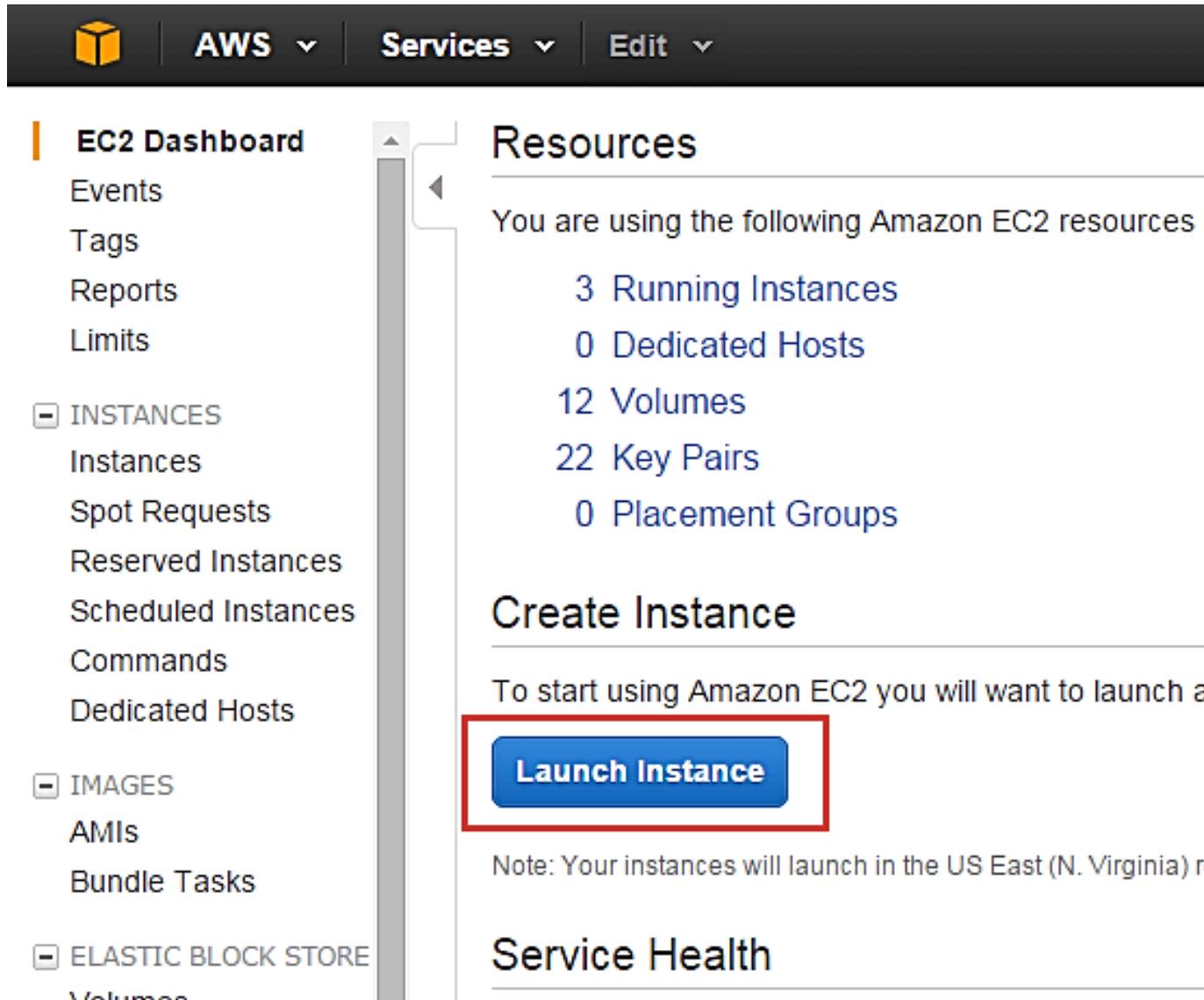
Here we are selecting N. Virginia. AWS provides 10 Regions all over the globe.



### Step 3) In this step

Once your desired Region is selected, come back to the EC2 Dashboard.

Click on 'Launch Instance' button in the section of Create Instance (as shown below).

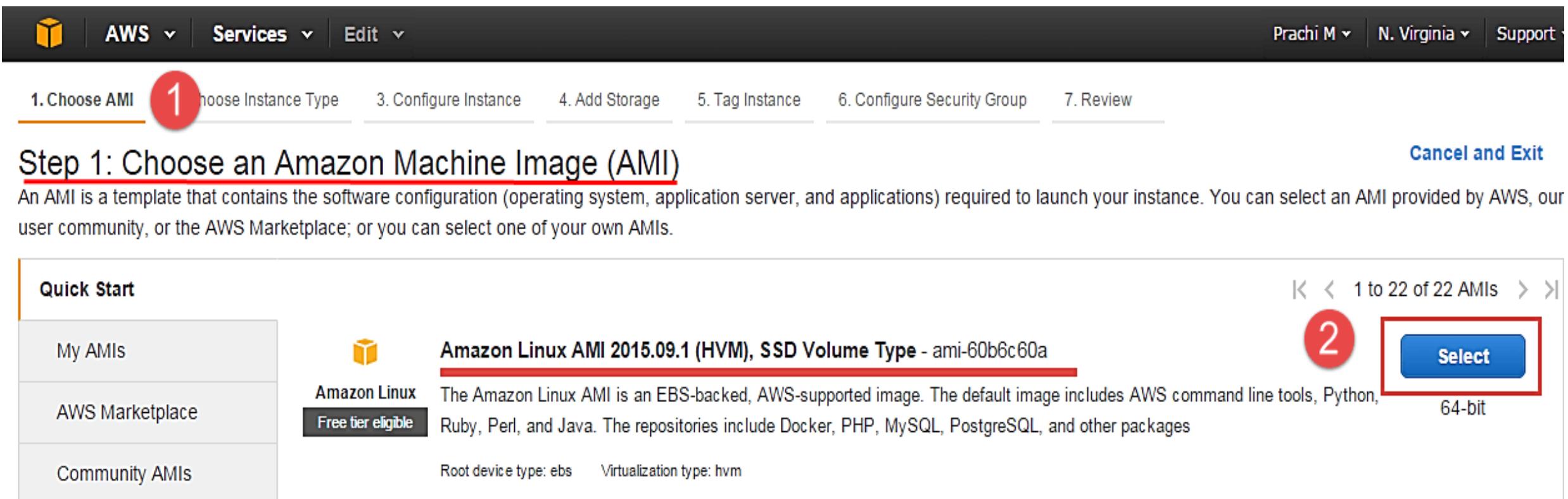


The screenshot shows the AWS EC2 Dashboard. The top navigation bar includes the AWS logo, a Services dropdown, and an Edit dropdown. The left sidebar has a 'EC2 Dashboard' section with links for Events, Tags, Reports, and Limits. Below this are sections for INSTANCES (Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts), IMAGES (AMIs, Bundle Tasks), and ELASTIC BLOCK STORE (Volumes). The main content area is titled 'Resources' and displays usage statistics: 3 Running Instances, 0 Dedicated Hosts, 12 Volumes, 22 Key Pairs, and 0 Placement Groups. A 'Create Instance' section follows, with the text 'To start using Amazon EC2 you will want to launch a' and a prominent blue 'Launch Instance' button, which is highlighted with a red border. A note at the bottom states 'Note: Your instances will launch in the US East (N. Virginia) region'.

# Choose AMI

In this step we will do,

You will be asked to choose an AMI of your choice. Once you launch an EC2 instance from your preferred AMI, the instance will automatically be booted with the desired OS.



AWS Services Edit Prachi M N. Virginia Support

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** K < 1 to 22 of 22 AMIs > >

My AMIs	Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-60b6c60a
Amazon Linux	 The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages
AWS Marketplace	<span style="float: right;">64-bit</span>
Community AMIs	Root device type: ebs Virtualization type: hvm

**My AMIs**

**Amazon Linux** Free tier eligible

**Community AMIs**

**Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-60b6c60a**

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages

Root device type: ebs Virtualization type: hvm

**Select**

## Choose EC2 Instance Types

In this step, you have to choose the type of instance you require based on your business needs. We will choose t2.micro instance type, which is a 1vCPU and 1GB memory server offered by AWS. Click on "Configure Instance Details" for further configurations

The screenshot shows the AWS EC2 instance creation wizard at step 2. The top navigation bar includes AWS, Services, Edit, and user Prachi M. The progress bar shows steps 1-7: Choose AMI, Choose Instance Type (underlined), Configure Instance, Add Storage, Tag Instance, Configure Security Group, and Review. The main content area is titled 'Step 2: Choose an Instance Type'. It explains that Amazon EC2 provides a wide selection of instance types optimized for different use cases. A table lists instance types by family, type, vCPUs, memory, storage, and network performance. The 't2.micro' row is highlighted with a red box and a green 'Free tier eligible' badge. A red circle with the number '1' is on the 't2.micro' cell. A red box with the number '2' surrounds the 'Next: Configure Instance Details' button at the bottom right.

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)

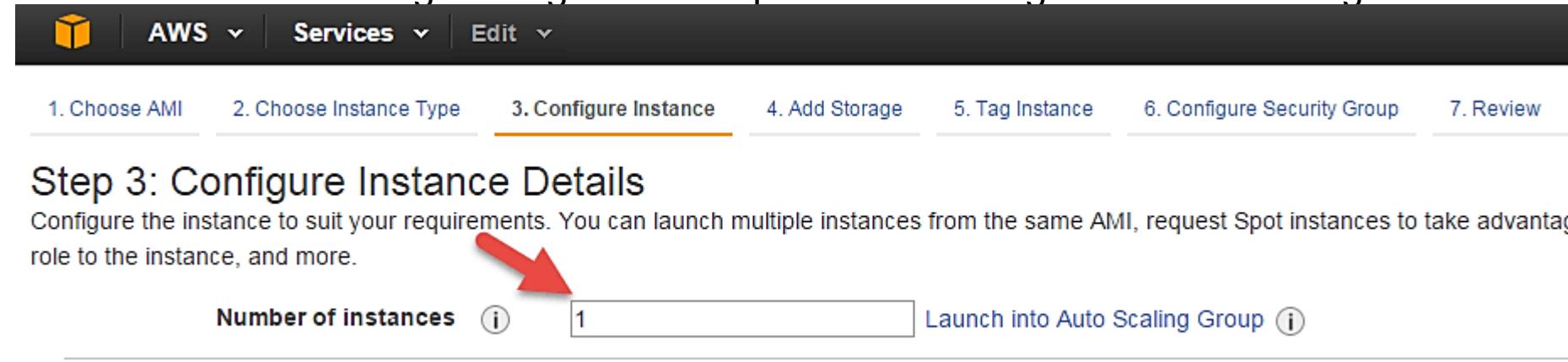
	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	m4.large	2	8	EBS only	Yes	Moderate

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

## Configure Instance

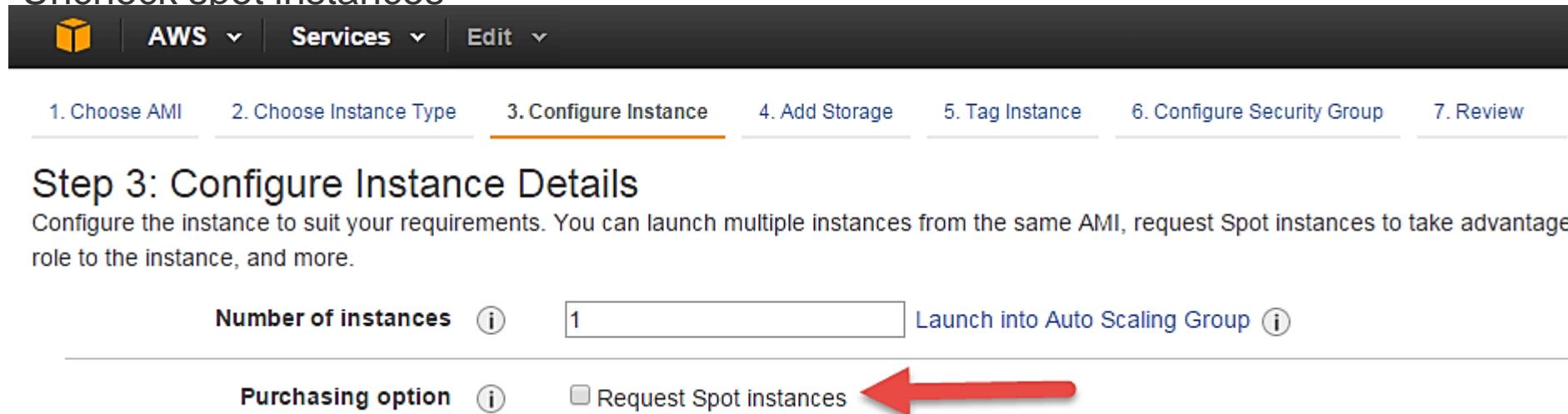
On the top right corner of the EC2 dashboard, choose the AWS Region in which you want to provision the EC2 server.

Here we are selecting N. Virginia. AWS provides 10 Regions all over the globe.



The screenshot shows the 'Configure Instance' step of the AWS EC2 wizard. The top navigation bar includes 'AWS' and 'Services' dropdowns, and 'Edit' and 'Review' buttons. Below the navigation is a horizontal progress bar with steps 1-7: '1. Choose AMI', '2. Choose Instance Type', '3. Configure Instance' (which is highlighted in orange), '4. Add Storage', '5. Tag Instance', '6. Configure Security Group', and '7. Review'. The main content area is titled 'Step 3: Configure Instance Details' with the sub-instruction: 'Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of low prices, attach data volumes to the instance, and more.' A red arrow points to the 'Number of instances' input field, which contains the value '1'. To the right of the input field is a 'Launch into Auto Scaling Group' link with an info icon.

### Uncheck spot instances



The screenshot shows the 'Configure Instance' step of the AWS EC2 wizard, identical to the previous one but with a specific step highlighted. The top navigation bar, progress bar, and main content area are the same. The 'Purchasing option' section is highlighted with a red arrow pointing to the 'Request Spot instances' checkbox, which is currently unchecked.

Next, we have to configure some basic networking details for our EC2 server. You have to decide here, in which VPC (Virtual Private Cloud) you want to launch your instance and under which subnets inside your VPC. It is better to determine and plan this prior to launching the instance.

Your AWS architecture set-up should include IP ranges for your subnets etc. pre-planned for better management. (We will see how to create a new VPC in Networking section of the tutorial. Subnetting should also be pre-planned. E.g.: If it's a web server you should place it in the public subnet and if it's a DB server, you should place it in a private subnet all inside your VPC.

Below,

Network section will give a list of VPCs available in our platform.

Select an already existing VPC

You can also create a new VPC

Here I have selected an already existing VPC where I want to launch my instance.

1. Network section will give a list of VPCs available in our platform.

2. Select an already existing VPC

3. You can also create a new VPC

Here I have selected an already existing VPC where I want to launch my instance.

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 current market price, attach data volumes to the instance, and more.

**Number of instances** (i)  Launch into Auto Scaling Group (i)

**Purchasing option** (i)  Request Spot instances

**Network** (i) **Subnet** (i)

**Auto-assign Public IP** (i) **IAM role** (i) None

**Create new VPC**

**Create new subnet**

**Create new IAM role**

Subnet	IP Range	VPC
vpc-d5194fb0 (192.168.0.0/16)	192.168.0.0/16	Prachi_Test - VPC
vpc-621a5e07 (172.20.0.0/16)	172.20.0.0/16	POC_vpc
vpc-d5194fb0 (192.168.0.0/16)	192.168.0.0/16	Prachi_Test - VPC
vpc-8452bce0 (172.20.0.0/16)	172.20.0.0/16	POC_vpc
vpc-823e39e7 (172.22.0.0/16)	172.22.0.0/16	TVPC
vpc-4c51bf28 (10.0.0.0/16)	10.0.0.0/16	POC_vpc3

- A VPC consists of subnets, which are IP ranges that are separated for restricting access.
- 1.Under Subnets, you can choose the subnet where you want to place your instance.
  - 2.I have chosen an already existing public subnet.
  - 3.You can also create a new subnet in this step.

AWS Services Edit

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 low role to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	<p>vpc-d5194fb0 (192.168.0.0/16)   Prachi_Test - VPC</p> <p><b>Subnet</b></p> <p>1</p> <p>subnet-b3e3d0ea(192.168.2.0/24)   Prachi_Test-Pl</p> <p>subnet-0eeef779(192.168.3.0/24)   Prachi_Test_Public subnet 3   us-east-1a</p> <p>subnet-a94427de(192.168.1.0/24)   Prachi_Test- Public Subnet   us-east-1a</p> <p>subnet-b3e3d0ea(192.168.2.0/24)   Prachi_Test-Public subnet2   us-east-1b</p>	<p><b>C</b> Create new VPC</p> <p><b>C</b> Create new subnet</p>
Auto-assign Public IP		
IAM role	None	<p><b>C</b> Create new IAM role</p>

Shutdown behavior: Stop

- You can choose if you want AWS to assign it an IP automatically, or you want to do it manually later. You can enable/ disable 'Auto assign Public IP' feature here likewise.
- Here we are going to assign this instance a static IP called as EIP (Elastic IP) later. So we keep this feature disabled as of now.

AWS Services Edit

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 current market price, assign an IAM role to the instance, and more.

**Number of instances** (i)  Launch into Auto Scaling Group (i)

**Purchasing option** (i)  Request Spot instances

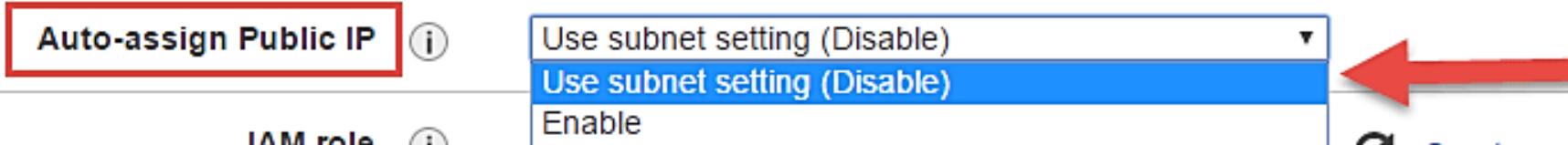
**Network** (i) vpc-d5194fb0 (192.168.0.0/16) | Prachi\_Test - VPC  Create new VPC

**Subnet** (i) subnet-b3e3d0ea(192.168.2.0/24) | Prachi\_Test-PI  Create new subnet  
251 IP Addresses available

**Auto-assign Public IP** (i)

**IAM role** (i)  Create new IAM role

**Shutdown behavior** (i)





AWS

Services

Edit

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 low role to the instance, and more.

**Number of instances**

1

Launch into Auto Scaling Group

**Purchasing option** Request Spot instances**Network**

vpc-d5194fb0 (192.168.0.0/16) | Prachi\_Test - VPC



Create new VPC

**Subnet**

subnet-b3e3d0ea(192.168.2.0/24) | Prachi\_Test-PI



Create new subnet

**Auto-assign Public IP**

Use subnet setting (Disable)

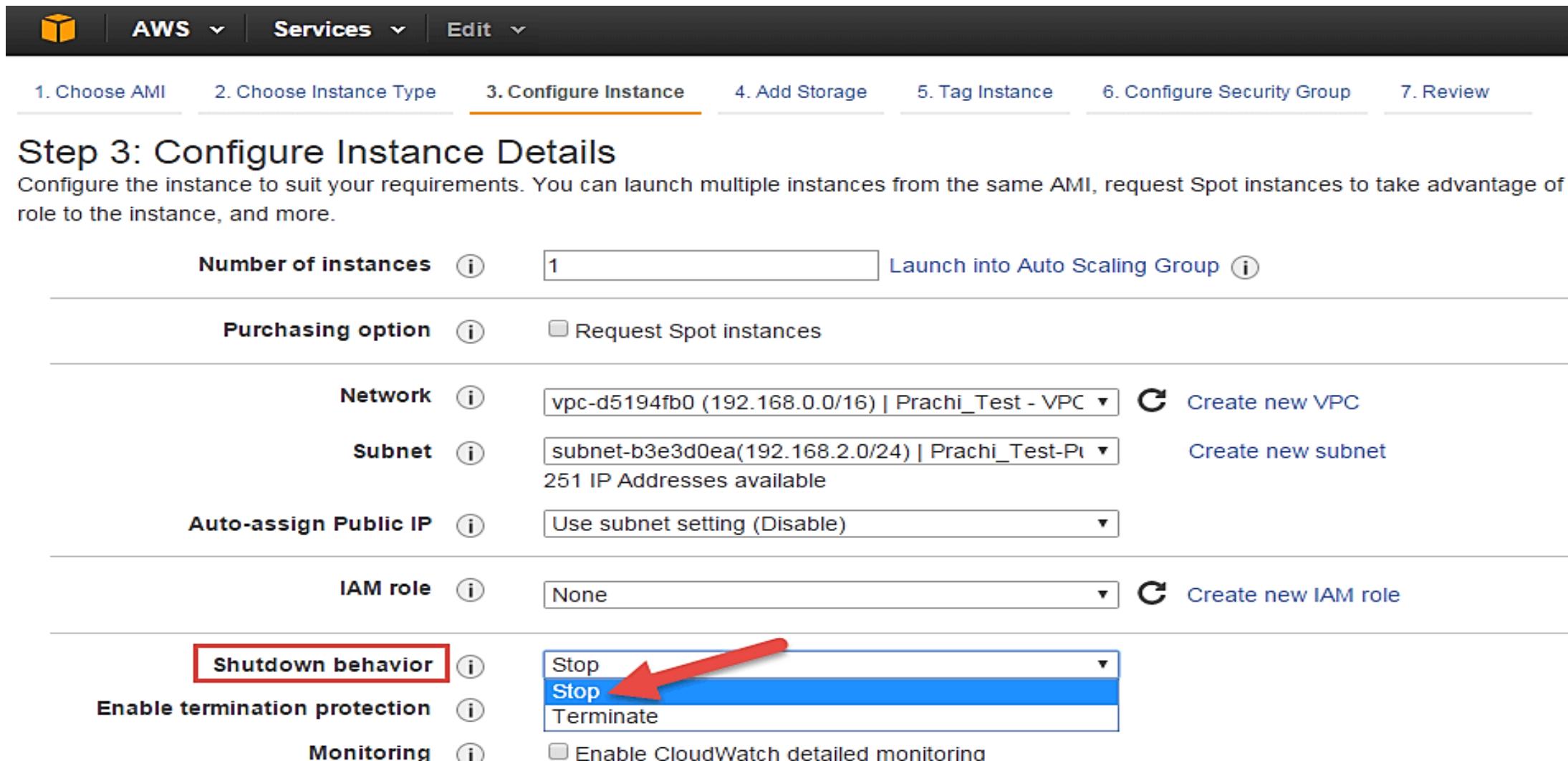
**IAM role**

None



Create new IAM role

- Shutdown Behavior – when you accidentally shut down your instance, you surely don't want it to be deleted but stopped.
- Here we are defining my shutdown behavior as Stop.



The screenshot shows the AWS CloudFormation 'Step 3: Configure Instance Details' page. The 'Number of instances' is set to 1. Under 'Purchasing option', there is a checkbox for 'Request Spot instances' which is unchecked. The 'Network' section shows 'vpc-d5194fb0 (192.168.0.0/16) | Prachi\_Test - VPC' with a 'Create new VPC' button. The 'Subnet' section shows 'subnet-b3e3d0ea(192.168.2.0/24) | Prachi\_Test-PI' with a 'Create new subnet' button and a note that 251 IP Addresses are available. Under 'Auto-assign Public IP', the setting is 'Use subnet setting (Disable)'. The 'IAM role' is set to 'None' with a 'Create new IAM role' button. The 'Shutdown behavior' dropdown is highlighted with a red box and a red arrow pointing to the 'Stop' option, which is selected. The 'Enable termination protection' and 'Monitoring' sections are also visible.

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 role to the instance, and more.

Number of instances (i)  Launch into Auto Scaling Group (i)

Purchasing option (i)  Request Spot instances

Network (i)  C Create new VPC

Subnet (i)  C Create new subnet  
251 IP Addresses available

Auto-assign Public IP (i)

IAM role (i)  C Create new IAM role

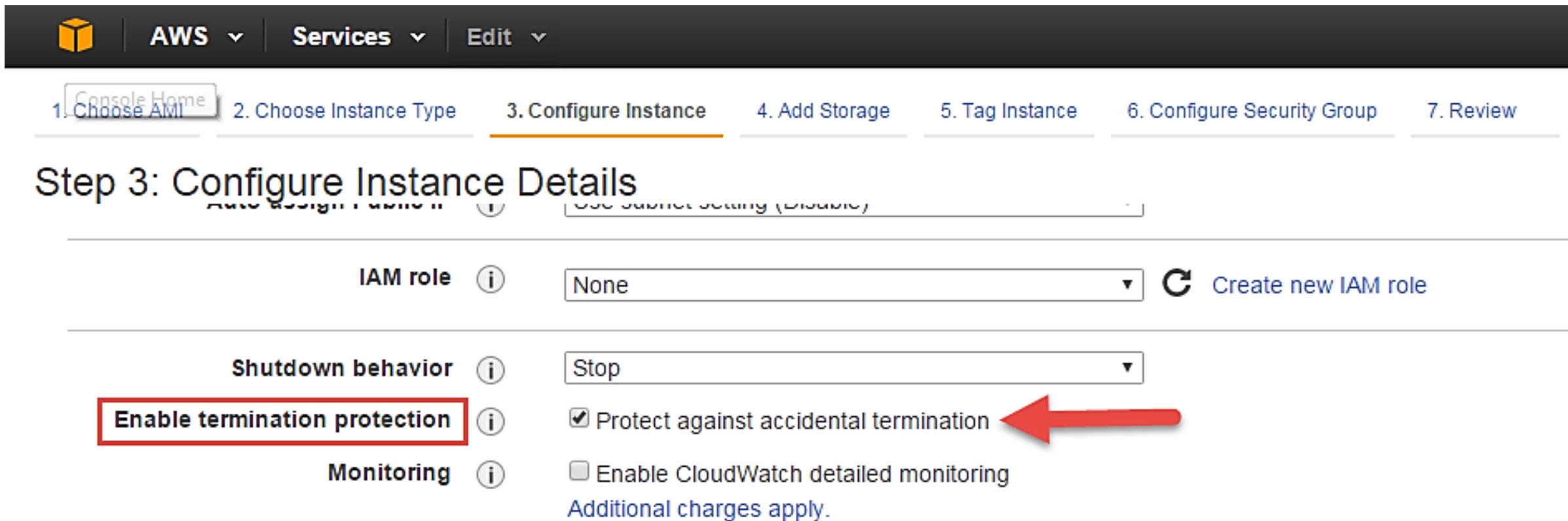
Shutdown behavior (i)  C Create new shutdown behavior

Enable termination protection (i)

Monitoring (i)  Enable CloudWatch detailed monitoring

- In case, you have accidentally terminated your instance, AWS has a layer of security mechanism. It will not delete your instance if you have enabled accidental termination protection.

- Here we are checking the option for further protecting our instance from accidental termination.



The screenshot shows the AWS CloudFormation 'Create New Stack' wizard at Step 3: Configure Instance Details. The top navigation bar includes 'AWS', 'Services', 'Edit', and a progress bar with steps 1 through 7. The main section is titled 'Step 3: Configure Instance Details'. It contains several configuration options: 'IAM role' set to 'None' with a 'Create new IAM role' link; 'Shutdown behavior' set to 'Stop'; 'Enable termination protection' (checkbox checked, highlighted with a red box and a red arrow); and 'Monitoring' (checkbox unchecked). A note below monitoring states 'Additional charges apply.'

- **Under Monitoring**- you can enable Detailed Monitoring if your instance is a business critical instance. Here we have kept the option unchecked. AWS will always provide Basic monitoring on your instance free of cost. We will visit the topic of monitoring in AWS Cloud Watch part of the tutorial.
- **Under Tenancy**- select the option if shared tenancy. If your application is a highly secure application, then you should go for dedicated capacity. AWS provides both options.

The screenshot shows the AWS Step 3: Configure Instance Details wizard. The top navigation bar includes the AWS logo, Services dropdown, and Edit dropdown. Below the navigation, a progress bar shows steps 1 through 7: 1. Choose AMI, 2. Choose Instance Type, 3. Configure Instance (highlighted in orange), 4. Add Storage, 5. Tag Instance, 6. Configure Security Group, and 7. Review.

**Step 3: Configure Instance Details**

**Monitoring**

Enable CloudWatch detailed monitoring  
Additional charges apply.

**Tenancy**

Shared - Run a shared hardware instance  
 Shared - Run a shared hardware instance (highlighted with a red arrow)  
 Dedicated - Run a Dedicated instance  
 Dedicated host - Launch this instance on a Dedicated host

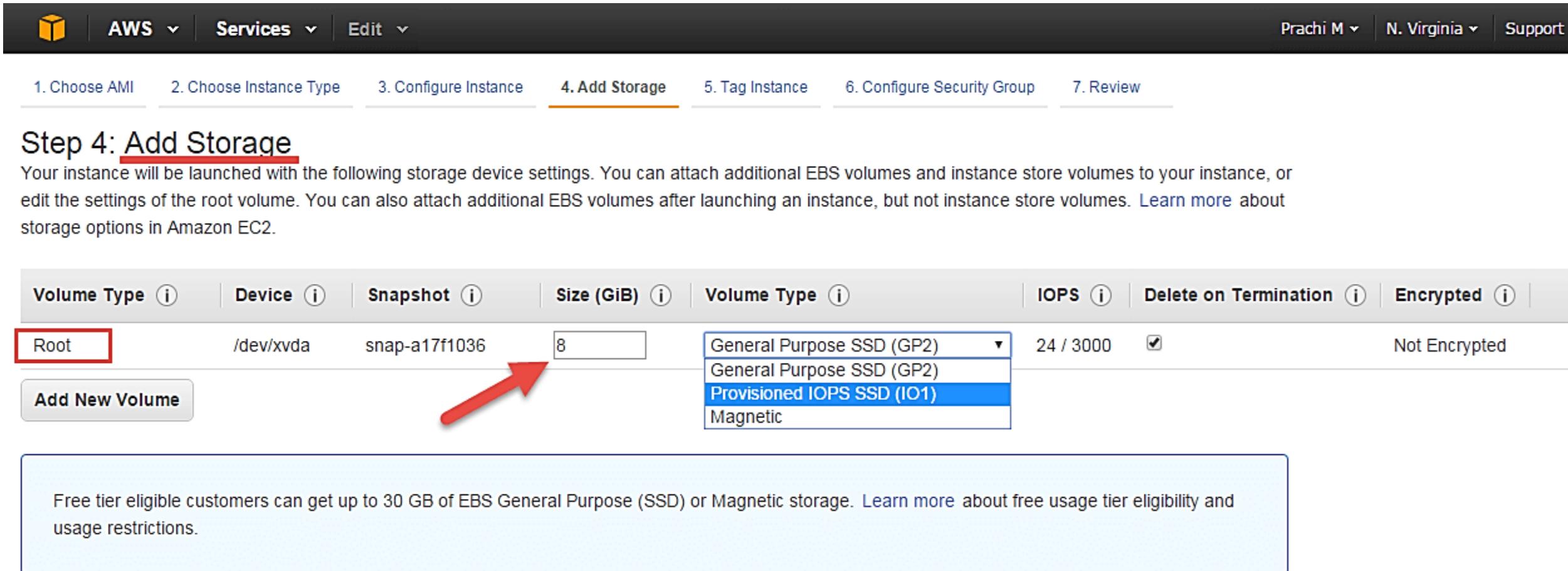
**Network interfaces**

## Add Storage

In the Add Storage step, you'll see that the instance has been automatically provisioned a General Purpose SSD root volume of 8GB

You can change your volume size, add new volumes, change the volume type, etc.

AWS provides 3 types of EBS volumes- Magnetic, General Purpose SSD, Provisioned IOPs. You can choose a volume type based on your application's IOPs needs.



The screenshot shows the AWS EC2 instance creation wizard at Step 4: Add Storage. The top navigation bar includes AWS Services, Edit, and user information (Prachi M, N. Virginia, Support). The progress bar shows steps 1-7. The '4. Add Storage' step is active, indicated by an orange underline.

**Step 4: Add Storage**

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more about storage options in Amazon EC2.](#)

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/xvda	snap-a17f1036	8	General Purpose SSD (GP2)	24 / 3000	<input checked="" type="checkbox"/>	Not Encrypted
<b>Add New Volume</b>							

A red arrow points to the 'Size (GiB)' input field, which is currently set to 8. A dropdown menu for 'Volume Type' is open, showing four options: General Purpose SSD (GP2) (selected), General Purpose SSD (GP2), Provisioned IOPS SSD (IO1), and Magnetic.

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more about free usage tier eligibility and usage restrictions.](#)

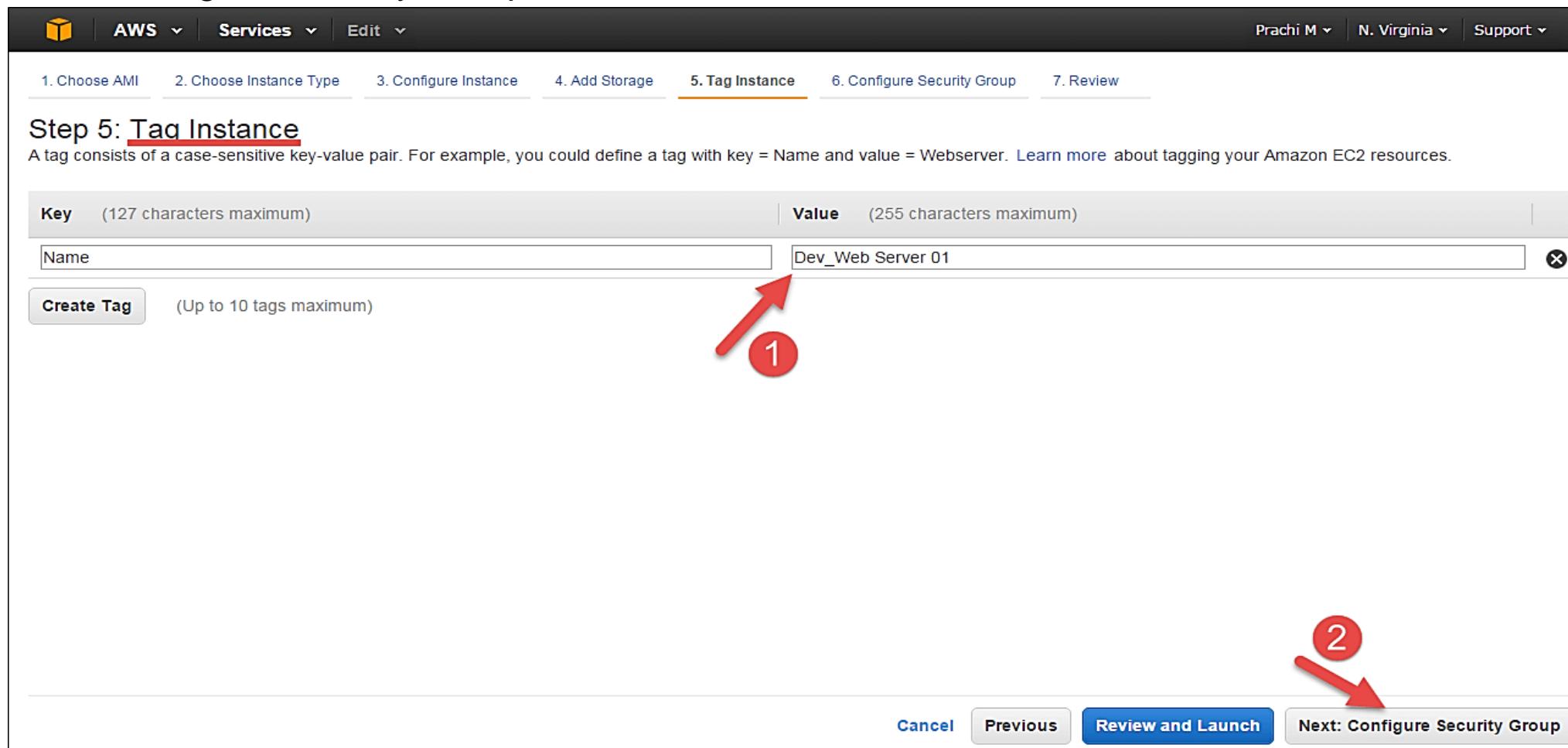
# TAG INSTANCE

You can tag your instance with a key-value pair. This gives visibility to the AWS account administrator when there are lot number of instances.

The instances should be tagged based on their department, environment like Dev/SIT/Prod. Etc. this gives a clear view of the costing on the instances under one common tag.

Here we have tagged the instance as a **Dev\_Web server 01**

Go to configure Security Groups later



The screenshot shows the AWS EC2 instance creation process at Step 5: Tag Instance. The 'Name' tag is highlighted with a red arrow and circled with a red number '1'. The 'Value' field contains 'Dev\_Web Server 01'. A red arrow and circled number '2' points to the 'Next: Configure Security Group' button at the bottom.

Step 5: Tag Instance

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	(127 characters maximum)	Value	(255 characters maximum)
Name	Dev_Web Server 01	<input type="button" value="X"/>	

[Create Tag](#) (Up to 10 tags maximum)

Cancel Previous [Review and Launch](#) Next: Configure Security Group

# Security Groups Configuration

In this next step of configuring Security Groups, you can restrict traffic on your instance ports. This is an added firewall mechanism provided by AWS apart from your instance's OS firewall. You can define open ports and IPs.

Since our server is a webserver=, we will do following things

- Creating a new Security Group
- Naming our SG for easier reference
- Defining protocols which we want enabled on my instance
- Assigning IPs which are allowed to access our instance on the said protocols
- Once, the firewall rules are set- Review and launch

# Creating Security Group

AWS Services Edit Prachi M N. Virginia Support

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 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: Web Server SG

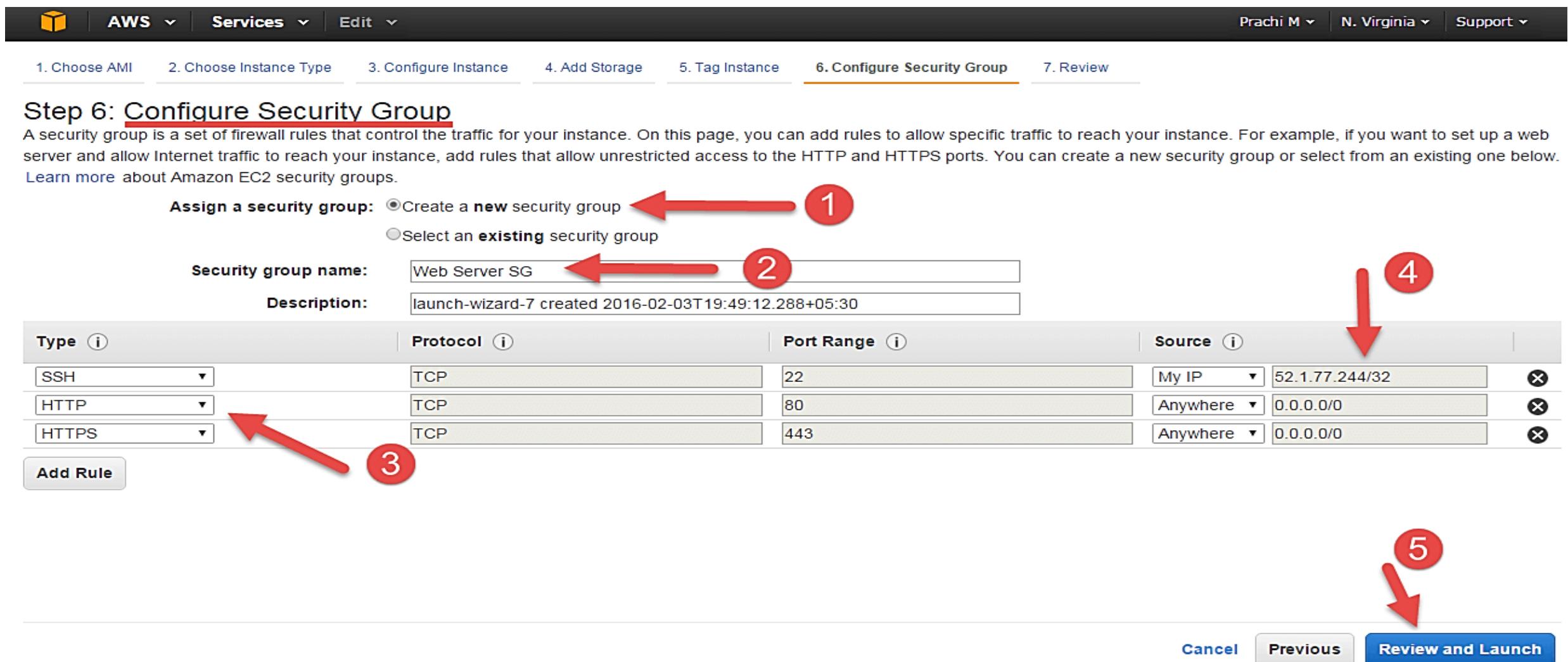
Description: launch-wizard-7 created 2016-02-03T19:49:12.288+05:30

Type	Protocol	Port Range	Source
SSH	TCP	22	My IP 52.1.77.244/32
HTTP	TCP	80	Anywhere 0.0.0.0/0
HTTPS	TCP	443	Anywhere 0.0.0.0/0

Add Rule

1 2 3 4 5

Cancel Previous Review and Launch



# Review

AWS Services Edit Prachi M N. Virginia Support

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

### Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

**AMI Details** [Edit AMI](#)

**Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-60b6c60a**

**Free tier eligible** The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages

Root Device Type: ebs Virtualization type: hvm

**Instance Type** [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

**Security Groups** [Edit security groups](#)

**Security group name** Web Server SG  
**Description** launch-wizard-7 created 2016-02-03T19:49:12.288+05:30

Type [i](#) | Protocol [i](#) | Port Range [i](#) | Source [i](#)

[Cancel](#) [Previous](#) **Launch**



In the next step you will be asked to create a key pair to login to your instance. A key pair is a set of public-private keys.

AWS stores the private key in the instance, and you are asked to download the public key. Make sure you download the key and keep it safe and secured; if it is lost you cannot download it again.

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch.

**AMI Details**

**Amazon Linux AMI 2015.0**  
The Amazon Linux AMI is an E...  
Free tier eligible  
Docker, PHP, MySQL, PostgreSQL  
Root Device Type: ebs Virtualization: HVM

**Instance Type**

Instance Type	ECUs
t2.micro	Variable

**Security Groups**

Security group name	Description
Web Server	launch-... Type: i

**Select an existing key pair or create a new key pair**

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

1. Create a new key pair  
2. Key pair name (Dev Key)  
3. Download Key Pair

You have to download the **private key file** (\*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Cancel **Launch Instances**

Click on the 'Instances' option on the left pane where you can see the status of the instance as 'Pending' for a brief while.



The screenshot shows the AWS EC2 Instances page. The left sidebar has a 'Instances' option selected, indicated by a yellow bar. The main content area shows a table with one row of data. The columns are: Name, Instance ID, Instance Type, Availability Zone, Instance State, and Status Checks. The 'Instance State' column for the first row is highlighted with a red box and contains the word 'pending' with a yellow exclamation mark icon. The 'Status Checks' column for the same row contains the word 'Initializing' with a clock icon.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
Dev_Web Server 01	i-4c2c3cff	t2.micro	us-east-1b	<span>pending</span>	<span>Initializing</span>

Once your instance is up and running, you can see its status as 'Running' now.  
Note that the instance has received a Private IP from the pool of AWS.

The screenshot shows the AWS EC2 Instances page. The left sidebar shows navigation links for EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts, Images, AMIs, and Bundle Tasks. The Instances section is currently selected. The main content area shows a table of instances. A search bar at the top of the table area contains the text "i-4c2c3cff". The table columns are Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, and Alarm Status. A single row is visible for "Dev\_Web Server 01", with the Instance ID "i-4c2c3cff", Instance Type "t2.micro", Availability Zone "us-east-1b", and Instance State "running" (highlighted with a red box). Below the table, the instance details are expanded. The "Description" tab is selected, showing the Instance ID "i-4c2c3cff", Instance state "running", Instance type "t2.micro", Private DNS "ip-192-168-2-167.ec2.internal", and Private IP "192.168.2.167". A red arrow points to the Private IP. The "Status Checks" tab is also visible. To the right, detailed instance information is listed in a grid format, including Public DNS (empty), Public IP (empty), Elastic IP (empty), Availability zone "us-east-1b", Security groups "Web Server SG. view rules", Scheduled events "No scheduled events", AMI ID "amzn-ami-hvm-2015.09.1.x86\_64-gp2 (ami-60b6c60a)", Platform (empty), IAM role (empty), Key pair name "Dev Key", Owner "018511290429", and Launch time "February 3, 2016 at 7:52:22 PM UTC+5:30 (less than one hour)".

