CODER UNIVERSE
TECHNOLOGY

**https://coderuniversetechnology.info/**
**https://www.facebook.com/profile.php?id=61569717476118**
**https://www.instagram.com/coder_universe_technology/**
**Contact us on: 7709701108**

## Linux, Aws & Devops Session

Cloud Computing:

It is on demand delivery of compute power, database storage, application and other IT resources through a cloud services platform with pas as you go.

NIST : It is responsible for  Developing standards and guideline.

## Service Model

*SaaS : Software as Service

*Paas: Platform as Service

*Iaas: Infrastructure as Service)

*Amazon web services is IaaS service

->It is service provider

-> https://aws.amazon.com     (internet to connect)

-> Two Types of Account

1. Root Account
2. IAM account (Identity Access Management)

-> In the year 2006 AWS started providing this services IT Infrastructure

-> Type of Services AWS provider

- Machine
- Servers
- Database
- Storage
- Security
- Analysis
- Monitoring

-> 190 + countries AWS provide the services

-> Region : It is an Geographical locations

->Data centre: Availability zone ,a room with server having complex network connections.

Challenge Before cloud

- Power back issue
- Natural Disaster
- Security
- Physical Damage

## Benefits after Cloud

- No need to buy servers/machine
- No physical damage
- Pay for what we use

## AWS

-> AWS stand for Amazon web services

-> Amazon is the company name

-> Amazon is cloud providing Infrastructure as a service(Iaas)

->Infrastructure we need to host application we can take from AWS on rent.

->Services provided by

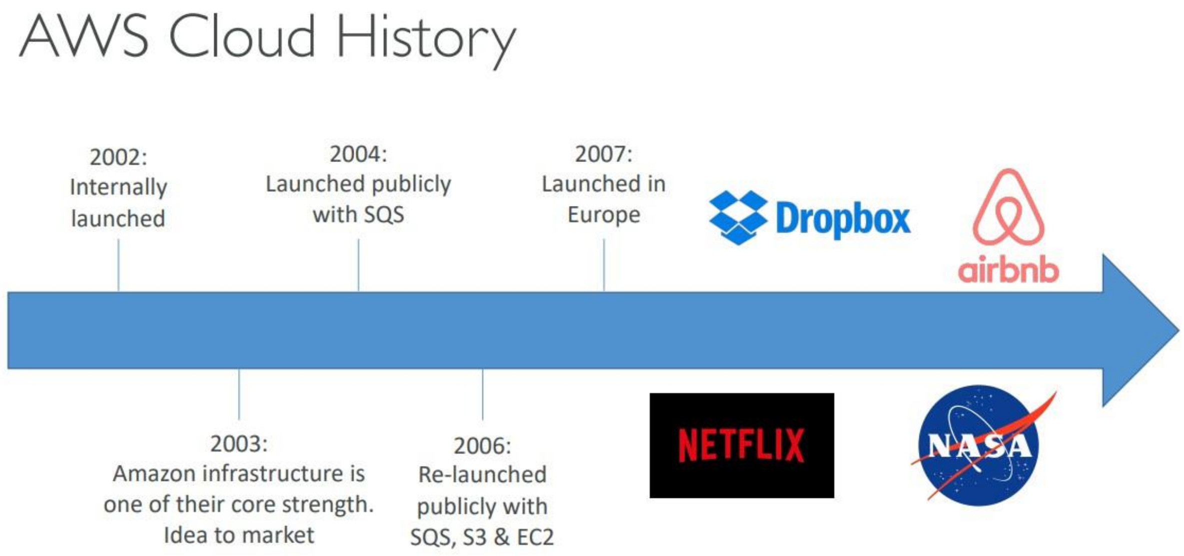
- Machines
- Servers
- DB
- Storages
- Network Security
- Analytics & Monitoring

->AWS provide services across the globe using Region & Availabilty Zone

->Region is nothing but geographical location.