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
Dev_Web Server 01	i-4c2c3cff	t2.micro	us-east-1b	running	Initializing	None

Instance: i-4c2c3cff (Dev\_Web Server 01) Private IP: 192.168.2.167

Description	Status Checks	Monitoring	Tags
Instance ID: i-4c2c3cff			
Instance state: running			
Instance type: t2.micro			
Private DNS: ip-192-168-2-167.ec2.internal			
Private IP: 192.168.2.167			
Secondary private IPs:			
VPC ID: vpc-d5194fb0			
Subnet ID: subnet-b3e3d0ea			
Network interfaces: eth0			
Source/dest. check: True			
ClassicLink: -			
EBS-optimized: False			

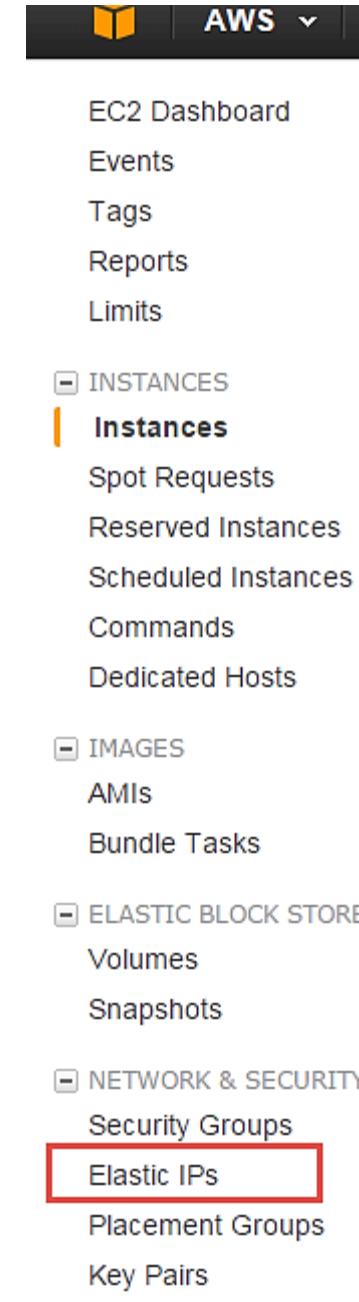
Public DNS	-
Public IP	
Elastic IP	-
Availability zone	us-east-1b
Security groups	Web Server SG. view rules
Scheduled events	No scheduled events
AMI ID	amzn-ami-hvm-2015.09.1.x86_64-gp2 (ami-60b6c60a)
Platform	-
IAM role	-
Key pair name	Dev Key
Owner	018511290429
Launch time	February 3, 2016 at 7:52:22 PM UTC+5:30 (less than one hour)

# Creating Elastic IP address

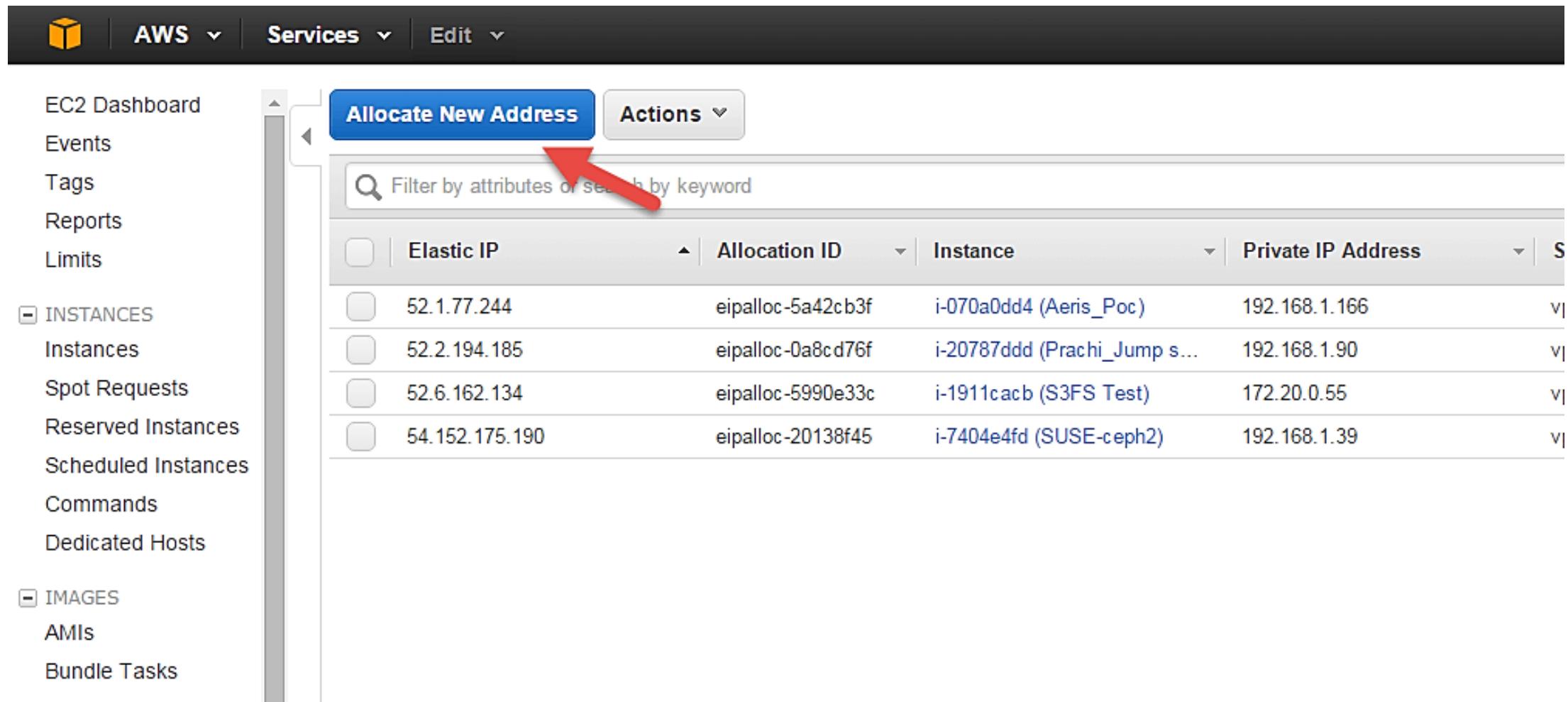
An EIP is a static public IP provided by AWS. It stands for Elastic IP. Normally when you create an instance, it will receive a public IP from the AWS's pool automatically.

If you stop/reboot your instance, this public IP will change- it's dynamic. In order for your application to have a static IP from where you can connect via public networks, you can use an EIP.

- On the left pane of EC2 Dashboard, you can go to 'Elastic IPs' as shown below.



## Allocate a new Elastic IP Address.



The screenshot shows the AWS EC2 Dashboard. The top navigation bar includes the AWS logo, a Services dropdown, and an Edit dropdown. The left sidebar contains links for EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES (with sub-links for Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, and Dedicated Hosts), and IMAGES (with sub-links for AMIs and Bundle Tasks). The main content area has a title 'Allocate New Address' in a blue button, followed by an 'Actions' dropdown and a search bar with the placeholder 'Filter by attributes or search by keyword'. Below these are two tabs: 'Elastic IP' (selected) and 'Allocation ID'. A table lists five Elastic IP entries with columns: Elastic IP, Allocation ID, Instance, and Private IP Address. The table includes a header row and five data rows. A red arrow points to the search bar.

Elastic IP	Allocation ID	Instance	Private IP Address
52.1.77.244	eipalloc-5a42cb3f	i-070a0dd4 (Aeris_Poc)	192.168.1.166
52.2.194.185	eipalloc-0a8cd76f	i-20787ddd (Prachi_Jump s...)	192.168.1.90
52.6.162.134	eipalloc-5990e33c	i-1911cacb (S3FS Test)	172.20.0.55
54.152.175.190	eipalloc-20138f45	i-7404e4fd (SUSE-ceph2)	192.168.1.39

Allocate this IP to be used in a VPC scope.

The screenshot shows the AWS EC2 Dashboard with the 'Allocate New Address' button highlighted. A modal dialog box is open, asking 'Are you sure you want to allocate a new IP address?'. The dropdown menu for 'EIP used in:' shows 'VPC' selected. The dialog box has 'Cancel' and 'Yes, Allocate' buttons.

EC2 Dashboard

Events

Tags

Reports

Limits

**INSTANCES**

- Instances
- Spot Requests
- Reserved Instances
- Scheduled Instances
- Commands
- Dedicated Hosts

**IMAGES**

- AMIs
- Bundle Tasks

**ELASTIC BLOCK STORE**

- Volumes

Allocate New Address

Actions

Filter by attributes or search by keyword

Elastic IP	Allocation ID	Instance	Private IP Address
52.1.77.244	eipalloc-5a42cb3f	i-070a0dd4 (Aens_Poc)	192.168.1.166
52.2.194.185	eipalloc-0a8cd76f	i-20787ddd (Prachi_Jump s...)	192.168.1.90
52.6.162.134	eipalloc-5990e33c	i-1911cach (S3FS Test)	172.20.0.55
54.152.175.190			1.39

Allocate New Address

Are you sure you want to allocate a new IP address?

EIP used in: **VPC**

EC2

VPC

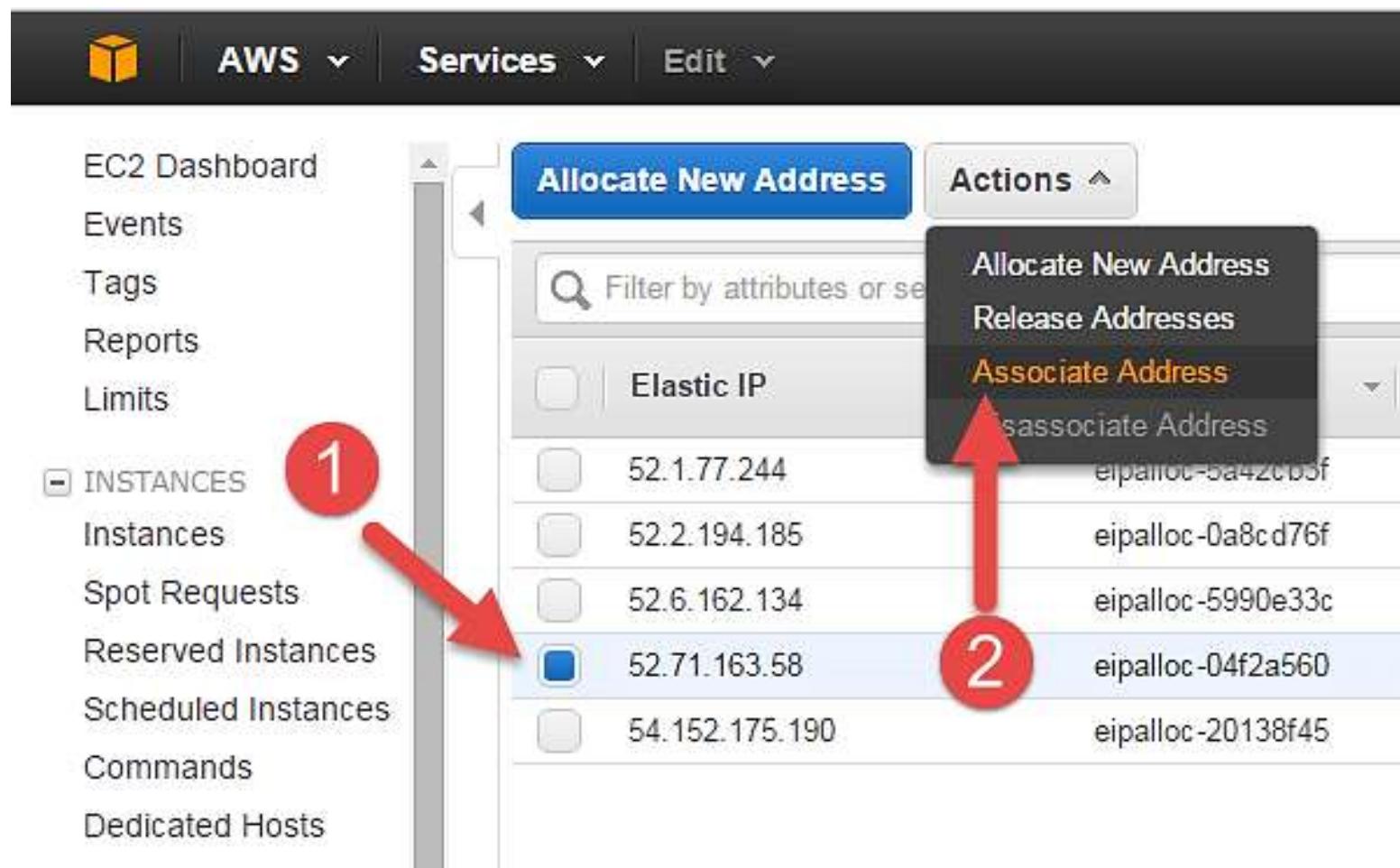
Cancel Yes, Allocate

Select an address above

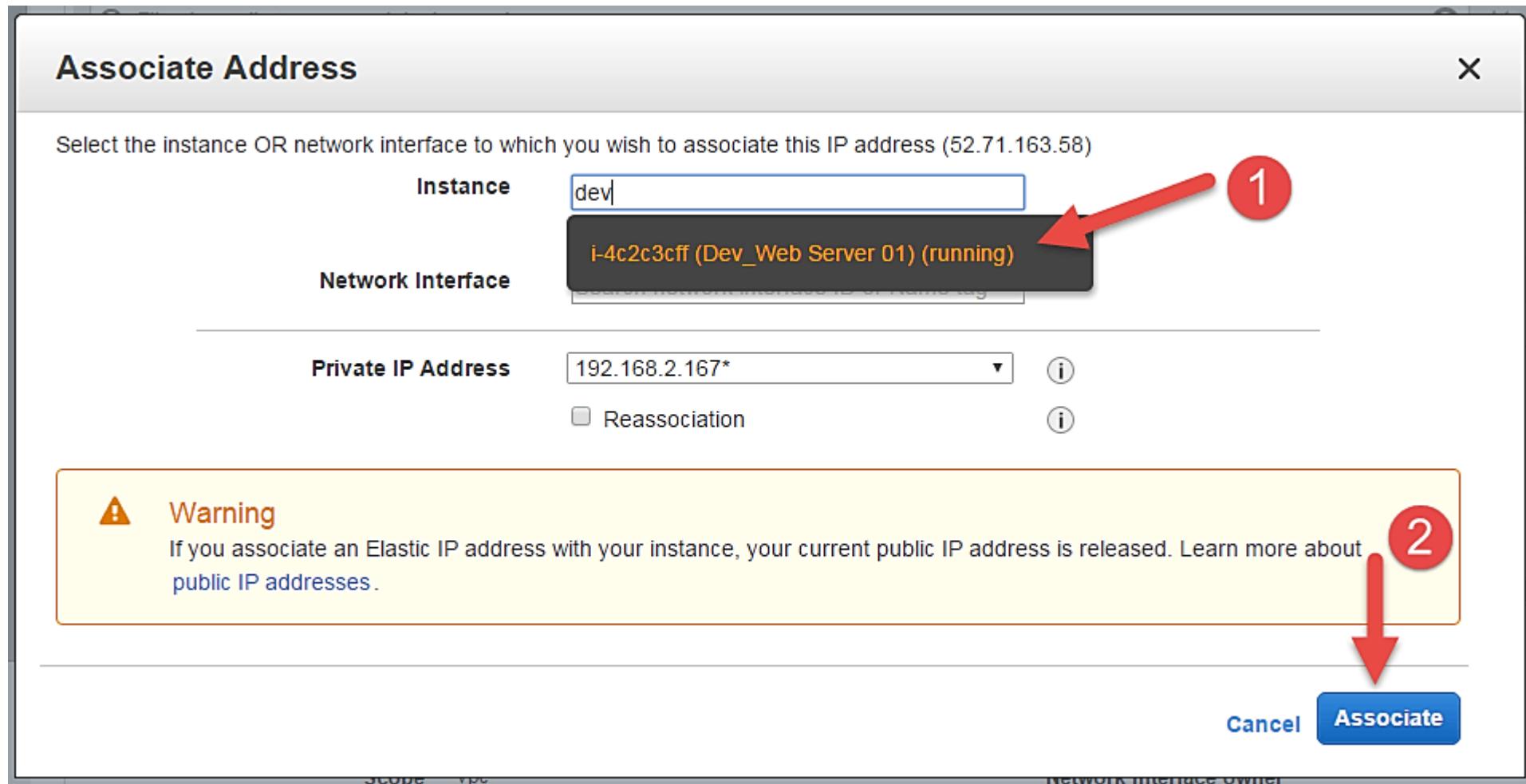
Now assign this IP to your instance.

Select the said IP

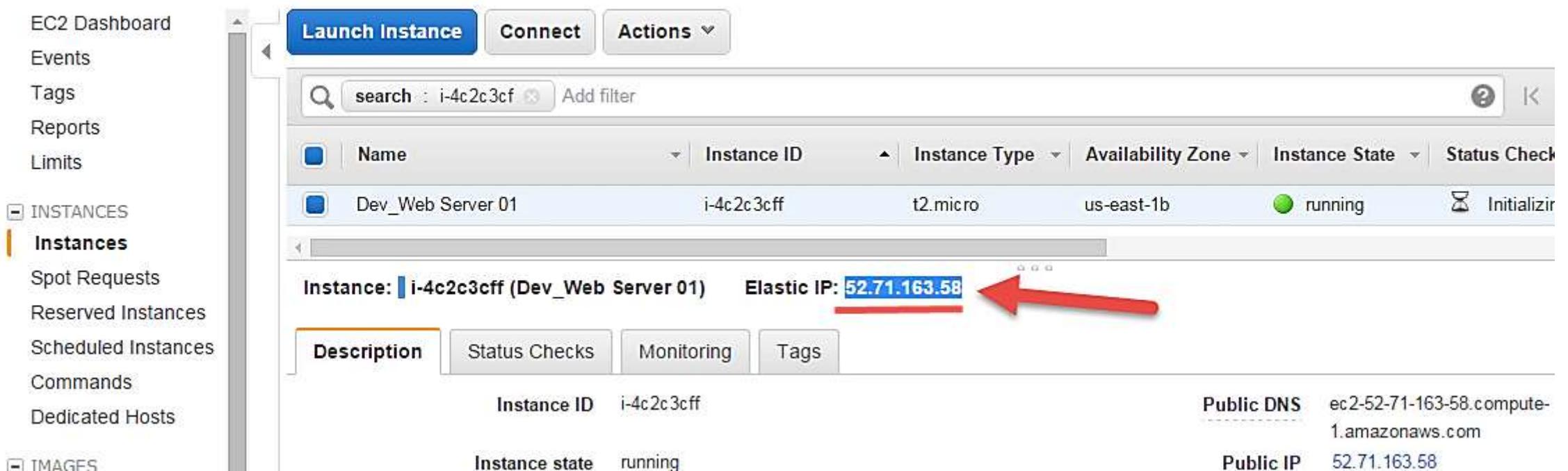
Click on Actions -> Associate Address



Search for your instance and  
Associate the IP to it.



Come back to your instances screen, you'll see that your instance has received your EIP.



The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area has a search bar with 'i-4c2c3cff' and a 'Launch Instance' button. The table lists one instance: 'Dev\_Web Server 01' (Instance ID: i-4c2c3cff, Instance Type: t2.micro, Availability Zone: us-east-1b, State: running). Below the table, the instance details are shown: Instance: i-4c2c3cff (Dev\_Web Server 01), Elastic IP: 52.71.163.58, Public DNS: ec2-52-71-163-58.compute-1.amazonaws.com, Public IP: 52.71.163.58. A red arrow points to the Elastic IP address.

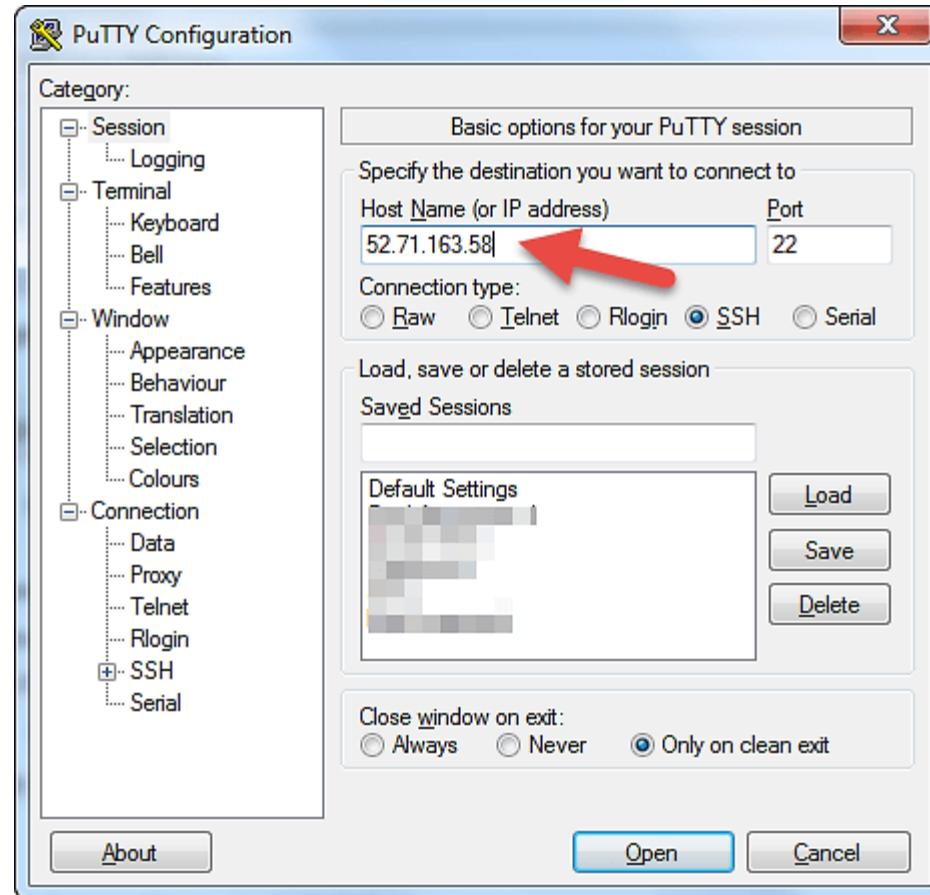
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
Dev_Web Server 01	i-4c2c3cff	t2.micro	us-east-1b	running	Initializir

Instance: i-4c2c3cff (Dev\_Web Server 01)    Elastic IP: 52.71.163.58

Instance ID: i-4c2c3cff    Public DNS: ec2-52-71-163-58.compute-1.amazonaws.com

Instance state: running    Public IP: 52.71.163.58

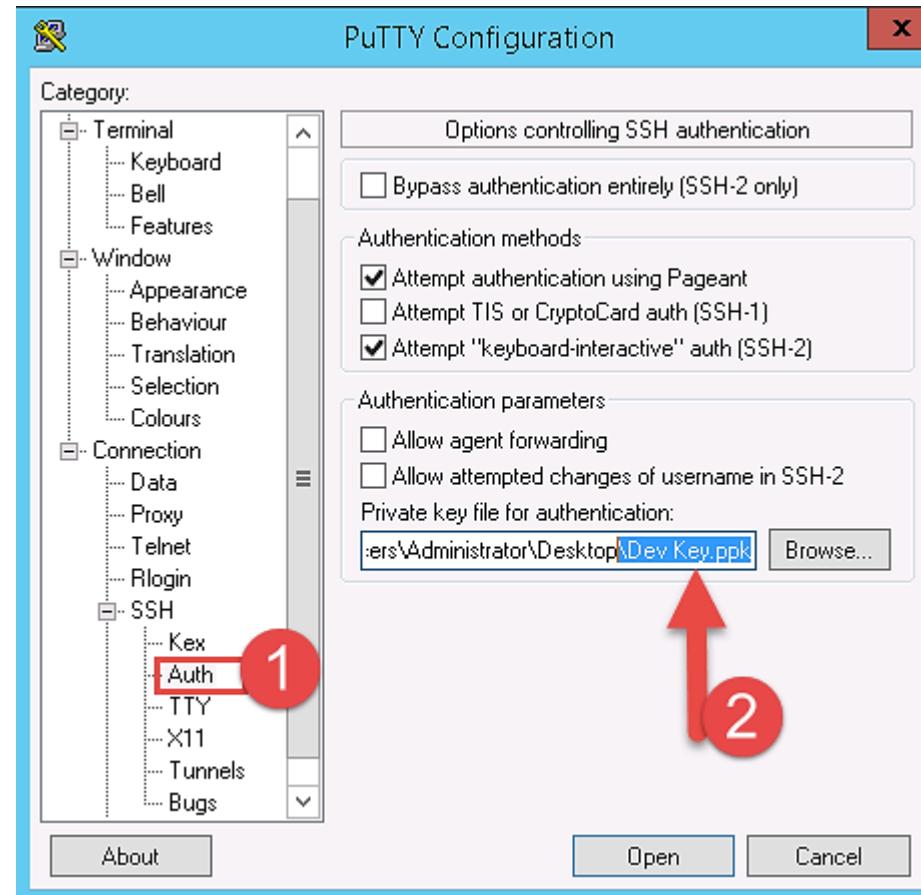
Now open putty from your programs list and add your same EIP in there as below.



Add your private key in putty for secure connection

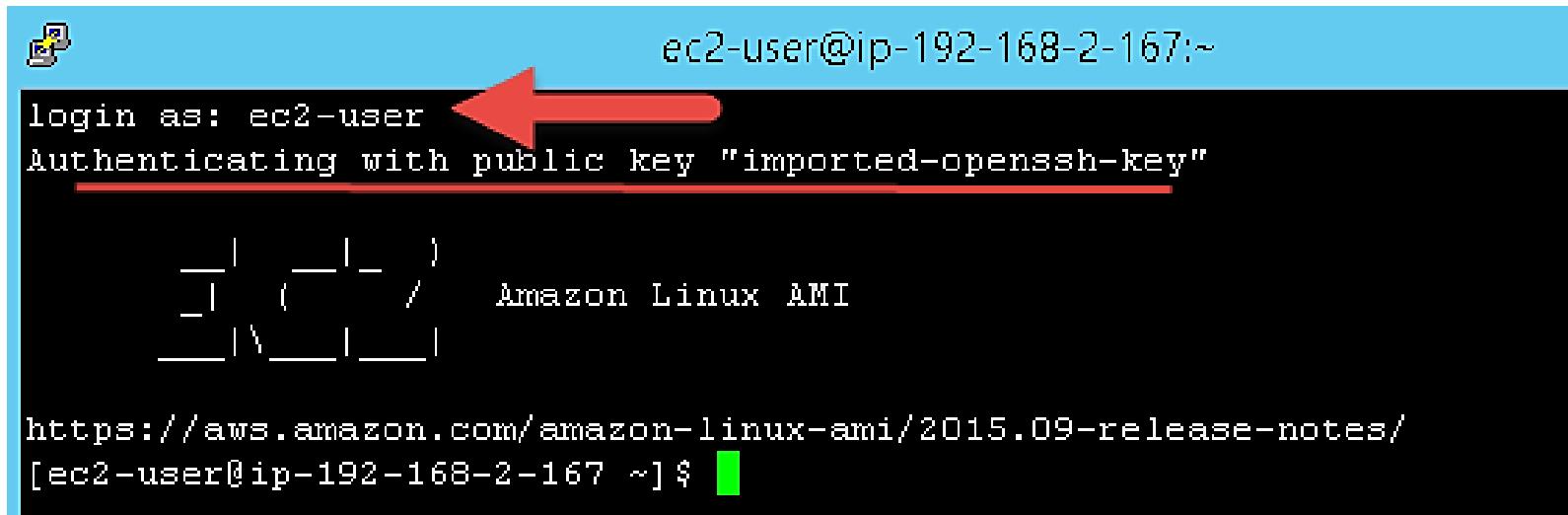
- Go to Auth
- Add your private key in .ppk (putty private key) format. You will need to convert pem file from AWS to ppk using puttygen

Once done click on "Open" button



Once you connect, you will successfully see the [Linux](#) prompt.

Please note that the machine you are connecting from should be enabled on the instance Security Group for SSH (like in the steps above).



```
ec2-user@ip-192-168-2-167:~$  
login as: ec2-user ←  
Authenticating with public key "imported-openssh-key"  
[ec2-user@ip-192-168-2-167 ~]$  
Amazon Linux AMI  
https://aws.amazon.com/amazon-linux-ami/2015.09-release-notes/  
[ec2-user@ip-192-168-2-167 ~]$
```

Once you become familiar with the above steps for launching the instance, it becomes a matter of 2 minutes to launch the same!

You can now use your on-demand EC2 server for your applications.



**KEEP  
CALM**

**it**

**is**

**FINISHED**

# Security & Storage Images & Snapshots

Rise 'n' Shine Technologies

By  
Reyaz

# An overview of security groups

- Security Groups are simple, yet powerful ways using which you can secure your entire EC2 environment.
- Use Security Groups to restrict and filter out both the ingress and egress traffic of an instance using a set of firewall rules.
- Each rule can allow traffic based on a particular protocol—TCP or UDP or Http or SSH

# Accessing Security Groups

- From the EC2 dashboard, select the **Security Groups** option located under the **Network & Security** section.
- Each Security Group is provided with a unique identifier called the **Group ID** and a **Group Name**.



# Default Security Group

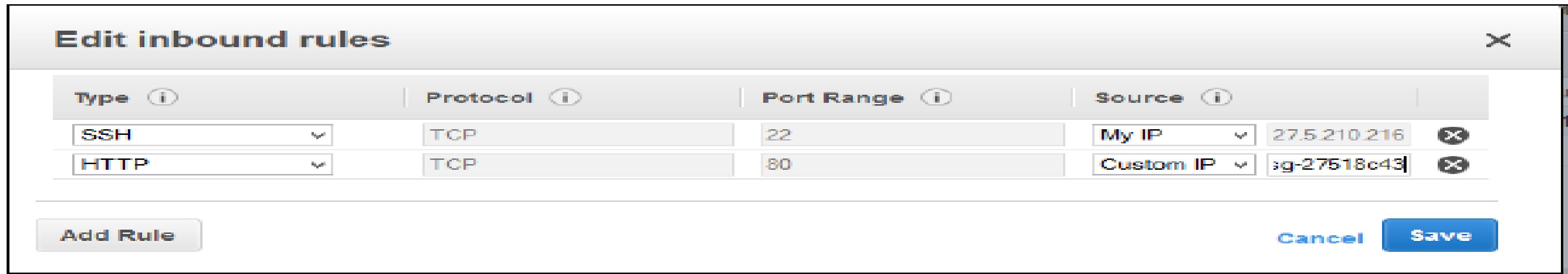
- This default Security Group is created by AWS when you first start and sign up for the EC2 service.
- The default Security Group has no ingress (inbound) traffic rules set;
- There is only one egress (outbound) rule, which allows your instances to connect to the outside world using any port and any protocol.
- You can add, delete, and modify any rules from this group; however, you cannot delete the default Security Group.

# Edit Security Groups

- You can modify the firewall rules of your Security Groups any time, even when your instance is running.
- From the dashboard, select a particular Security Group you wish to modify. Next, from the Actions drop-down list, select the option Edit inbound rules, as shown:



# Edit Security Groups



- **Type field:** which specifies the type of application for which you need to allow access. By default, AWS already has provided a list of common application types to choose from, which includes SSH, RDP, HTTP, HTTPS, POP3, IMAP, MySQL, SMTP, and so on so forth. You can additionally create custom TCP/ UDP application types using this same drop-down list as well.

# Edit Security Groups

- **Source field:** where you can specify any of these three options:
- **Anywhere:** Using this option as the source, your particular application port will be accessible from any and all networks out there (0.0.0.0/0). This is not a recommended configuration for any production environment and should be avoided at all times.
- **My IP:** As the name suggest, AWS will try and autofill the IP address of your local computer here. Your computer's IP address should not be based on a DHCP network.
- **Custom IP:** Perhaps the most preferable out of the three options, Specify your own custom source IP address or IP range as per your requirements. For example, allow access only via traffic coming from the network 203.20.31.0/24 CIDR.

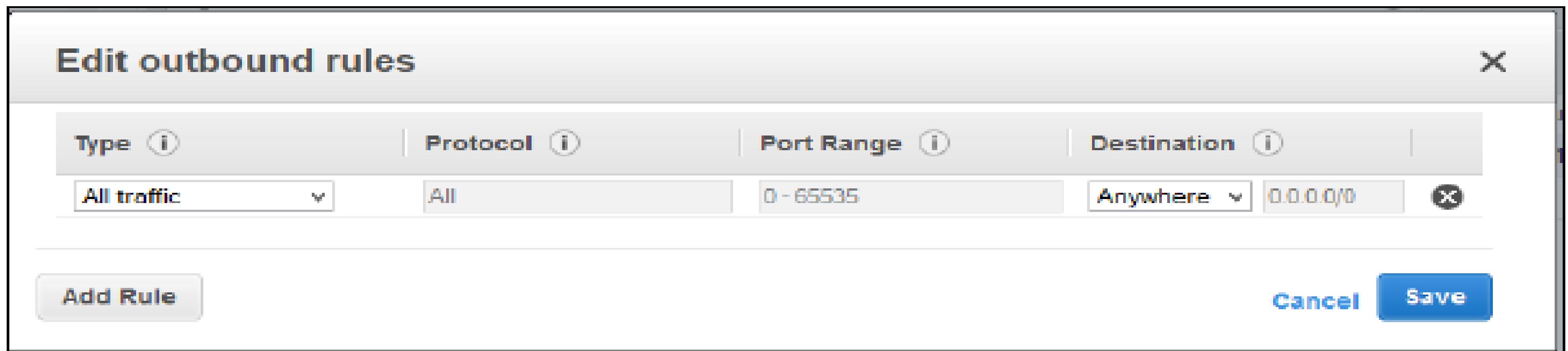
# Edit outbound rules

- You should see the default allow all access outbound rule, as shown here:

**Edit outbound rules**

Type	Protocol	Port Range	Destination
All traffic	All	0 - 65535	Anywhere 0.0.0.0/0

**Add Rule** **Cancel** **Save**

A screenshot of a web-based interface for managing outbound rules. The title bar says 'Edit outbound rules'. Below it is a table with four columns: 'Type', 'Protocol', 'Port Range', and 'Destination'. The 'Type' column has a dropdown menu set to 'All traffic'. The 'Protocol' column has a dropdown menu set to 'All'. The 'Port Range' column has an input field set to '0 - 65535'. The 'Destination' column has a dropdown menu set to 'Anywhere' with an IP address '0.0.0.0/0' next to it. At the bottom left is a 'Add Rule' button, and at the bottom right are 'Cancel' and 'Save' buttons.

# Create Security Group

- Security group name and Description
- Select the default VPC subnet from the VPC drop-down list.
- You can create up to 100 Security Groups in a VPC, with each Security Group having up to fifty firewall rules.

**Create Security Group**

Security group name  (i)

Description  (i)

VPC  (i) \* denotes default VPC

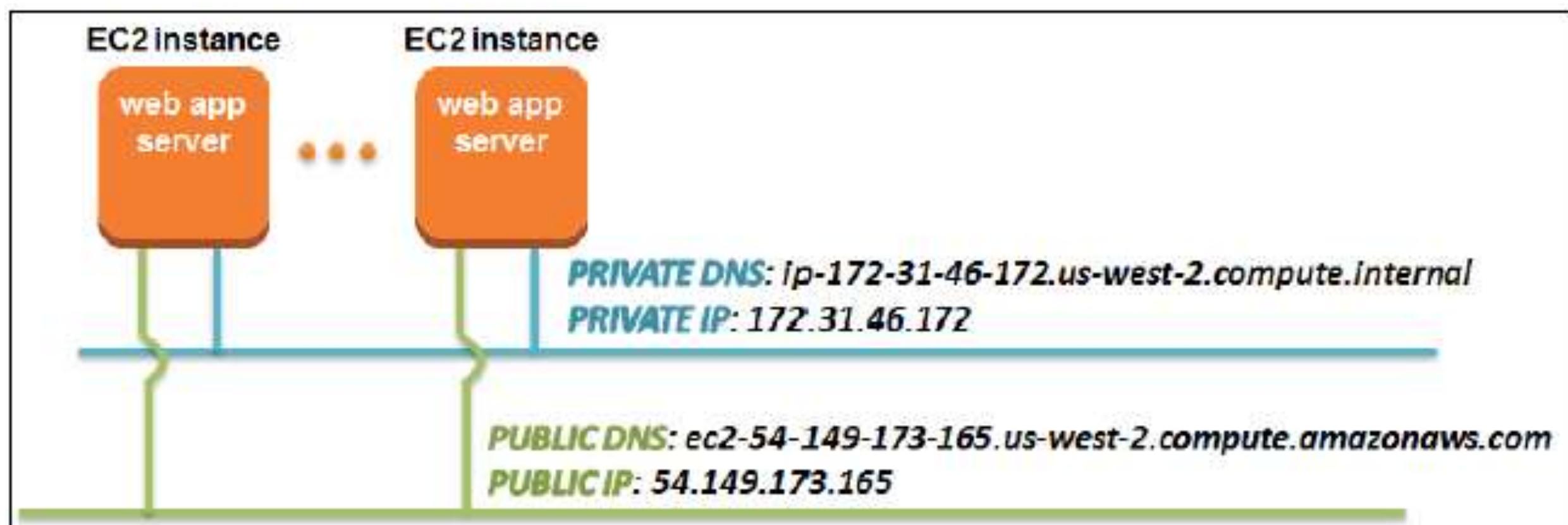
Security group rules:

Inbound

Type <small>(i)</small>	Protocol <small>(i)</small>	Port Range <small>(i)</small>	Source <small>(i)</small>
Custom TCP Rule <input type="button" value="X"/>	TCP <input type="text" value="232"/>	<input type="text" value="Custom IP"/> <input type="text" value="203.20.31.0/24"/>	<input type="button" value="X"/>
Custom TCP Rule <input type="button" value="X"/>	TCP <input type="text" value="3306"/>	<input type="text" value="Custom IP"/> <input type="text" value="203.20.31.102/32"/>	<input type="button" value="X"/>

# Private and Public IP address

- EC2 launches with two unique IP addresses called a private and public IP address.
- This is the default behavior of an instance and is not under your control by default, unless you are working with a VPC



# Private IP Address

- Private IP address to communicate with the instances present in the same network and not for outside (Internet).
- Along with the private IP address, you also get an internal DNS hostname for your instance.
- DNS resembles something like this string, ip-172-31-46-172.us-west-2.compute.internal
- DNS hostname resolves a private IP of 172-31-46-172 and also this particular instance is currently deployed in the us-west-2 region.

# Public IP Address

- This particular IP address is reachable from the Internet and can be used to communicate with the outside world.
- AWS maps the public IP address of an instance to its corresponding private IP address using simple NAT
- The public DNS resembles something like this string, ec2-54-149-173-165.us-west-2.compute.amazonaws.com

# Determining Instances IP Addresses

- The simplest by far is using the Description tab from the EC2 dashboard
- Select any particular running instance from the EC2 dashboard and view the instance's Private DNS, Private IPs, Public DNS, and Public IP.
- you should see an additional row called Secondary private IPs as well. These are the additional private IPs that you can allocate to your instance as per your needs. If you don't see these additional rows, then don't worry! You are probably running your instances from an EC2-Classic account and that's fine for now.

# Instance metadata

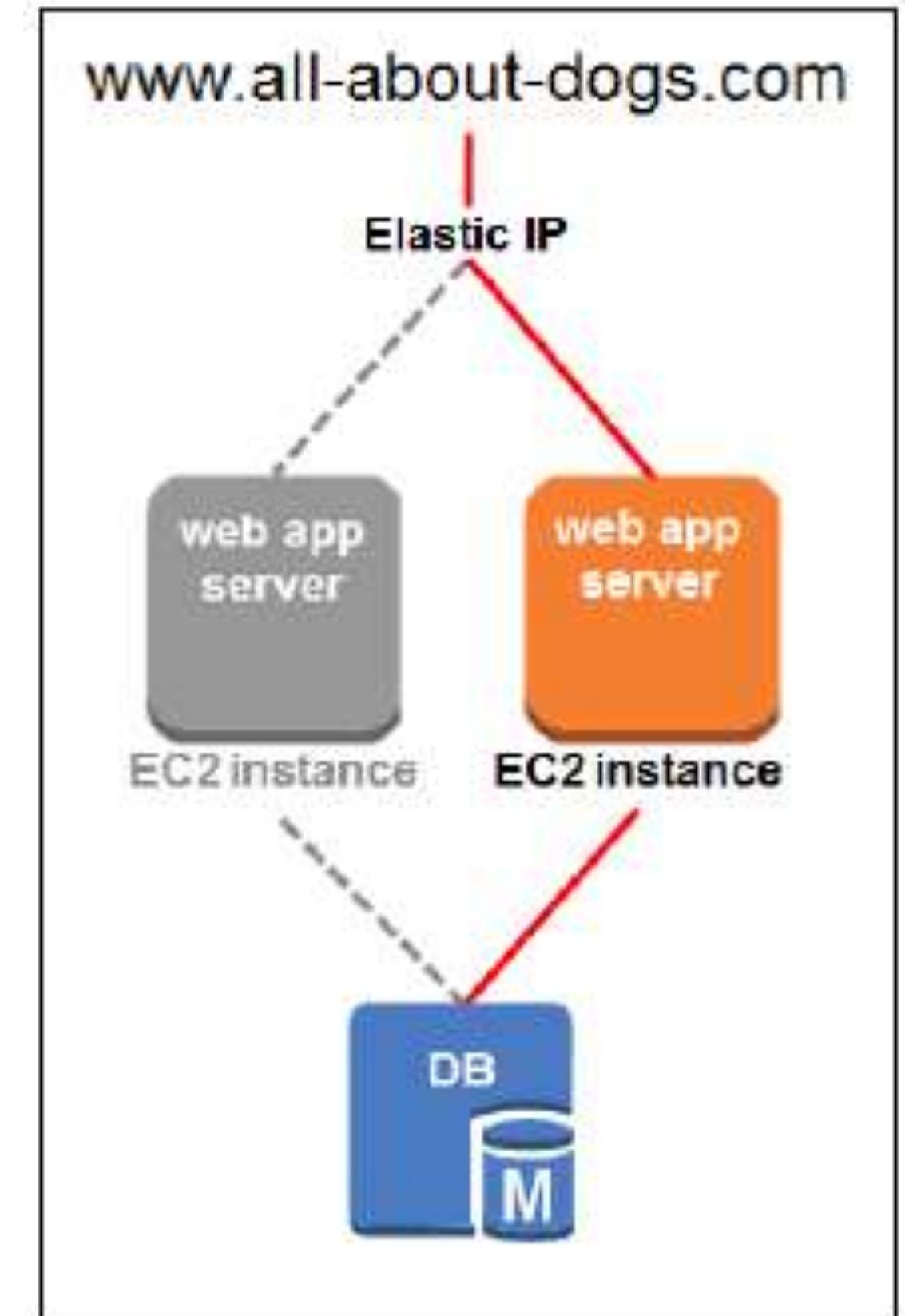
- Instance metadata is simply data about your instance.
- Information such as your instance's AMI ID, instance's hostname, block device mapping, network details, and a lot more can be obtained by querying against the instance's metadata.
- To determine your instance's IP addresses using instance metadata, simply connect to your running instance and run the following command:
- `# curl http://169.254.169.254/latest/meta-data/local-ipv4`
- `# curl http://169.254.169.254/latest/meta-data/public-ipv4`
- Running a Windows instance? You can still query its instance metadata by substituting curl with wget and running the command in your Windows command prompt.
-

# Elastic IP addresses

- Public and private IP addresses do not persist with the instance when it is powered off.
- If you want to assign a static IP address to your instance, need to use something called an **Elastic IP Address (EIP)**.
- EIPs are nothing but a bunch of static public IP addresses that AWS allocates to your account, not to your instances.
- Each AWS account can be associated with up to five EIPs.
- EIP is that it can be reassigned to a different running instance dynamically as and when needed.

# Use Case

- hosting a customer's website on AWS
- As with all websites, this design calls for a web server and a database server to begin with.
- We created and allocated an EIP to the web server instance.
- This EIP can then be mapped to a proper website name, such as all-about-dogs.com, using any DNS service, such as AWS Route 53 and so on.



# Use Case Continue..

- Now, if the web server instance undergoes any upgrades or maintenance activities, you can simply create a new, similar web server instance and point your EIP to it. Once the scheduled maintenance activity is over, simply swap the EIP back to the previous web server instance.
- When you add an EIP to your instance, AWS automatically releases that instance's public IP address to the general IP pool. On disassociating the EIP from your instance, AWS will once again provide your instance with a new public IP address from the general IP pool. All this happens really quickly, just a matter of minutes!

# EIP Charge

- How is an EIP charged? Well, for the first EIP that you attach to a running instance, you don't have to pay anything. However, you will need to shell out a minimum of \$0.005 per additional EIP for each instance on a per hourly basis.
- AWS imposes a small hourly charge (approx. \$0.005) on EIPs if they are attached to instances in a stopped state or not associated with running instances. This is just to make sure that the EIPs are used efficiently and not wasted.

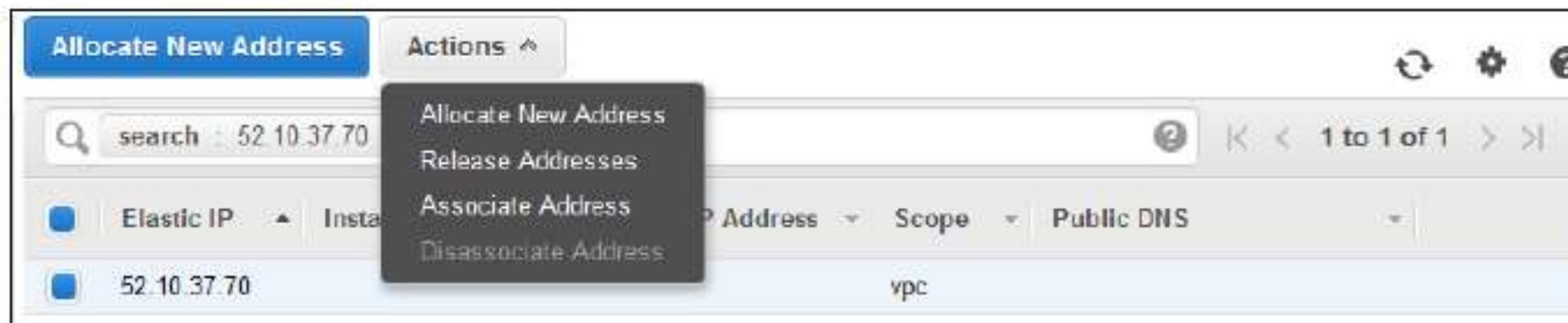
# Create an Elastic IP address

- Select the EC2 service option as EIPs are a part of the EC2 services. Next, from the navigation pane, select the Elastic IPs option.
- This will bring up the Elastic IP management dashboard as shown here. Since this is going to be our first EIP, simply go ahead and select the Allocate New Address option. In the confirmation dialog box, select Yes, Allocate to complete the process



# Allocating Elastic IP addresses

- Once your EIP has been created, you can go ahead and allocate it to any running instance from your current EC2 scope. Scope here can mean either EC2-Classic or a VPC environment
- To allocate the EIP, select the EIP, and from the Actions tab, select the option Associate Address, as shown:



# Allocating EIP

- You should see the Allocate New Address pop-up dialog box as shown. There are two ways in which you can allocate your EIPs to your instances, either by providing their Instance ID or by providing the instance's Network Interface information.
- Optionally, you can even select the Re association checkbox if you wish to re-allocate an EIP from one attached instance to a new instance.

# Disassociating and releasing an Elastic IP address

- Disassociating an EIP from an instance is an equally important task and can be performed quite easily using the EIP management dashboard. Select the particular EIP from the dashboard and from the **Actions tab**. Then select the **Disassociate Address** option. This will pop up a confirmation box detailing the EIP and its associated instance ID information, as shown here. Select Yes, Disassociate to complete the process:
- To release the EIP back to the pool, select the EIP from the dashboard. From the Actions tab, select the **Release Addresses** option. You will be provided with a confirmation box describing the current EIP address. Select Yes, Release to complete the process

# Understanding EBS volumes

- EBS volumes are nothing more than **block-level storage** devices that you can attach to your EC2 instances.
- They are highly durable and can provide a host of additional functionalities to your instances, such as data persistence, encryption, snapshotting capabilities, and so on.
- Majority of the time, these EBS volumes are used for storing data for a variety of applications and databases, however you can use it just as a normal hard drive as well.
- The best part of EBS volumes is that they can persist independently from your instances. So powering down an instance or even terminating it will not affect the state of your EBS volumes. Your data will stay on it unless and until you explicitly delete it.

# Features and Benefits Of EBS volumes

- High availability:
  - Unlike your instance store-backed drives, EBS volumes are automatically replicated by AWS within the availability zone in which they are created.
  - You can create an EBS volume and attach it to any instance present in the same availability zone;
  - One EBS volume cannot be attached to multiple instances at the same time.
  - A single instance, however, can have multiple EBS volumes attached to it at any given time.

- **Encryption capabilities:**
  - EBS volumes provide an add-on feature using which you can encrypt your volumes using standard encryption algorithms, such as AES-256, and keys as well.
  - These keys are autogenerated the first time you employ encryption on a volume using the AWS Key Management Service (KMS).
- **Snapshot capabilities:**
  - The state of an EBS volume can be saved using point-in-time snapshots.
  - These snapshots are all stored incrementally on your Amazon S3 account and can be used for a variety of purposes, such as creating new volumes based on an existing one, resizing volumes, backup and data recovery, and so on.
- **Note:**
  - EBS volumes cannot be copied from one AWS region to another. In such cases, you can take a snapshot of the volume and copy the snapshot over to a different region using the steps mentioned at <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-copy-snapshot.html>.

# EBS volume types

- There are three different types of EBS volumes
- **General purpose volumes (SSD):**
- **Provisioned IOPS volumes (SSD):**
- **Magnetic volumes:**

# General purpose volumes (SSD):

- Commonly used EBS volume types as they provide a good balance between cost and overall performance.
- By default, this volume provides a standard 3 IOPS per GB of storage
- So a 10 GB general purpose volume will get approximately 30 IOPS and so on so forth, with a max value of 10,000 IOPS.
- Range in size from 1 GB to a maximum of 16 TB.
- Such volumes can be used for a variety of purposes, such as instance root volumes, data disks for dev and test environments, database storage, and so on.

# Provisioned IOPS volumes (SSD):

- These are a specialized set of SSDs that can consistently provide a minimum of 100 IOPS burstable up to 20,000 IOPS.
- You can create Provisioned IOPS Volumes that range in size from a minimum of 4 GB all the way up to 16 TB.
- Such volumes are ideally suited for applications that are IO intensive, such as databases, parallel computing workloads such as Hadoop, and so on.

# Magnetic volumes

- Very similar to traditional tape drives and magnetic disks, these volumes are a good match for workloads where data is accessed infrequently, such as log storage, data backup and recovery, and so on.
- On an average, these volumes provide up to a 100 IOPS with an ability to burst up to 1,000 IOPS.
- You can create Magnetic volumes that range in size from a minimum of 1 GB all the way up to 1 TB.

# Getting started with EBS Volumes

- To view and access your account's EBS Volumes using AWS Management Console, simply select the Volumes option from the EC2 dashboard's navigation pane
- This will bring up the Volume Management dashboard
- Each EBS-backed instance's volume will appear here in the Volume Management dashboard.

# EBS Dashboard

- You can view the volume's ID, Size, Created date, the volume's current State as well as its Attachment information, which displays the volume's mount point on a particular instance

The screenshot shows the AWS EBS Dashboard. At the top, there are buttons for 'Create Volume' and 'Actions'. Below is a search bar with the placeholder 'Filter by tags and attributes or search by keyword'. A table lists two volumes:

	Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created	Availability Zone	State	Alarm Status	Attachment Information
<input type="checkbox"/>	vol-0462ad8...	8 GiB	gp2	100	snap-00f00b3a...	December 18, 2018...	ap-south-1a	<span>in-use</span>	None		i-089c3070dc713f23...
<input checked="" type="checkbox"/>	vol-07d2534...	8 GiB	gp2	100	snap-00f00b3a...	December 18, 2018...	ap-south-1a	<span>in-use</span>	None		i-0bf8b55f6ada13356...

Below the table, a message says 'Volumes: vol-07d2534e8570d47bf'. A detailed view for the selected volume (vol-07d2534e8570d47bf) is shown with tabs for 'Description', 'Status Checks', 'Monitoring', and 'Tags'. The 'Description' tab is selected. The volume details are:

Volume ID	vol-07d2534e8570d47bf	Alarm status	None
Size	8 GiB	Snapshot	<a href="#">snap-00f00b3a3718745e9</a>
Created	December 18, 2018 at 7:28:28 PM UTC+5:30	Availability Zone	ap-south-1a
State	in-use	Encrypted	Not Encrypted
Attachment information	<a href="#">i-0bf8b55f6ada13356 (Main):/dev/xvda (attached)</a>	KMS Key ID	
Volume type	gp2	KMS Key Aliases	
Product codes	marketplace:	KMS Key ARN	
IOPS	100		

# Creating EBS volumes

- **Type:** From the Type drop-down list, select either General Purpose (SSD), Provisioned IOPS (SSD), or Magnetic as per your requirements.
- **Size (GiB):** Provide the size of your volume in GB. Here, I provided 10 GB.
- **IOPS:** This field will only be editable if you have selected Provisioned IOPS (SSD) as the volume's type. Enter the max IOPS value as per your requirements.
- **Availability Zone:** Select the appropriate availability zone in which you wish to create the volume. Remember, an EBS volume can span availability zones, but not regions.

# Creating EBS Volumes

- **Snapshot ID:** This is an optional field. You can choose to populate your EBS volume based on a third party's snapshot ID. In this case, we have left this field blank.
- **Encryption:** As mentioned earlier, you can choose whether or not you wish to encrypt your EBS Volume. Select Encrypt this volume checkbox if you wish to do so.
- **Master Key:** On selecting the Encryption option, AWS will automatically create a default key pair for the AWS's KMS. You can make a note of the KMS Key ID as well as the KMS Key ARN as these values will be required during the volume's decryption process as well.

# Creating EBS volumes

## Create Volume

Volume Type  ⓘ

Size (GiB)  (Min: 1 GiB, Max: 16384 GiB) ⓘ

IOPS 300 / 3000 (Baseline of 3 IOPS per GiB with a minimum of 100 IOPS, burstable to 3000 IOPS) ⓘ

Availability Zone\*  ⓘ

Throughput (MB/s) Not applicable ⓘ

Snapshot ID  ⓘ C ⓘ

Encryption  Encrypt this volume ⓘ

Master Key  ⓘ C

Key (127 characters maximum) | Value (255 characters maximum) |

This resource currently has no tags

Choose the Add tag button or [click to add a Name tag](#)

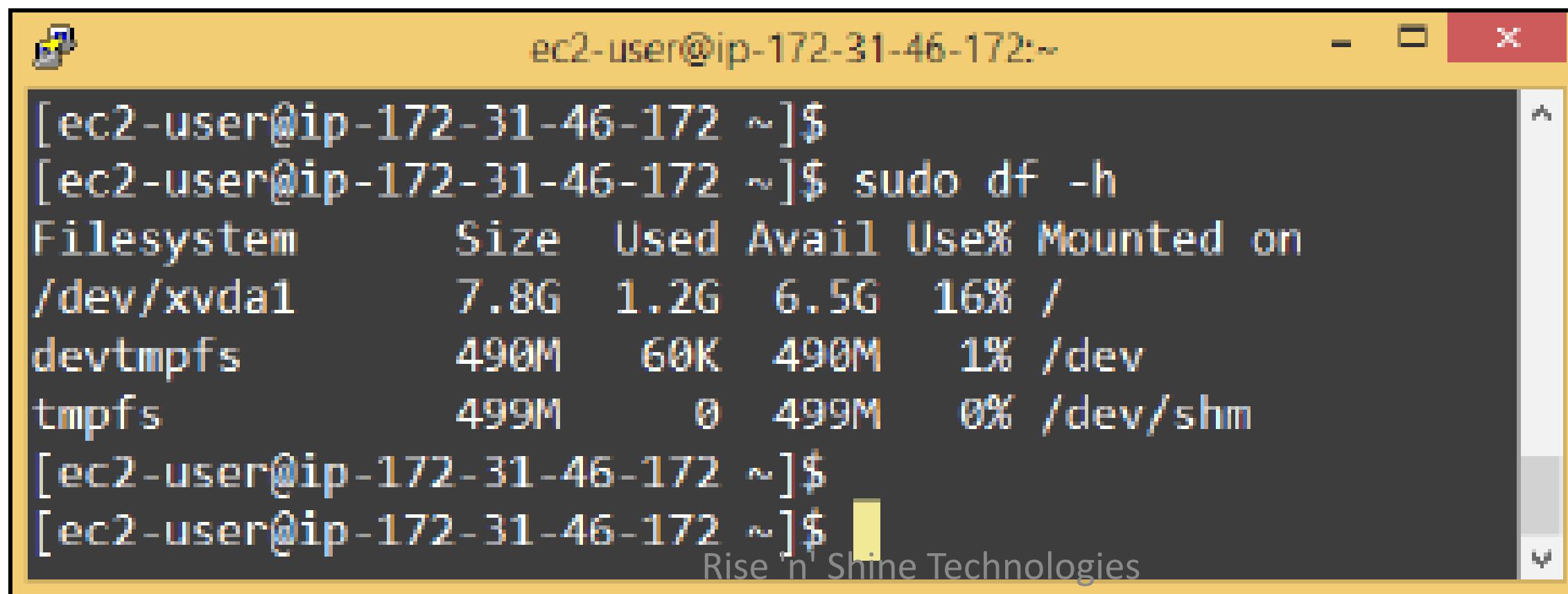
50 remaining (Up to 50 tags maximum)

# Attaching EBS volumes

- Once the EBS volume is created, make sure it is in the **available state**
- You can attach **multiple volumes** to a **single instance** at a time, with each volume having a unique device name.
- Some of these device names are reserved, for example, **/dev/sda1** is reserved for the **root device** volume.
- The complete list of potential and recommended device names at
  - [http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/device\\_naming.html](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/device_naming.html).

# Accessing volumes from an instance

- First up connect to your running instance using putty or any other SSH client
- Check the current disk partitioning of your instance
  - **# sudo df -h**
  - You should see a **/dev/xvda1** like filesystem mounted on the root (/) partition along with few other temp filesystems



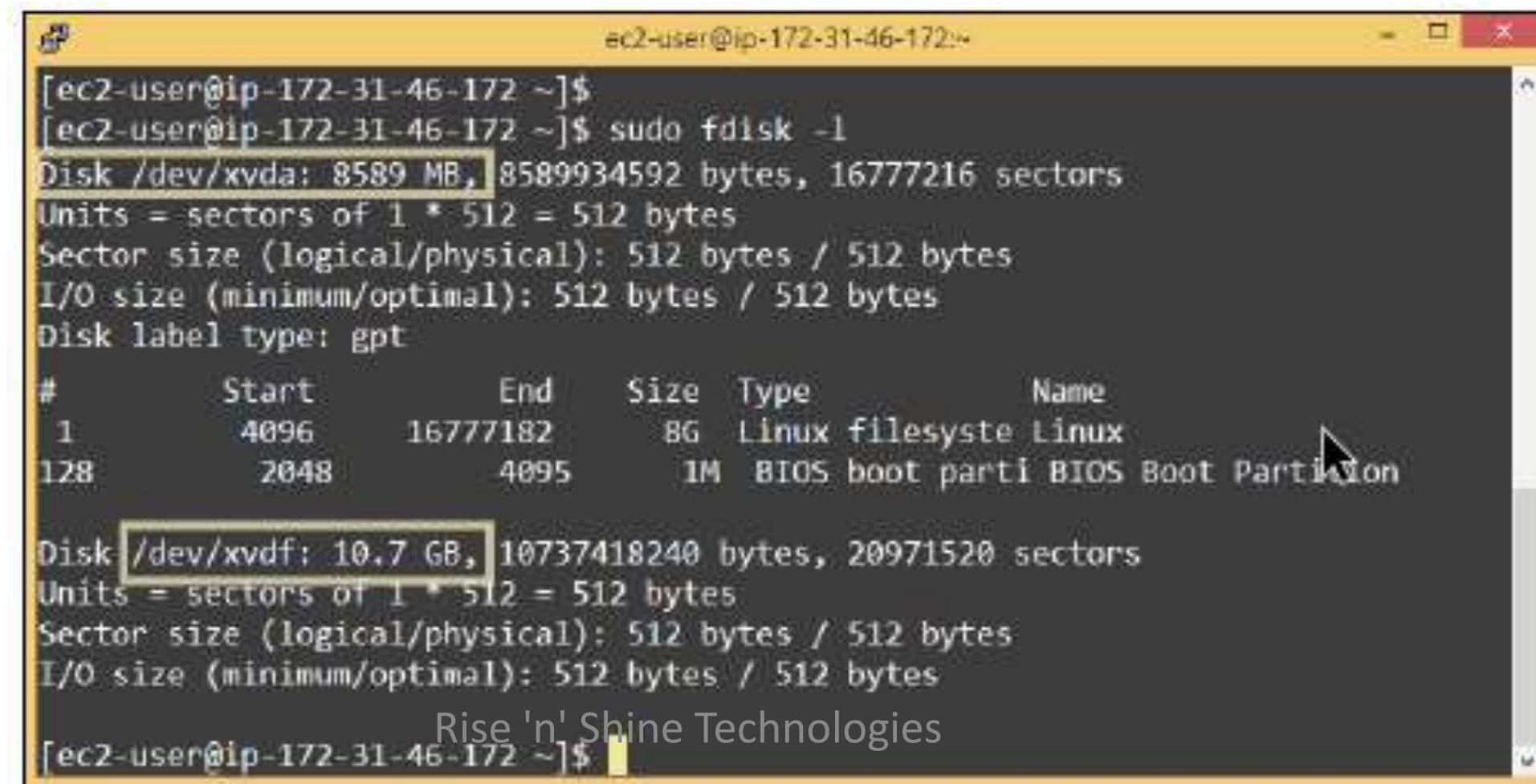
The screenshot shows a terminal window with the title bar 'ec2-user@ip-172-31-46-172:~'. The window contains the following text:

```
[ec2-user@ip-172-31-46-172 ~]$  
[ec2-user@ip-172-31-46-172 ~]$ sudo df -h  
Filesystem      Size  Used Avail Use% Mounted on  
/dev/xvda1      7.8G  1.2G  6.5G  16% /  
devtmpfs        490M   60K  490M   1% /dev  
tmpfs          499M     0  499M   0% /dev/shm  
[ec2-user@ip-172-31-46-172 ~]$  
[ec2-user@ip-172-31-46-172 ~]$
```

At the bottom of the terminal window, the text 'Rise n' Shine Technologies' is visible.

# Checking the Volumes from Instance

- Command to list out partitions on your current instance
  - **# sudo fdisk -l**
- You should see a default /dev/xvda partition along with its partition table and an unformatted disk partition with the name /dev/xvdf



```
[ec2-user@ip-172-31-46-172 ~]$  
[ec2-user@ip-172-31-46-172 ~]$ sudo fdisk -l  
Disk /dev/xvda: 8589 MB, 8589934592 bytes, 16777216 sectors  
Units = sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disk label type: gpt  


| #   | Start | End      | Size | Type             | Name                          |
|-----|-------|----------|------|------------------|-------------------------------|
| 1   | 4096  | 16777182 | 8G   | Linux filesystem | linux                         |
| 128 | 2048  | 4095     | 1M   | BIOS boot        | partition BIOS Boot Partition |

  
Disk /dev/xvdf: 10.7 GB, 10737418240 bytes, 20971520 sectors  
Units = sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
[ec2-user@ip-172-31-46-172 ~]$
```

Rise 'n' Shine Technologies

- Once you have verified the name of your newly added disk, you can go ahead and format with a filesystem of your choice. In this case, I have gone ahead and used the ext4 filesystem for my new volume:
  - **# sudo mkfs -t ext4 /dev/xvdf**
- Now that your volume is formatted, you can create a new directory on your Linux instance and mount the volume to it using your standard Linux commands:
  - **# sudo mkdir /my-new-dir**
  - **# sudo mount /dev/xvdf /my-new-dir**

# Detaching EBS volumes

- You will first need to unmount the volume from your instance and then simply detach it using **Volume Management dashboard**.
- Run the following command to unmount the EBS volume from the instance:
  - `# sudo umount /dev/sdf`
- **Note:** Make sure you are unmounting the correct volume from the instance. *Do not try and unmount the /dev/sda or any other root partitions.*
- Once the volume is successfully unmounted from the instance, detach the volume by selecting the **Detach Volume** option from the **Actions** tab

# Backing up volumes using EBS snapshots

- AWS automatically replicates EBS volumes so that your data is preserved even in case the complete drive fails.
- But this replication is limited only to the availability zone in which the drive or EBS volume was created, which means if that particular availability zone was to go down for some reason, then there is no way for you to back up your data.
- Fortunately for us, AWS provides a very simple yet highly efficient method of backing EBS volumes, called as EBS snapshots.

# EBS to S3

- An EBS snapshot in simple terms is a state of your volume at a particular point in time.
- Each snapshot that you take is stored incrementally in Amazon S3, but, you will not be able to see these snapshots in your S3 buckets; they are kind of hidden away and stored separately.

# Tasks using snapshots

- **Create new volumes based on existing ones:** Snapshots are a great and easy way to spin up new volumes. A new volume spawned from a snapshot is an exact replica of the original volume, down to the last detail.
- **Expand existing volumes:** Snapshots can also be used to expand an existing EBS Volume's size as well. It is a multistep process, which involves you taking a snapshot of your existing EBS volume and creating a larger new volume from the snapshot.
- **Share your volumes:** Snapshots can be shared within your own account (*private*) as well publicly.
- **Backup and disaster recovery:** Snapshots are a handy tool when it comes to backing up your volumes. You can create multiple replicates of an existing volume within an AZ, across AZs that belong to a particular region, as well as across regions, using something called an EBS Snapshot copy mechanism.

# SNAPSHOTS

- To create a snapshot of your volumes, all you need to do is select the particular **volume from the Volume Management dashboard**. Click on the Actions tab and select the Create Snapshot option
- **Note:**
  - It is really a good practice to stop your instance before taking a snapshot if you are taking a snapshot of its root volume. This ensures a consistent and complete snapshot of your volume at all times.

- You should see the Create Snapshot dialog box as shown in the following screenshot.
- Provide a suitable Name and Description for your new snapshot.
- If the original volume was not encrypted, neither will the snapshot be encrypted. Snapshots of encrypted volumes are automatically encrypted.
- Even new volumes created from an encrypted snapshot are encrypted automatically.
- Once you have finished providing the details, click on Create to complete the snapshot process:

- Once the snapshot process is completed, you can use this particular snapshot and **Create Volume**, **Copy this snapshot from one region to another**, and **Modify Snapshot Permissions** to private or public as you see fit. These options are all present in the **Actions** tab of your **Snapshot Management** dashboard:

# Create Image using SnapShot

- you can use snapshots to create AMIs as well. From the **Actions** tab, select the **Create Image** option. You should see the **Create Image from EBS Snapshot** wizard as shown here. Fill in the required details and click on **Create** to create your very first AMI.

**Create Image from EBS Snapshot**

Name	US-P-WebServer-Image-v1.0	Description	US Prod WebServer Image v1.0				
Architecture	i386_64	Virtualization type	Paravirtual				
Root device name	/dev/sda1	Kernel ID	Use default				
RAM disk ID	Use default						
<b>Block Device Mappings:</b>							
Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/sda1	snap-d8763091	8	General Purpose (S)	24 / 3000	<input checked="" type="checkbox"/>	Not Encrypted
<b>Add New Volume</b>							

**Rise 'n' Shine Technologies**

**Create**

- The details contain the following options:
- **Name:** Provide a suitable and meaningful name for your AMI.
- **Description:** Provide a suitable description for your new AMI.
- **Architecture:** You can either choose between **i386 (32 bit)** or **x86\_64 (64 bit)**.
- **Root device name:** Enter a suitable name for your root device volume. Ideally, a root device volume should be labelled as **/dev/sda1** as per EC2's device naming best practices.

- **Virtualization type:** You can choose whether the instances launched from this particular AMI will support Paravirtualization (PV) or Hardware Virtual Machine (HVM) virtualization.
- Note: You can read more about the various Virtualization types supported by EC2 at [http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/virtualization\\_types.html](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/virtualization_types.html).
- **RAM disk ID, Kernel ID:** You can select and provide your AMI with its own RAM disk ID (ARI) and Kernel ID (AKI); however, in this case I have opted to keep the default ones.
- **Block Device Mappings:** You can use this dialog to either expand your root volume's size or add additional volumes to it. You can change the Volume Type from General Purpose (SSD) to Provisioned IOPS (SSD) or Magnetic as per your AMI's requirements. For now, I have left these to their default values.
- An important point to note here is that you will not be able to delete this particular EBS Snapshot now as it is in use by your AMI. You will have to deregister your AMI first from the AMI Management dashboard and then try and delete the snapshot.

# Recommendations and best practices

- Create and use IAM policies and allow only a particular set of users from accessing your EBS volumes.
- Create and take periodic snapshots of your volumes. Always remember to provide suitable names and descriptions for your snapshots so that they can be easily identified and re-used.
- Always take snapshots during the nonbusiness hours of your application.
- Clean up unused or older snapshots to save on unnecessary costs.
- Encrypt your EBS volumes if you have some sensitive data stored on them.
- Select and use the correct type of EBS volume as per your application's needs. Use performance-optimized volumes for your high-performance applications and magnetic volumes for applications that do not need a lot of data read and write.

**LET'S START!**

# Need more Space? Storage?

- The need for **storage** is increasing every day, so building and maintaining your own repositories, therefore, becomes a tedious and tiresome job because knowing the amount of capacity you may need in the future is difficult to predict.
- You may either over-utilize it leading to an application failure because of not having sufficient space or you may end up buying stacks of storage which will then be under-utilized.
- Keeping all these hassles in mind, Amazon came up with an internet storage service called *AWS S3*.

# Amazon Simple Storage Service (S3)



- **Amazon S3** (Simple Storage Service) is a scalable, high-speed, low-cost web-based service designed for online backup and archiving of data and application programs.
- It allows to upload, store, and download any type of files up to 5 GB in size. This service allows the subscribers to access the same systems that Amazon uses to run its own web sites. The subscriber has control over the accessibility of data, i.e. privately/publicly accessible.

# Advantages of S3



Amazon S3 is intentionally built with a minimal feature set that focuses on simplicity and robustness. Following are some of advantages of the Amazon S3 service:

- **Create Buckets** – Create and name a bucket that stores data. Buckets are the fundamental container in Amazon S3 for data storage.
- **Store data in Buckets** – Store an infinite amount of data in a bucket. Upload as many objects as you like into an Amazon S3 bucket. Each object can contain up to 5 TB of data. Each object is stored and retrieved using a unique developer-assigned key.
- **Download data** – Download your data or enable others to do so. Download your data any time you like or allow others to do the same.
- **Permissions** – Grant or deny access to others who want to upload or download data into your Amazon S3 bucket. Grant upload and download permissions to three types of users. Authentication mechanisms can help keep data secure from unauthorized access.
- **Standard interfaces** – Use standards-based REST and SOAP interfaces designed to work with any Internet-development toolkit.

# Amazon S3 Concepts



This section describes key concepts and terminology you need to understand to use Amazon S3 effectively. They are presented in the order you will most likely encounter them.

- Buckets
- Objects
- Keys
- Regions

# Buckets



- A bucket is a container for objects stored in Amazon S3. Every object is contained in a bucket.
- For example, if the object named photos/puppy.jpg is stored in the johnsmith bucket, then it is addressable using the URL <http://johnsmith.s3.amazonaws.com/photos/puppy.jpg>
- You can configure buckets so that they are created in different regions.
- You can also configure a bucket so that every time an object is added to it, Amazon S3 generates a unique version ID and assigns it to the object.

# Objects



- Objects are the fundamental entities stored in Amazon S3.
- Objects consist of object data and metadata.
- The metadata is a set of name-value pairs that describe the object. These include some default metadata, such as the date last modified, and standard HTTP metadata, such as Content-Type. You can also specify custom metadata at the time the object is stored.
- An object is uniquely identified within a bucket by a key (name) and a version ID

# Keys



- A key is the unique identifier for an object within a bucket.
- Every object in a bucket has exactly one key. Because the combination of a bucket, key, and version ID uniquely identify each object.
- Amazon S3 can be thought of as a basic data map between "bucket + key + version" and the object itself. Every object in Amazon S3 can be uniquely addressed through the combination of the web service endpoint, bucket name, key, and optionally, a version.
- For example, in the URL <http://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl>, "doc" is the name of the bucket and "2006-03-01/AmazonS3.wsdl" is the key.

# Regions



- You can choose the geographical region where Amazon S3 will store the buckets you create.
- You might choose a region to optimize latency, minimize costs, or address regulatory requirements. Objects stored in a region never leave the region unless you explicitly transfer them to another region. For example, objects stored in the EU (Ireland) region never leave it.

# How is data organized in S3?

Data in S3 is organized in the form of buckets.

- A Bucket is a logical unit of storage in S3.
- A Bucket contains objects which contain the data and metadata.

Before adding any data in S3 the user has to create a bucket which will be used to store objects



# Amazon Simple Storage Service (S3)



- Amazon Simple Storage Service (S3) is a storage for the internet. It is designed for large-capacity, low-cost storage provision across multiple geographical regions. Amazon S3 provides developers and IT teams with **Secure, Durable and Highly Scalable** object storage.

# Amazon S3 Features



- **Low cost and Easy to Use** – Using Amazon S3, the user can store a large amount of data at very low charges.
- **Secure** – Amazon S3 supports data transfer over SSL and the data gets encrypted automatically once it is uploaded. The user has complete control over their data by configuring bucket policies using AWS IAM.
- **Scalable** – Using Amazon S3, there need not be any worry about storage concerns. We can store as much data as we have and access it anytime.
- **Durable** – It regularly verifies the integrity of data stored using checksums e.g. if S3 detects there is any corruption in data, it is immediately repaired with the help of replicated data.
- **Higher performance** – Amazon S3 is integrated with Amazon CloudFront, that distributes content to the end users with low latency and provides high data transfer speeds without any minimum usage commitments.
- **Integrated with AWS services** – Amazon S3 integrated with AWS services include Amazon CloudFront, Amazon CloudWatch, Amazon Kinesis, Amazon RDS, Amazon Route 53, Amazon VPC, AWS Lambda, Amazon EBS, Amazon Dynamo DB, etc.

# What kind and how much of data one can store in AWS S3?

You can store virtually any kind of data, in any format, in S3 and when we talk about capacity, the volume and the number of objects that we can store in S3 are unlimited.

\*An object is the fundamental entity in S3. It consists of data, key and metadata.

When we talk about data, it can be of two types-

- Data which is to be accessed frequently.
- Data which is accessed not that frequently.

Therefore, Amazon came up with 3 storage classes to provide its customers the best experience and at an affordable cost.

# Let's understand the 3 storage classes with a “health-care” use case:

- **Amazon S3 Standard for frequent data access**

This is suitable for performance sensitive use cases where the latency should be kept low. e.g. in a hospital, frequently accessed data will be the data of admitted patients, which should be retrieved quickly.

- **Amazon S3 Standard for infrequent data access**

This is suitable for use cases where the data is long lived and less frequently accessed, i.e for data archival but still expects high performance. e.g. in the same hospital, people who have been discharged, their records/data will not be needed on a daily basis, but if they return with any complication, their discharge summary should be retrieved quickly.

- **Amazon Glacier**

Suitable for use cases where the data is to be archived, and high performance is not required, it has a lower cost than the other two services. e.g. in the hospital, patients' test reports, prescriptions, MRI, X Ray, Scan docs etc. that are older than a year will not be needed in the daily run and even if it is required, lower latency is not needed.

Characteristics	Standard	Standard - Infrequent Access	Glacier
Durability	99.99%	99.99%	99.99%
Availability	99.99%	99.90%	N/A
Minimum Object Size	No limit	128KB	No limit
Minimum Storage Duration	No minimum duration	30 Days	90 Days
First Byte Latency	milliseconds	milliseconds	4 hours
Retrieval Fee	No Fee	per GB retrieved	per GB retrieved

# Where is your data stored geographically?

You can self-choose where or in which region your data should be stored. Making a decision for the region is important and therefore it should be planned well.

These are the 4 parameters to choose the optimal region –

- Pricing
- User/Customer Location
- Latency
- Service Availability

# AWS S3 Features

- Storage Class
- Bucket Policies
- AWS Identity and Access Management
- Access Control Lists
- Versioning
- Operations

# AWS S3 Features

- **Storage Class**
- Amazon S3 offers a range of storage classes designed for different use cases.
- These include Amazon S3 STANDARD for general-purpose storage of frequently accessed data.
- Amazon S3 STANDARD\_IA for long-lived, but less frequently accessed data,
- And GLACIER for long-term archive.

# AWS S3 Features

## Bucket Policies

- Bucket policies provide centralized access control to buckets and objects based on a variety of conditions,
- Including Amazon S3 operations, requesters, resources, and aspects of the request (e.g., IP address). The policies are expressed in our access policy language and enable centralized management of permissions.
- The permissions attached to a bucket apply to all of the objects in that bucket.

# AWS S3 Features

## Object Versioning

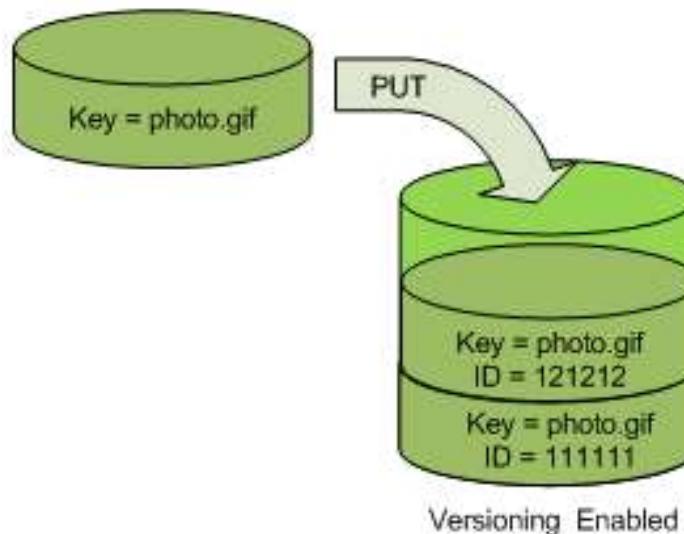
Use versioning to keep multiple versions of an object in one bucket. For example, you could store my-image.jpg(version 1111111) and my-image.jpg(version 2222222) in a single bucket. Versioning protect you from the consequences of unintended overwrites and deletions. You can also use versioning to archive objects so you have access to previous versions.

- You must explicitly enable versioning on your bucket. By default, versioning is disabled. Regardless of whether you have enabled versioning, each object in your bucket has a version ID.
- If you have not enabled versioning, Amazon S3 sets the value of the version ID to null. If you have enabled versioning, Amazon S3 assigns a unique version ID value for the object. When you enable versioning on a bucket, objects already stored in the bucket are unchanged. The version IDs (null), contents, and permissions remain the same.

# AWS S3 Features

## Object Versioning

You PUT an object in a versioning-enabled bucket, the noncurrent version is not overwritten. The following figure shows that when a new version of photo.gif is PUT into a bucket that already contains an object with the same name, the original object (ID = 111111) remains in the bucket, Amazon S3 generates a new version ID (121212), and adds the newer version to the bucket

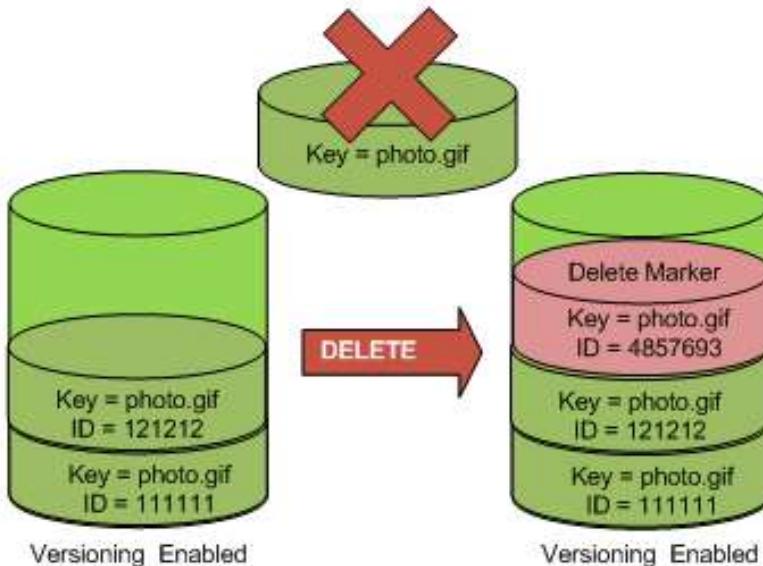


# AWS S3 Features

## Object Versioning

This functionality prevents you from accidentally overwriting or deleting objects and affords you the opportunity to retrieve a previous version of an object.

When you **DELETE** an object, all versions remain in the bucket and Amazon S3 inserts a delete marker, as shown in the following figure.



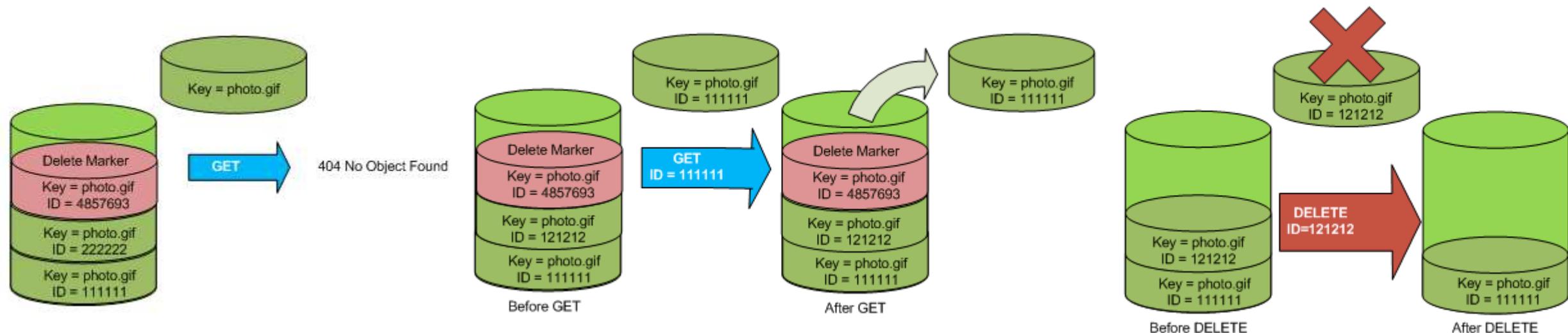
# AWS S3 Features

## Object Versioning

The delete marker becomes the current version of the object. By default, GET requests retrieve the most recently stored version. Performing a simple GET Object request when the current version is a delete marker returns a 404 Not Found error, as shown in the following figure.

You can, however, GET a noncurrent version of an object by specifying its version ID. In the following figure, we GET a specific object version, 111111. Amazon S3 returns that object version even though it's not the current version.

You can permanently delete an object by specifying the version you want to delete. Only the owner of an Amazon S3 bucket can permanently delete a version. The following figure shows how DELETE versionId permanently deletes an object from a bucket and that Amazon S3 doesn't insert a delete marker.



# AWS S3 Features

## Operations

### Common Operations

- Create a Bucket – Create and name your own bucket in which to store your objects.
- Write an Object – Store data by creating or overwriting an object. When you write an object, you specify a unique key in the namespace of your bucket. This is also a good time to specify any access control you want on the object.
- Read an Object – Read data back. You can download the data via HTTP or BitTorrent.
- Deleting an Object – Delete some of your data.
- Listing Keys – List the keys contained in one of your buckets. You can filter the key list based on a prefix.

# Lets do a small Project: AWS Use Case

## Hosting a Static website on AWS S3



# Project Statement

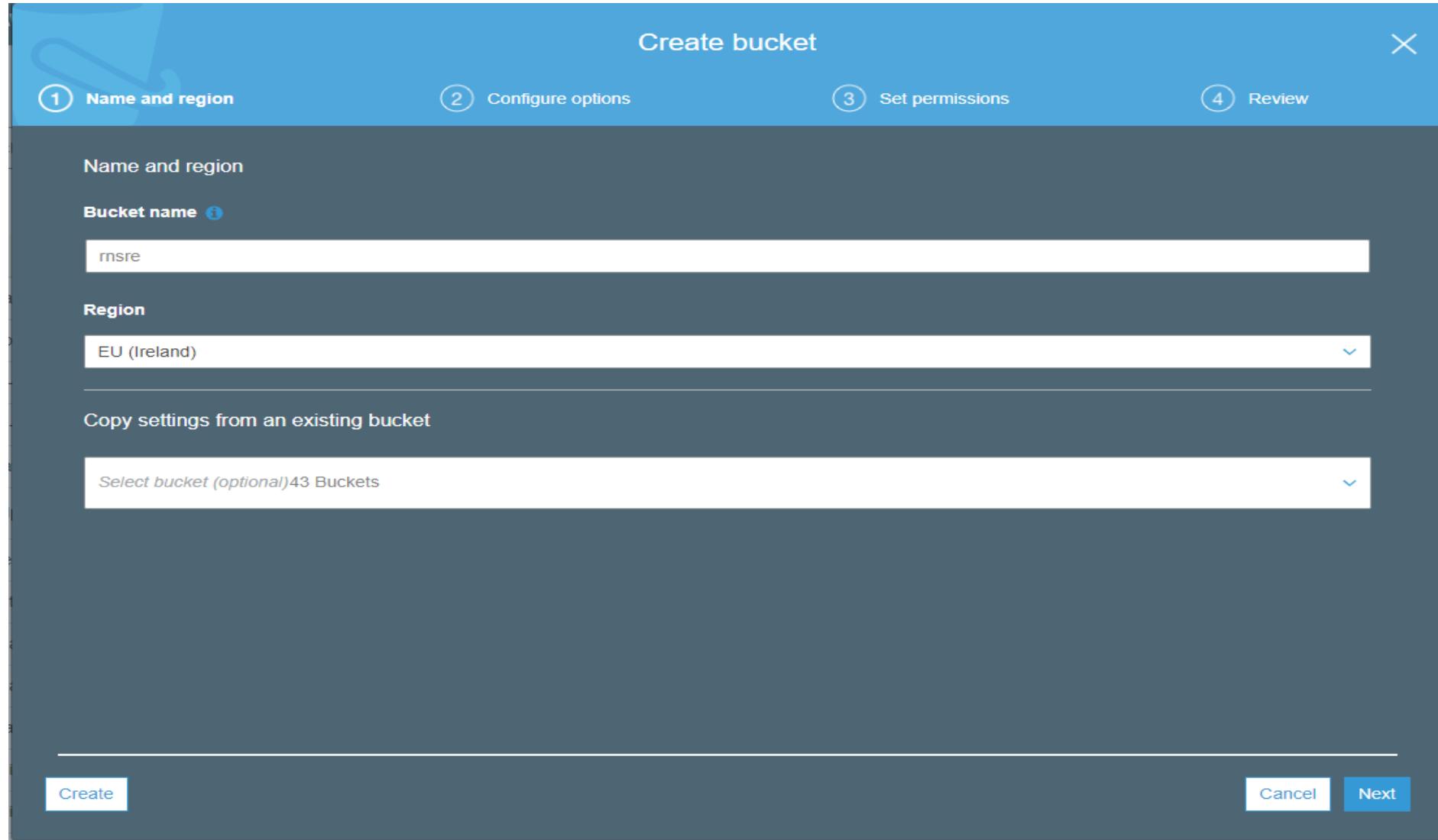
## Project Statement – Hosting a Static Website on Amazon S3

- Let's first understand: What is a static website?
- In short, it's a website comprised of only HTML, CSS, and/or JavaScript. That means server-side scripts aren't supported, so if you want to host a Rails or PHP app, you'll need to look elsewhere.
- For simpler purposes, welcome to the wonderful world of hosting websites on AWS S3!



## Step 1: Create a bucket

- To create a bucket, navigate to S3 in the AWS Management Console and hit Create Bucket. You'll be prompted to enter a name and a region.



## Step 2: Bucket Properties

Create bucket X

① Name and region ② Configure options ③ Set permissions ④ Review

**Versioning**  
 Keep all versions of an object in the same bucket. [Learn more](#)

**Server access logging**  
 Log requests for access to your bucket. [Learn more](#)

**Tags**  
You can use tags to track project costs. [Learn more](#)

Key  Value   
+ Add another

**Object-level logging**  
 Record object-level API activity using AWS CloudTrail for an additional cost. See [CloudTrail pricing](#) or [learn more](#)

**Default encryption**  
 Automatically encrypt objects when they are stored in S3. [Learn more](#)

► Advanced settings

**Management**

**CloudWatch request metrics**  
 Monitor requests in your bucket for an additional cost. See [CloudWatch pricing](#) or [learn more](#)

[Previous](#) [Next](#)

## Step 3: Permissions

Create bucket X

1 Name and region 2 Configure options 3 Set permissions 4 Review

Note: You can grant access to specific users after you create the bucket.

**Public access settings for this bucket**

Use the Amazon S3 block public access settings to enforce that buckets don't allow public access to data. You can also configure the Amazon S3 block public access settings at the account level. [Learn more](#)

**Manage public access control lists (ACLs) for this bucket** [i](#)

Block new public ACLs and uploading public objects (Recommended) [i](#)

Remove public access granted through public ACLs (Recommended) [i](#)

**Manage public bucket policies for this bucket** [i](#)

Block new public bucket policies (Recommended) [i](#)

Block public and cross-account access if bucket has public policies (Recommended) [i](#)

**Manage system permissions**

Do not grant Amazon S3 Log Delivery group write access to this bucket ▼

[Previous](#) [Next](#)

# Step 1: Create a bucket

Create bucket X

1 Name and region 2 Configure options 3 Set permissions 4 Review

**Bucket name** rnsre **Region** EU (Ireland)

**Options** Edit

Versioning	Disabled
Server access logging	Disabled
Tagging	0 Tags
Object-level logging	Disabled
Default encryption	None
CloudWatch request metrics	Disabled
Object lock	Disabled

**Permissions** Edit

Block new public ACLs and uploading public objects	True
Remove public access granted through public ACLs	True
Block new public bucket policies	True
Block public and cross-account access if bucket has public policies	True
System permissions	Disabled

Previous Create bucket

## Find your bucket by searching.

S3 buckets

[Discover the new console](#) [Quick tips](#)

<input type="text" value="reyaz"/> <span>All access types</span>				
<a href="#">Create bucket</a> <a href="#">Edit public access settings</a> <a href="#">Empty</a> <a href="#">Delete</a>		1 Buckets	1 Regions	
<input type="checkbox"/> Bucket name	Access	Region	Date created	
<input type="checkbox"/> reyazrns	Bucket and objects not public	EU (Ireland)	Dec 14, 2018 12:04:32 PM GMT+0530	

Find your bucket and click on it and you will see the below screen

Amazon S3 > reyazrms

**Overview** **Properties** **Permissions** **Management**

---

**Versioning**  
Keep multiple versions of an object in the same bucket.  
[Learn more](#)  
 Disabled

**Server access logging**  
Set up access log records that provide details about access requests.  
[Learn more](#)  
 Disabled

**Static website hosting**  
Host a static website, which does not require server-side technologies.  
[Learn more](#)  
 Disabled

**Object-level logging**  
Record object-level API activity using the CloudTrail data events feature (additional cost).  
[Learn more](#)  
 Disabled

**Default encryption**  
Automatically encrypt objects when stored in Amazon S3  
[Learn more](#)  
 Disabled

---

**Advanced settings**

**Object lock**  
Prevent objects from being deleted.  
[Learn more](#)  
 Disabled

**Tags**  
Use tags to track your cost against projects or other criteria.  
[Learn more](#)  
 0 Tags

**Transfer acceleration**  
Enable fast, easy and secure transfers of files to and from your bucket.  
[Learn more](#)  
 Suspended

**Events**  
Receive notifications when specific events occur in your bucket.  
[Learn more](#)  
 0 Active notifications

**Requester pays**  
The requester (instead of the bucket owner) will pay for requests and data transfer.  
[Learn more](#)  
 Disabled

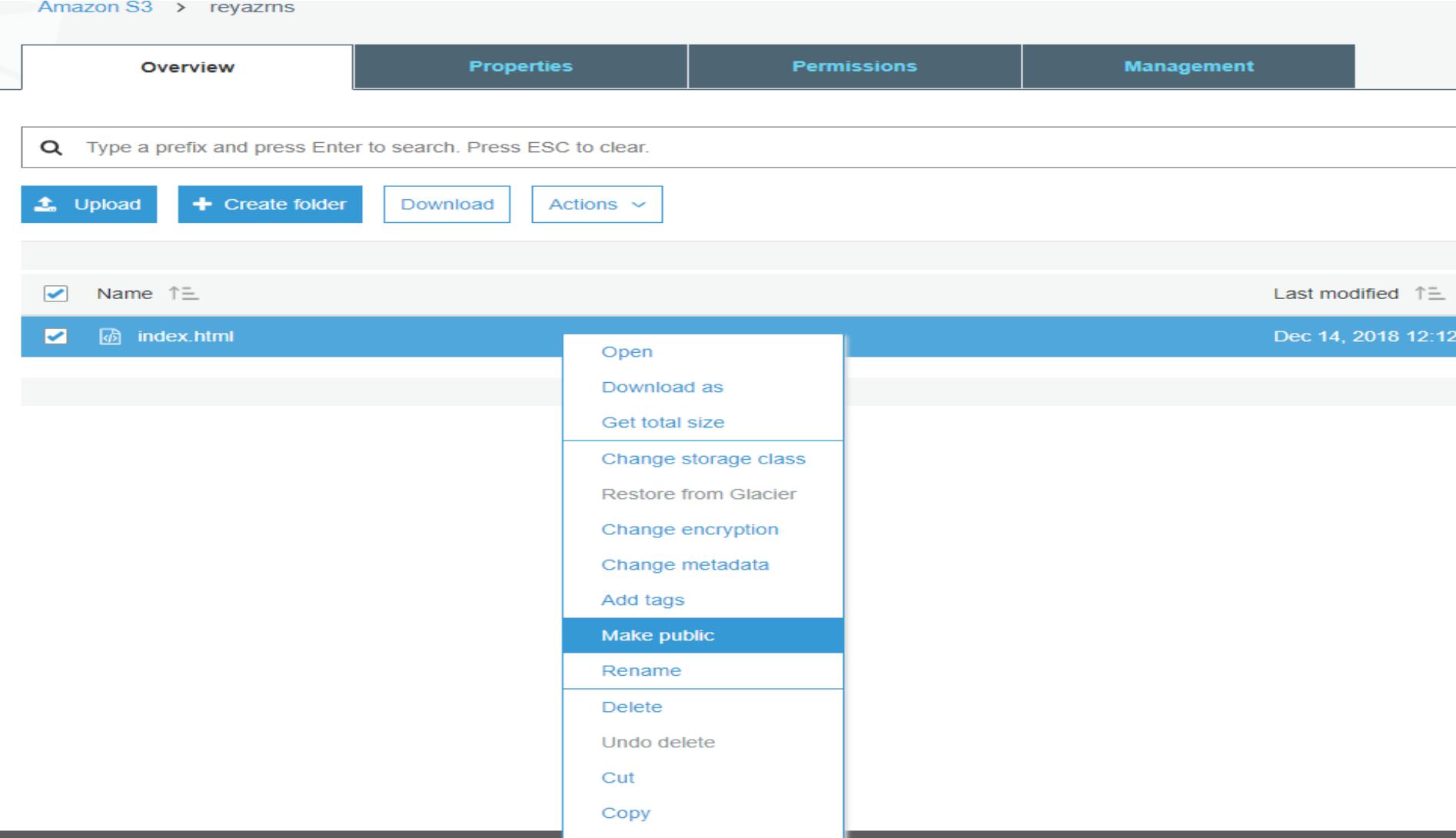
Click on Static website hosting and select “Use this bucket to host a website”

The screenshot shows the Amazon S3 console for the bucket 'reyazrns'. The 'Management' tab is selected. A modal dialog box titled 'Static website hosting' is open, containing the following configuration fields:

- Endpoint: <http://reyazrns.s3-website-eu-west-1.amazonaws.com>
- Use this bucket to host a website [Learn more](#)
- Index document: `index.html`
- Error document: `error.html`
- Redirection rules (optional): An empty text area.
- Redirect requests [Learn more](#)
- Disable website hosting

At the bottom of the dialog are 'Cancel' and 'Save' buttons. In the background, other tabs like 'Overview', 'Properties', and 'Permissions' are visible, along with sections for 'Versioning', 'Server access logging', and 'Object-level logging', all of which are currently disabled.