->AZ : avaialbility zone means data centre

-> Data centre:  It is an room with server



AWS Cloud History

2002: Internally launched

2003: Amazon infrastructure is one of their core strength. Idea to market

2004: Launched publicly with SQS

2006: Re-launched publicly with SQS, S3 & EC2

2007: Launched in Europe

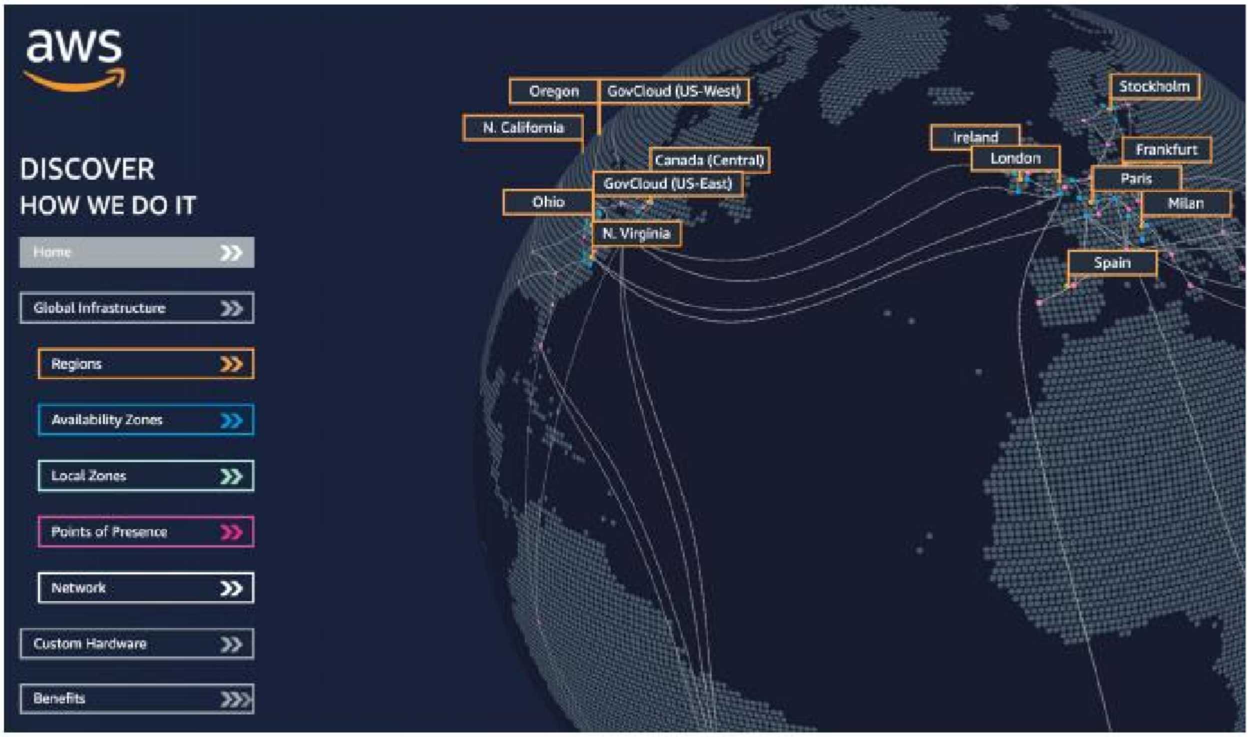
## AWS Cloud Use Cases

- AWS enables you to build sophisticated, scalable applications
- Applicable to a diverse set of industries
- Use cases include
- Enterprise IT, Backup & Storage, Big Data analytics
- Website hosting, Mobile & Social Apps
- Gaming

## AWS Global Infrastructure

- AWS Regions
- AWS Availability Zones

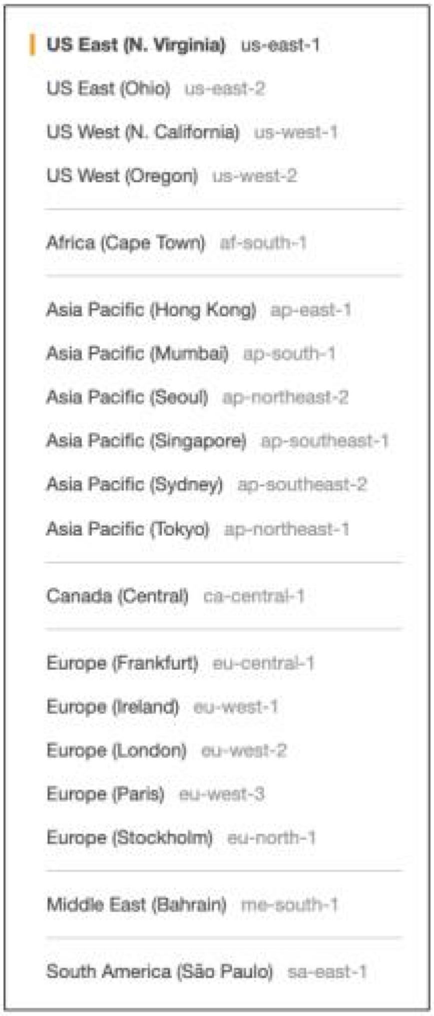
- AWS Data Centers
- AWS Edge Locations / Points of Presence