You don't have index.html. Create a simple html file with name index.html and upload to the bucket and Make public.  
Access the bucket link from the properties



The screenshot shows the Amazon S3 console interface. The top navigation bar includes 'Amazon S3' and the bucket name 'reyazrns'. Below the navigation is a horizontal menu bar with four tabs: 'Overview' (light gray), 'Properties' (dark blue, selected), 'Permissions' (light gray), and 'Management' (light gray). A search bar below the menu bar contains the placeholder text 'Type a prefix and press Enter to search. Press ESC to clear.' Under the search bar are four buttons: 'Upload', 'Create folder', 'Download', and 'Actions ▾'. The main content area displays a list of files in the bucket. The first file listed is 'index.html', which is selected, indicated by a blue background and a checked checkbox in the 'Name' column. To the right of the file list are two columns: 'Last modified' and 'Dec 14, 2018 12:12:1'. A context menu is open over the 'index.html' file, showing the following options: 'Open', 'Download as', 'Get total size', 'Change storage class', 'Restore from Glacier', 'Change encryption', 'Change metadata', 'Add tags', 'Make public' (which is highlighted in blue), 'Rename', 'Delete', 'Undo delete', 'Cut', 'Copy', and a separator line followed by a dash. The background of the S3 interface is white, and the overall layout is clean and modern.



<https://s3-eu-west-1.amazonaws.com/reyazrns/index.html>

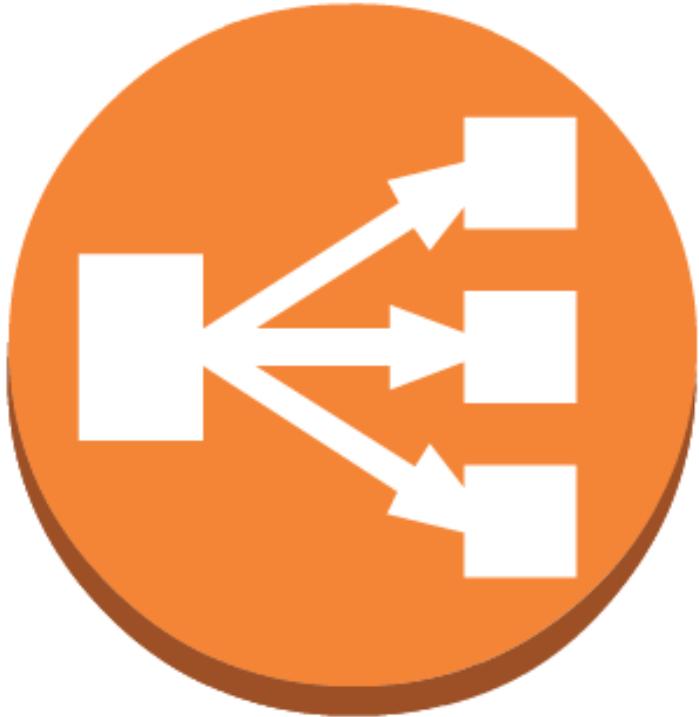
# My first S3 website

I can't believe it was that easy!

**Congratulations!** You have just hosted a html website in AWS using S3.



# Elastic Load Balancer



**AWS Elastic Load Balancer**

By  
Reyaz Shaik

# What is ELB?

- Amazon ELB allows you to make your applications highly available by using health checks and distributing traffic across a number of instances

# Example

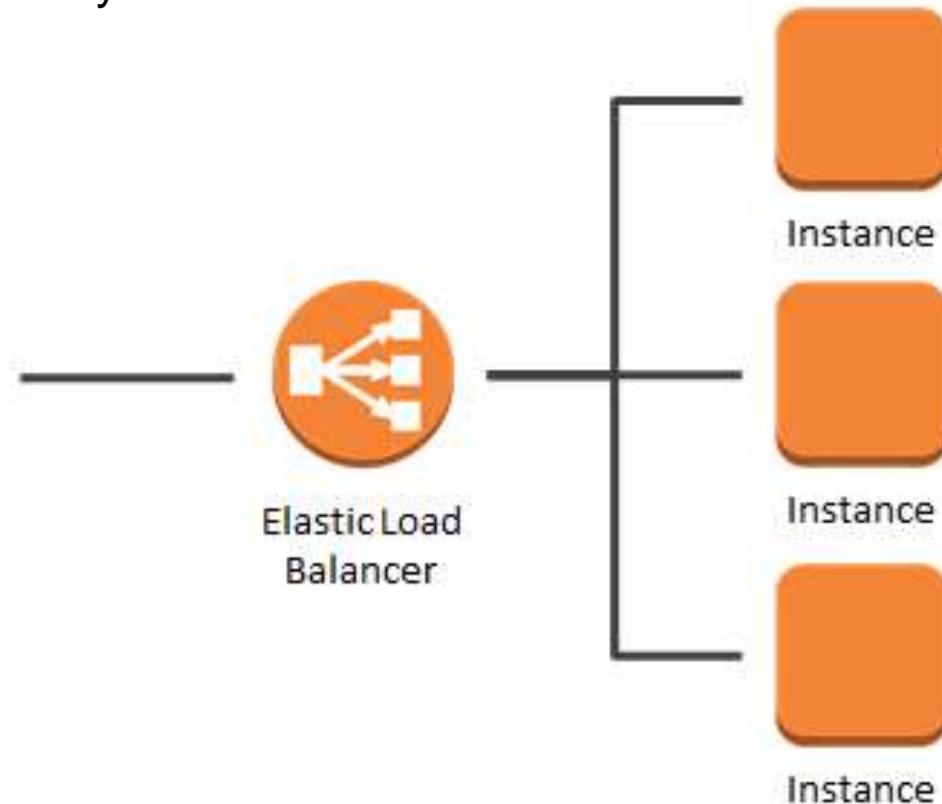
- Consider you have a blog which is running on a single t2.micro EC2 instance.
- Now you publish an article, it goes viral and your site gets hundreds of thousands of requests. Since you are using a single t2-micro, your website will probably crash.
- So, what can you do to avoid this?

# Example Cont..

- You may decide to launch a larger instance like an m5-large in place of t2-micro. This is called vertical scaling when you replace an instance with a more powerful instance.
- But vertical scaling isn't always economical.
- Another approach can be to use a bunch of smaller instances like t2-micros and distribute the website traffic between them. And Elastic Load Balancer allows you to do just that.
- It distributes incoming application or network traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses, in multiple Availability Zones.

# ELB

It uses health checks to detect which instances are healthy and directs traffic only across those instances.



# Types of Elastic Load Balancers

There are three types of load balancers available. You can use the one that best fits your use case

- 1. Classic Load Balancer (CLB)
- 2. Application Load Balancer (ALB)
- 3. Network Load Balancer (NLB)

# Types of Elastic Load Balancers

There are three types of load balancers available. You can use the one that best fits your use case

## 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 and containers.

[Learn more >](#)

**Network Load Balancer**



**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-Classic network.

[Learn more >](#)

# Classic Load Balancer(CLB)

- Classic Load Balancer provides basic load balancing across multiple Amazon EC2 instances and operates at both the request level and connection level.
- Classic Load Balancer is intended for applications that were built within the EC2-Classic network. We recommend Application Load Balancer for Layer 7 and Network Load Balancer for Layer 4 when using Virtual Private Cloud (VPC).

# Application Load Balancer(ALB)

- This load balancer is specially designed for web applications with HTTP and HTTPS traffic.
- There is a networking model called the OSI Model (Open Systems Interconnection) that is used to explain how computer networks work. This model has 7 layers and the top layer is the Application Layer.
- This load balancer works at this Application Layer, hence the name.
- Application Load Balancer simplifies and improves the security of your application, by ensuring that the latest SSL/TLS ciphers and protocols are used at all times.
- It also provides advanced routing features such as **host-based** and **path-based routing** and also works with containers and microservices.

# Host Based and Path Based Routing

## Host-based Routing

- Suppose you have two websites **medium.com** and **admin.medium.com**. Each website is hosted on two EC2 instances for high availability and you want to distribute the incoming web traffic between them.
- If you were using the CLB you would have to create two load balancers, one for each website.
- But you can do the same thing using a single ALB!
- Hence you will be saving money as you will only be paying for a single ALB instead of two CLBs.

## Path-based Routing

- Suppose the website of your company is **payzello.com** and the company's blog is hosted on **payzello.com/blog**. The operations team has decided to host the main website and the blog on different instances.
- Using ALB you can route traffic based on the path of the requested URL. So again a single ALB is enough to handle this for you.

# Network Load Balancer(NLB)

- This load balancer operates at the Network layer of the OSI model, hence the name.
- Suppose your company's website is running on four m4-xlarge instances and you are using an ALB to distribute the traffic among them.
- Now your company launched a new product today which got viral and your website starts to get millions of requests per second.
- In this case, the ALB may not be able to handle the sudden spike in traffic.
- This is where the NLB really shines. It has the capability to handle a sudden spike in traffic since it works at the connection level.
- It also provides support for static IPs.

I hope you have got a rough idea about load balancers.

Now, enough talking, let's go practical.



**Launch 2 EC2 Instances and make it WebServers(Apache)**

**Create 2 Targets groups Main and Blog**

**Create Application Load Balancer**

# Sample Project

- We will handle a case of path-based routing. We will be handling two paths here, “/” and “/blog”.
- We will launch two instances, one for handling each path. Let’s get started!

# Launching EC2 Instances

- Launch two EC2 instances
- When launching, give a Name tag to your instances.
- For the first instance, give a tag with **Name** as key and **Main** as the value.
- For the second instance, give a tag with **Name** as key and **Blog** as the value. This will help us in distinguishing between them.
- After launching the two instances, your dashboard should look like this.



The screenshot shows the AWS EC2 Instances dashboard. At the top, there is a search bar labeled "Filter by tags and attributes or search by keyword" and a pagination indicator "1 to 2 of 2". The table has the following columns: Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). There are two rows of data:

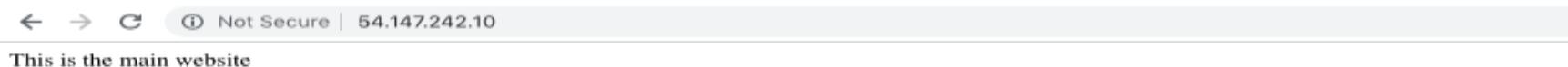
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
Blog	i-04c4fcc4f4f73a1d6	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-52-91-104-140.co...
Main	i-0d90c9c4fe3dffbb8	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-147-242-10.co...

# Install Apache server on instances

Now SSH into the first instance (with name Main) and run the following commands to install and start the apache server.

```
sudo yum update -y
sudo yum install -y httpd
sudo service httpd start
sudo chkconfig httpd on
cd /var/www/html
sudo su
echo "This is the Main Website" > index.html
```

Now paste the IP address of the instance in the browser and hit Enter.  
You should see something as shown in the picture below.



Now SSH into the second instance (with name Blog) and run the same commands except the last command. Instead, run the following command.

```
echo "This is the Blog Website" > blog
```

Paste the IP address of this instance with **/blog** as the suffix in the browser and hit Enter. You should see something like below.



# Create Target Groups

- A target group allows you to tell the load balancer which protocol and port will receive the traffic on the registered instances.
- 1. In the left navigation bar, scroll down and click on **Target Groups**. Now click on **Create target group** at the top.

**Create target group**

Your load balancer routes requests to the targets in a target group using the target group settings that you specify, and performs health checks on the targets using the health check settings that you specify.

Target group name

Target type  Instance  IP

Protocol

Port

VPC

Health check settings

Protocol

Path

[Advanced health check settings](#)

[Cancel](#) [Create](#)

- Give your target group a name **Main** and click **Create** button..

The screenshot shows the AWS Lambda Target Groups console. At the top, there is a table with columns: Name, Port, Protocol, Target type, Load Balancer, VPC ID, and Monitoring. A single row is selected, showing 'Main' as the name, port 80, protocol HTTP, target type instance, and VPC ID 'vpc'. Below this, the main content area is titled 'Target group: Main'. It has tabs for Description, Targets (which is selected and highlighted in orange), Health checks, Monitoring, and Tags. An 'Edit' button is visible. The 'Registered targets' section has a table with columns: Instance ID, Name, Port, Availability Zone, and Status. A message states 'There are no targets registered to this target group'. The 'Availability Zones' section has a table with columns: Availability Zone, Target count, and Healthy?. A message states 'There are no targets registered to this target group'.

Name	Port	Protocol	Target type	Load Balancer	VPC ID	Monitoring
Main	80	HTTP	instance		vpc	

Target group: Main

Description Targets Health checks Monitoring Tags

Edit

Registered targets

Instance ID	Name	Port	Availability Zone	Status
There are no targets registered to this target group				

Availability Zones

Availability Zone	Target count	Healthy?
There are no targets registered to this target group		

- Now, navigate to the **Targets** tab at the bottom, click on **Edit**, select the **Main** instance, click **Add to registered** and click **Save**.

**Register and deregister targets**

**Registered targets**

To deregister instances, select one or more registered instances and then click Remove.

<input type="checkbox"/>	Instance	Name	Port	State	Security groups	Zone
<input type="checkbox"/>	i-0d90c9c4fe3dffbb8	Main	80	<span style="color: green;">●</span> running	MyWebDMZ	us-east-1b

**Instances**

To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.

on port

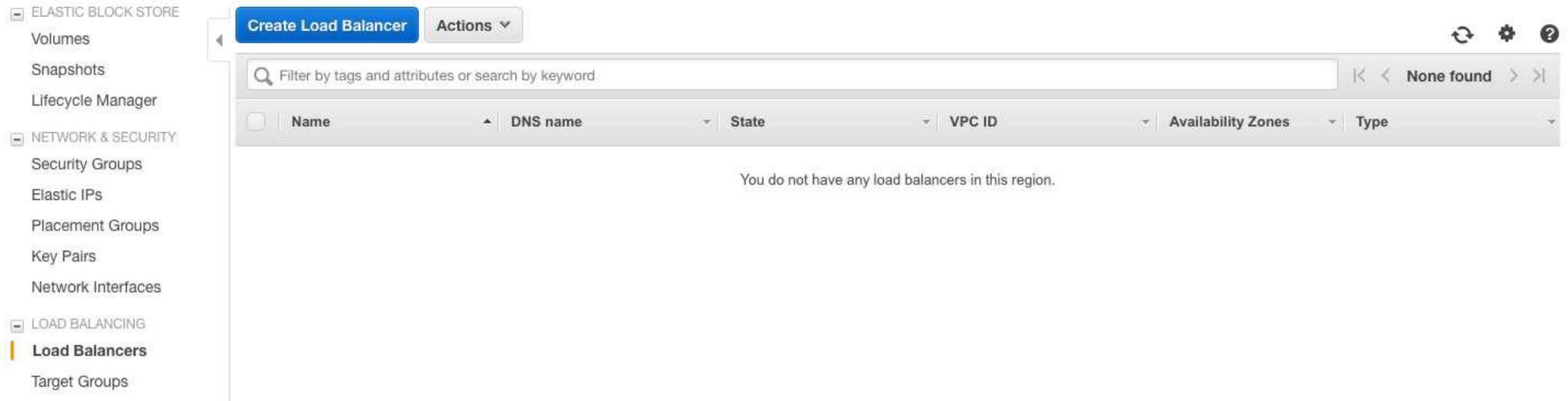
<input type="checkbox"/>	Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
<input type="checkbox"/>	i-04c4fcc4f4f73a1d6	Blog	<span style="color: green;">●</span> running	MyWebDMZ	us-east-1b	subnet-263ffa09	172.31.80.0/20
<input checked="" type="checkbox"/>	i-0d90c9c4fe3dffbb8	Main	<span style="color: green;">●</span> running	MyWebDMZ	us-east-1b	subnet-263ffa09	172.31.80.0/20

Create another target group with the name **Blog** and add the **Blog instance** to it as we did above.

# Creating an Application Load Balancer

## Creating and configuring the Application Load Balancer

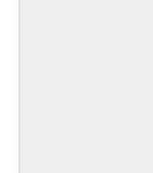
- Now, in the left navigation scroll down and click on **Load Balancers**. Click on the **Create Load Balancer** button at the top.



- 1. Choose the Application 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	Network Load Balancer	Classic Load Balancer
 <a href="#">Create</a>  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 and containers.  <a href="#">Learn more &gt;</a>	 <a href="#">Create</a>  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.  <a href="#">Learn more &gt;</a>	 <b>PREVIOUS GENERATION</b> for HTTP, HTTPS, and TCP  <a href="#">Create</a>  Choose a Classic Load Balancer when you have an existing application running in the EC2-Classic network.  <a href="#">Learn more &gt;</a>

- 2. Give a name to your load balancer and select at least two availability zones for high availability and click on the **Next: Configure Security Settings** button.

# Configure Security Settings

1. Configure Load Balancer   2. Configure Security Settings   3. Configure Security Groups   4. Configure Routing   5. Register Targets   6. Review

## Step 1: Configure Load Balancer

Name  ⓘ

Scheme  internet-facing  internal ⓘ

IP address type  ⓘ

## Listeners

A listener is a process that checks for connection requests, using the protocol and port that you configured.

Load Balancer Protocol	Load Balancer Port
HTTP	80

[Add listener](#) ×

## Availability Zones

Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to increase the availability of your load balancer.

VPC  ⓘ (172.31.0.0/16) (default)

Availability Zone	Subnet ID	Subnet IPv4 CIDR	Name
<input checked="" type="checkbox"/> us-east-1a	subnet-	172.31.0.0/20	

[Cancel](#) [Next: Configure Security Settings](#)

- 3. You may see a warning message but that is because we are only listening for HTTP traffic which is fine for our case, so click on the **Next: Configure Security Groups** button again.
- 4. Here select the existing group option and select the same security group that you assigned to the instances you launched. Once done click on **Next: Configure Routing** button.
- 5. In Target groups, select the existing target group. In the name select **Main** and click **Next**.

## Step 4: Configure Routing

Your load balancer routes requests to the targets in this target group using the protocol and port that you specify, and performs health checks on the targets using these health check settings. Note that each target group can be associated with only one load balancer.

### Target group

Target group ⓘ Existing target group

Name ⓘ Main

Target type  Instance  IP

Protocol ⓘ HTTP

Port ⓘ 80

### Health checks

Protocol ⓘ HTTP

Path ⓘ /

▶ Advanced health check settings

[Cancel](#) [Previous](#) [Next: Register Targets](#)

6. Click Next again, review and click **Create**.

Name	DNS name	State	VPC ID	Availability Zones	Type
my-load-balancer	my-load-balancer-21507358...	provisioning	vpc-	us-east-1f, us-east-1b, ...	application

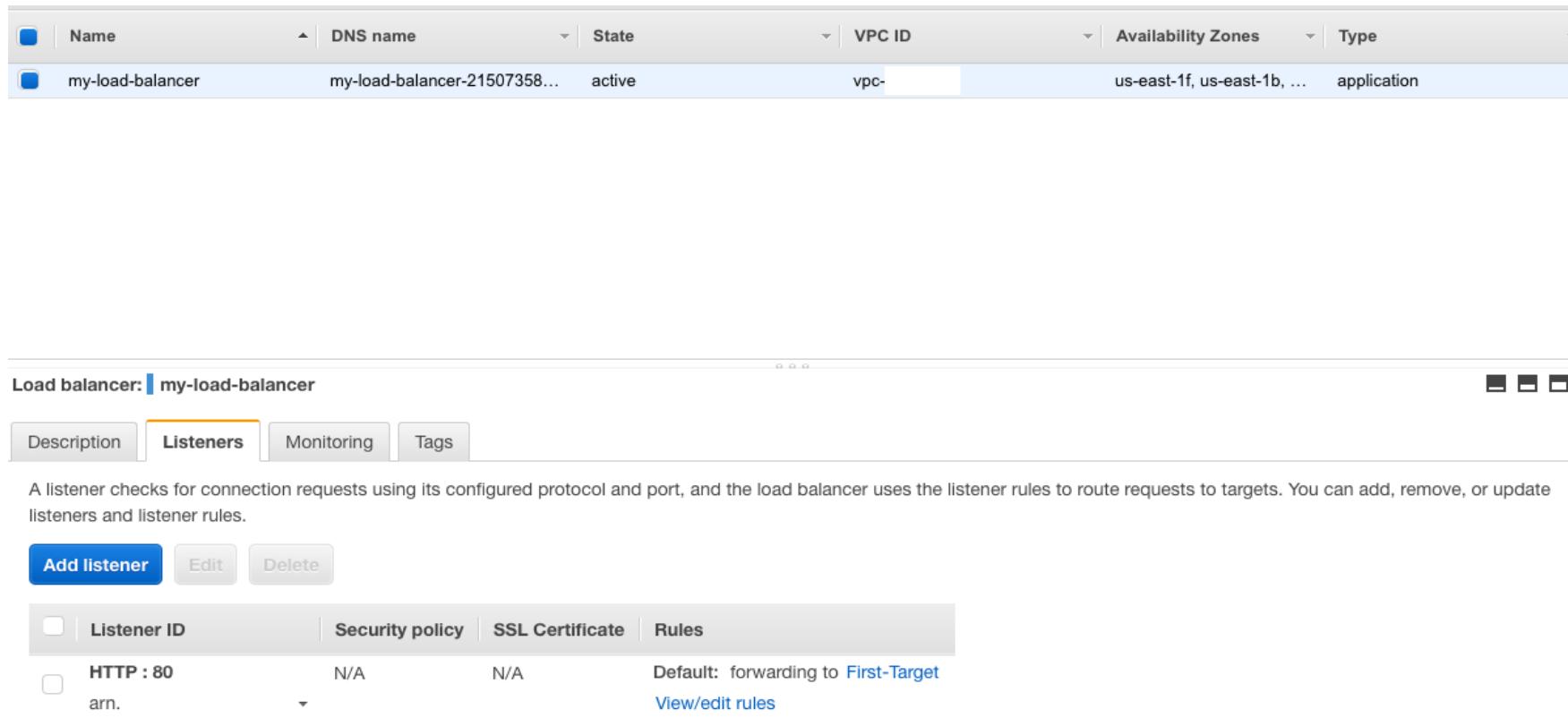
Load balancer: my-load-balancer

Description    Listeners    Monitoring    Tags

### Basic Configuration

Name:	my-load-balancer	Creation time:	November 19, 2018 at 11:07:43 PM UTC+5:30
ARN:	arn:aws:elasticloadbalancing:us-east-1:ba	Hosted zone:	Z35SXDOTRQ7X7K
DNS name:	my-load-balancer-215073587.us-east-1.elb.amazonaws.com	State:	provisioning
Scheme:	internet-facing	VPC:	vpc-
Type:	application	IP address type:	ipv4
		AWS WAF Web ACL:	

- Congratulations, you have just created an Application Load Balancer!
- But we still have to configure our Blog instance so let's continue. Take a note of the **DNS name** of the Load balancer here. We will need it at the end.
- 7. Select the **Listeners** tab and Click on **View/edit rules**



The screenshot shows the AWS Application Load Balancer (ALB) configuration page. At the top, there is a table with columns: Name, DNS name, State, VPC ID, Availability Zones, and Type. One row is visible for 'my-load-balancer'.

Name	DNS name	State	VPC ID	Availability Zones	Type
my-load-balancer	my-load-balancer-21507358...	active	vpc-	us-east-1f, us-east-1b, ...	application

Below the table, the page title is 'Load balancer: my-load-balancer'. The navigation tabs are 'Description', 'Listeners' (which is selected and highlighted in orange), 'Monitoring', and 'Tags'. A descriptive text explains that a listener checks for connection requests using its configured protocol and port, and the load balancer uses the listener rules to route requests to targets. It also mentions that listeners and listener rules can be added, removed, or updated.

At the bottom of the page, there are buttons for 'Add listener', 'Edit', and 'Delete'. A table lists the listeners, showing columns: Listener ID, Security policy, SSL Certificate, and Rules. One listener is listed: 'HTTP : 80' with 'arn.' as the Listener ID, 'N/A' for Security policy and SSL Certificate, and 'Default: forwarding to First-Target' under Rules. A link 'View/edit rules' is located next to the 'Default' text.

- 8. Click the + sign at the top to add a rule. In Add Condition select “Path is” and type **/blog**.
- 9. Then in Add Action select **Forward to** and select **Blog** and then click **Save**.

Click a location for your new rule. Each rule must include one action of type forward, redirect, fixed response.

my-load-balancer | HTTP:80 (2 rules)

Insert Rule

RULE ID	IF (all match)	THEN
1 A rule ID (ARN) is generated when you save your rule.	<input checked="" type="checkbox"/> Path is /blog <input type="checkbox"/> + Add condition	<input checked="" type="checkbox"/> 1. Forward to Blog <input type="checkbox"/> + Add action

last: HTTP 80: default action  
 This rule cannot be moved or deleted

IF  
 Requests otherwise not routed

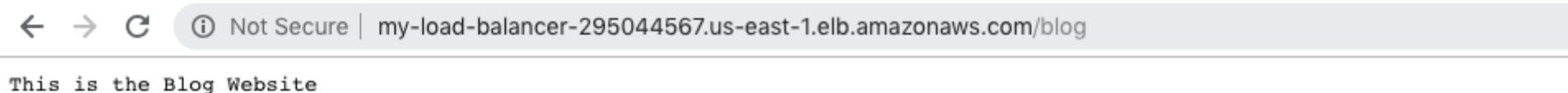
THEN  
 Forward to [Main](#)

- Now, we can use the **DNS name** of our load balancer to visit the two different paths and see the results.

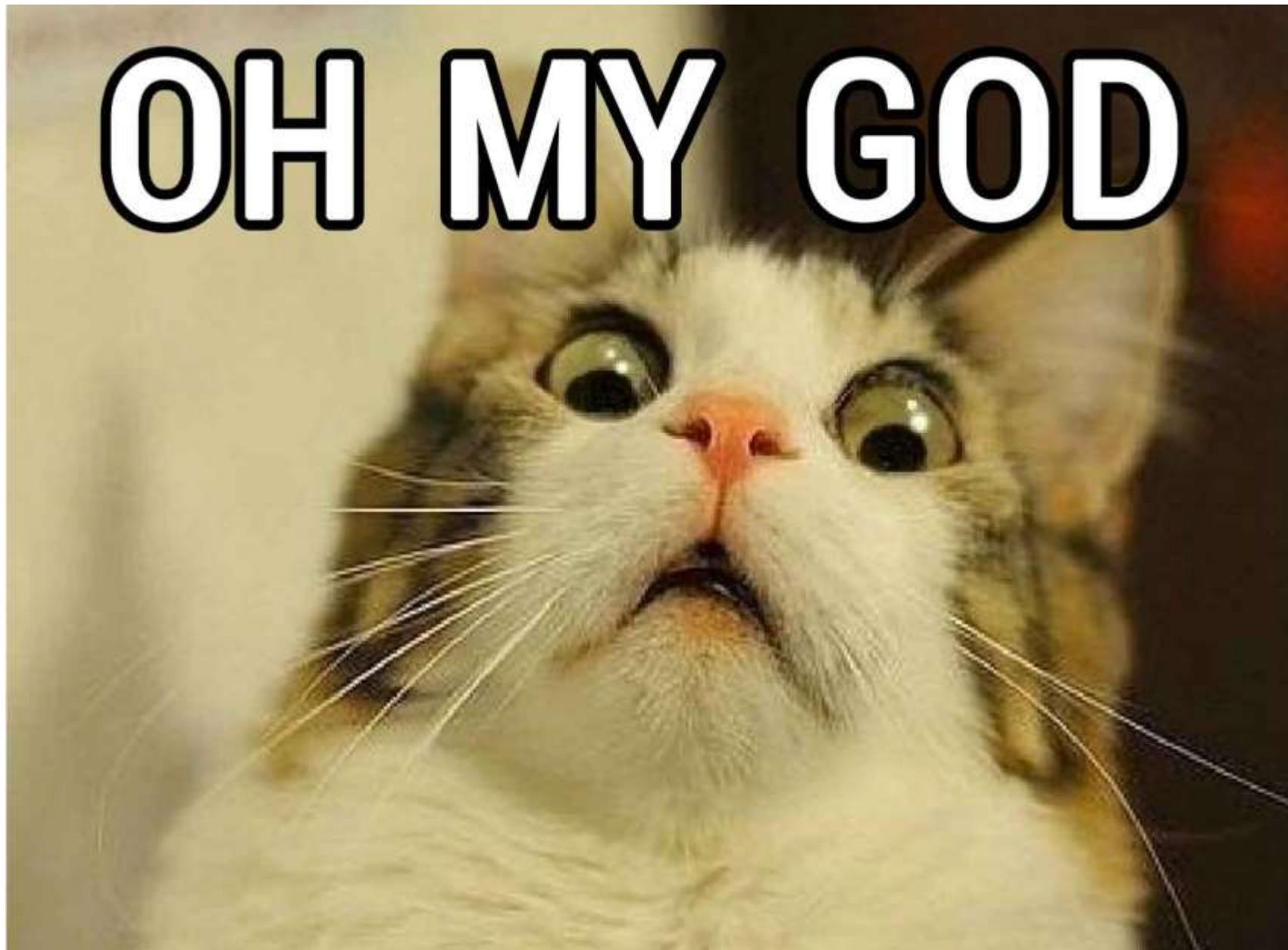
## For /



## For /blog



OH MY GOD



# Security



# Security and Clouds

- Security is a core requirement for any application whether it is hosted on an on-premise data center or a cloud such as AWS.
- It is a fundamental service that protects your applications and data from a variety of cyber-attacks, security breaches, accidental or deliberate data deletions, theft, and much more.

# Is AWS really secure?

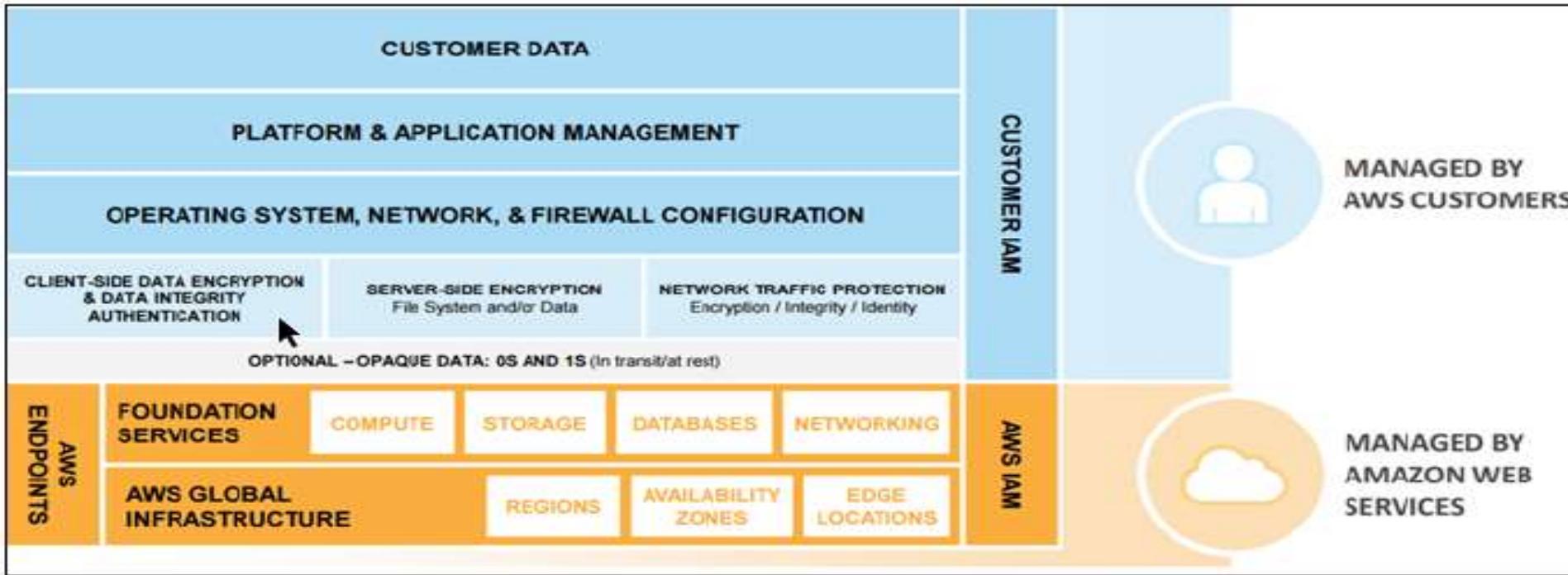
Different layers of security that AWS uses

- 1. Physical data center security:
  - The AWS infrastructure is designed and managed according to security best practices and compliance guides.
  - The data centers themselves are housed at non-disclosed locations and entry to them is strictly controlled, managed, logged, and audited on a regular basis.
- 2. Virtualization and OS security:
  - AWS regularly patches and updates virtualization and operating systems against a variety of attacks such as DDoS, and so on.

# Is AWS really secure?

- **3. Regulatory compliances:**
  - The AWS infrastructure is certified against security and data protection in accordance with various industry and government requirements. Here are a few compliances that AWS is certified against:
  - SOC 1 (formerly SAS 70 Type II),
  - SOC 2, and SOC 3
  - FISMA, DIACAP, and FedRAMP
  - ISO 27001
  - HIPAA
- To read the complete list, visit the AWSrisk and compliance whitepaper at <http://aws.amazon.com/security/>.

# Shared Model for AWS's Services



AWS provides a few services and products that are specifically designed to help you secure your infrastructure on the cloud, such as **IAM**, **AWS Multi-Factor Authentication (AWS MFA)**, **AWS Cloud Trail**, and much more.

# Amazon IAM(Identity & Access Management)



AWS IAM

# Identity and Access Management

- You can use IAM to create users and groups, assigning users specific permissions and policies, and a lot more.
- The best part of all this is that IAM is completely FREE. Yup! Not a penny is required to use it.

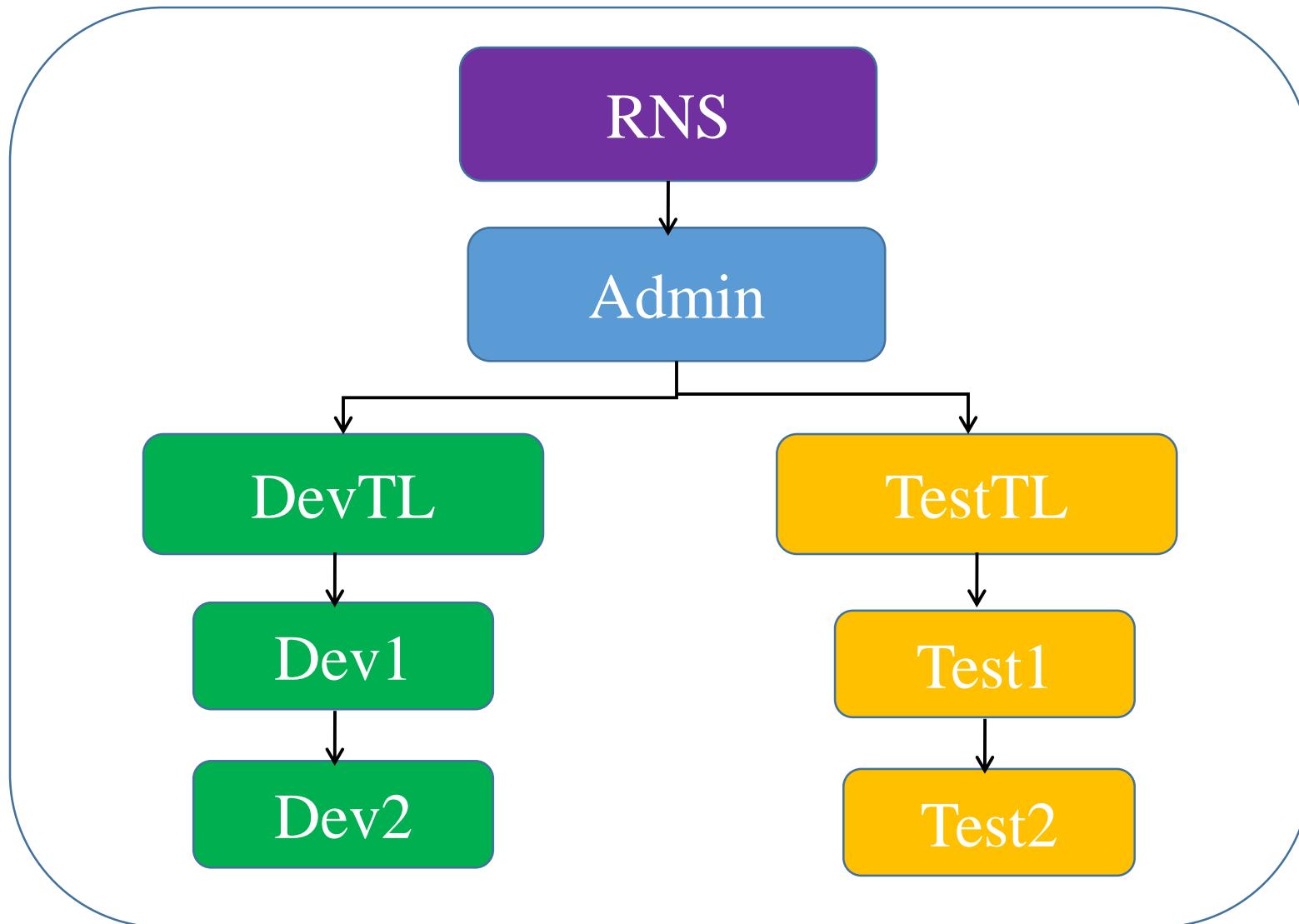
# IAM features

- **Shared access to a single account:**
  - you can create and provide users with shared access to your single account with real ease.
- **Multi-factor authentication:**
  - along with your password, you will also have to provide a secret key/pin from a special hardware device, or even from software apps such as Google Authenticator.

# IAM features

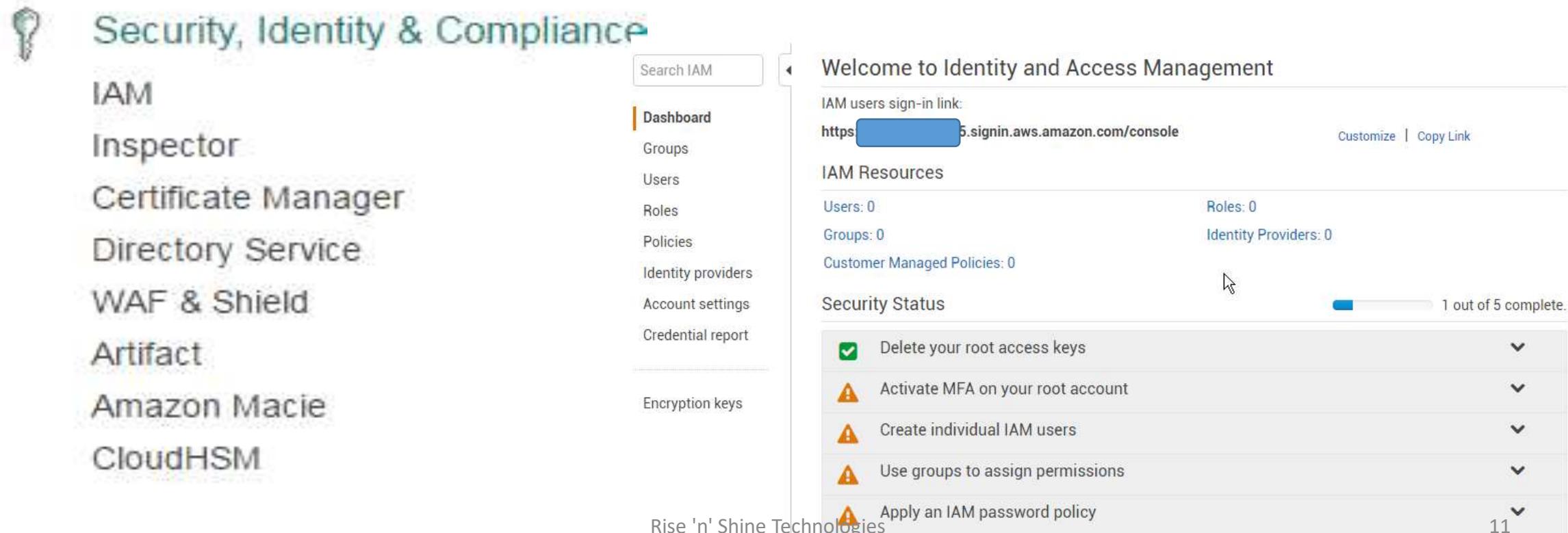
- **Integration with other AWS products:**
  - can be used to provide granular access rights and permissions to each service as required.
- **Identity federation:**
  - IAM can be integrated with an on-premise AD to provide access to your AWS account
- **Global reach:**
  - IAM is the Global, not specific to the Region.
- **Access mechanisms:**
  - IAM can be accessed using a variety of different tools, AWS Management Console, AWS CLI, via SDKs that support different platforms and programming languages such as Java, .NET, Python.

# Business use case scenario



# Getting started with the IAM Console

- To begin with, sign in to the AWS Management Console using <https://console.aws.amazon.com/>



The screenshot shows the AWS IAM console dashboard. On the left, a sidebar lists various services: IAM, Inspector, Certificate Manager, Directory Service, WAF & Shield, Artifact, Amazon Macie, and CloudHSM. The 'IAM' service is selected. The main content area is titled 'Welcome to Identity and Access Management'. It displays the following statistics: IAM users sign-in link (https://[REDACTED].signin.aws.amazon.com/console), IAM Resources (Users: 0, Groups: 0, Policies: 0, Identity providers: 0, Customer Managed Policies: 0), and Security Status. The security status section contains five items, each with a checkbox and a dropdown arrow:

- Delete your root access keys
- ⚠ Activate MFA on your root account
- ⚠ Create individual IAM users
- ⚠ Use groups to assign permissions
- ⚠ Apply an IAM password policy

At the bottom right, a progress bar indicates '1 out of 5 complete.' The footer of the page reads 'Rise 'n' Shine Technologies' and '11'.

# Getting started with the IAM Console

- Delete your root access keys. Now why would you want to do? What are root access keys?
  - Root keys simply consist of an access ID and a secret key
  - Can be used to programmatically access any AWS service.
  - Each user that you create gets its own set of keys.
  - The secret key has to be protected and kept under lock and key at all costs.

# Getting started with the IAM Console

- The IAM URL contains the following format:
  - `https://<AWS_Account_ID>.signin.aws.amazon.com/console/`
- Select the **Customize** option, Provide a suitable alias name for your account.



# Creating Admin User Account for AWS

- Create an Admin User Account instead of using the root user account
- Step 1: Create an user with the following info
  - Name
  - Password
  - Security Key and Access Key ID
  - Attach Existing Policy as ‘AdministratorAccess’

# Creating users and groups

- From the IAM dashboard, select the
  - IAM -> Users -> Add User
  - Create all the Users with the following info
    - UserName
    - Password
    - Security Key and Access Key ID
- Note: Access keys are unique to each user and should not be shared with anyone under any circumstances. Save them in a secure place.

# Groups and Policies

- Group is a collection of IAM users that has a particular set of permissions assigned to it.
  - For example, a set of users who perform admin tasks can be clubbed under a common group called as **administrators**.
- A policy is a document that lists one or more permissions. You can attach policies to virtually anything in AWS, from users and groups to individual AWS resources as well.
- **Create New Group**
  - Group Name
  - Select one or more policies to attach. Each group can have up to 10 policies attached.
    - Ex: AdministratorAccess

# Understanding permissions and policies

- Permissions provide you with access to and control of various AWS resources.
- They are also responsible for controlling actions that you can perform on the resources.
- Permissions can be classified into two main classes
  - **User-based permissions**
  - **Resource-based permissions**

# User-based permissions:

- These permissions are attached to IAM users and allow them to perform some action over an AWS resource.
- User-based permissions can be applied to groups as well.
- Two Categories
  - **inline policies**
    - created and managed completely by you
  - **Managed Policies**
    - created and managed more by AWS itself

# Resource-based permissions:

- User has specific level of access to a particular AWS resource along with what actions they can perform on it.
- These categories of permissions are only inline-based this means that they are completely managed and created by you.

# Understanding permissions and policies

## User Based Permissions

### Admin

All Actions on All Resources

### DevTL

List, Read, Write on EC2

### TestTL

Read on EC2

## Resource Based Permissions

### S3 Bucket

Admin: List, Read, Write

DevTL: Read, Write

TestTL: Read, Write

# Policy – JSON Format

- Let's look at a simple policy for our reference:
- {
- **"Version": "2012-10-17",**
- **"Statement": [**
- {
- **"Effect": "Allow",**
- **"Action": [**
- **"ec2:DescribeInstances",**
- **"ec2:DescribeImages"**
- **],**
- **"Resource": "arn:aws:iam::012345678910:user/admin"**
- }
- ]
- }

# Creating and assigning policies

- IAM → Policies
- filter and list existing policies (*both inline and manage policies*) using the **Filter and Search options**
- **Create Policy**
  - Copy an AWS Managed Policy:
  - Policy Generator:
  - Create your Own Policy:

# Roles and Policies

- Roles are nothing but a group of permissions that grant users access to some particular AWS resources and services.
- Diff:
  - Policies are applied to users and groups that belong to a particular AWS account
  - Roles are applied to users who are generally not a part of your AWS account
  - Use roles to delegate access to users, applications, and services that do not have access to your AWS resources.
  - Use roles to create federated identities where a user from your organization's corporate directory gets access to your AWS resources on a temporary basis.

# AWS Other Services: Identity Provider

- To provide external users access to some resources.
- Facebook or Google credentials to log in to site.
- It can use either SAML 2.0 or OpenID Connect to establish trust between your AWS account and your external source of identity provider.

# AWS Other Services: AWSCloudTrail:

- Administrator, to log and record each and every API call that is made from within your account.
- These logs can contain information such as the API's request and response parameters, who made the API call, the time of the API call, and so on.
- These details are vital and can be used during security audits, compliance tracking, and so on.

# AWS Other Services: AWSConfig

- AWS Config is a fully automated service that enables you to take a complete snapshot of all your AWS resource's configurations for compliance and auditing purposes.
- It can also be used as a change management tool to find out when your AWS resources were created, updated, and destroyed.

# AWS Key Management Service:

- To manage your account's keys more effectively and efficiently.
- It also provides add-on functionality such as centralized key management.
- One click encryption of your data
- Automatic key rotations, and so on so forth.

# IAM - Best Practices

- Get rid of the Root Account, use IAM wherever necessary. Hide away the Root key and avoid using it unless it's the end of the world!
- Create a separate IAM users for your organization, each with their own sets of access and Secret Keys. *DO NOT SHARE YOUR KEYS OR PASSWORDS!* *Sharing such things is never a good idea and can cause serious implications and problems.*
- Create separate administrators for each of the AWS services that you use.
- Use roles and groups to assign individual IAM users permissions. Provide only the required level of access and permissions that the task demands.

# IAM - Best Practices

- Leverage multi-factor authentication (MFA) wherever possible.
- Rotate your passwords and keys on a periodic basis. Create keys only if there is a requirement for it.
- Maintain logs and history of your AWS account and its services. Use AWSCloudTrail for security and compliance auditing.
- Use temporary credentials (IAM Roles) rather than sharing your account details with other users and applications.
- Leverage AWS Key Management Service to encrypt data and your keys wherever necessary.

# What you have on the IAM Dashboard

- Users
- Groups
- Roles
- Policies
- Identity Providers
- Accounts Settings
- Credentials Report
- Encryption keys

# Users

- An AWS Identity and Access Management (IAM) *user* is an entity that you create in AWS to represent the person or application that uses it to interact with AWS. A user in AWS consists of a name and credentials.
- An IAM user with administrator permissions is not the same thing as the AWS account root user
- An Amazon Resource Name (ARN) for the user. You use the ARN when you need to uniquely identify the user across all of AWS.
  - `arn:aws:rds:<region>:<account number>:<resourcetype>:<name>`  
`arn:aws:iam::account-ID-without-hyphens:user/Richard`

# Users and Credentials

You can access AWS in different ways depending on the user credentials:

- Console password: A password that the user can type to sign in to interactive sessions such as the AWS Management Console.
- Access keys: A combination of an access key ID and a secret access key. You can assign two to a user at a time. These can be used to make programmatic calls to AWS. For example, you might use access keys when using the API for code or at a command prompt when using the AWS CLI or the AWS PowerShell tools.
- By default, a brand new IAM user has no permissions to do anything
- Each IAM user is associated with one and only one AWS account
- There's a limit to the number of IAM users you can have in an AWS account.

# Limits

## Default limits for IAM entities:

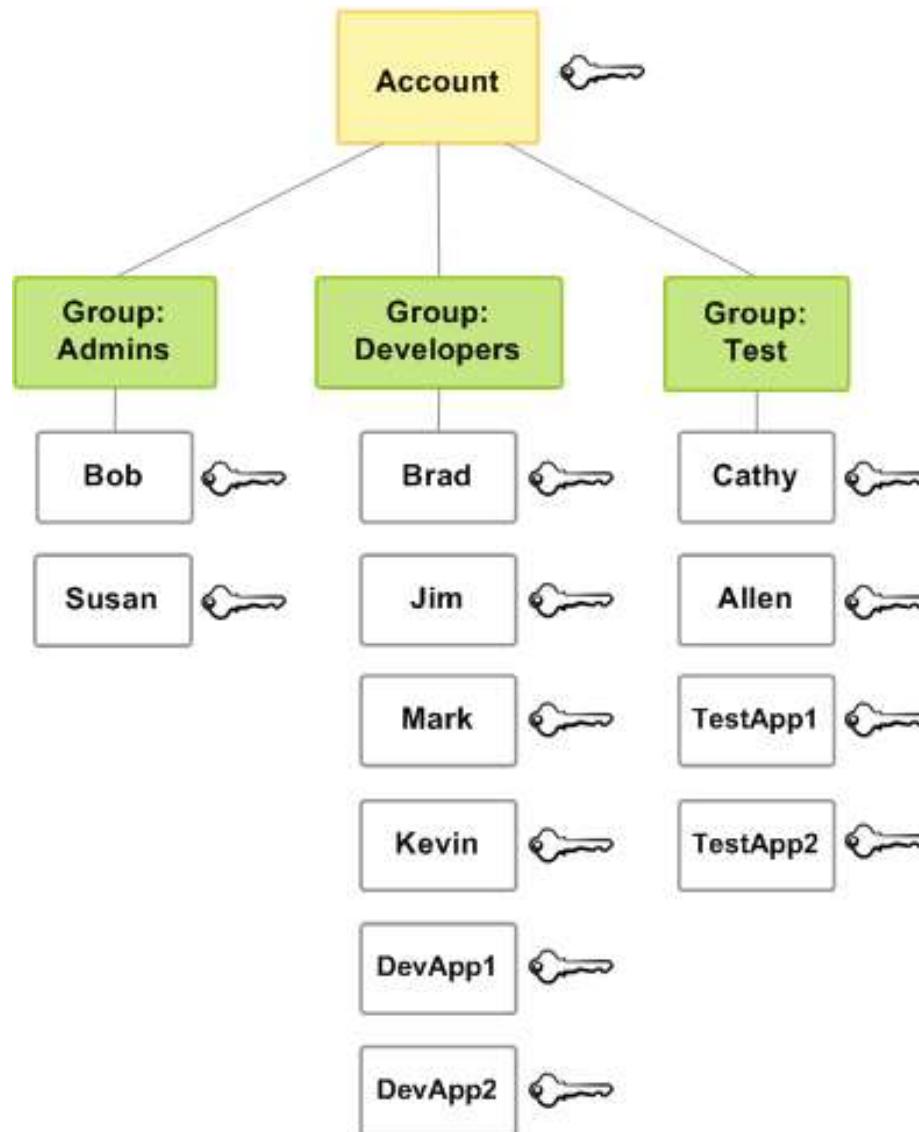
Resource	Default Limit
Customer managed policies in an AWS account	1500
Groups in an AWS account	300
Roles in an AWS account	1000
Managed policies attached to an IAM role	10
Managed policies attached to an IAM user	10
Virtual MFA devices (assigned or unassigned) in an AWS account	Equal to the user quota for the account
Instance profiles in an AWS account	1000
Server certificates stored in an AWS account	20

You cannot request a limit increase for the following limits like access keys 2. You can refer AWS documentation

# Groups

- An IAM group is a collection of IAM users.
- Groups let you specify permissions for multiple users, which can make it easier to manage the permissions for those users.
- A group can contain many users, and a user can belong to multiple groups.
- Groups can't be nested; they can contain only users, not other groups.
- There's no default group that automatically includes all users in the AWS account. If you want to have a group like that, you need to create it and assign each new user to it.
- There's a limit to the number of groups you can have, and a limit to how many groups a user can be in.

# Groups



# IAM Roles

- IAM roles are designed so that our applications can securely make API requests from our instances, without requiring us to manage the security credentials that the applications use. Instead of creating and distributing our AWS credentials, we can delegate permission to make API requests using IAM roles

# Policies

- You manage access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an entity or resource, defines their permissions.
- AWS evaluates these policies when a principal, such as a user, makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents.



# IAM Dashboard

Search IAM

## Dashboard

Groups

Users

Roles

Policies

Identity providers

Account settings

Credential report

Encryption keys

## Welcome to Identity and Access Management

IAM users sign-in link:

[https://\[REDACTED\].signin.aws.amazon.com/console](https://[REDACTED].signin.aws.amazon.com/console) 

| Customize

### IAM Resources

Users: 28

Roles: 41

Groups: 11

Identity Providers: 0

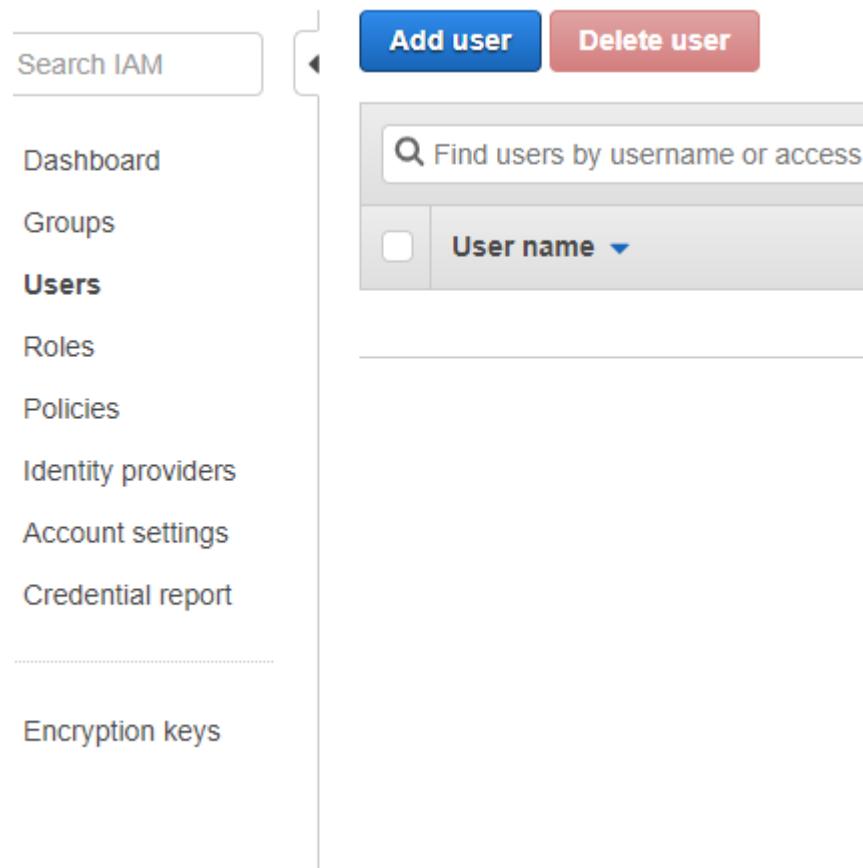
Customer Managed Policies: 16

### Security Status

5 out of 5 complete.

- Activate MFA on your root account 
- Create individual IAM users 
- Use groups to assign permissions 
- Apply an IAM password policy 
- Rotate your access keys 

# Add User- Demo



Search IAM

Add user   Delete user

Find users by username or access key

User name ▾

Dashboard

Groups

**Users**

Roles

Policies

Identity providers

Account settings

Credential report

Encryption keys

## Add user

1 2 3 4 5

### Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name\*  [Add another user](#)

### Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type\*  **Programmatic access**  
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

**AWS Management Console access**  
Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password\*  Autogenerated password  
 Custom password

Require password reset  User must create a new password at next sign-in  
Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

\* Required

[Cancel](#) [Next: Permissions](#)

# Add User - Demo

## Add user

1 2 3 4 5

### ▼ Set permissions



Add user to group



Copy permissions from existing user



Attach existing policies directly

#### Get started with groups

You haven't created any groups yet. Using groups is a best-practice way to manage users' permissions by job functions, AWS service access, or your custom permissions. Get started by creating a group. [Learn more](#)

[Create group](#)

### ▼ Set permissions boundary

Set a permissions boundary to control the maximum permissions this user can have. This is an advanced feature used to delegate permission management to others. [Learn more](#)

- Create user without a permissions boundary
- Use a permissions boundary to control the maximum user permissions

[Cancel](#)

[Previous](#)

[Next: Tags](#)

# Add User - Demo

Add user

1 2 3 4 5

## Add tags (optional)

IAM tags are key-value pairs you can add to your user. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this user. [Learn more](#)

Key	Value (optional)	Remove
Team	Developer	×
Add new key		

You can add 49 more tags.

# Add User - Demo

## Add user

1 2 3 4 5

### Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.



#### This user has no permissions

You haven't given this user any permissions. This means that the user has no access to any AWS service or resource. Consider returning to the previous step and adding some type of permissions.

### User details

User name	rns
AWS access type	AWS Management Console access - with a password
Console password type	Autogenerated
Require password reset	No
Permissions boundary	Permissions boundary is not set

### Tags

The new user will receive the following tag

Key	Value
Team	Developer

# Add User - Success

Add user

1 2 3 4 5

## Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: <https://reyazrns.signin.aws.amazon.com/console>

 Download .csv

	User	Password	Email login instructions
	 rns	***** <a href="#">Show</a>	<a href="#">Send email</a> 

# Login to Console for more users properties

# Groups

Search IAM

Create New Group

Group Actions ▾

Filter

Group Name ▾

No records found.

Dashboard

**Groups**

Users

Roles

Policies

Identity providers

Account settings

Credential report

Encryption keys

# Create Group- Demo

## Create New Group Wizard

**Step 1** : Group Name

Step 2 : Attach Policy

Step 3 : Review

**Click Next →**

## Set Group Name

Specify a group name. Group names can be edited any time.

**Group Name:**

Developers

Example: Developers or ProjectAlpha

Maximum 128 characters

# Create Group- Demo

## Create New Group Wizard

Step 1 : Group Name

**Step 2 : Attach Policy**

Step 3 : Review

## Attach Policy

Select one or more policies to attach. Each group can have up to 10 policies attached.

Filter: Policy Type		Filter
	Policy Name	Attached Entities
<input type="checkbox"/>	 AdministratorAccess	0
<input type="checkbox"/>	 AlexaForBusinessDeviceSetup	0
<input type="checkbox"/>	 AlexaForBusinessFullAccess	0
<input type="checkbox"/>	 AlexaForBusinessGatewayExecution	0
<input type="checkbox"/>	 AlexaForBusinessReadOnlyAccess	0
<input type="checkbox"/>	 AmazonAPIGatewayAdministrator	0
<input type="checkbox"/>	 AmazonAPIGatewayInvokeFullAccess	0
<input type="checkbox"/>	...	...

# Create Group- Demo

## Create New Group Wizard

[Step 1 : Group Name](#)

[Step 2 : Attach Policy](#)

**Step 3 : Review**

## Review

Review the following information, then click **Create Group** to proceed.

**Group Name** Developers

[Edit Group Name](#)

**Policies** arn:aws:iam::aws:policy/AdministratorAccess

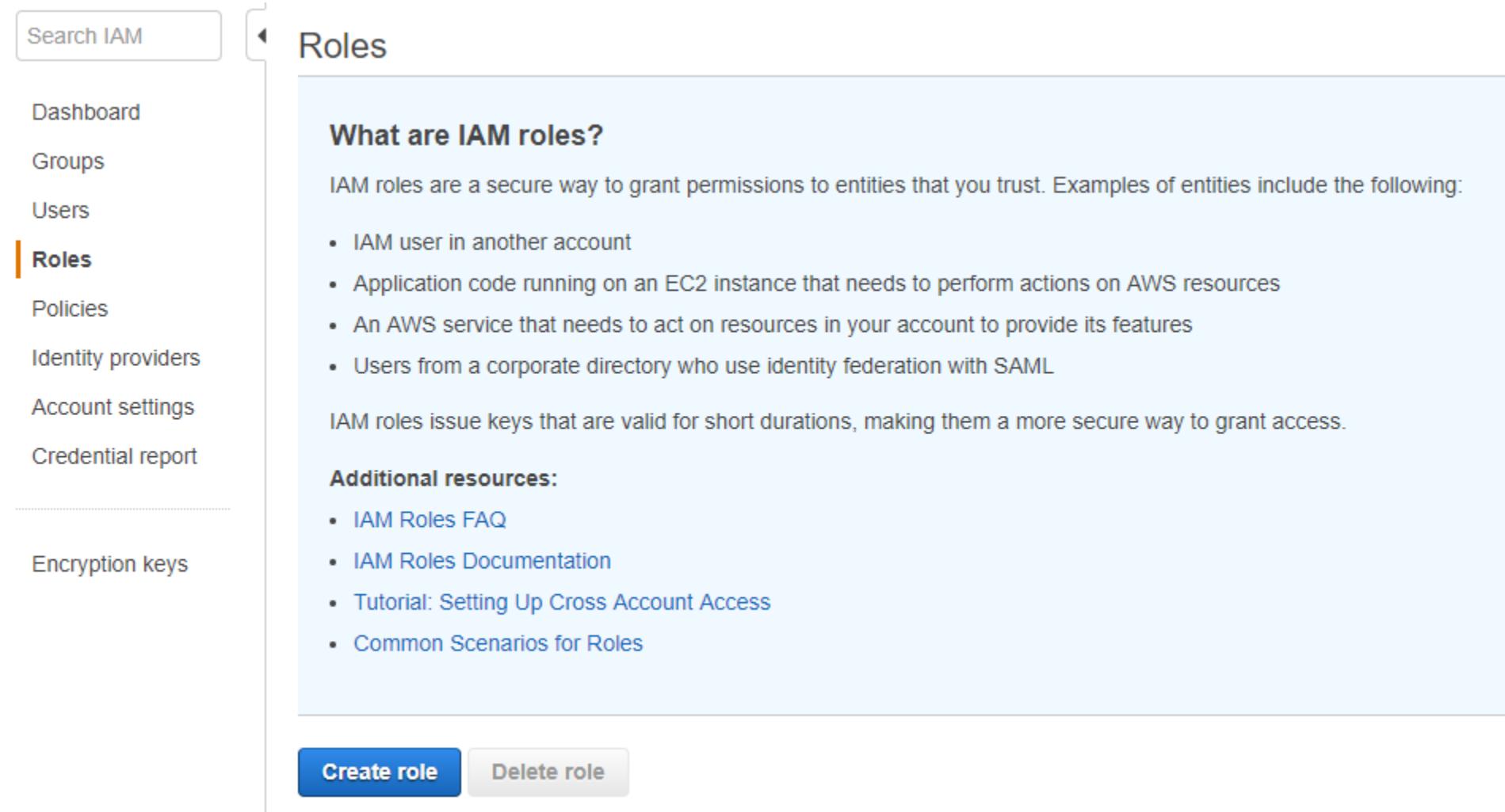
[Edit Policies](#)

**Create Group**

# Create Group- Demo

The screenshot shows the AWS IAM Groups page. The left sidebar includes a 'Search IAM' bar and links for Dashboard, Groups (which is selected and highlighted in orange), Users, Roles, Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main content area has a 'Create New Group' button and a 'Group Actions' dropdown menu. The 'Group Actions' menu is open, showing options: 'Add Users to Group', 'Delete Group', 'Edit Group Name', and 'Remove Users from Group'. The 'Edit Group Name' option is currently selected. The 'Group Name' field contains 'Developers'. The 'Users' section shows a table with 0 users listed. A 'Filter' input field is also present.

# Create Role- Demo



Search IAM

Roles

Dashboard

Groups

Users

**Roles**

Policies

Identity providers

Account settings

Credential report

Encryption keys

**What are IAM roles?**

IAM roles are a secure way to grant permissions to entities that you trust. Examples of entities include the following:

- IAM user in another account
- Application code running on an EC2 instance that needs to perform actions on AWS resources
- An AWS service that needs to act on resources in your account to provide its features
- Users from a corporate directory who use identity federation with SAML

IAM roles issue keys that are valid for short durations, making them a more secure way to grant access.

**Additional resources:**

- [IAM Roles FAQ](#)
- [IAM Roles Documentation](#)
- [Tutorial: Setting Up Cross Account Access](#)
- [Common Scenarios for Roles](#)

**Create role** **Delete role**

# Create Role- Demo

## Create role

1 2 3 4

### Select type of trusted entity



Allows AWS services to perform actions on your behalf. [Learn more](#)

### Choose the service that will use this role

#### EC2

Allows EC2 instances to call AWS services on your behalf.

#### Lambda

Allows Lambda functions to call AWS services on your behalf.

API Gateway	CodeBuild	<b>EC2 - Fleet</b>	IoT	Rekognition
AWS Support	CodeDeploy	EKS	Kinesis	S3
Amplify	Config	EMR	Lambda	SMS
AppSync	Connect	ElastiCache	Lex	SNS
Application Auto Scaling	DMS	Elastic Beanstalk	License Manager	SWF
Application Discovery Service	Data Lifecycle Manager	Elastic Container Service	Machine Learning	SageMaker
Auto Scaling	Data Pipeline	Elastic Transcoder	Macie	Service Catalog
Batch	DataSync	Elastic Load Balancing	MediaConvert	Step Functions
CloudFormation	DeepLens	Glue	OpsWorks	Storage Gateway
CloudHSM	Directory Service	Greengrass	RAM	Trusted Advisor
CloudTrail	DynamoDB	GuardDuty	RDS	VPC
CloudWatch Events	EC2	Inspector	Redshift	

\* Required

Cancel

Next: Permissions

# Create Role- Demo

## Create role

1 2 3 4

The service-linked role that you selected requires the following policy.

Filter policies   Search			Showing 1 result
Policy name 	Used as	Description	
 AWSEC2FleetServiceRolePolicy	None	Allows EC2 Fleet to launch and manage ins...	
<b>AWSEC2FleetServiceRolePolicy</b> Allows EC2 Fleet to launch and manage instances.			
 			
<pre>1 { 2   "Version": "2012-10-17", 3   "Statement": [ 4     { 5       "Effect": "Allow", 6       "Action": [ 7         "ec2:DescribeImages", 8         "ec2:DescribeSubnets", 9         "ec2:RequestSpotInstances", 10        "ec2:DescribeInstanceStatus", 11        "ec2:RunInstances" 12      ], 13      "Resource": [ 14        "*" 15      ] 16    } 17  ] 18}</pre>			

**Next: Tags**

# Create Role- Demo

## Create role

- 1
- 2
- 3
- 4

### Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. [Learn more](#)

**Service-Linked Roles doesn't support tagging during creation.**

# Create Role- Demo

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

**Role name** AWSServiceRoleForEC2Fleet Optional suffix

Use alphanumeric and '+=,.@-\_ ' characters. Maximum 64 characters.

**Role description** Allows EC2 Fleet to launch and manage EC2 instances on your behalf.

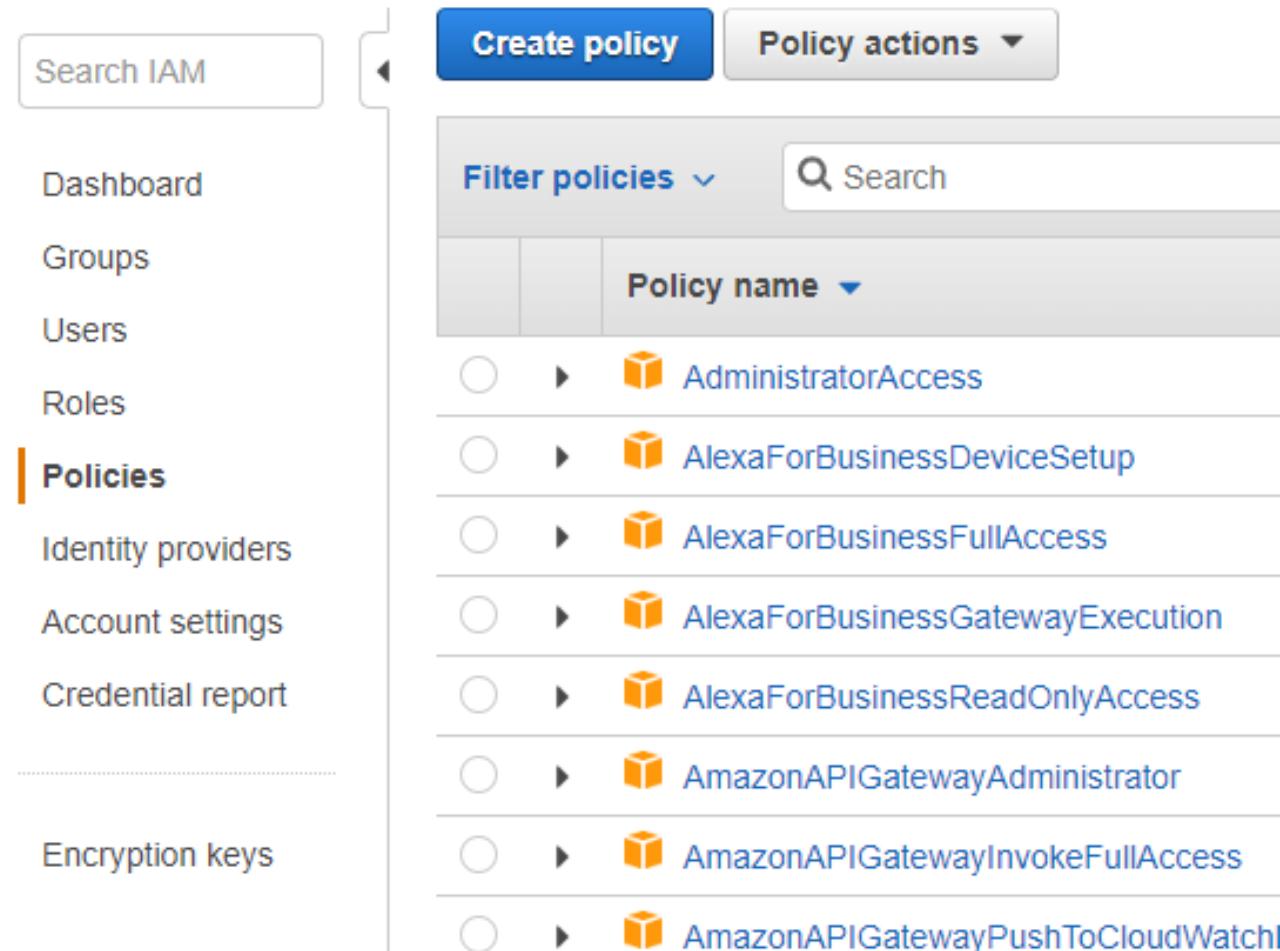
Maximum 1000 characters. Use alphanumeric and '+=,.@-\_ ' characters.

**Trusted entities** AWS service: ec2fleet.amazonaws.com

**Policies**  AWSEC2FleetServiceRolePolicy 

**Create Role**

# Create Policy - Demo



The screenshot shows the AWS IAM 'Create Policy' page. The left sidebar has a 'Search IAM' input field and links for Dashboard, Groups, Users, Roles, Policies (which is selected and highlighted in orange), Identity providers, Account settings, Credential report, and Encryption keys. The main area has a 'Create policy' button, a 'Policy actions' dropdown, a 'Filter policies' dropdown, and a search bar. A table lists existing policies with columns for Policy name, ARN, and Description (which is not visible in the screenshot). The policies listed are: AdministratorAccess, AlexaForBusinessDeviceSetup, AlexaForBusinessFullAccess, AlexaForBusinessGatewayExecution, AlexaForBusinessReadOnlyAccess, AmazonAPIGatewayAdministrator, AmazonAPIGatewayInvokeFullAccess, and AmazonAPIGatewayPushToCloudWatch.

Policy name	ARN	Description
AdministratorAccess	arn:aws:iam::aws:policy/AdministratorAccess	
AlexaForBusinessDeviceSetup	arn:aws:iam::aws:policy/AlexaForBusinessDeviceSetup	
AlexaForBusinessFullAccess	arn:aws:iam::aws:policy/AlexaForBusinessFullAccess	
AlexaForBusinessGatewayExecution	arn:aws:iam::aws:policy/AlexaForBusinessGatewayExecution	
AlexaForBusinessReadOnlyAccess	arn:aws:iam::aws:policy/AlexaForBusinessReadOnlyAccess	
AmazonAPIGatewayAdministrator	arn:aws:iam::aws:policy/AmazonAPIGatewayAdministrator	
AmazonAPIGatewayInvokeFullAccess	arn:aws:iam::aws:policy/AmazonAPIGatewayInvokeFullAccess	
AmazonAPIGatewayPushToCloudWatchLogs	arn:aws:iam::aws:policy/AmazonAPIGatewayPushToCloudWatchLogs	

# Create Policy- Demo

## Create policy

1

2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor [JSON](#) [Import managed policy](#)

[Expand all](#) | [Collapse all](#)

▼ Select a service [Clone](#) | [Remove](#)

▶ **Service** [Choose a service](#)

**Actions** Choose a service before defining actions

**Resources** Choose actions before applying resources

**Request conditions** Choose actions before specifying conditions

[+ Add additional permissions](#)

# Create Role- Demo

## Create policy

1 2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor JSON

Import managed policy

[Expand all](#) | [Collapse all](#)

▼ S3

▶ Service S3

Clone | Remove

▼ Actions Specify the actions allowed in S3 [?](#)

[close](#)

Filter actions

[Switch to deny permissions](#) [i](#)

**Manual actions (add actions)**

All S3 actions (s3:\*)

**Access level**

▶  List

▶  Read

▶  Write

▶  Permissions management

[Expand all](#) | [Collapse all](#)

**Resources** Choose actions before applying resources

**Request conditions** Choose actions before specifying conditions

[+ Add additional permissions](#)

# Create Policy - Demo

## Create policy

1

2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

JSON

Import managed policy

[Expand all](#) | [Collapse all](#)

▼ S3 (All actions)

▶ Service S3

Clone | Remove

▼ Actions Specify the actions allowed in S3 [?](#)

[close](#)  Filter actions

[Switch to deny permissions](#) [i](#)

Manual actions (add actions)

All S3 actions (s3:\*)

Access level

▶  List (3 selected)

▶  Read (33 selected)

▶  Write (29 selected)

▶  Permissions management (8 selected)

Expand all | Collapse all

▶ Resources All resources

▶ Request conditions Specify request conditions (optional)

[+ Add additional permissions](#)

# Create Policy - Demo

Create policy

1 2

Review policy

Name\*

S3FullAccess|

Use alphanumeric and '+,.,@-\_ characters. Maximum 128 characters.

Description

Maximum 1000 characters. Use alphanumeric and '+,.,@-\_ characters.

Summary

Service ▾				Access level	Resource	Request condition
Allow (1 of 165 services) <a href="#">Show remaining 164</a>						
S3	Full access		All resources	None		

**Create Policy**

# MFA

## Customized sign in URL





# Amazon CloudWatch

By  
Reyaz Shaik

# What is Cloud Watch?

- Amazon CloudWatch is a monitoring service for AWS cloud resources and the applications you run on AWS. It is used to collect and track metrics, collect and monitor log files, and set alarms.

# What is Cloud Watch?

- You can monitor your instances using Amazon CloudWatch, which collects and processes raw data from Amazon EC2 into readable, near real-time metrics. These statistics are recorded for a period of 15 months, so that you can access historical information and gain a better perspective on how your web application or service is performing.
- By default, Amazon EC2 sends metric data to CloudWatch in 5-minute periods.(Basic Monitoring)
- To send metric data for your instance to CloudWatch in 1-minute periods, you can enable detailed monitoring on the instance(Detailed Monitoring)
- The Amazon EC2 console displays a series of graphs based on the raw data from Amazon CloudWatch. Depending on your needs, you might prefer to get data for your instances from Amazon CloudWatch instead of the graphs in the console.

# EC2 Cloud Watch Console

Description Status Checks **Monitoring** Tags

▶ CloudWatch alarms: ✓ No alarms configured

Create A

CloudWatch metrics: Basic monitoring. Enable Detailed Monitoring

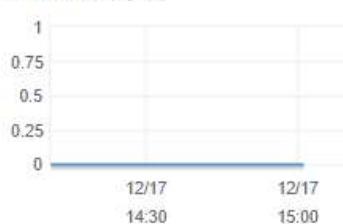
Showing data for: Last Hour

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All times shown are in UTC. [View all CloudWatch metrics](#)

CPU Utilization (Percent)



Disk Reads (Bytes)



Disk Read Operations (Operations)



Disk Writes (Bytes)



Disk Write Operations (Operations)



Network In (Bytes)



Network Out (Bytes)



Network Packets In (Count)



Network Packets Out (Count)



Status Check Failed (Any) (Count)



Status Check Failed (Instance) (Count)



Status Check Failed (System) (Count)



CPU Credit Usage (Count)



CPU Credit Balance (Count)



# Monitoring

Before getting started with Amazon CloudWatch it's important to know the items it enables the user to monitor:

- Amazon EC2 instances
- Amazon EBS volumes
- Elastic Load balancers
- AutoScaling groups
- Amazon RDS database instances in real-time
- Amazon SQS queues, SNS topics,
- EMR job flows, Storage Gateway, DynamoDB tables
- Estimated AWS charges

# Monitoring

With Amazon CloudWatch, the user can get:

- Up-to-minute statistics
- View graphs
- Set alarms for your metric data
- Use Auto Scaling to add/remove resources based on CloudWatch Metrics

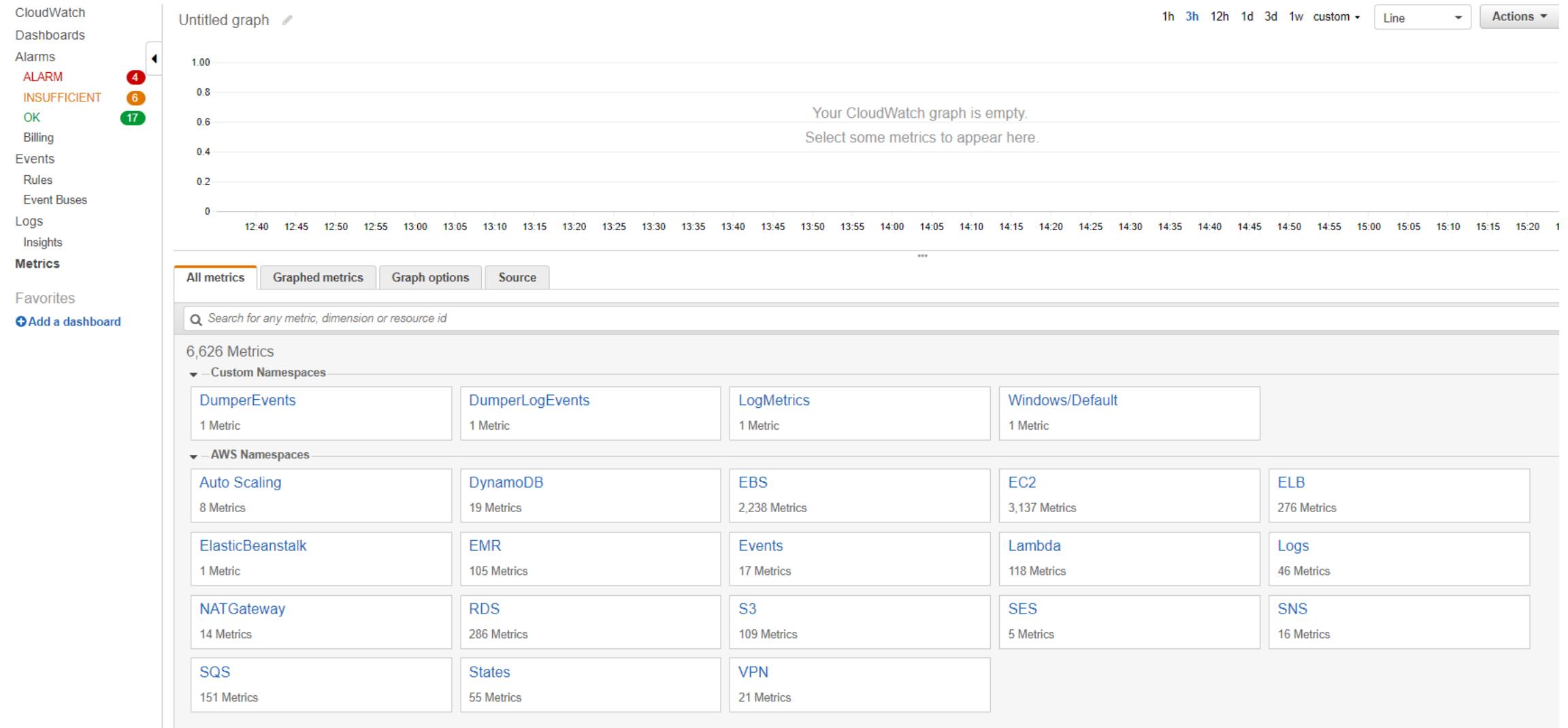
# CloudWatch Monitoring:

- **Basic Monitoring for Amazon EC2 instances:** Ten pre-selected metrics at five-minute frequency, free of charge
- **Detailed Monitoring for Amazon EC2 instances:** Seven pre-selected metrics at one-minute frequency, for an additional charge
- **Amazon EBS volumes:** Ten pre-selected metrics at five-minute frequency, free of charge
- **Elastic Load Balancers:** Ten pre-selected metrics at one-minute frequency, free of charge
- **Auto Scaling groups:** Seven pre-selected metrics at one-minute frequency, optional and charged at standard pricing
- **Amazon RDS DB instances:** Thirteen pre-selected metrics at one-minute frequency, free of charge.

# Metrics

- **Namespaces:** It is a grouping to know what this metric belongs to. For example: AWS/EC2, AWS/AutoScaling/ AWS/ELB
- **Dimensions:** Dimension is a name/value pair that you uniquely identify a metric. For example: AutoScalingGroupName, ImageId, InstanceID, InstanceType, Volume ID.
- **Timestamps:** To know what timestamp it had captured.
- **Units:** Unit represents the statistic's unit of measure. For example: EC2 NetworkIn metric in bytes.

# Cloud Watch DashBoard



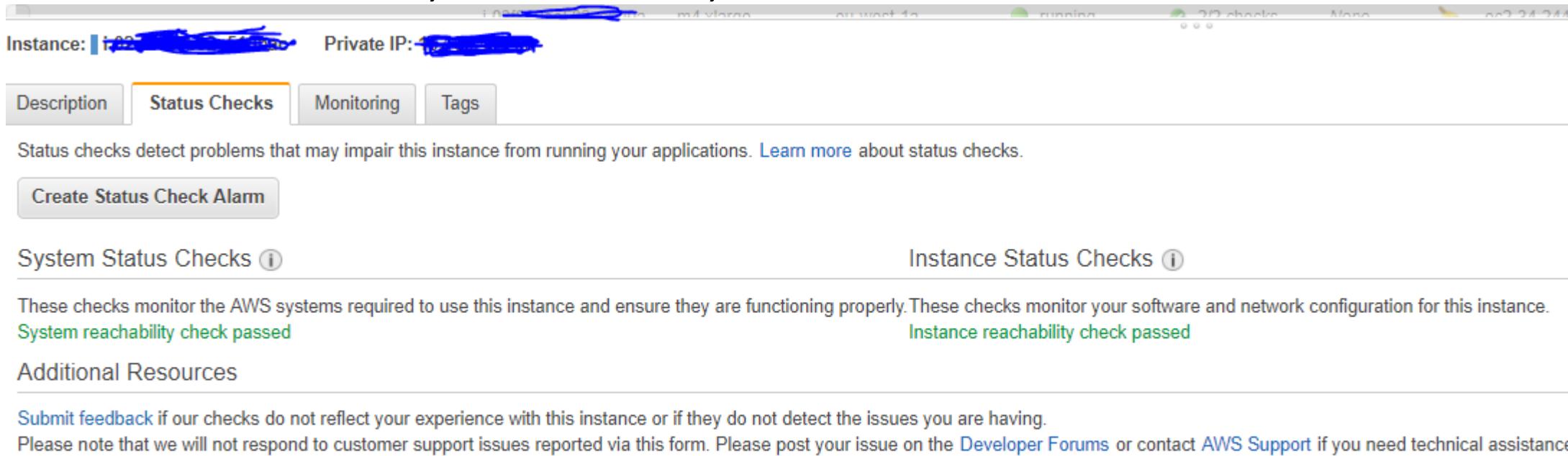
# Alarms

## EC2 Instance status check Alarm

Select any EC2 instance and go to “Status Checks” tab

## Create Status Check Alarm

Alarm states: Alarm, Insufficient , OK



Instance: [REDACTED] Private IP: [REDACTED]

Description Status Checks Monitoring Tags

Status checks detect problems that may impair this instance from running your applications. [Learn more](#) about status checks.

[Create Status Check Alarm](#)

**System Status Checks** ⓘ

These checks monitor the AWS systems required to use this instance and ensure they are functioning properly. These checks monitor your software and network configuration for this instance.

System reachability check passed

**Instance Status Checks** ⓘ

Instance reachability check passed

**Additional Resources**

Submit feedback if our checks do not reflect your experience with this instance or if they do not detect the issues you are having.

Please note that we will not respond to customer support issues reported via this form. Please post your issue on the [Developer Forums](#) or contact [AWS Support](#) if you need technical assistance

# Alarms

**Create Alarm** X

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

**Send a notification to:**  [create topic](#)

**Take the action:**  Recover this instance (i)  
 Stop this instance (i)  
 Terminate this instance (i)  
 Reboot this instance (i)

**Whenever:**  ▼

**Is:** Failing

**For at least:**  consecutive period(s) of  ▼

**Name of alarm:**

**Status Check Failed (Any) Count**

Time	Count
10:00	0
12:00	0
14:00	1

[Cancel](#) Create Alarm

Create topic if you don't have any notification enabled. Next slide has the screen shot of create topic

# Alarms

Click on Create Topic

### Create Alarm

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

**Send a notification to:**  [cancel](#)

**With these recipients:**

**Take the action:**  Recover this instance [i](#)  
 Stop this instance [i](#)  
 Terminate this instance [i](#)  
 Reboot this instance [i](#)

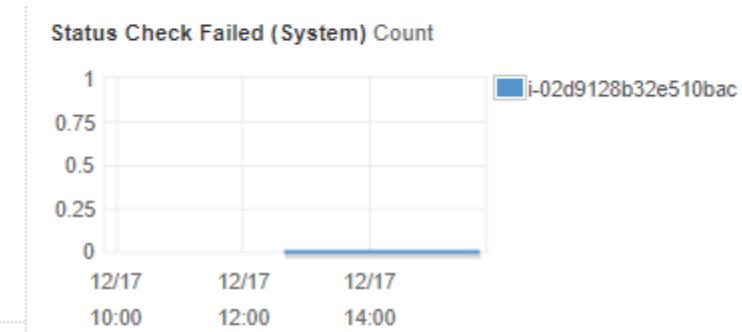
**Whenever:**  [▼](#)

**Is:** Failing

**For at least:**  consecutive period(s) of  [▼](#)

**Name of alarm:**

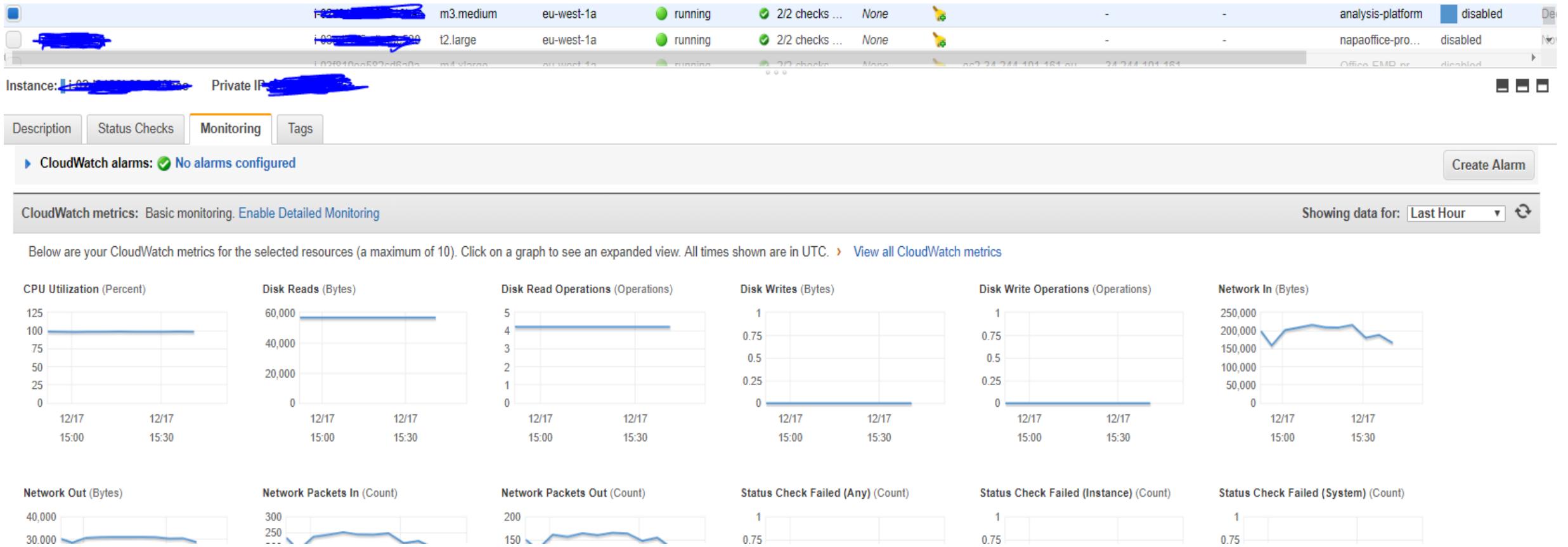
[Cancel](#) [Create Alarm](#)



# Create Alarms for EC2 Metrics

# EC2 Alarms

Select any EC2 instance and go to Monitoring tab → right top Create Alarm



# EC2 Alarms

**Create Alarm** X

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

**Send a notification to:**  [create topic](#)

**Take the action:**  Recover this instance [i](#)  
 Stop this instance [i](#)  
 Terminate this instance [i](#)  
 Reboot this instance [i](#)

**Whenever:**  of

**Is:**   Percent

**For at least:**  consecutive period(s) of

**CPU Utilization Percent**

**Name of alarm:** awsec2-i-02d9128b32e510bac-CPU-Utilization

[Cancel](#) Create Alarm

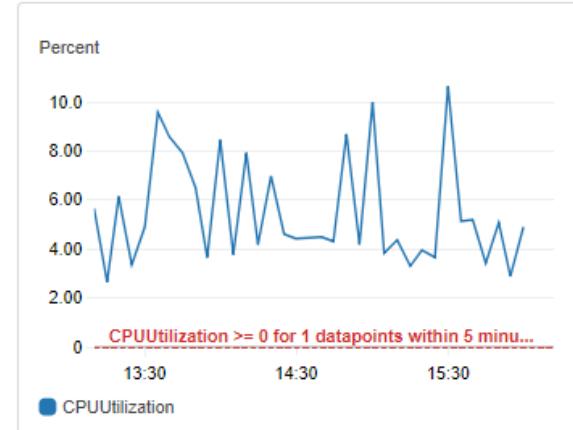
# Create Alarm from the Cloudwatch Dashboard

- → Create Alarm
- Select Metric
- Provide Alarm Details
- Provide CPUUtilization Condition

## Create new alarm

### Metric [Edit](#)

This alarm will trigger when the blue line goes up to or above the red line for 1 datapoints within 5 minutes



Namespace: AWS/EC2  
Metric Name: CPUUtilization  
InstanceId: i-00000000000000000  
InstanceName: [REDACTED]  
Period: 5 Minutes  
Statistic: Average

## Alarm details

Provide the details and threshold for your alarm. Use the graph to help set the appropriate threshold.

Name:

Description:

Whenever: CPUUtilization

is:  $\geq$

for: 1  out of  datapoints [i](#)

## Additional settings

Provide additional configuration for your alarm.

Treat missing data as:  [i](#)

# Alarm Actions

- Notification
- AutoScalingAction
- EC2 Actions
- → Create Alarm

## Additional settings

Provide additional configuration for your alarm.

Treat missing data as:  

## Actions

Define what actions are taken when your alarm changes state.

### Notification

[Delete](#)

Whenever this alarm:  

Send notification to:   [New list](#) [Enter list](#) 

### AutoScaling Action

[Delete](#)

Whenever this alarm:  

From resource type:  

From the:  

Take this action:  

### EC2 Action

[Delete](#)

Whenever this alarm:  

Take this action:  Recover this instance 

Stop this instance 

Terminate this instance 

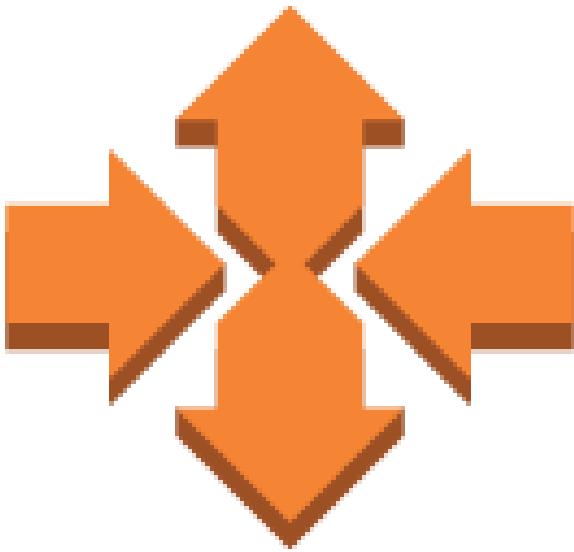
Reboot this instance 

[+ Notification](#)

[+ AutoScaling Action](#)

[+ EC2 Action](#)

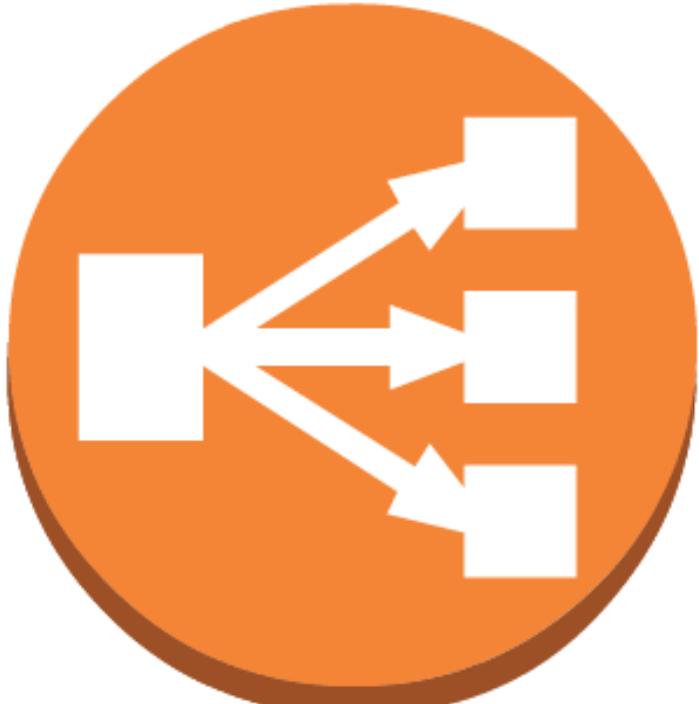
# Auto Scaling



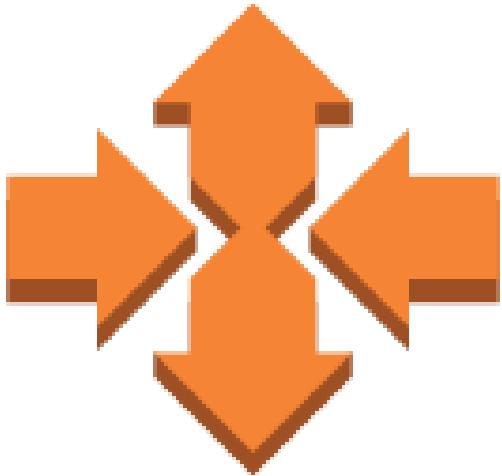
# Before AutoScaling

**Lets Talk about ELB(Elastic Load Balancer)**

# Elastic Load Balancer



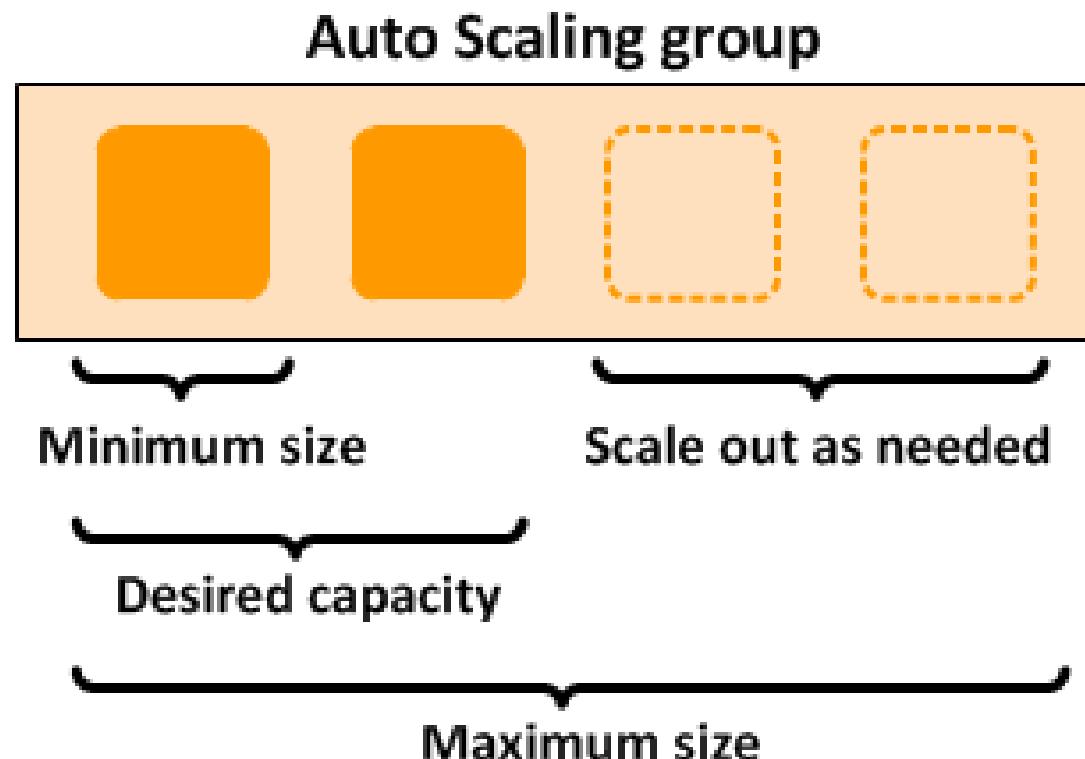
# AutoScaling



By  
Reyaz Shaik

# What is AutoScaling

- Autoscaling scales up and down a group of servers based on computing or traffic demand by provisioning new services.
- AWS autoscaling allows us to increase/decrease the number of EC2 instances within our application's architecture.
- With AWS autoscaling, we create collections of EC2 instances, called **Auto Scaling groups (ASG)**.



# AWS Auto Scaling Components

The key components of Amazon EC2 Auto Scaling.

## Groups

- Your EC2 instances are organized in to *groups* so that they can be treated as a logical unit for the purposes of scaling and management. When you create a group, you can specify its minimum, maximum, and, desired number of EC2 instances. For more information,

## Configuration templates

- Your group uses a *launch template* or a *launch configuration* as a configuration template for its EC2 instances. You can specify information such as the AMI ID, instance type, key pair, security groups, and block device mapping for your instances.

## Scaling options

- Amazon EC2 Auto Scaling provides several ways for you to scale your Auto Scaling groups. For example, you can configure a group to scale based on the occurrence of specified conditions (dynamic scaling) or on a schedule.

# Project



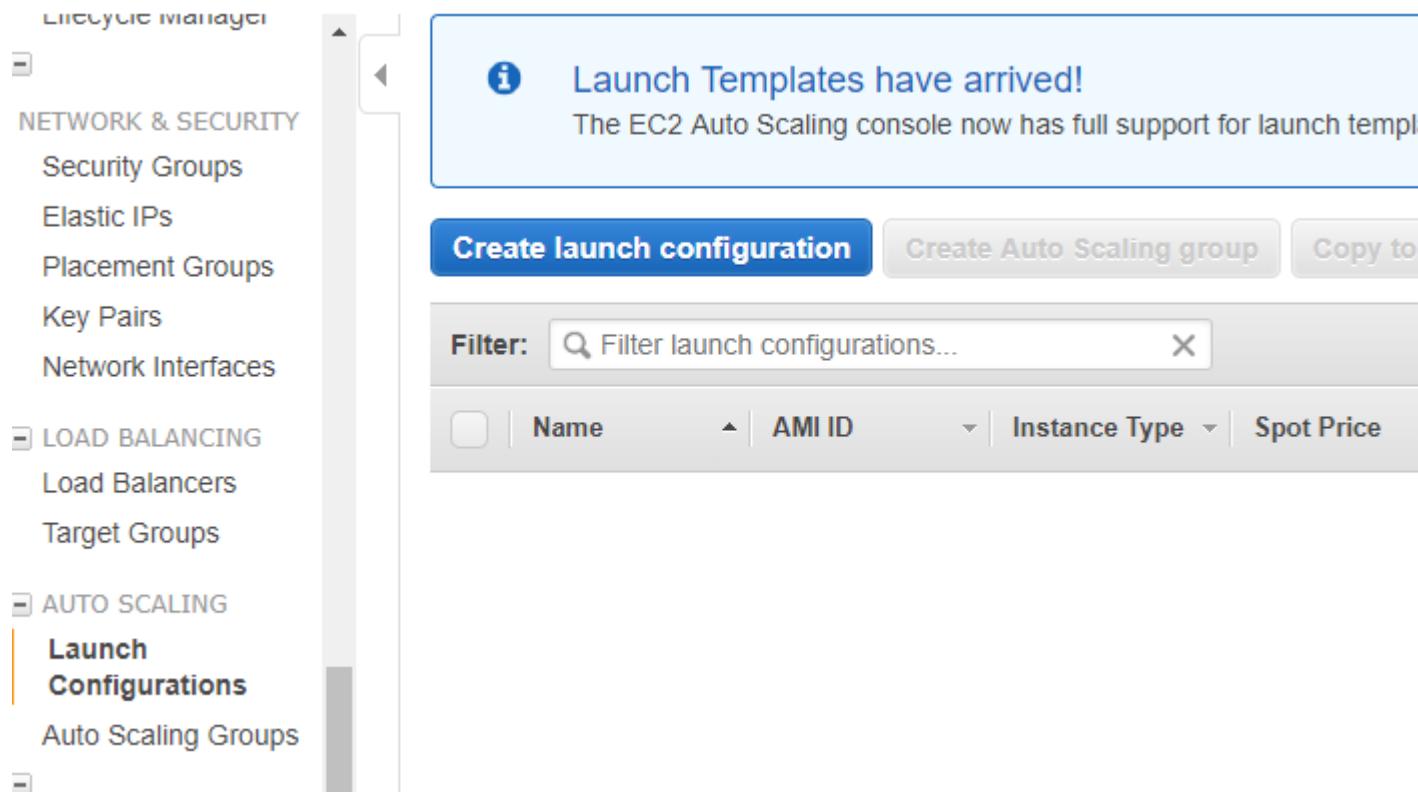
# Project Requirements

we will go through the following steps and this is a high-level overview of what you'll be doing:

- Create a launch configuration with 1 Node app with user data to start an HTTP server (config from lab 2, user data has modified Node Hello World app)
- Create an autoscaling group
- Create an autoscaling policy to increase instances by 1 when CPU load is > 15% for 1 min
- Load test it with loadtest npm module to see a new instance is created
- Remove autoscaling group
- Terminate instances

# AWS Auto Scaling Implementation

Log in to the web console (use Mumbai region) and navigate to the EC2 dashboard. Select “Launch Configuration” to start the wizard.



# Configuration Wizard

- The Launch Configuration wizard is *very similar* to the Launch instance wizard. You will need to specify image(s), instance(s) type(s), volume(s), etc.
- On the first screen of the instance wizard, find in Quick Start Amazon Linux. We recommend using “*Amazon Linux 64-bit, HVM, SSD, EBS*” because it’s eligible for free tier on **t2.micro**.

## Steps

- → Select Image
- → Select instance type
- → Name the configuration → provide “user data” with some predefined scripts under advance settings
- → Add Storage
- → Configuration Security Groups
- → Review & Create Launch configuration

# Script for user data

- `#!/bin/bash`
- `yum install httpd -y`
- `service httpd start`
- `mkdir /var/www/html`
- `echo 'Your AutoScaling and ELB test page!' > /var/www/html/index.html`

- Once you finished launch Configuration wizard. You will see the below screen

Launch configuration creation status

✓ Successfully created launch configuration: Test [View creation log](#)

▼ View [View your launch configurations](#) [View your Auto Scaling groups](#)

▶ Here are some helpful resources to get you started

[Create an Auto Scaling group using this launch configuration](#) [Close](#)



Click “Create an Auto Scaling Group using this launch configuration”

# Create Auto scaling Group

1. Configure Auto Scaling group details   2. Configure scaling policies   3. Configure Notifications   4. Configure Tags   5. Review

Create Auto Scaling Group

Launch Configuration [\(i\)](#) Test

Group size [\(i\)](#) Start with  instances

Network [\(i\)](#)  [\(x\)](#) [C](#) Create new VPC

Subnet [\(i\)](#)  [\(x\)](#)  
 [\(x\)](#)  
[Create new subnet](#)

Each instance in this Auto Scaling group will be assigned a public IP address. [\(i\)](#)

[▼ Advanced Details](#)

Load Balancing [\(i\)](#)  Receive traffic from one or more load balancers [Learn about Elastic Load Balancing](#)

Classic Load Balancers [\(i\)](#)

Target Groups [\(i\)](#)

Health Check Type [\(i\)](#)  ELB  EC2

Health Check Grace Period [\(i\)](#)  seconds

Monitoring [\(i\)](#) Amazon EC2 Detailed Monitoring metrics, which are provided at 1 minute frequency, are not enabled for the launch configuration Test. Instances launched from it will use Basic Monitoring metrics, provided at 5 minute frequency.  
[Learn more](#)

Instance Protection [\(i\)](#)

Service-Linked Role [\(i\)](#)  [\(x\)](#) [C](#) [View Role in IAM](#)

# Scaling Policies

1. Configure Auto Scaling group details   2. **Configure scaling policies**   3. Configure Notifications   4. Configure Tags   5. Review

Create Auto Scaling Group

Keep this group at its initial size  
 Use scaling policies to adjust the capacity of this group

Scale between **1** and **2** instances. These will be the minimum and maximum size of your group.

**Increase Group Size**

Name:  Execute policy when:  [Add new alarm](#)

Take the action:  **0** instances [Add step](#)

Instances need:  seconds to warm up after each step

[Create a simple scaling policy](#)

**Decrease Group Size**

Name:  Execute policy when:  [Add new alarm](#)

Take the action:  **0** instances [Add step](#)

[Create a simple scaling policy](#)

[Scale the Auto Scaling group using a target tracking scaling policy](#)

Scale between 1 and 2 instances  
Execute policy when: Add new alarm

Adding new alarm -> see next slide

# Create Alarm

**Create Alarm** X

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

**Send a notification to:**  arn:aws:kinesis:...:topic

**Whenever:**  of  Percent

**Is:**   Percent

**For at least:**  consecutive period(s) of

**Name of alarm:**

**CPU Utilization Percent**

10  
7.5  
5  
2.5  
0

12/19 12/19 12/19  
04:00 06:00 08:00

MyAutoScalingGroup

If you don't have topic, create one

## Create Auto Scaling Group

Keep this group at its initial size

Use scaling policies to adjust the capacity of this group

Scale between  and  instances. These will be the minimum and maximum size of your group.

### Increase Group Size

Name:

Execute policy when:  C Add new alarm

breaches the alarm threshold: CPUUtilization  $\geq 15$  for 60 seconds  
for the metric dimensions AutoScalingGroupName = CPUgreaterthan10

Take the action: Add  instances   $\leq \text{CPUUtilization} < +\infty$

[Add step](#) i

Instances need:  seconds to warm up after each step

[Create a simple scaling policy](#) i

### Decrease Group Size

Name:

Execute policy when:  C Add new alarm

breaches the alarm threshold: CPUUtilization  $\geq 15$  for 60 seconds  
for the metric dimensions AutoScalingGroupName = CPUgreaterthan10

Take the action: Remove  instances   $\leq \text{CPUUtilization} < +\infty$

[Add step](#) i

Select that alarm for “Execute policy when”  
Add 1 instance when cpu is greater than 10 %  
Instance need 1 seconds to warm(just for demo)

Do the same thing for Decrease Group Size  
Decrease the instance when cpu is less

## Add Notifications

### Create Auto Scaling Group

Configure your Auto Scaling group to send notifications to a specified endpoint, such as an email address, whenever a specified event takes place, incl  
If you created a new topic, check your email for a confirmation message and click the included link to confirm your subscription. Notifications can only b

Send a notification to:  [create topic](#)

Whenever instances:  launch  
 terminate  
 fail to launch  
 fail to terminate

---

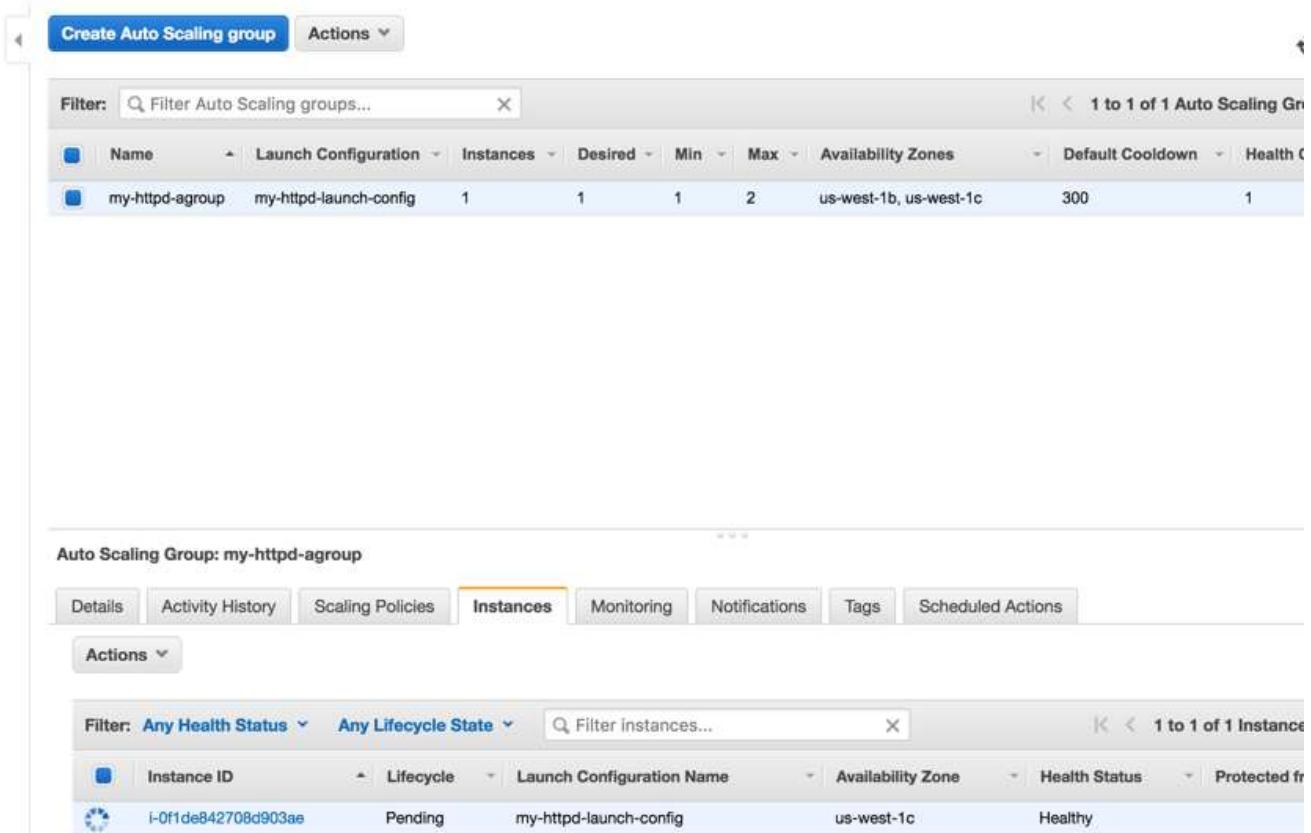
[Add notification](#)

Next Create Tags, review and Finish

# Creating Auto scaling Group Done

- Navigate to Auto Scaling Group from the EC2 dashboard
- Once you select the created Autoscaling group
- Go to Instances tab, you see a minimum instance has being created.

Access the instance public IP you will see the website



The screenshot shows the EC2 Auto Scaling Groups and Instances tabs.

**Auto Scaling Groups Tab:**

Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health C
my-htpd-agroup	my-htpd-launch-config	1	1	1	2	us-west-1b, us-west-1c	300	1

**Auto Scaling Group: my-htpd-agroup Instances Tab:**

Actions	Details	Activity History	Scaling Policies	Instances	Monitoring	Notifications	Tags	Scheduled Actions
Actions	Details	Activity History	Scaling Policies	Instances	Monitoring	Notifications	Tags	Scheduled Actions
Actions	Filter: Any Health Status Any Lifecycle State	Filter instances...	Instances	Instance ID: i-0f1de842708d903ae	Lifecycle: Pending	Launch Configuration Name: my-htpd-launch-config	Availability Zone: us-west-1c	Health Status: Healthy

# Now create Application Load Balancer

- Create Application Load Balancer
  - Name: Provide proper name
  - Scheme: internet facing
  - IP address type: ipv4
  - Listeners: allow ports as per your application. Ex: Port 80
  - Availability Zones: Select both subnets
- After this, configure security Groups
- Create Target Groups
- Register Instances

# Configure Load Balancer

1. Configure Load Balancer   2. Configure Security Settings   3. Configure Security Groups   4. Configure Routing   5. Register Targets   6. Review

## Step 1: Configure Load Balancer

### Basic Configuration

To configure your load balancer, provide a name, select a scheme, specify one or more listeners, and select a network. The default configuration is an Internet-facing load balancer in the selected net

Name i

Scheme i  internet-facing  internal

IP address type i

### Listeners

A listener is a process that checks for connection requests, using the protocol and port that you configured.

Load Balancer Protocol	Load Balancer Port
<input type="text" value="HTTP"/>	<input type="text" value="80"/>

**Add listener**

### Availability Zones

Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. Yo  
balancer.

VPC i

<input type="checkbox"/> Availability Zone	Subnet ID	Subnet IPv4 CIDR	Name
<input checked="" type="checkbox"/> ap-south-1a	subnet-a6c089ce	172.31.16.0/20	
<input checked="" type="checkbox"/> ap-south-1b	subnet-dd942a91	172.31.0.0/20	

# Target Groups

1. Configure Load Balancer   2. Configure Security Settings   3. Configure Security Groups   4. Configure Routing   5. Register Targets   6. Review

## Step 4: Configure Routing

Your load balancer routes requests to the targets in this target group using the protocol and port that you specify, and performs health checks on the targets.

### Target group

**Target group** ⓘ

**Name** ⓘ

**Target type**  Instance  
 IP  
 Lambda function

**Protocol** ⓘ

**Port** ⓘ

### Health checks

**Protocol** ⓘ

**Path** ⓘ

► Advanced health check settings

# Register instance

The screenshot shows the 'Register Targets' step in the AWS CloudFront configuration wizard. The top navigation bar includes 'Services', 'Resource Groups', a star icon, and account information for 'Azat' in 'N. California'. The step navigation bar shows '1. Configure Load Balancer', '2. Configure Security Settings', '3. Configure Security Groups', '4. Configure Routing', '5. Register Targets' (which is underlined in yellow), and '6. Review'. The main content area is titled 'Step 5: Register Targets' with the sub-instruction: 'Register targets with your target group. If you register an instance running in an enabled Availability Zone, the load balancer starts routing requests to the instance as soon as the registration process completes and the instance passes the initial health checks.' Below this, a 'Registered instances' section shows a table with one instance registered on port 80. The 'Instances' section shows a search interface to add more instances to the target group. A large 'Create' button is at the bottom right.

Step 5: Register Targets

Register targets with your target group. If you register an instance running in an enabled Availability Zone, the load balancer starts routing requests to the instance as soon as the registration process completes and the instance passes the initial health checks.

Registered instances

To deregister instances, select one or more registered instances and then click Remove.

Instance	Name	Port	State	Security groups	Zone
i-0f1de842708d903ae		80	running	open-sg	us-west-1c

Instances

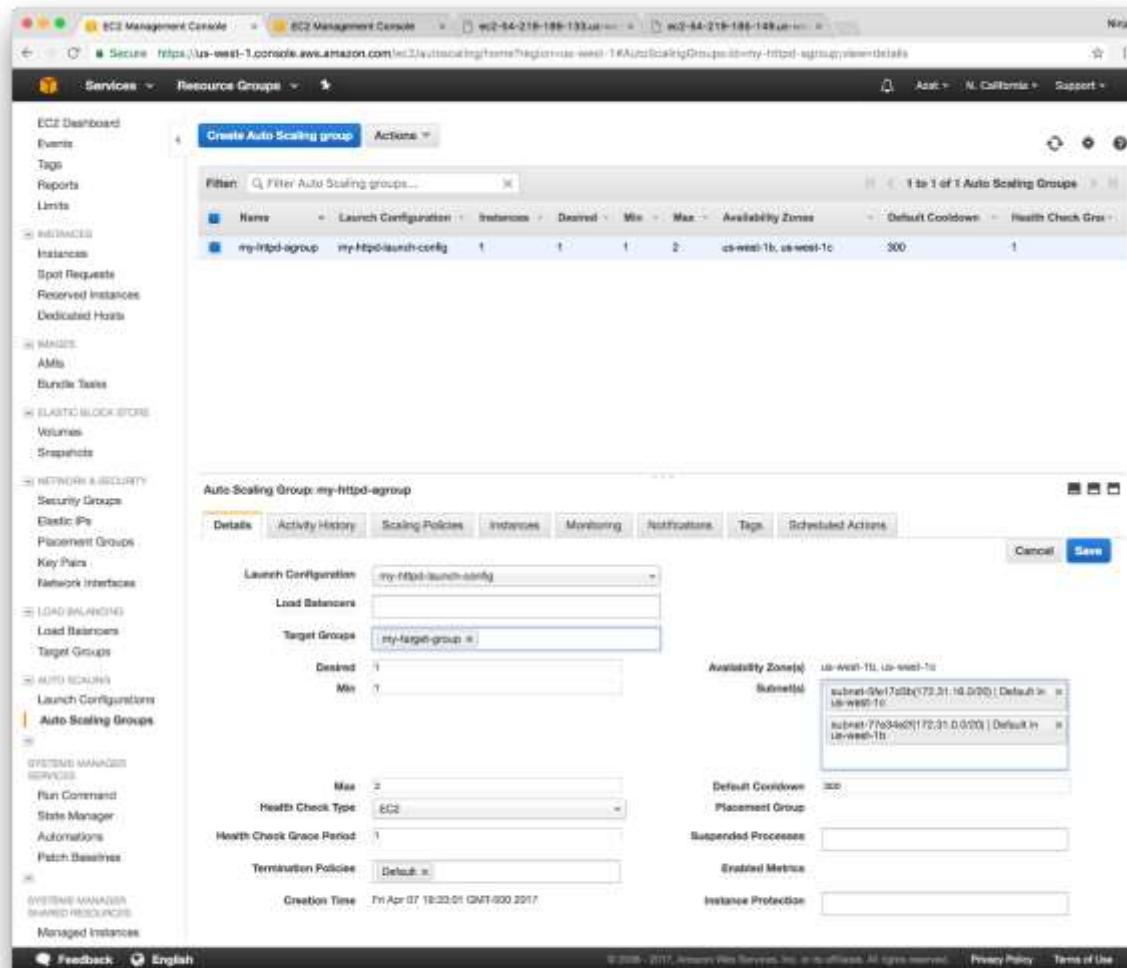
To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.

Add to registered on port 80

Search Instances	X					
Instance	Name	State	Security	Zone	Subnet ID	Subnet CIDR
i-023c012ed834e3cb6		running	open-sg	us-west-1b	subnet-77e34e2f	172.31.0.0/20
i-0f1de842708d903ae		running	open-sg	us-west-1c	subnet-5fe17d3b	172.31.16.0/20

Create

- Lastly, associate this target group (ELB) with your autoscaling group by editing your autoscaling group and adding the target group by name (there would be an auto complete drop down).
- Stress the webserver and see autoscaling works meaning a new instance will be created or not
- To Stress the server you will find many utilities





# S3 Static Website Hosting and Route 53 DNS Failover to redirect to maintenance page

ref:

<http://javaworld-abhinav.blogspot.com/2017/04/s3-static-website-hosting-and-route-53.html>

We will do following activities:

- Setup S3 Bucket and host static maintenance page.
- Launch 2 EC2 instances and setup a small web app on both instances.
- Launch and ELB and attach the EC2 instances with it.
- Configure Route 53 DNS failover to redirect to maintenance page when instances are unhealthy/down.

Create 2 EC2 Instances

Setup S3 bucket(web.cloudrsh.com) maintenance. Html

Make public access & enable hosting

Create target & register both instances

Create load balancer & register both instances

Create route53 healthcheck

Create A record with web.cloudrsh.com with ELB IP

Create A record with Alias of S3 website hosting

Stop apache on Ec2 instance

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "PublicReadForGetBucketObjects",  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": "s3:GetObject",  
      "Resource":  
        "arn:aws:s3:::web.abhinav.com/*"  
    }  
  ]  
}
```

[Back to Hosted Zones](#)[Create Record Set](#)[Import Zone File](#)[Delete Record Set](#)[Test Record Set](#)

	Name	Type	Value	Evaluate Target Health	Health Check ID	TTL	Region	Weight	Geolocation
	cloudrsh.com.	NS	ns-1181.awsdns-19.org. ns-508.awsdns-63.com. ns-1745.awsdns-26.co.uk. ns-611.awsdns-12.net.	-	-	172800			
	cloudrsh.com.	SOA	ns-1181.awsdns-19.org. awsdns-hostmaster.amazon	-	-	900			
<input checked="" type="checkbox"/>	web.cloudrsh.com.	A	52.66.60.186	-	4fbe48b2-ac9d-4b6f-86af-cf11cb914ce6	300			
	web.cloudrsh.com.	A	ALIAS s3-website.ap-south-1.amazonaws.com. (z11	No	-				

**Edit Record Set**

**Name:** web.cloudrsh.com.

**Type:** A – IPv4 address

**Alias:**  Yes  No

**TTL (Seconds):**

**Value:** 52.66.60.186

IPv4 address. Enter multiple addresses on separate lines.

Example:  
192.0.2.235  
198.51.100.234

**Routing Policy:** Failover

Route 53 responds to queries using primary record sets if any are healthy, or using secondary record sets otherwise. [Learn More](#)

**Failover Record Type:**  Primary  Secondary

**Set ID:** web-Primary

**Associate with Health Check:**  Yes  No

When responding to queries, Route 53 can omit resources that fail health checks. [Learn More](#)

**Health Check to Associate:** myhealthcheck

## Configure health check



Route 53 health checks let you track the health status of your resources, such as web servers or mail servers, and take action when an outage occurs.

**Name** myhealthcheck

**What to monitor** Endpoint

---

**Monitor an endpoint**

Multiple Route 53 health checkers will try to establish a TCP connection with the following resource to determine whether it's healthy. [Learn more](#)

**Specify endpoint by** Domain name

**Protocol** HTTP

**Domain name \*** failover-403065813.ap-south-1.elb.amazonaws.com

**Port \*** 80

**Path** /images

### ► Advanced configuration

**URL** <http://failover-403065813.ap-south-1.elb.amazonaws.com:80/>

\* Required

[Cancel](#)

[Save](#)

[Back to Hosted Zones](#)[Create Record Set](#)[Import Zone File](#)[Delete Record Set](#)[Test Record Set](#)

Record Set Name  Any Type  Aliases Only  Weighted Only

Displaying 1 to 4 out of 4 Record Sets

	Name	Type	Value	Evaluate Target Health	Health Check ID	TTL	Region	Weight	Geolocation
<input type="checkbox"/>	cloudrsh.com.	NS	ns-1181.awsdns-19.org. ns-508.awsdns-63.com. ns-1745.awsdns-26.co.uk. ns-611.awsdns-12.net.	-	-	172800			
<input type="checkbox"/>	cloudrsh.com.	SOA	ns-1181.awsdns-19.org. awsdns-hostmaster.amazon	-	-	900			
<input type="checkbox"/>	web.cloudrsh.com.	A	52.66.60.186	-	4fbe48b2-ac9d-4b6f-86af-cf11cb914ce6	300			
<input checked="" type="checkbox"/>	web.cloudrsh.com.	A	ALIAS s3-website.ap-south-1.amazonaws.com. (z11)	No	-				

### Edit Record Set

Name: [web.cloudrsh.com.](#)

Type: A – IPv4 address

Alias:  Yes  No

Alias Target: [s3-website.ap-south-1.amazonaws.co](#)

Alias Hosted Zone ID: Z11RGJOFQNVJUP

You can also type the domain name for the resource. Examples:  
- CloudFront distribution domain name: d111111abcdef8.cloudfront.net  
- Elastic Beanstalk environment CNAME: example.elasticbeanstalk.com  
- ELB load balancer DNS name: example-1.us-east-2.elb.amazonaws.com  
- S3 website endpoint: s3-website.us-east-2.amazonaws.com  
- Resource record set in this hosted zone: www.example.com  
- VPC endpoint: example.us-east-2.vpc.amazonaws.com  
- API Gateway custom regional API: d-abcde12345.execute-api.us-wes2.amazonaws.com

[Learn More](#)

Routing Policy: [Failover](#)

Route 53 responds to queries using primary record sets if any are healthy using secondary record sets otherwise. [Learn More](#)

Failover Record Type:  Primary  Secondary

Set ID: [web-Secondary](#)

Evaluate Target Health:  Yes  No

Associate with Health Check:  Yes  No

[Save Record Set](#)

Search for buckets  All access types

[+ Create bucket](#) [Edit public access settings](#) [Empty](#) [Delete](#) 1 Buckets 1 Regions [Edit](#)

<input type="checkbox"/> Bucket name <a href="#">▼</a>	Access <a href="#">i</a> <a href="#">▼</a>	Region <a href="#">▼</a>	Date created <a href="#">▼</a>
<input type="checkbox"/>  web.cloudrsh.com	Public	Asia Pacific (Mumbai)	Dec 22, 2018 8:42:52 PM GMT+0530

Amazon S3 > web.cloudrsh.com

[Overview](#) [Properties](#) [Permissions](#) [Public](#) [Management](#)

Type a prefix and press Enter to search. Press ESC to clear.

[Upload](#) [+ Create folder](#) [Download](#) [Actions \[▼\]\(#\)](#) As

<input type="checkbox"/> Name <a href="#">▼</a>	Last modified <a href="#">▼</a>	Size <a href="#">▼</a>	Storage class <a href="#">▼</a>
<input type="checkbox"/>  maintenance.html	Dec 22, 2018 8:43:11 PM GMT+0530	165.0 B	Standard

Amazon S3 > web.cloudrsh.com

Overview Properties Permissions Management

Public access settings Access Control List Bucket Policy Public CORS configuration

Bucket policy editor ARN: arn:aws:s3:::web.cloudrsh.com

Type to add a new policy or edit an existing policy in the text area below.

```
1  "Version": "2012-10-17",
2  "Statement": [
3    {
4      "Sid": "PublicReadForGetBucketObjects",
5      "Effect": "Allow",
6      "Principal": "*",
7      "Action": "s3:GetObject",
8      "Resource": "arn:aws:s3:::web.cloudrsh.com/*"
9    }
10  ]
11 ]
12 }
```

Overview    Properties    **Permissions** Public    Management

**Versioning**  
Keep multiple versions of an object in the same bucket.  
[Learn more](#)  
 Disabled

**Server access logging**  
Set up access log records that provide details about access requests.  
[Learn more](#)  
 Disabled

**Static website hosting**

Endpoint : <http://web.cloudrsh.com.s3-website.ap-south-1.amazonaws.com>

Use this bucket to host a website [Learn more](#)

Index document [i](#)

Error document [i](#)

Redirection rules (optional) [i](#)

Redirect requests [Learn more](#)

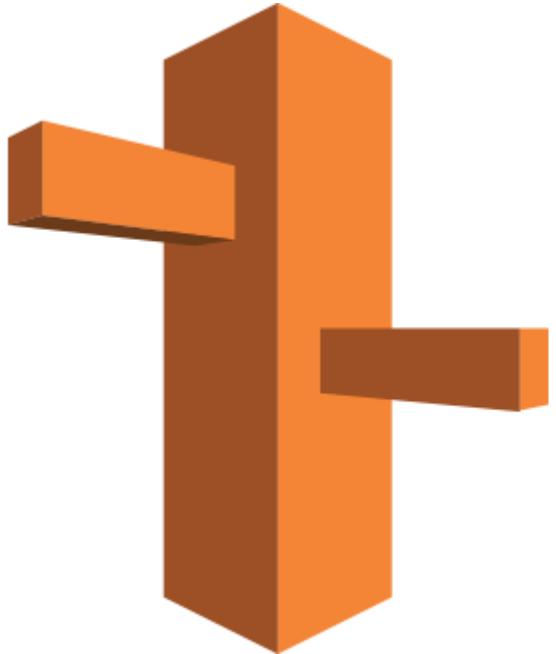
Disable website hosting

[Cancel](#) [Save](#)

**Object-level logging**  
Record object-level API activity using the CloudTrail data events feature (additional cost).  
[Learn more](#)  
 Disabled

COOL!

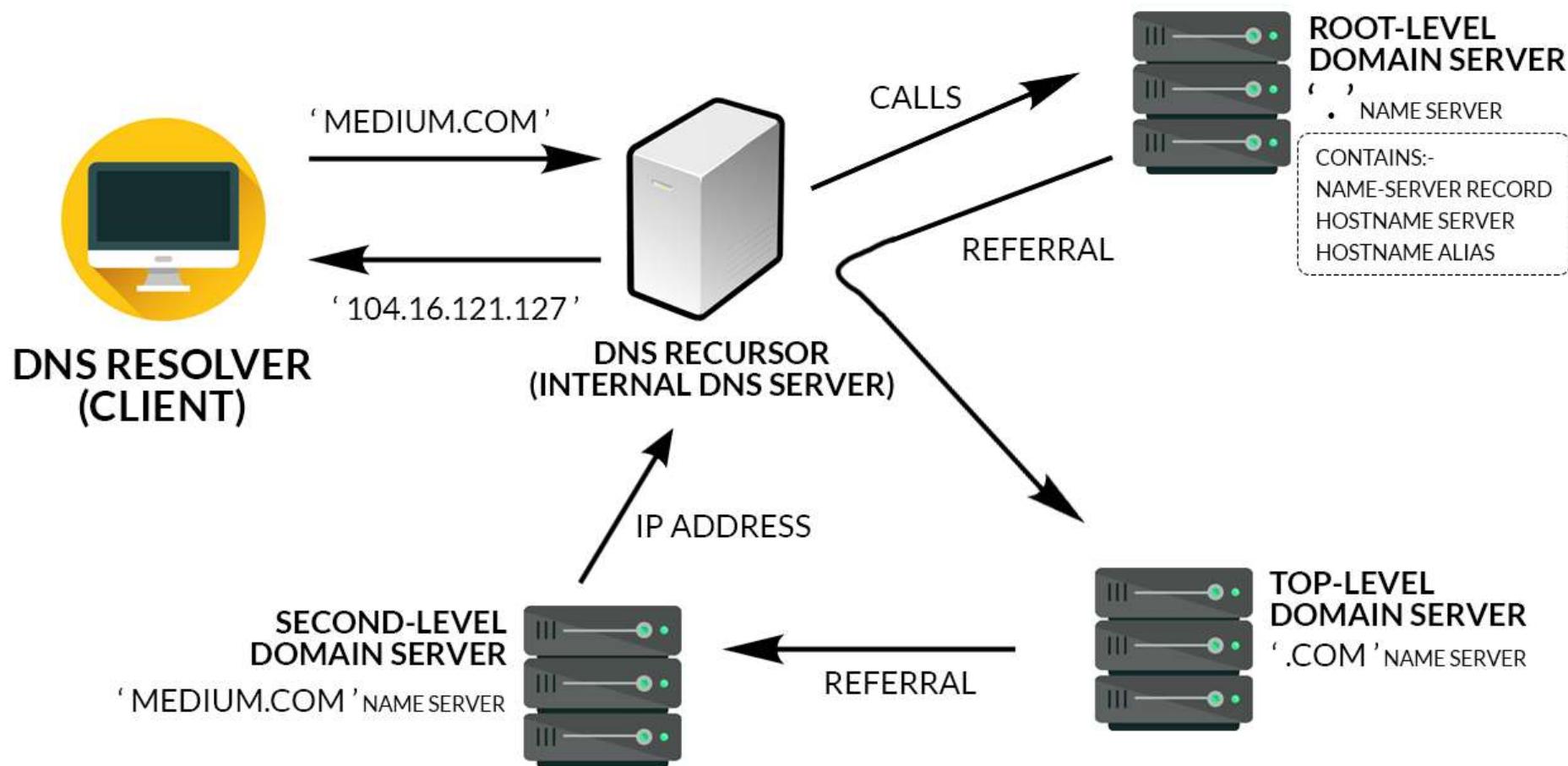




# Route53

By  
Reyaz Shaik

Why only Route53? *Why not any other number?*





# What is Route53

- Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. You can use Route 53 to perform three main functions in any combination: domain registration, DNS routing, and health checking
- Amazon Route 53 is an authoritative Domain Name System (DNS) service. DNS is the system that translates human-readable domain names (example.com) into IP addresses (192.0.2.0). With authoritative name servers in data centers all over the world, Route 53 is reliable, scalable, and fast.

# Route53 – Landing Page



## Amazon Route 53

You can use Amazon Route 53 to register new domains, transfer existing domains, route traffic for your domains to your AWS and external resources, and monitor the health of your resources.



### DNS management

If you already have a domain name, such as example.com, Route 53 can tell the Domain Name System (DNS) where on the Internet to find web servers, mail servers, and other resources for your domain.

[Learn More](#)



### Traffic management

Route 53 traffic flow provides a visual tool that you can use to create and update sophisticated routing policies to route end users to multiple endpoints for your application.

[Learn More](#)



### Availability monitoring

Route 53 can monitor the health and performance of your application as well as your web servers and other resources. Route 53 can also redirect traffic to healthy resources.

[Learn More](#)



### Domain registration

If you need a domain name, you can find an available name and register it by using Route 53. You can also make Route 53 the registrar for existing domains that you registered with other registrars.

[Learn More](#)

[Get started now](#)

[Get started now](#)

[Get started now](#)

[Get started now](#)

# DNS Management

- Hosted Zones
- Health Checks
- Traffic Policies

# DNS Management – Hosted Zones

- A hosted zone is a container for records, and records contain information about how you want to route traffic for a specific domain, such as example.com, and its subdomains (apex.example.com, acme.example.com). A hosted zone and the corresponding domain have the same name. There are two types of hosted zones:
- ***Public hosted zones*** contain records that specify how you want to route traffic on the internet
- ***Private hosted zones*** contain records that specify how you want to route traffic in an Amazon VPC

# DNS Management – Create Hosted Zones

## To create a hosted zone using the Route 53 console

- Sign in to the AWS Management Console and open the Route 53 console at <https://console.aws.amazon.com/route53/>.
- If you're new to Route 53, choose **Get Started Now** under **DNS Management**.
- If you're already using Route 53, choose **Hosted zones** in the navigation pane.
- Choose **Create Hosted Zone**.
- In the **Create Hosted Zone** pane, enter the name of the domain that you want to route traffic for. You can also optionally enter a comment.
- For information about how to specify characters other than a-z, 0-9, and - (hyphen) and how to specify internationalized domain names, see [DNS Domain Name Format](#).
- For **Type**, accept the default value of **Public Hosted Zone**.
- Choose **Create**.

# Working with Records

When you create a record, you choose a routing policy, which determines how Amazon Route 53 responds to queries:

- **Simple routing policy** – Use for a single resource that performs a given function for your domain, for example, a web server that serves content for the example.com website.
- **Failover routing policy** – Use when you want to configure active-passive failover.
- **Geolocation routing policy** – Use when you want to route traffic based on the location of your users.
- **Geoproximity routing policy** – Use when you want to route traffic based on the location of your resources and, optionally, shift traffic from resources in one location to resources in another.
- **Latency routing policy** – Use when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the best latency.
- **Multivalue answer routing policy** – Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.
- **Weighted routing policy** – Use to route traffic to multiple resources in proportions that you specify.

# Supported DNS Records Types

- **A Record Type:** The value for an A record is an IPv4 address in dotted decimal notation
- AAAA Record Type
- CAA Record Type
- **CNAME Record Type:** The value element is the same as the domain name
- MX Record Type
- NAPTR Record Type
- **NS Record Type:** An NS record identifies the name servers for the hosted zone. The value for an NS record is the domain name of a name server. For more information about NS records
- PTR Record Type
- **SOA Record Type:** A start of authority (SOA) record provides information about a domain and the corresponding Amazon Route 53 hosted zone.
- SPF Record Type
- SRV Record Type
- TXT Record Type

# Routing Internet Traffic to Your AWS Resources

You can use Amazon Route 53 to route traffic to a variety of AWS resources.

- Routing Traffic to an Amazon CloudFront Web Distribution by Using Your Domain Name
- *Routing Traffic to an Amazon EC2 Instance*
- *Routing Traffic to an AWS Elastic Beanstalk Environment*
- *Routing Traffic to an ELB Load Balancer*
- Opening Connections to an Amazon RDS Database Instance Using Your Domain Name
- *Routing Traffic to a Website that Is Hosted in an Amazon S3 Bucket*
- Routing Traffic to Amazon WorkMail

# Routing Traffic to an EC2 Instance

# Routing Traffic to Ec2 Instance

- Amazon EC2 provides scalable computing capacity in the AWS cloud. You can launch an EC2 virtual computing environment (an instance) using a preconfigured template (an Amazon Machine Image, or AMI). When you launch an EC2 instance, EC2 automatically installs the operating system (Linux or Microsoft Windows) and additional software included in the AMI, such as web server or database software.
- If you're hosting a website or running a web application on an EC2 instance, you can route traffic for your domain, such as example.com, to your server by using Amazon Route 53.

# Prerequisites

Before you get started, you need the following:

An Amazon EC2 instance. For information about launching an EC2 instance

- Linux or Microsoft Windows

## Important

- We recommend that you also create an Elastic IP address and associate it with your EC2 instance. An Elastic IP address ensures that the IP address of your Amazon EC2 instance will never change.
- A registered domain name. You can use Amazon Route 53 as your domain registrar, or you can use a different registrar.
- Route 53 as the DNS service for the domain. If you register your domain name by using Route 53, AWS will automatically configure Route 53 as the DNS service for the domain.

# Configuring Amazon Route 53 to Route Traffic to an Amazon EC2 Instance

## To route traffic to an Amazon EC2 instance

- Get the IP address for the Amazon EC2 instance:
  - Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
  - In the regions list in the upper right corner of the console, choose the region that you launched the instance in.
  - In the navigation pane, choose **Instances**.
  - In the table, choose the instance that you want to route traffic to.
  - In the bottom pane, on the **Description** tab, get the value of **Elastic IPs**.
  - If you didn't associate an Elastic IP with the instance, get the value of **IPv4 Public IP**.
- Open the Route 53 console at <https://console.aws.amazon.com/route53/>.
- In the navigation pane, choose **Hosted zones**.
- Choose the name of the hosted zone that matches the name of the domain that you want to route traffic for.
- Choose **Create Record Set**.

# Configuring Amazon Route 53 to Route Traffic to an Amazon EC2 Instance

Specify the following values:

## **Name**

Enter the domain name that you want to use to route traffic to your EC2 instance. The default value is the name of the hosted zone. For example, if the name of the hosted zone is example.com and you want to use acme.example.com to route traffic to your EC2 instance, enter **acme**.

## **Type**

Choose **A – IPv4 address**.

## **Alias**

Accept the default value of **No**.

## **TTL (Seconds)**

Accept the default value of **300**.

## **Value**

Enter the IP address that you got in step 1.

## **Routing Policy**

Accept the default value, **Simple**.

2. Choose **Create**.

Changes generally propagate to all Route 53 servers within 60 seconds. When propagation is done, you'll be able to route traffic to your EC2 instance by using the name of the record that you created in this procedure.

Access the DNS name that you had given in the record set

# Advantages of Route53

- Amazon Route 53 effectively connects user requests to infrastructure running in AWS – such as Amazon EC2 instances, Elastic Load Balancing load balancers, or Amazon S3 buckets
- You can use Amazon Route 53 to configure DNS health checks to route traffic to healthy endpoints or to independently monitor the health of your application and its endpoints.
- Amazon Route 53 Traffic Flow makes it easy for you to manage traffic globally through a variety of routing types, including Latency Based Routing, Geo DNS, Geoproximity, and Weighted Round Robin—all of which can be combined with DNS Failover in order to enable a variety of low-latency, fault-tolerant architectures.
- Using Amazon Route 53 Traffic Flow's simple visual editor, you can easily manage how your end-users are routed to your application's endpoints—whether in a single AWS region or distributed around the globe.
- Amazon Route 53 also offers Domain Name Registration – you can purchase and manage domain names such as example.com and Amazon Route 53 will automatically configure DNS settings for your domains.

# DNS Management – Health check

- Route 53 health checks monitor the health and performance of your application's servers, or endpoints, from a network of health checkers in locations around the world. You can specify either a domain name or an IP address and a port to create HTTP, HTTPS, and TCP health checks that check the health of the endpoint. To get started, click **Create health check**.



Dashboard

**Hosted zones**

Health checks

Traffic flow

Traffic policies

Policy records

Domains

Registered domains

Pending requests

Resolver

VPCs

Inbound endpoints

Outbound endpoints

Rules

**Create Hosted Zone**

[Go to Record Sets](#)

[Delete Hosted Zone](#)



Amazon Route 53 is an authoritative Domain Name System (DNS) that translates human-readable domain names (example.com) into IP addresses. For your website or application to work, it needs to connect to servers in data centers all over the world. Route 53 is the easiest way to do this.

If you already have a domain name, such as example.com, Route 53 can tell the Domain Name System (DNS) resources for your domain.

[Learn More](#)

**Create Hosted Zone**

[Create Hosted Zone](#) [Go to Record Sets](#) [Delete Hosted Zone](#) [Help](#) [Feedback](#)

Search all fields  All Types

Domain Name	Type	Record Set Count	Comment	Hosted Zone ID
You have no hosted zones				

**Create Hosted Zone**

A hosted zone is a container that holds information about how you want to route traffic for a domain, such as example.com, and its subdomains.

**Domain Name:**

**Comment:**

**Type:**

A public hosted zone determines how traffic is routed on the Internet.

**Create**

[Back to Hosted Zones](#)[Create Record Set](#)[Import Zone File](#)[Delete Record Set](#)[Test Record Set](#)

	Name	Type	Value	Evaluate Target Health	Health Check ID	TTL	Region	Weight	Geolocation	Multivalue Answer
<input type="checkbox"/>	risenshine.com.	NS	ns-1509.awsdns-60.org. ns-1572.awsdns-04.co.uk. ns-234.awsdns-29.com. ns-892.awsdns-47.net.	-	-	172800				
<input type="checkbox"/>	risenshine.com.	SOA	ns-1509.awsdns-60.org. awsdns-hostmaster.amazon	-	-	900				

**Create Record Set**

**Name:**  .risenshine.com.

**Type:**  A - IPv4 address

**Alias:**  Yes  No

**TTL (Seconds):**  300  1m  5m  1h  1d

**Value:**  198.51.100.234

IPv4 address. Enter multiple addresses on separate lines.

Example:  
192.0.2.235  
198.51.100.234

**Routing Policy:**  Simple

Route 53 responds to queries based only on the values in this record. [Learn More](#)

# Configuring ELB to Route53

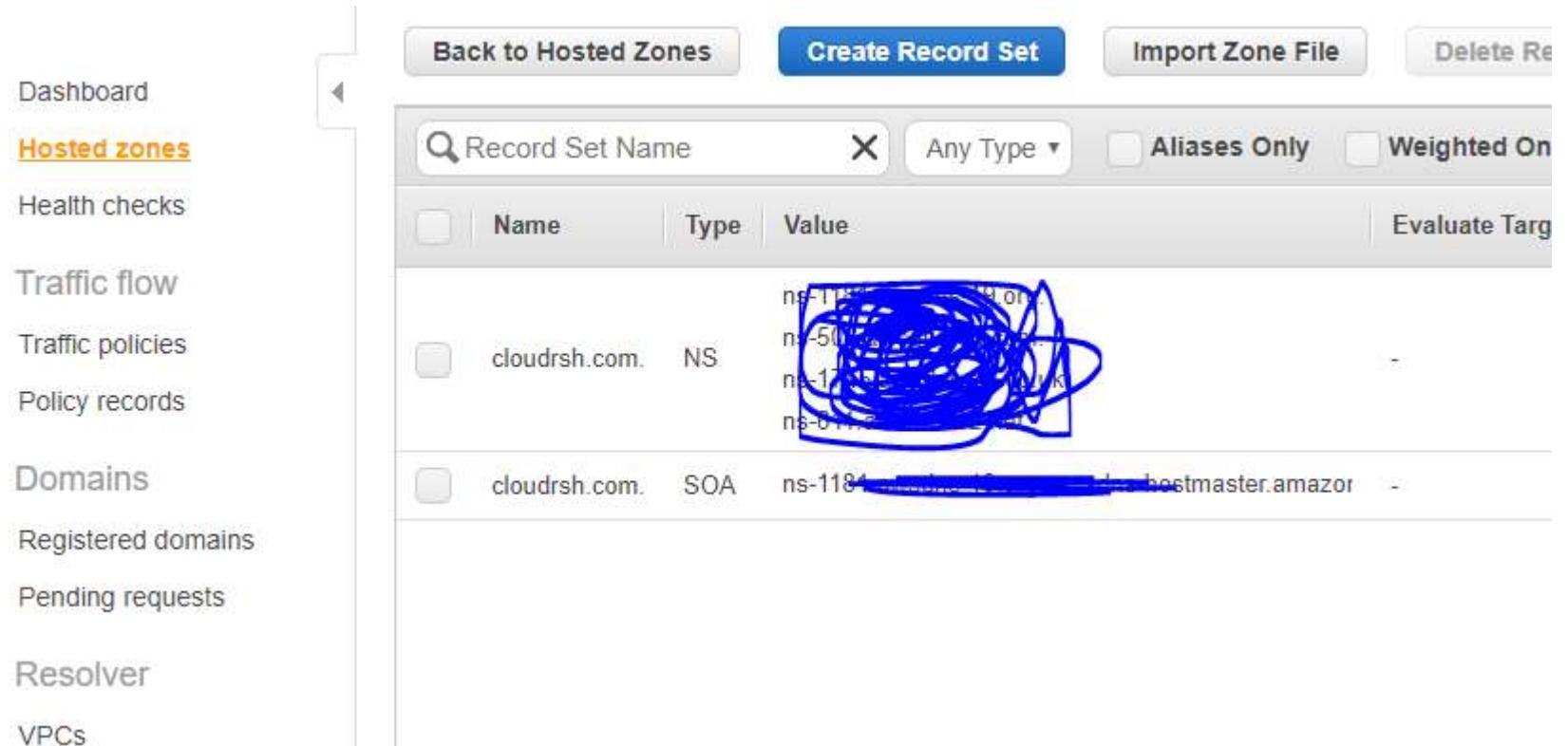
- Assume you have EC2 Instance running with apache and with sample website
- You normally access it through the Public IP address of the instance.



- Usually its very difficult to remember all the IP address of the EC2 instances
- Instead let us create a route53 record to access it through the hostname

# Configuring ELB to Route53

- Go to Route53
- Create Record Set



The screenshot shows the AWS Route 53 Hosted Zones interface. The left sidebar lists navigation options: Dashboard, Hosted zones (which is selected and highlighted in orange), Health checks, Traffic flow, Traffic policies, Policy records, Domains, Registered domains, Pending requests, Resolver, and VPCs. The main content area is titled 'Create Record Set' and shows a table of existing record sets for the domain 'cloudrsh.com.'. The table has columns for 'Name', 'Type', and 'Value'. One NS record for 'ns-1184' has its value redacted with a blue scribble. The table also shows an SOA record for 'ns-1184'. At the top of the main area are buttons for 'Back to Hosted Zones', 'Create Record Set' (which is blue and bold), 'Import Zone File', and 'Delete Re' (partially visible).

	Name	Type	Value	Evaluate Targ
<input type="checkbox"/>	cloudrsh.com.	NS	ns-1184-1.cloudrsh.com. ns-501-1.cloudrsh.com. ns-17-1.cloudrsh.com. ns-6-177-1.cloudrsh.com.	-
<input type="checkbox"/>	cloudrsh.com.	SOA	ns-1184-1.cloudrsh.com. hostmaster.amazon	-

# Configuring ELB to Route53

- Name: mywebsite
- Type: A- IPV4
- Value: EC2 Instance public IP

Create Record Set

Name: mywebsite.cloudrsh.com.

Type: A – IPv4 address

Alias:  Yes  No

TTL (Seconds): 300 1m 5m 1h 1d

Value: 13.127.147.74

IPv4 address. Enter multiple addresses on separate lines.

Example:  
192.0.2.235  
198.51.100.234

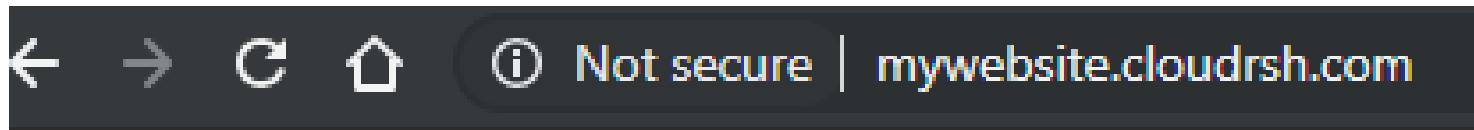
Back to Hosted Zones Create Record Set Import Zone File Delete Record Set Test Record Set Routing Policy: Simple

Record Set Name Any Type Aliases Only Weighted Only

Route 53 responds to queries based only on the values in this record. [Learn More](#)

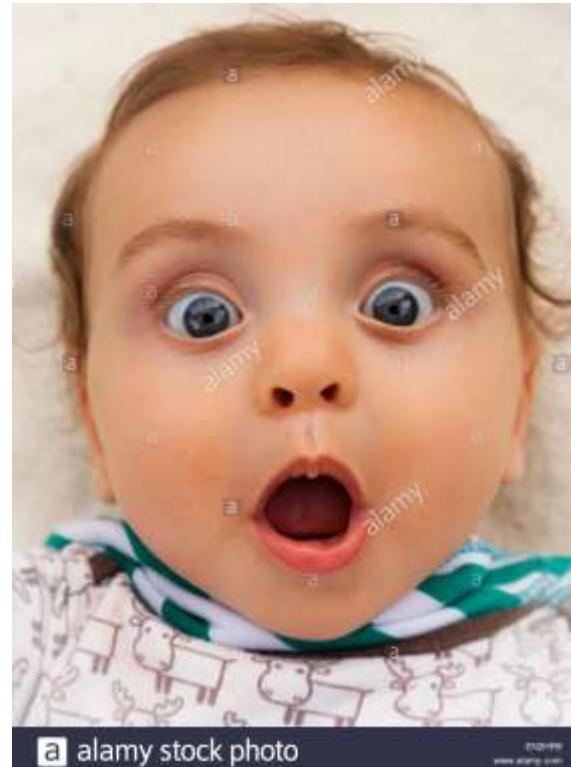
<input type="checkbox"/>	Name	Type	Value	Evaluate Target Health	Health Check ID	TTL	Regi
<input type="checkbox"/>	cloudrsh.com.	NS	ns-1[REDACTED].com. ns-5[REDACTED].com. ns-4[REDACTED].com.uk. ns-2[REDACTED].net.	-	-	172800	
<input type="checkbox"/>	cloudrsh.com.	SOA	ns-1[REDACTED].[REDACTED].amazon[REDACTED]	-	-	900	
<input checked="" type="checkbox"/>	mywebsite.cloudrsh.com.	A	13.127.147.74	-	-	300	

# Now Test <http://mywebsite.cloudrsh.com>



This is the Main website

## IT WORKED!!!!

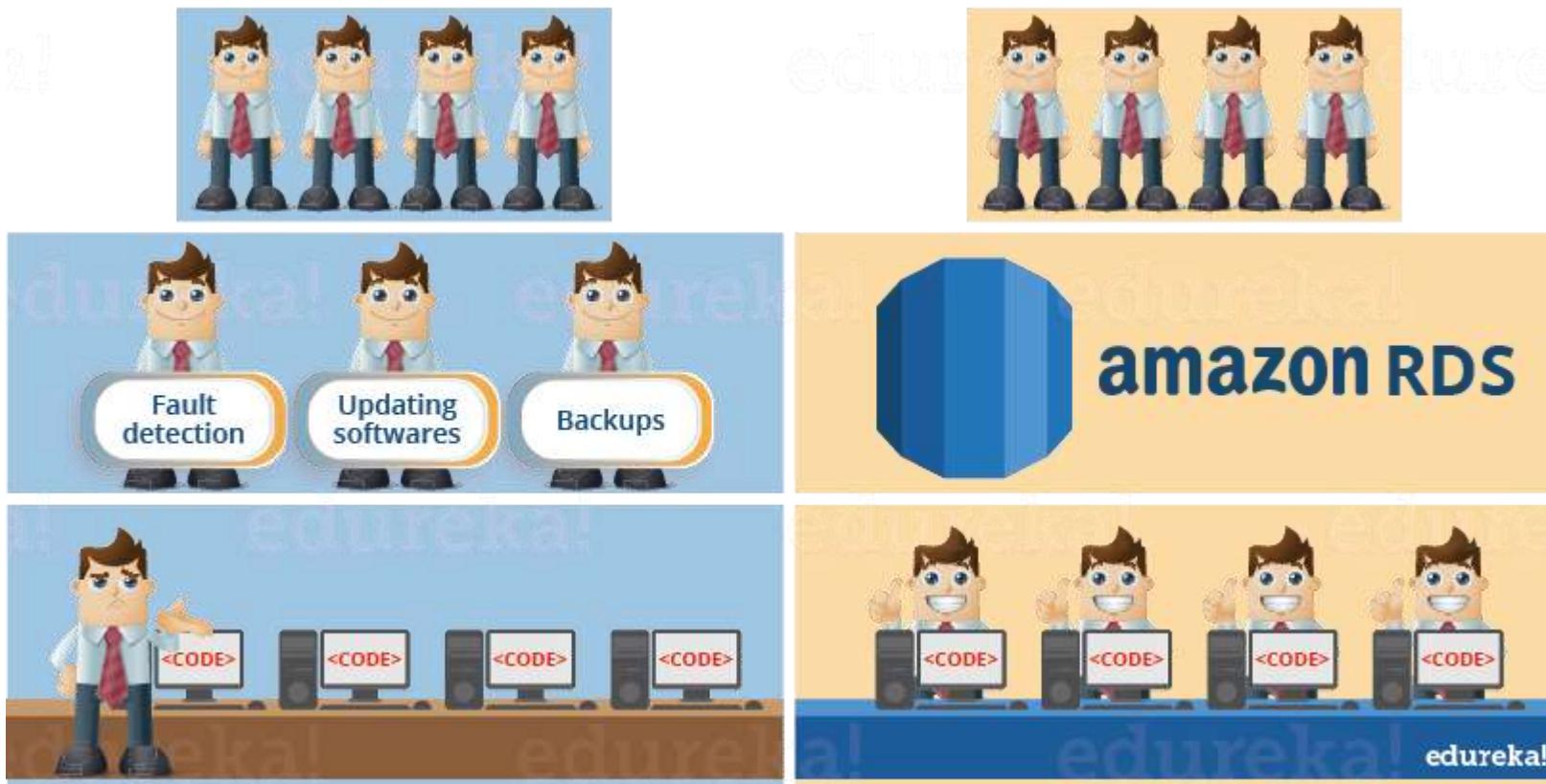








# Amazon RDS



# Amazon RDS



- Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.
- So people often develop a misconception, when they confuse RDS with a database.
- RDS is not a database, it's a service that manages databases, having said that, let's discuss the databases that RDS can manage as of now

# Amazon RDS DB Engines



## Engine options

Amazon Aurora

Amazon  
**Aurora**

MySQL



MariaDB



PostgreSQL



Oracle

**ORACLE**

Microsoft SQL Server

 Microsoft  
**SQL Server**

# Amazon RDS



- **Amazon Aurora:**

It is a relational database engine made by amazon which combines the speed and reliability of high-end commercial databases with the simplicity and cost-effectiveness of open source databases. Amazon claims that Aurora is 5x faster than RDS MySQL

- **PostgreSQL:**

PostgreSQL is yet another open source database management system which uses SQL to access the data

- **Oracle:**

It is object-relational database management system which was developed by Oracle Inc

- **MySql**

It is an open source database management system which uses SQL (Structured Query Language) to access the data stored in its system

- **SQL SERVER**

SQL Server is a Relational Database Management System, which was developed by Microsoft in 2005 for the enterprise environment.

- **Maria DB**

MariaDB is a community developed fork of MySQL DBMS. The reason for its fork, was the concern over the acquisition of Oracle over MySQL

# Overview of Amazon RDS



Why do you want a managed relational database service? Because Amazon RDS takes over many of the difficult or tedious management tasks of a relational database:

- When you buy a server, you get CPU, memory, storage, and IOPS, all bundled together. With Amazon RDS, these are split apart so that you can scale them independently. If you need more CPU, less IOPS, or more storage, you can easily allocate them.
- Amazon RDS manages backups, software patching, automatic failure detection, and recovery.
- To deliver a managed service experience, Amazon RDS doesn't provide shell access to DB instances, and it restricts access to certain system procedures and tables that require advanced privileges.
- You can have automated backups performed when you need them, or manually create your own backup snapshot. You can use these backups to restore a database. The Amazon RDS restore process works reliably and efficiently.
- You can get high availability with a primary instance and a synchronous secondary instance that you can fail over to when problems occur. You can also use MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.
- You can use the database products you are already familiar with: MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server.
- In addition to the security in your database package, you can help control who can access your RDS databases by using AWS Identity and Access Management (IAM) to define users and permissions. You can also help protect your databases by putting them in a virtual private cloud.

# RDS AWS Components

- DB Instances
- Regions and Availability Zones
- Security Groups
- DB Parameter Groups
- DB Option Groups

# DB Instance

- They are the building blocks of RDS. It is an isolated database environment in the cloud, which can contain multiple user-created databases, and can be accessed using the same tools and applications that one uses with a stand-alone database instance.
- A DB Instance can be created using the AWS Management Console , the Amazon RDS API, or the AWS Command line Interface .
- The computation and memory capacity of a DB Instance depends on the DB Instance class. For each DB Instance you can select from 5GB to 6TB of associated storage capacity.
- The DB Instances are of the following types:
  - Standard Instances (m4,m3)
  - Memory Optimised (r3)
  - Micro Instances (t2)

# Regions and Availability Zones

- The AWS resources are housed in highly available data centers, which are located in different areas of the world. This “area” is called a region.
- Each region has multiple Availability Zones (AZ), they are distinct locations which are engineered to be isolated from the failure of other AZs.
- You can deploy your DB Instance in multiple AZ, this ensures a failover i.e. in case one AZ goes down, there is a second to switch over to. The failover instance is called a standby, and the original instance is called the primary instance.

# Security Groups

- A security group controls the access to a DB Instance. It does so by specifying a range of IP addresses or the EC2 instances that you want to give access.
- Amazon RDS uses 3 types of Security Groups:
- VPC Security Group
  - It controls the DB Instance that is inside a VPC.
- EC2 Security Group
  - It controls access to an EC2 Instance and can be used with a DB Instance.
- DB Security Group
  - It controls the DB Instance that is not in a VPC.

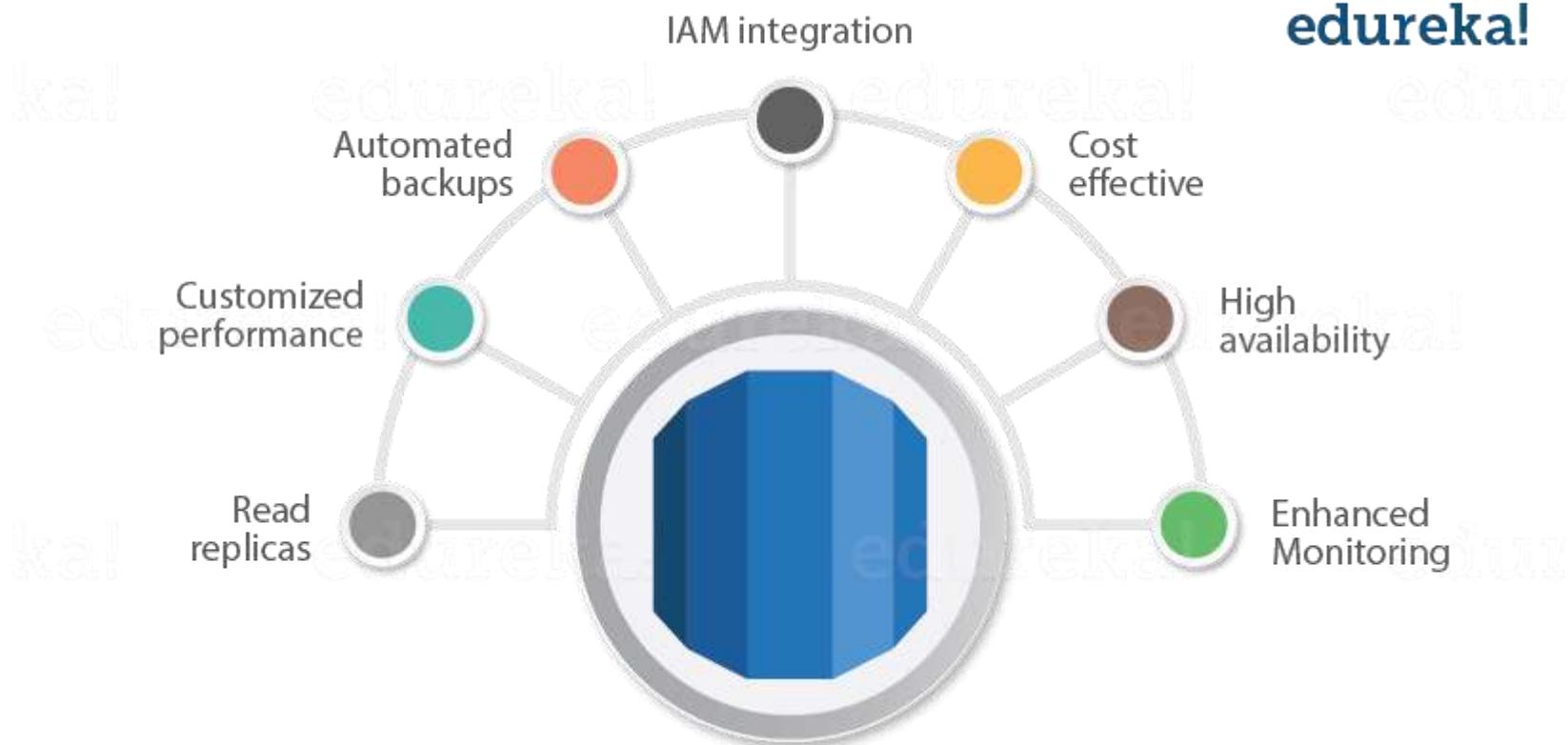
## **DB Parameter groups**

- It contains the engine configuration values that can be applied to one or more DB Instances of the same instance type.
- If you don't apply a DB Parameter group to your instance, you are assigned a default Parameter group which has the default values.

## **DB Option groups**

- Some DB engines offer tools that simplify managing your databases.
- RDS makes these tools available with the use of Option groups.

# RDS Advantages

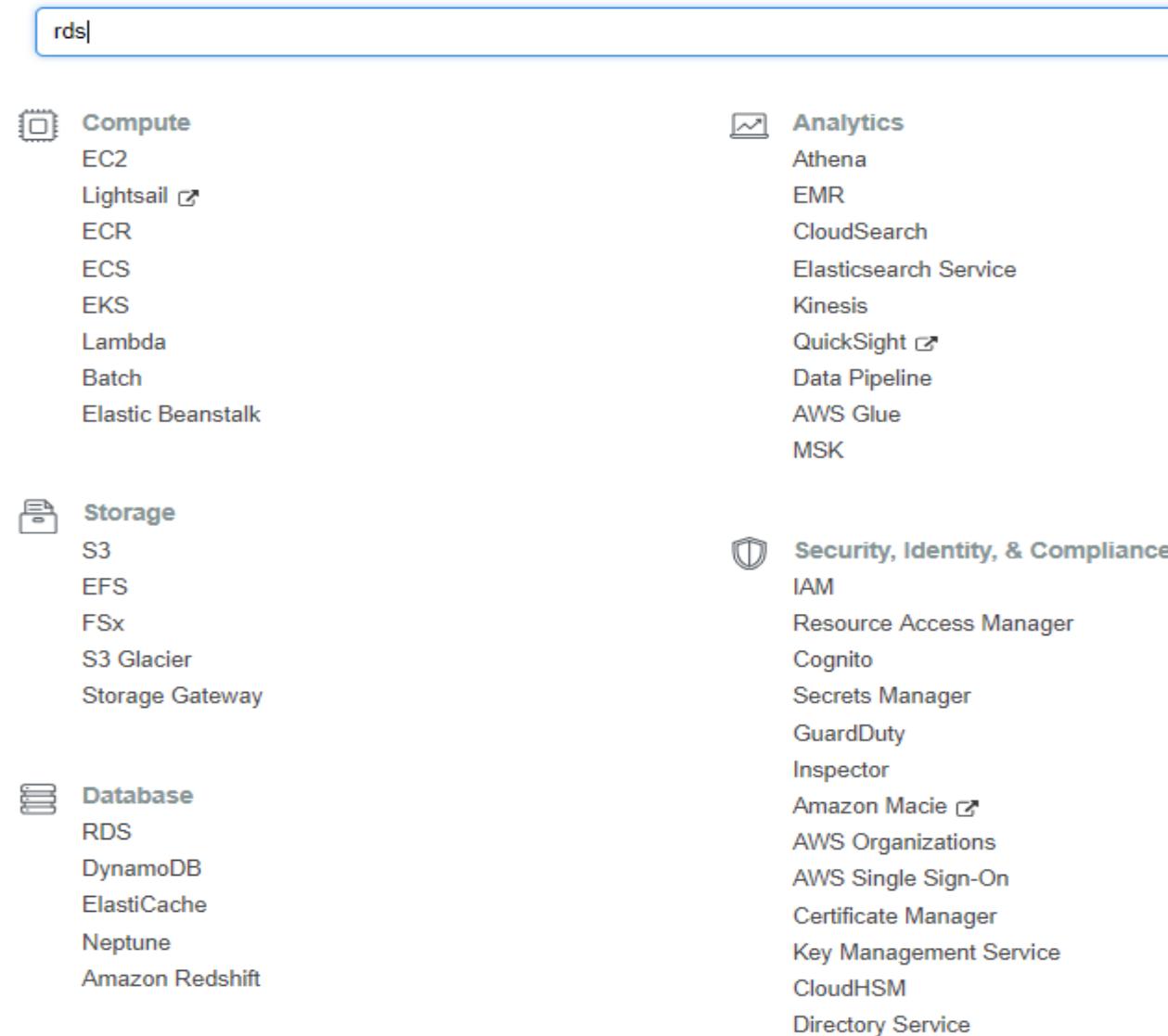


# Demo



# Hands- ON

First select RDS Service from the AWS management console



The screenshot shows the AWS Management Console search bar at the top with the text "rds" typed in. Below the search bar is a list of AWS services categorized into groups. The visible categories and their services are:

- Compute**: EC2, Lightsail, ECR, ECS, EKS, Lambda, Batch, Elastic Beanstalk
- Analytics**: Athena, EMR, CloudSearch, Elasticsearch Service, Kinesis, QuickSight, Data Pipeline, AWS Glue, MSK
- Storage**: S3, EFS, FSx, S3 Glacier, Storage Gateway
- Security, Identity, & Compliance**: IAM, Resource Access Manager, Cognito, Secrets Manager, GuardDuty, Inspector, Amazon Macie, AWS Organizations, AWS Single Sign-On, Certificate Manager, Key Management Service, CloudHSM, Directory Service
- Database**: RDS, DynamoDB, ElastiCache, Neptune, Amazon Redshift

# RDS Dashboard

## Create Database

Amazon RDS

[Dashboard](#)

[Databases](#)

[Performance Insights](#)

[Snapshots](#)

[Automated backups](#)

[Reserved instances](#)

[Security groups](#)

[Subnet groups](#)

[Parameter groups](#)

[Option groups](#)

[Events](#)

[Event subscriptions](#)

Recommendations (9)

**Amazon Aurora**  
Amazon Aurora is a MySQL- and PostgreSQL-compatible enterprise-class database, starting at <\$1/day. Aurora supports up to 64TB of auto-scaling storage capacity, 6-way replication, and more.

[Create database](#)

Or, [Restore Aurora DB cluster from S3](#)

**Resources**

Refresh

You are using the following Amazon RDS resources in the EU West (Ireland) region (used/quota)

DB Instances (4/40)	Parameter groups (8)
Allocated storage (320.00 GB/100.00 TB)	Default (7)
<a href="#">Click here to increase DB instances limit</a>	Custom (1/100)
Reserved instances (0/40)	Option groups (6)
Snapshots (98)	Default (6)
Manual (28/100)	Custom (0/20)
Automated (33)	Subnet groups (2/50)
Recent events (12)	Supported platforms EC2,VPC
Event subscriptions (0/20)	Default network none
DB Security groups (1/25)	

**Create database**

Amazon Relational Database Service (RDS) makes it easy to set up, operate, and scale a relational database in the cloud.

[Restore from S3](#) [Create database](#)

Note: your DB instances will launch in the EU West (Ireland) region

**Service health**

[View service health dashboard](#)

Current status

Details

# Select Engine

For Example, Select MySQL  
Click Next

## Select engine

### Engine options

Amazon Aurora

Amazon  
Aurora

MySQL



MariaDB



PostgreSQL



Oracle

ORACLE

Microsoft SQL Server

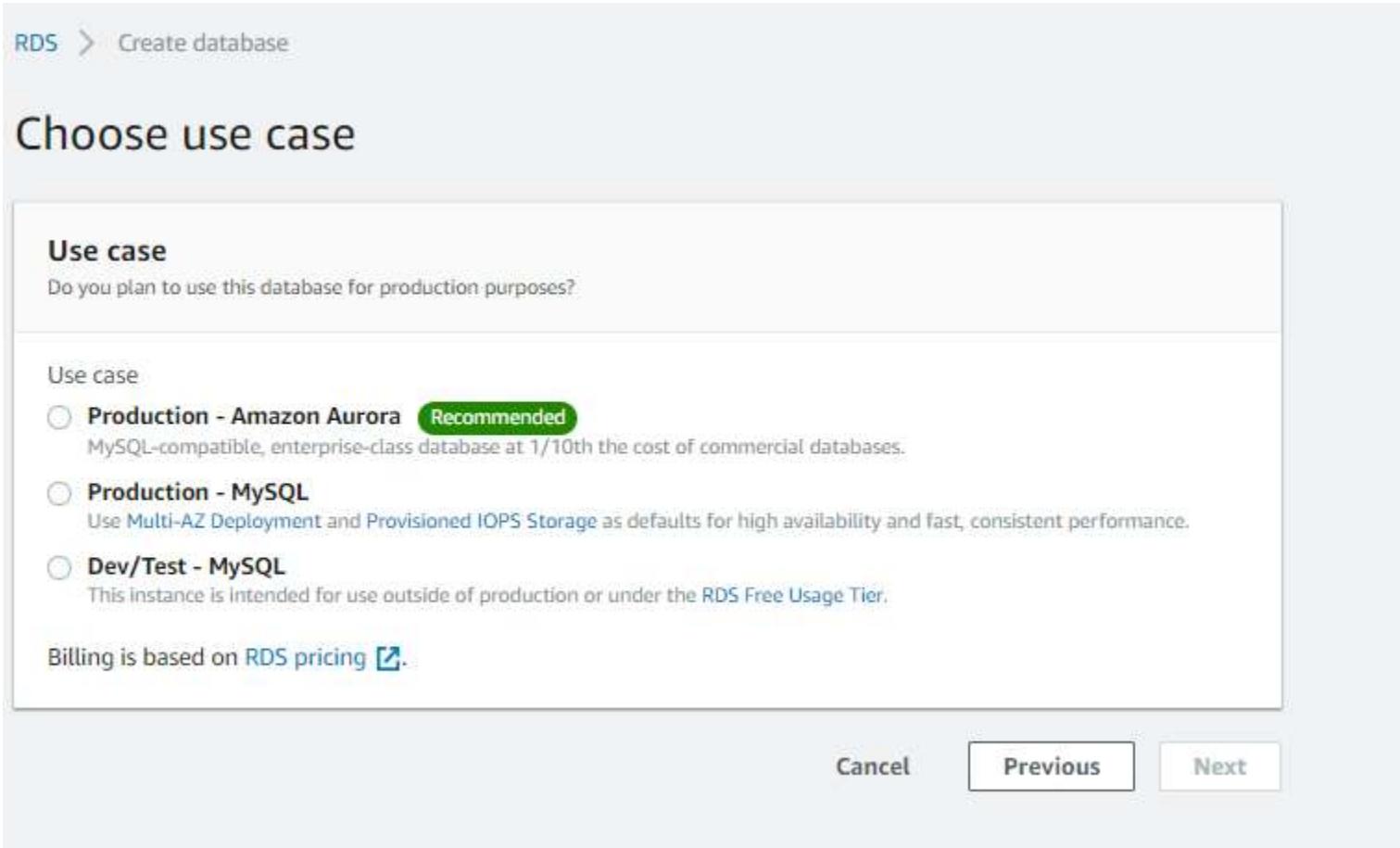


### MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 32 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 5 Read Replicas per instance, within a single Region or cross-region.

# Use-Case



RDS > Create database

## Choose use case

**Use case**  
Do you plan to use this database for production purposes?

Use case

- Production - Amazon Aurora** Recommended  
MySQL-compatible, enterprise-class database at 1/10th the cost of commercial databases.
- Production - MySQL**  
Use Multi-AZ Deployment and Provisioned IOPS Storage as defaults for high availability and fast, consistent performance.
- Dev/Test - MySQL**  
This instance is intended for use outside of production or under the RDS Free Usage Tier.

Billing is based on [RDS pricing](#).

Cancel Previous Next

# Specify DB Details

## Specify DB details

### Instance specifications

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#)

DB engine

MySQL Community Edition

License model [Info](#)

general-public-license

DB engine version [Info](#)

MySQL 5.6.40



#### Known Issues/Limitations

Review the [Known Issues/Limitations](#) to learn about potential compatibility issues with specific database versions.



#### Free tier

The Amazon RDS Free Tier provides a single db.t2.micro instance as well as up to 20 GiB of storage, allowing new AWS customers to gain hands-on experience with Amazon RDS. Learn more about the RDS Free Tier and the instance restrictions [here](#).

Only enable options eligible for RDS Free Usage Tier [Info](#)

DB instance class [Info](#)

db.r4.xlarge — 4 vCPU, 30.5 GiB RAM

Multi-AZ deployment [Info](#)

Create replica in different zone

Creates a replica in a different Availability Zone (AZ) to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.

No

# Storage Type, Size, DB Credentials

Storage type [Info](#)  
 ▼

Allocated storage  
 GiB  
(Minimum: 20 GiB, Maximum: 32768 GiB) Higher allocated storage may improve IOPS performance.

Provisioning less than 100 GiB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. [Click here](#) for more details.

**Estimated monthly costs**

DB Instance	386.90 USD
Storage	2.54 USD
<b>Total</b>	<b>389.44 USD</b>

Billing estimate is based on on-demand usage as described in [Amazon RDS Pricing](#). Estimate does not include costs for backup storage, IOs (if applicable), or data transfer.

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#)

**Settings**

DB instance identifier [Info](#)  
Specify a name that is unique for all DB instances owned by your AWS account in the current region.

DB instance identifier is case insensitive, but stored as all lower-case, as in "mydbinstance". Must contain from 1 to 63 alphanumeric characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Cannot end with a hyphen or contain two consecutive hyphens.

Master username [Info](#)  
Specify an alphanumeric string that defines the login ID for the master user.

Master Username must start with a letter. Must contain 1 to 16 alphanumeric characters.

Master password [Info](#)  Confirm password [Info](#)

# Advance Settings

RDS > Create database

## Configure advanced settings

### Network & Security

Virtual Private Cloud (VPC) [Info](#)  
VPC defines the virtual networking environment for this DB instance.

[C](#)

Only VPCs with a corresponding DB subnet group are listed.

Subnet group [Info](#)  
DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.

[C](#)

Public accessibility [Info](#)

Yes  
EC2 instances and devices outside of the VPC hosting the DB instance will connect to the DB instances. You must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.

No  
DB instance will not have a public IP address assigned. No EC2 instance or devices outside of the VPC will be able to connect.

Availability zone [Info](#)

[C](#)

VPC security groups

Security groups have rules authorizing connections from all the EC2 instances and devices that need to access the DB instance.

Create new VPC security group

Choose existing VPC security groups

[C](#)

[X](#)

# DB Options

## Database options

Database name [Info](#)

dbname

Note: If no database name is specified then no initial MySQL database will be created on the DB Instance.

Port [Info](#)

TCP/IP port the DB instance will use for application connections.

3306

DB parameter group [Info](#)

default:mysql5.6



Option group [Info](#)

default:mysql-5-6



IAM DB authentication [Info](#)

Enable IAM DB authentication

Manage your database user credentials through AWS IAM users and roles.

Disable

## Encryption

Encryption

Enable encryption [Learn more](#)

Select to encrypt the given instance. Master key ids and aliases appear in the list after they have been created using the Key Management Service(KMS) console.

Disable encryption

## Backup



Please note that automated backups are currently supported for InnoDB storage engine only. If you are using MyISAM, refer to detail [here](#).

# Backup Monitoring

### Backup

**⚠** Please note that automated backups are currently supported for InnoDB storage engine only. If you are using MyISAM, refer to detail here. [\[x\]](#)

**Backup retention period** [Info](#)  
Select the number of days that Amazon RDS should retain automatic backups of this DB instance.  
 [▼](#)

**Backup window** [Info](#)  
 Select window  
 No preference  
 Copy tags to snapshots

### Monitoring

**Enhanced monitoring**  
 Enable enhanced monitoring  
Enhanced monitoring metrics are useful when you want to see how different processes or threads use the CPU.  
 Disable enhanced monitoring

### Log exports

Select the log types to publish to Amazon CloudWatch Logs

Audit log  
 Error log  
 General log  
 Slow query log

**IAM role**  
The following service-linked role is used for publishing logs to CloudWatch Logs.  
[RDS Service Linked Role](#)

# Create Database

The screenshot shows the 'Create Database' wizard interface. The 'Maintenance' section is currently active, displaying options for auto minor version upgrade and maintenance window. The 'Deletion protection' section is shown below. At the bottom, there are 'Cancel', 'Previous', and 'Create database' buttons.

**Maintenance**

Auto minor version upgrade [Info](#)

Enable auto minor version upgrade  
Enables automatic upgrades to new minor versions as they are released. The automatic upgrades occur during the maintenance window for the DB instance.

Disable auto minor version upgrade

Maintenance window [Info](#)

Select the period in which you want pending modifications or patches applied to the DB instance by Amazon RDS.

Select window

No preference

**Deletion protection**

Enable deletion protection  
Protects the database from being deleted accidentally. While this option is enabled, you can't delete the database.

Cancel Previous **Create database**

## Connect using EndPoint

COOL!

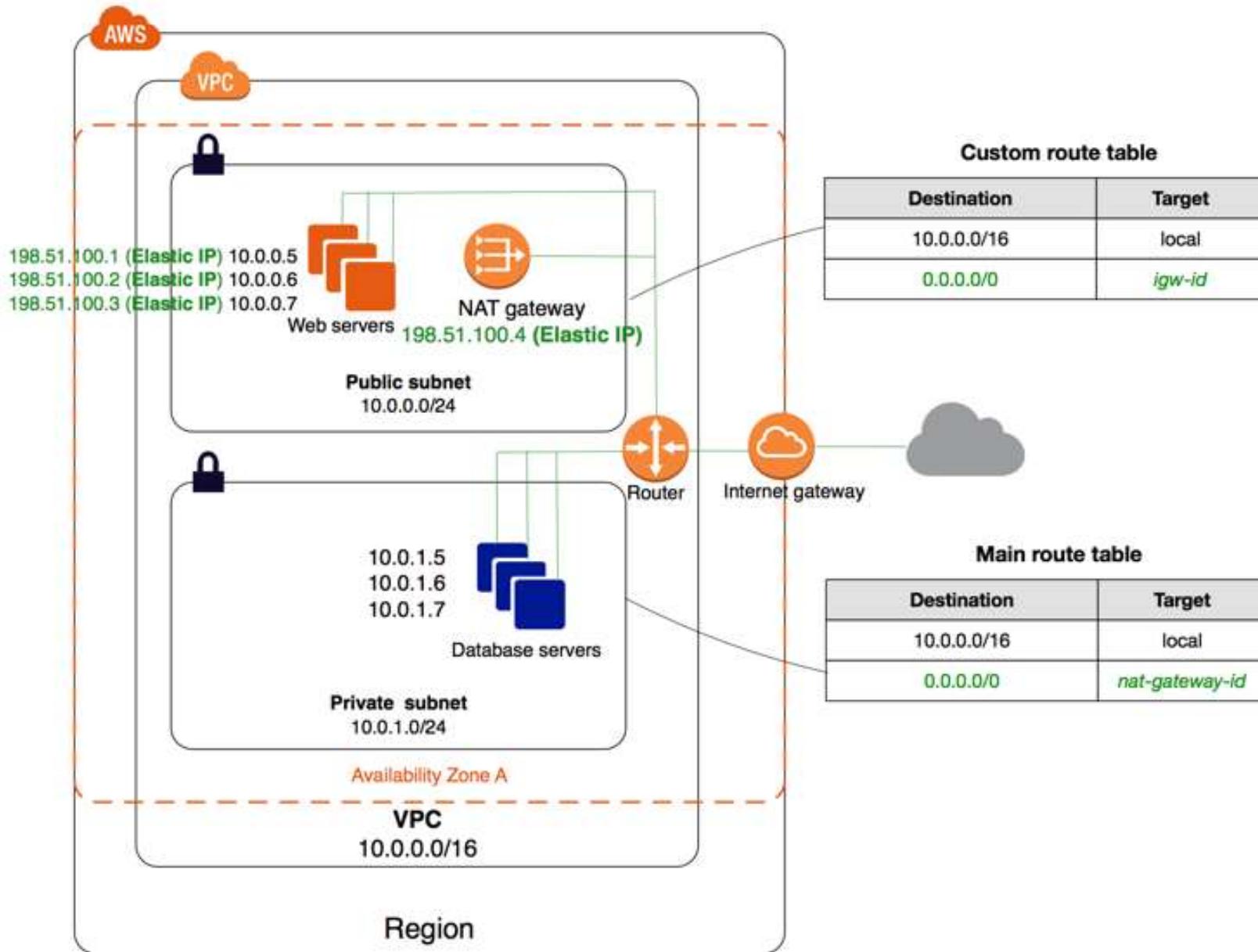




By  
Reyaz Shaik

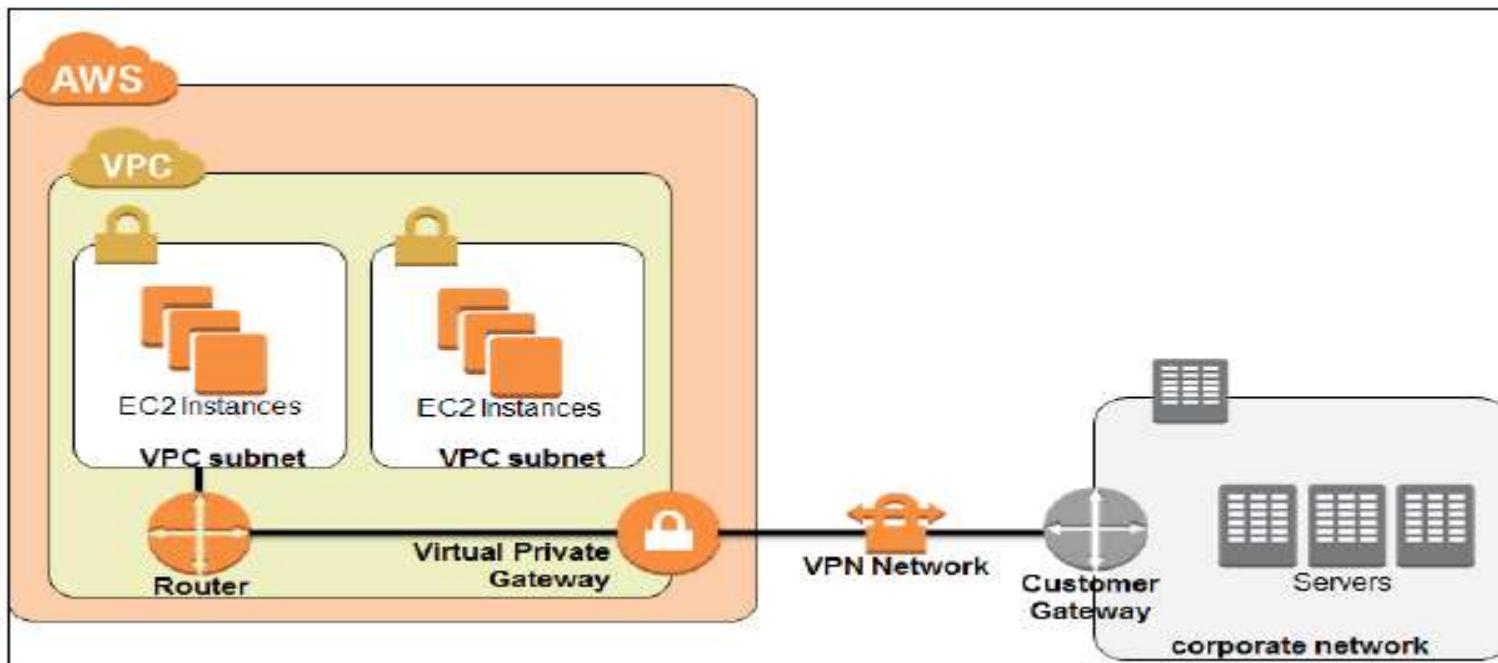
# What is VPC?

- Amazon VPC is your own private network inside Amazon's cloud infrastructure.
- It is an alternative to maintaining your own data center and is cheaper since it creates resources on demand.
- It is also more secure since Amazon takes care of the infrastructure security for you.



# VPC and VPN

- VPCs also provide an added functionality using which you can connect and extend your on-premise datacenters to the AWS cloud.
- This is achieved using an IPsec VPN tunnel that connects from your on-premise datacenter's gateway device to the VPC's **Virtual Private Gateway**



# VPC concepts and terminologies

- VPC is nothing more than a network service provided by AWS using which you can create logically isolated environments for your EC2 instances.
  - **CIDR Block**
  - **Subnets**
  - **Security groups and network ACLs**
  - **Routing tables**
  - **Internet Gateways**
  - **NAT instances/Gateways**

# CIDR

- CIDR or Classless Inter-Domain Routing is used to allocate IP address within a network.
- We will use CIDR blocks to mark a range of IP addresses for each subnet within a VPC.
- The VPC itself would have a CIDR block that lists all the IP addresses available with it.

# CIDR

VPC Dashboard

Filter by VPC:

Select a VPC

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Create VPC Actions

Filter by tags and attributes or search by keyword

<input type="checkbox"/>	Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
<input type="checkbox"/>	vpc-974efbed	available	172.31.0.0/16	-	

Previous

# Subnets

- The subnets are nothing more than a range of valid IP addresses that you specify.
- VPC provides you with two different subnet creation options: a publically or Internet routed subnet called as a

**Public subnet** and an isolated subnet called as a **private subnet**.

# Subnets

VPC Dashboard

Create subnet Actions

Filter by VPC:

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

The screenshot shows the AWS VPC Subnets dashboard. At the top, there are buttons for 'Create subnet' and 'Actions'. Below that is a search bar with the placeholder 'Filter by tags and attributes or search by keyword'. The main area is a table with the following columns: Name, Subnet ID, State, VPC, IPv4 CIDR, Available IPv4, IPv6 CIDR, Availability Zone, and Availability Zone ID. The table lists six subnets, all of which are currently 'available'. The VPC for all subnets is 'vpc-974efbed'. The IPv4 CIDR blocks are 172.31.32.0/20, 172.31.16.0/20, 172.31.0.0/20, 172.31.80.0/20, 172.31.48.0/20, and 172.31.64.0/20. Each subnet has 4091 available IPv4 addresses. The IPv6 CIDR column shows '-' for all subnets. The Availability Zone and Availability Zone ID columns show the following values: us-east-1a, use1-az6; us-east-1d, use1-az4; us-east-1b, use1-az1; us-east-1c, use1-az2; us-east-1f, use1-az5; and us-east-1e, use1-az3.

	Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 CIDR	Availability Zone	Availability Zone ID
<input type="checkbox"/>	subnet-0635505a		available	vpc-974efbed	172.31.32.0/20	4091	-	us-east-1a	use1-az6
<input type="checkbox"/>	subnet-655d752f		available	vpc-974efbed	172.31.16.0/20	4091	-	us-east-1d	use1-az4
<input type="checkbox"/>	subnet-7822bb1f		available	vpc-974efbed	172.31.0.0/20	4091	-	us-east-1b	use1-az1
<input type="checkbox"/>	subnet-ab315685		available	vpc-974efbed	172.31.80.0/20	4091	-	us-east-1c	use1-az2
<input type="checkbox"/>	subnet-b00557bf		available	vpc-974efbed	172.31.48.0/20	4091	-	us-east-1f	use1-az5
<input type="checkbox"/>	subnet-fba70bc5		available	vpc-974efbed	172.31.64.0/20	4091	-	us-east-1e	use1-az3

# Subnets

VPC Dashboard

Filter by VPC:

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Endpoint Services

NAT Gateways

Peering Connections

Security

Network ACLs

Security Groups

Virtual Private Network (VPN)

**Create subnet** **Actions**

Filter by tags and attributes or search by keyword

Name	Subnet ID	State	VPC	IPv4 CIDR	Available IP
subnet-0635505a	available	vpc-974efbed	172.31.32.0/20	4091	
subnet-655d752f	available	vpc-974efbed	172.31.16.0/20	4091	
subnet-7822bb1f	available	vpc-974efbed	172.31.0.0/20	4091	
subnet-ab315685	available	vpc-974efbed	172.31.80.0/20	4091	
subnet-b00557bf	available	vpc-974efbed	172.31.48.0/20	4091	
subnet-fba70bc5	available	vpc-974efbed	172.31.64.0/20	4091	

Subnet: subnet-0635505a

Description Flow Logs **Route Table** Network ACL Tags Sharing

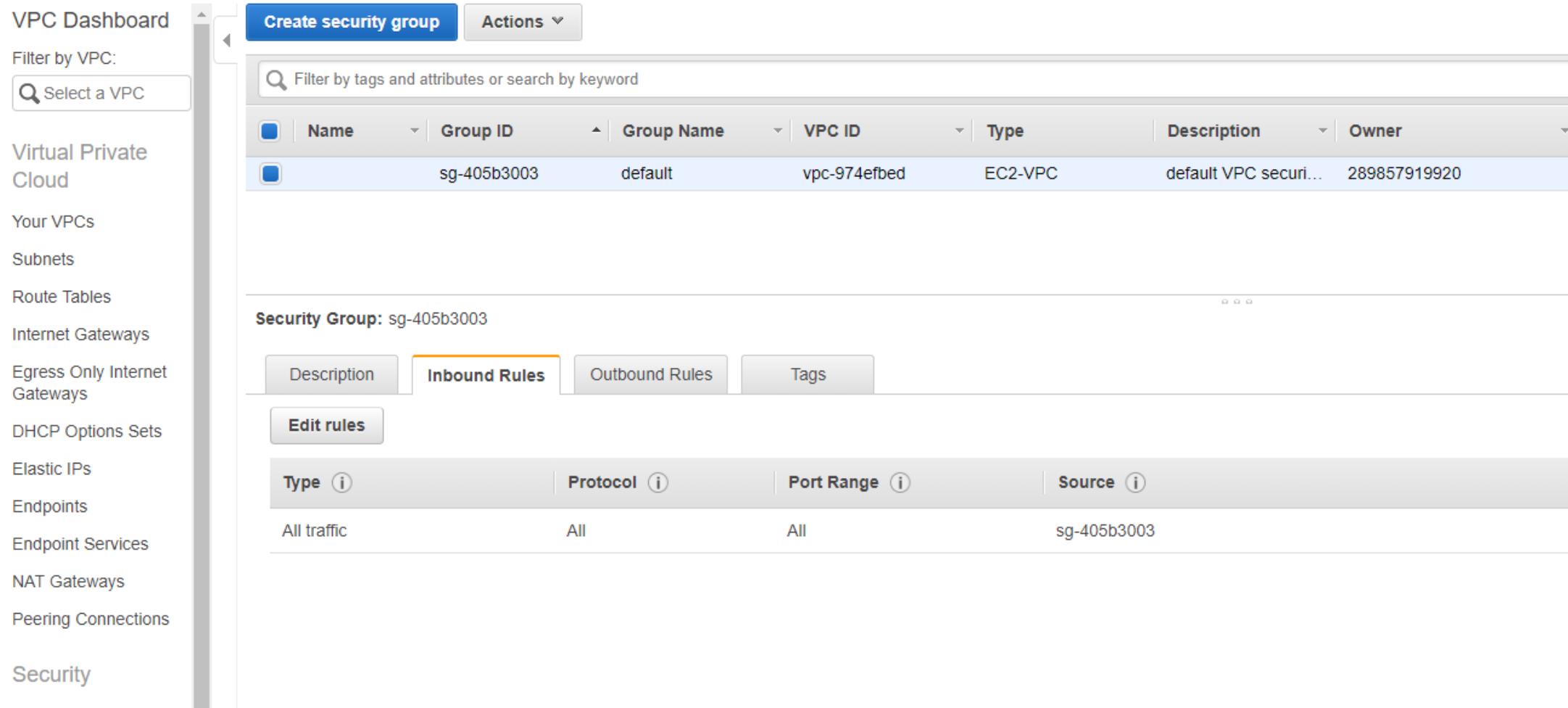
**Edit route table association**

Route Table: rtb-2dc2f252

1 to 2 of 2

Destination	Target
172.31.0.0/16	local
0.0.0.0/0	igw-af2038d7 <b>OR NAT</b>

# Security groups - Inbound



The screenshot shows the AWS VPC Dashboard with the 'Security' section selected. The main content area displays the 'Inbound Rules' tab for the security group 'sg-405b3003'. The table shows a single rule allowing all traffic on all ports from the source 'sg-405b3003'.

**VPC Dashboard**

**Filter by VPC:** Select a VPC

**Virtual Private Cloud**

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Endpoint Services

NAT Gateways

Peering Connections

Security

**Create security group** Actions ▾

Filter by tags and attributes or search by keyword

Name	Group ID	Group Name	VPC ID	Type	Description	Owner
sg-405b3003	default	vpc-974efbed	EC2-VPC	default VPC securi...	289857919920	

**Security Group: sg-405b3003**

Description Inbound Rules Outbound Rules Tags

Edit rules

Type	Protocol	Port Range	Source
All traffic	All	All	sg-405b3003

# Security groups - Outbound

VPC Dashboard

Filter by VPC:

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Endpoint Services

NAT Gateways

Peering Connections

**Create security group** **Actions**

Filter by tags and attributes or search by keyword

Name	Group ID	Group Name	VPC ID	Type	Description	Owner
sg-405b3003	default	vpc-974efbed	EC2-VPC	default VPC securi...	289857919920	

**Security Group: sg-405b3003**

**Outbound Rules**

**Edit rules**

Type	Protocol	Port Range	Destination
All traffic	All	All	0.0.0.0/0

# Security groups

- Security groups are nothing but simple firewall rules that you can configure to safeguard your instances.
- You can create a maximum of 100 security groups for a single VPC, with each Security Group containing up to 50 firewall rules in them.
- Also, it is very important to remember that a Security Group does not permit inbound traffic by default.
- You have to explicitly set inbound traffic rules to allow traffic to flow to your instance.
- However, all outbound traffic from the instance is allowed by default.

# Network ACLs

- These provide an added security measure over security groups as they are instance specific, whereas Network ACLs are subnet specific.
- Unlike your security groups, however, you can both allow and restrict inbound and outbound traffic using ACL rules.
- Each ACL rule is evaluated by AWS based on a number. The number can be anything from 100 all the way up to 32,766.

# Example of how ACL rules

The screenshot shows the AWS VPC Dashboard with the 'Network ACLs' section selected. A search bar at the top allows filtering by tags and attributes or keyword. A table lists existing Network ACLs, including their Name, Network ACL ID, Associated with (Subnets), Default status, VPC, and Owner. The table shows one entry: 'acl-93e9f5e9' associated with '6 Subnets', marked as 'Default', belonging to 'vpc-974efbed' and owner '289857919920'. Below this, the 'Inbound Rules' tab is selected for the 'Network ACL: acl-93e9f5e9'. It includes a 'Edit inbound rules' button and a 'View' dropdown set to 'All rules'. A table lists the inbound rules:

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

What do these rules mean?

- 100 = traffic to flow from any protocol running on any port in and out of the subnet.
- \* = That you drop any packets that do not match the ACL's rules.

# Routing Table

- A route table contains rules for routing traffic within a subnet and from the subnet to outside world.
- Amongst other things, we use routing tables to add internet gateways and NAT gateways to the subnet.

# Routing Table

VPC Dashboard Actions

Filter by VPC:  Select a VPC

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Endpoint Services

NAT Gateways

Peering Connections

Security

Network ACLs

Security Groups

Virtual Private Network (VPN)

Customer Gateways

**Create route table**

Route Table ID: rtb-2dc2f252

Explicitly Associated with: Main

VPC ID: vpc-974efbed

Owner: 289857919920

Route Table: rtb-2dc2f252

Summary **Routes** Subnet Associations Route Propagation Tags

Edit routes

View: All routes

Destination	Target	Status	Propagated
172.31.0.0/16	local	active	No
0.0.0.0/0	igw-af2038d7	active	No

# Internet Gateways

- Internet Gateways, as the name suggest, are primarily used to provide Internet connectivity to your VPC instances.
- An Internet Gateway allows you to make a subnet public by providing a route to the internet.
- All instances within the subnet can access the internet only through this gateway. Also, resources from the internet can access the instances in your subnet using this gateway.

# Internet Gateways

VPC Dashboard

Filter by VPC:

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways **Internet Gateways**

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Endpoint Services

NAT Gateways

Peering Connections

**Create internet gateway** Actions ▾

Filter by tags and attributes or search by keyword

	Name	ID	State	VPC	Owner
	igw-af2038d7	igw-af2038d7	attached	vpc-974efbed	289857919920

Internet gateway: igw-af2038d7

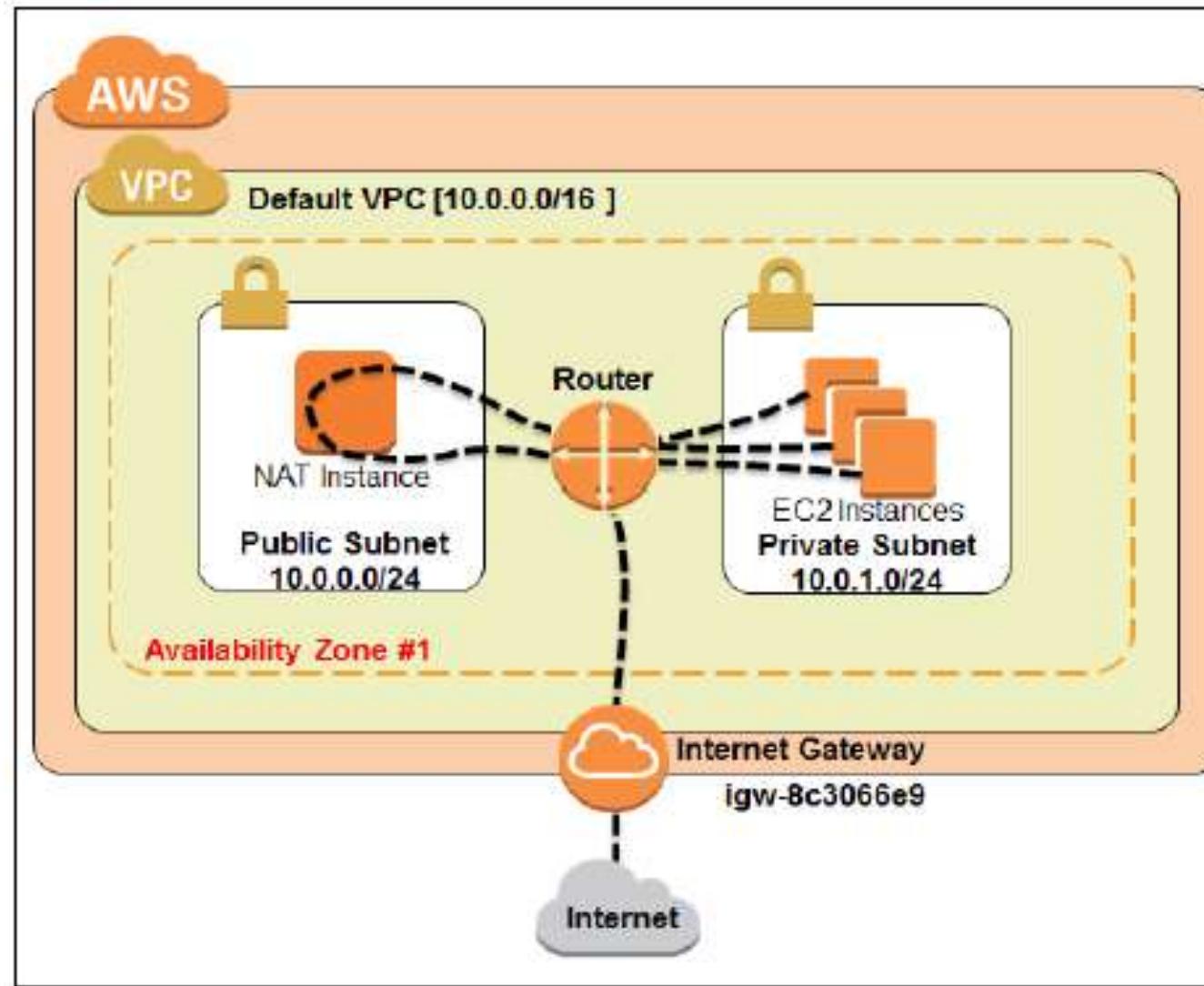
Description Tags

ID	igw-af2038d7	Attached VPC ID	vpc-974efbed
State	attached	Owner	289857919920

# NAT Gateway

- You can allow instances from your private subnet to connect to the internet using a NAT gateway.
- The instances in the private subnet do not have an public IP address, so the NAT gateway translates the private IP to a public IP before routing the traffic out to the internet.
- NAT stands for Network Address Translation and it does just that – translates private IPs to public IP.

# NAT Gateway



# NAT Gateway

# VPC deployment scenarios

- VPC provides a simple, easy-to-use wizard that can spin up a fully functional VPC within a couple of minutes. All you need to do is select a particular deployment scenario out of the four scenarios provided and configure a few basic parameters such as subnet information, availability zones in which you want to launch your subnets, and so on, and the rest is all taken care of by AWS itself.
- **VPC with a single public subnet**
- **VPC with public and private subnets (NAT)**
- **VPC with public and private subnets and hardware VPN access**
- **VPC with a private subnet only and hardware VPN access**

# VPC with a Single Public Subnet

## Step 1: Select a VPC Configuration

### VPC with a Single Public Subnet

VPC with Public and Private Subnets

VPC with Public and Private Subnets and Hardware VPN Access

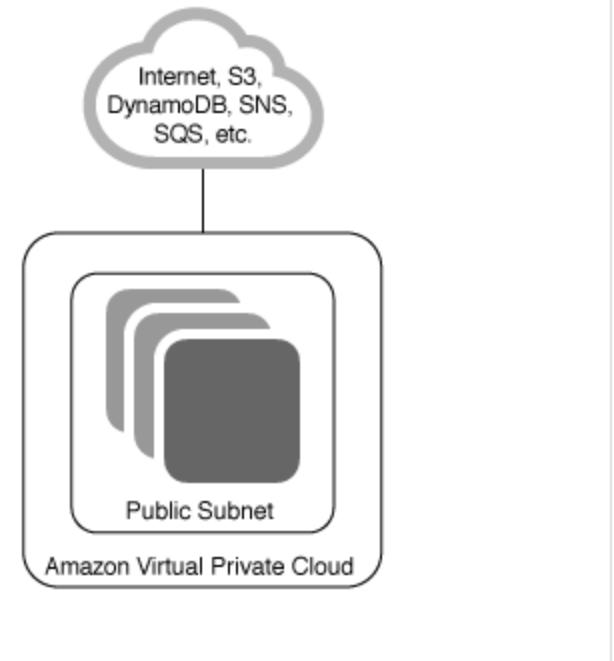
VPC with a Private Subnet Only and Hardware VPN Access

Your instances run in a private, isolated section of the AWS cloud with direct access to the Internet. Network access control lists and security groups can be used to provide strict control over inbound and outbound network traffic to your instances.

#### Creates:

A /16 network with a /24 subnet. Public subnet instances use Elastic IPs or Public IPs to access the Internet.

Select



# VPC with Public and Private Subnets

## Step 1: Select a VPC Configuration

VPC with a Single Public Subnet

**VPC with Public and Private Subnets**

VPC with Public and Private Subnets and Hardware VPN Access

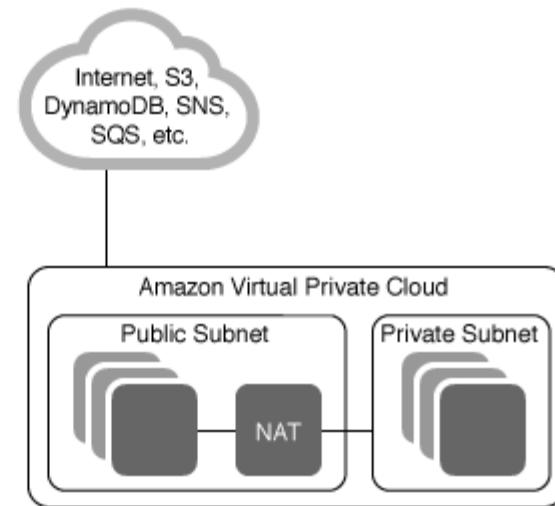
VPC with a Private Subnet Only and Hardware VPN Access

In addition to containing a public subnet, this configuration adds a private subnet whose instances are not addressable from the Internet. Instances in the private subnet can establish outbound connections to the Internet via the public subnet using Network Address Translation (NAT).

### Creates:

A /16 network with two /24 subnets. Public subnet instances use Elastic IPs to access the Internet. Private subnet instances access the Internet via Network Address Translation (NAT). (Hourly charges for NAT devices apply.)

Select



# VPC with Public and Private Subnets and Hardware VPN Access

## Step 1: Select a VPC Configuration

VPC with a Single Public Subnet

VPC with Public and Private Subnets

**VPC with Public and Private Subnets and Hardware VPN Access**

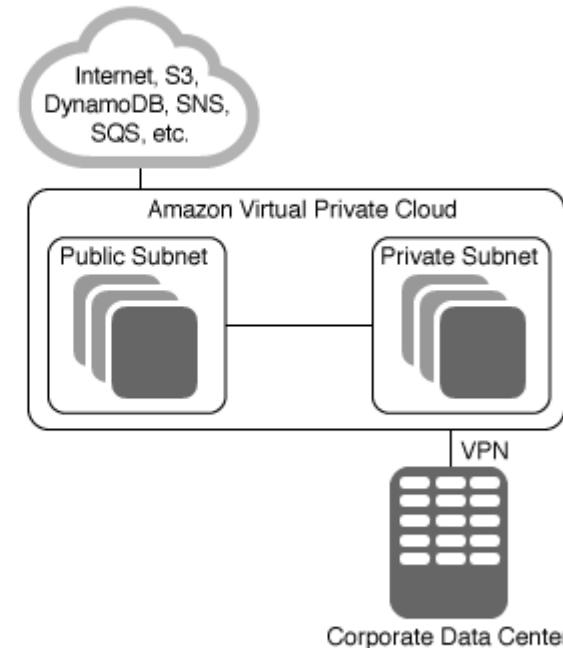
VPC with a Private Subnet Only and Hardware VPN Access

This configuration adds an IPsec Virtual Private Network (VPN) connection between your Amazon VPC and your data center - effectively extending your data center to the cloud while also providing direct access to the Internet for public subnet instances in your Amazon VPC.

### Creates:

A /16 network with two /24 subnets. One subnet is directly connected to the Internet while the other subnet is connected to your corporate network via IPsec VPN tunnel. (VPN charges apply.)

Select



# VPC with a Private Subnet Only and Hardware VPN Access

## Step 1: Select a VPC Configuration

VPC with a Single Public Subnet

VPC with Public and Private Subnets

VPC with Public and Private Subnets and Hardware VPN Access

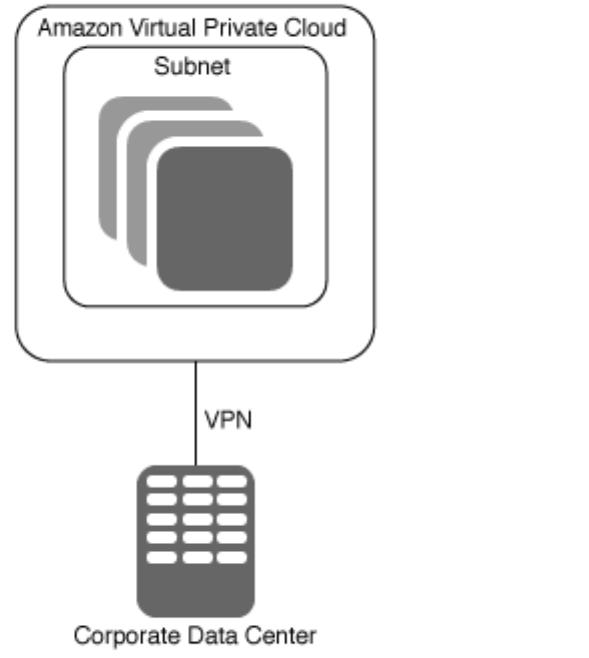
**VPC with a Private Subnet Only and Hardware VPN Access**

Your instances run in a private, isolated section of the AWS cloud with a private subnet whose instances are not addressable from the Internet. You can connect this private subnet to your corporate data center via an IPsec Virtual Private Network (VPN) tunnel.

**Creates:**

A /16 network with a /24 subnet and provisions an IPsec VPN tunnel between your Amazon VPC and your corporate network. (VPN charges apply.)

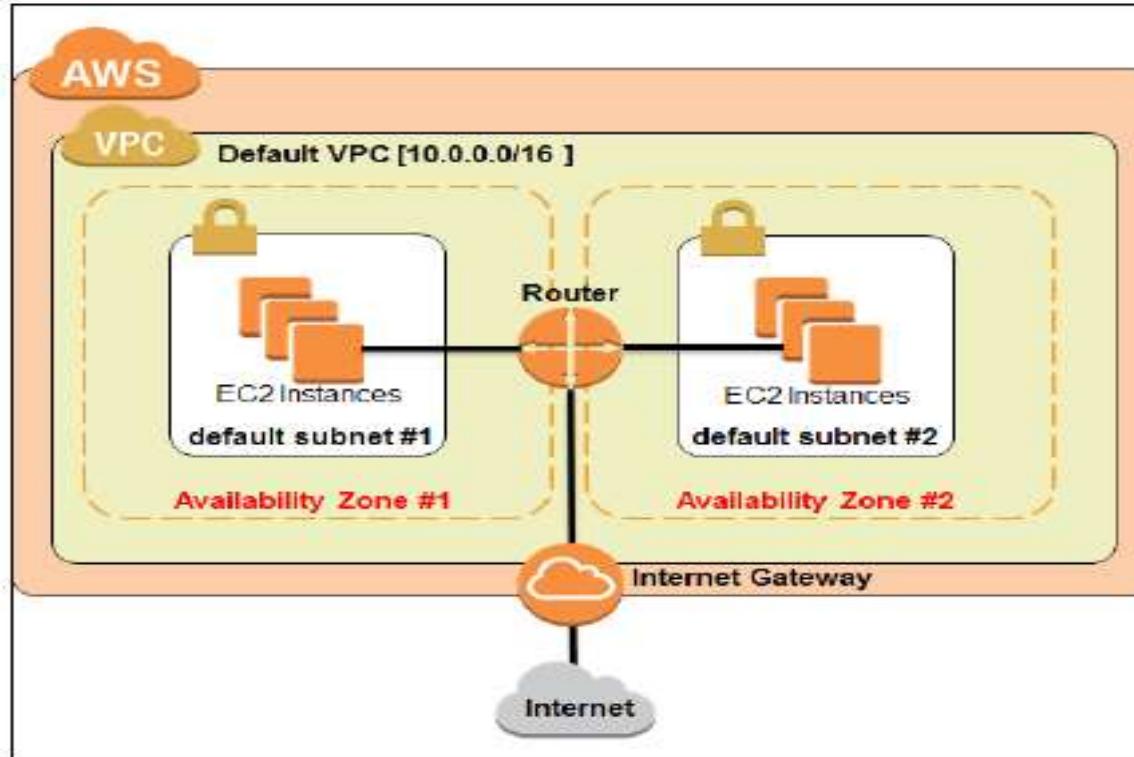
Select



# The Default VPC

- The default VPC comes preconfigured with the following set of configurations:
- The default VPC is always created with a CIDR block of /16, which means it supports 65,536 IP addresses in it.
- A default subnet is created in each AZ of your selected region. Instances launched in these default subnets have both a public and a private IP address by default as well.
- An Internet Gateway is provided to the default VPC for instances to have Internet connectivity.
- A few necessary route tables, security groups, and ACLs are also created by default that enable the instance traffic to pass through to the Internet.  
Refer to the following figure:

# The Default VPC



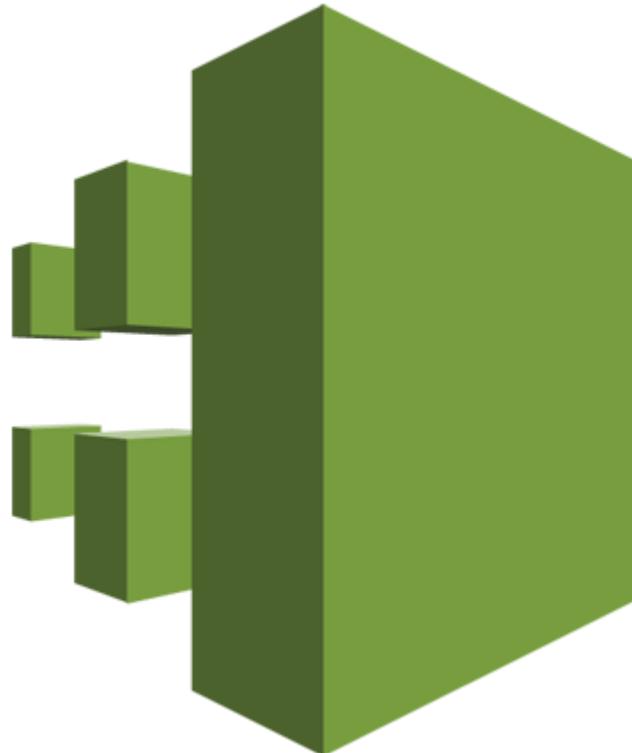
You can use this default VPC just as any other VPC by creating additional subnets in it, provisioning route tables, security groups, and so on.

Note: Any other VPC that you create besides the default VPC is called as the non-default VPC. Each non-default VPC in turn contains non-default subnets, and so on and so forth.

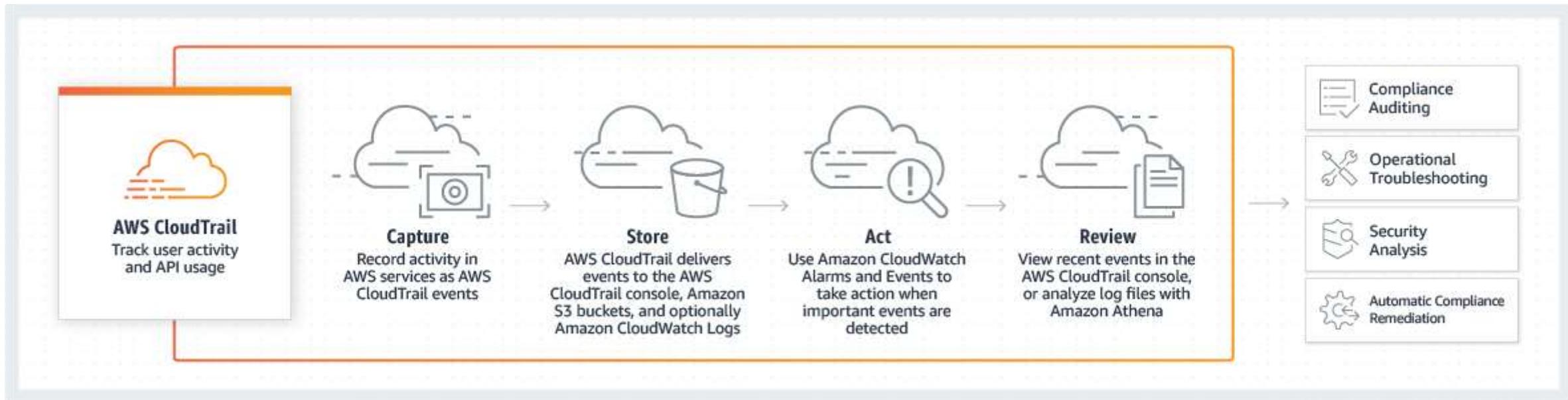
Tooooo Much ?



# CLOUD TRAIL



# CloudTrail



# What is Cloud Trail?

- AWS Cloud Trail is a service that enables governance, compliance, operational auditing, and risk auditing of your AWS account.
- With Cloud Trail, you can log, continuously monitor, and retain account activity related to actions across your AWS infrastructure.
- Cloud Trail provides event history of your AWS account activity, including actions taken through the AWS Management Console, AWS SDKs, command line tools, and other AWS services.
- This event history simplifies security analysis, resource change tracking, and troubleshooting

Trail name\*

Apply trail to all regions  Yes  No

Creates the same trail in all regions and delivers log files for all regions

## Management events

Management events provide insights into the management operations that are performed on resources in your AWS account. [Learn more](#)

Read/Write events  All  Read-only  Write-only  None 

## Data events

Data events provide insights into the resource operations performed on or within a resource. Additional [charges](#) apply. [Learn more](#)

S3

Lambda

You can record S3 object-level API activity (for example, GetObject and PutObject) for individual buckets, or for all current and future buckets in your AWS account. Additional [charges](#) apply. [Learn more](#)

Showing 0 of 0 resources

Bucket name	Prefix	Read	Write	
<input type="checkbox"/> Select all S3 buckets in your account 		<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Write	

No resources found

 [Add S3 bucket](#)

## Storage location

Create a new S3 bucket  Yes  No

S3 bucket\*  

[CloudTrail](#)[Dashboard](#)[Event history](#)[Trails](#)

## Event history

Your event history contains the activities taken by people, groups, or AWS services in [supported services](#) in your AWS account. By default, the view filters out read-only events. You can change or remove that filter, or a

You can view the last 90 days of events. Choose an event to view more information about it. To view a complete log of your CloudTrail events, create a trail and then go to your Amazon S3 bucket or CloudWatch Logs.

Can't find what you're looking for? [Run advanced queries in Amazon Athena](#)

Filter:	Read only	false	Time range:	Select time range			
	Event time	User name	Event name	Resource type	Resource name		
▶	2018-12-22, 09:32:02 PM	weather-visualization-sta...	StartExecution				
▶	2018-12-22, 09:32:02 PM	weather-visualization-sta...	StartExecution				
▶	2018-12-22, 09:32:02 PM	weather-visualization-sta...	StartExecution				
▶	2018-12-22, 09:32:01 PM	weather-visualization-sta...	StartExecution				
▶	2018-12-22, 09:32:01 PM	weather-visualization-sta...	StartExecution				
▶	2018-12-22, 09:31:14 PM	i-cb3ecc66	UpdateInstanceInformation				
▶	2018-12-22, 09:30:49 PM	i-0e9e9829dbc96c7ce	UpdateInstanceInformation				
▼	2018-12-22, 09:30:15 PM	reyaz	REST.GET.OBJECT_LOCK_CO...				
AWS access key <a href="#">A██████████</a>			Event time	2018-12-22, 09:30:15 PM			
AWS region eu-west-1			Read only	false			
Error code <a href="#">ObjectLockConfigurationNotFoundError</a>			Request ID	342DD33C655EFA19			
Event ID 7ffe27f2-ee6f-491f-9be0-6060a37285c0			Source IP address	183.82.196.37			
Event name REST.GET.OBJECT_LOCK_CONFIGURATION			User name	reyaz			
Event source s3.amazonaws.com							
Resources Referenced (0)							
<a href="#">View event</a>							

# Simple Email Service



**Amazon SES**

# What is SES?

- Amazon Simple Email Service (Amazon SES) cloud bases email sending service
- Lets you send transactional email, marketing messages, or any other type of high-quality content to your customer.

Click on Verify a New Email Address

SES Home

Identity Management

Domains

Email Addresses

Email Sending

Sending Statistics

Reputation Dashboard

Dedicated IPs

Configuration Sets

SMTP Settings

Suppression List Removal

Cross-Account Notifications

Email Templates

Email Receiving

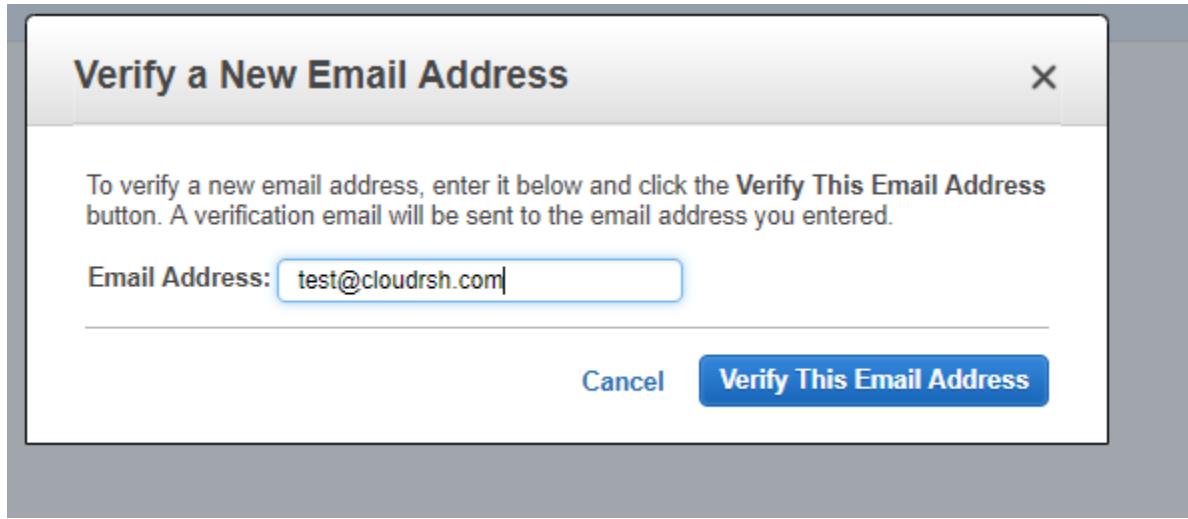
Rule Sets

IP Address Filters

Verify a New Email Address   Send a Test Email   Remove   View Details

Search email addresses   All identities

	Email Address Identities	Verification Status
<input checked="" type="checkbox"/>	noreply@██████████	verified



1. First Add a email address
2. Click on Create SMTP Credentials

SES Home

Identity Management

Domains

Email Addresses

Email Sending

Sending Statistics

Reputation Dashboard

Dedicated IPs

Configuration Sets

**SMTP Settings**

Suppression List Removal

Cross-Account Notifications

Email Templates

## Using SMTP to Send Email with Amazon SES

You can send email through Amazon SES by using a variety of SMTP-enabled programming languages.

To send email using SMTP, you will need to know the following:

**Server Name:** email-smtp.eu-west-1.amazonaws.com

**Port:** 25, 465 or 587

**Use Transport Layer Security (TLS):** Yes

**Authentication:** Your SMTP credentials - see below.

To send email through Amazon SES using SMTP, you must create SMTP credentials. SMTP credentials are available when SES is available.

To obtain your SMTP credentials, click the button below. For more information about SMTP credentials, see the [Using SMTP to Send Email with Amazon SES](#) topic.

**Create My SMTP Credentials**

**Note:** Your SMTP user name and password are not the same as your AWS access key ID and secret key.

Once click on SMTP Credentials it takes to IAM smtp user creation  
After clicking on create you will get access key and secret key

---

This form lets you create an IAM user for SMTP authentication with Amazon SES. Enter the name of a new IAM user or accept the default and click Create to set up yo

**IAM User Name:**

Maximum 64 characters

[▼ Hide More Information](#)

Amazon SES uses AWS Identity and Access Management (IAM) to manage SMTP credentials. The IAM user name is case sensitive and may contain only alphanumeric characters. SMTP credentials consist of a username and a password. When you click the Create button below, SMTP credentials will be generated for you.

The new user will be granted the following IAM policy:

```
"Statement": [{ "Effect": "Allow", "Action": "ses:SendRawEmail", "Resource": "*" }]
```

**Create**

Once you got the keys, those keys would be your username and password for SMTP

## Using SMTP to Send Email with Amazon SES

You can send email through Amazon SES by using a variety of SMTP-enabled programming languages.

To send email using SMTP, you will need to know the following:

Server Name: email-smtp.eu-west-1.amazonaws.com  
Port: 25, 465 or 587  
Use Transport Layer Security (TLS): Yes  
Authentication: Your SMTP credentials - see below.

To send email through Amazon SES using SMTP, you must create SMTP credentials. SES is available.

To obtain your SMTP credentials, click the button below. For more information about SMTP configuration, see the following sections:

Below are the configuration of SMTP on your application. Username= access key, password = secret key

```
SMTP hosts = email-smtp.us-east-1.amazonaws.com
SMTP port = 465
SMTP security = SSL
SMTP username: USERNAME
SMTP password: PASSWORD
From address: EMAIL_ADDRESS
```