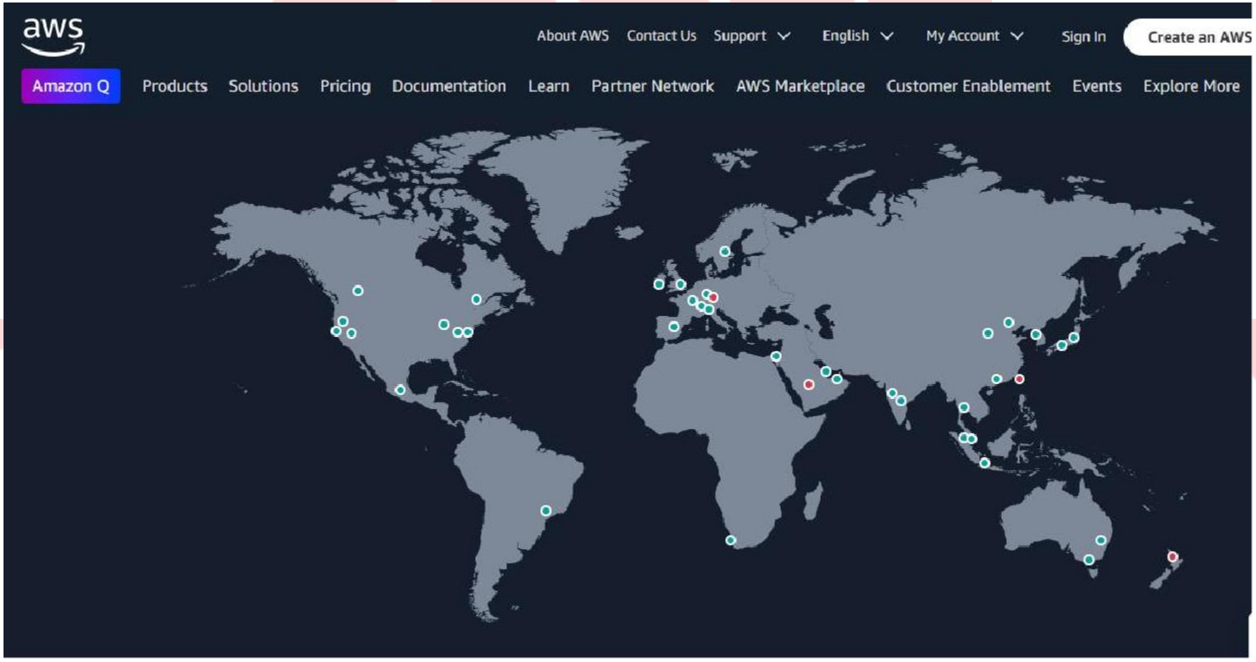
## AWS Regions

- AWS has Regions all around the world
- Names can be us-east-1, eu-west-3…
- A region is a cluster of data centers
- Most AWS services are region-scoped

Global Infrastructure - AWS



| | |
|---|---|
| **US East (N. Virginia)** | us-east-1 |
| US East (Ohio) | us-east-2 |
| US West (N. California) | us-west-1 |
| US West (Oregon) | us-west-2 |
| Africa (Cape Town) | af-south-1 |
| Asia Pacific (Hong Kong) | ap-east-1 |
| Asia Pacific (Mumbai) | ap-south-1 |
| Asia Pacific (Seoul) | ap-northeast-2 |
| Asia Pacific (Singapore) | ap-southeast-1 |
| Asia Pacific (Sydney) | ap-southeast-2 |
| Asia Pacific (Tokyo) | ap-northeast-1 |
| Canada (Central) | ca-central-1 |
| Europe (Frankfurt) | eu-central-1 |
| Europe (Ireland) | eu-west-1 |
| Europe (London) | eu-west-2 |
| Europe (Paris) | eu-west-3 |
| Europe (Stockholm) | eu-north-1 |
| Middle East (Bahrain) | me-south-1 |
| South America (São Paulo) | sa-east-1 |

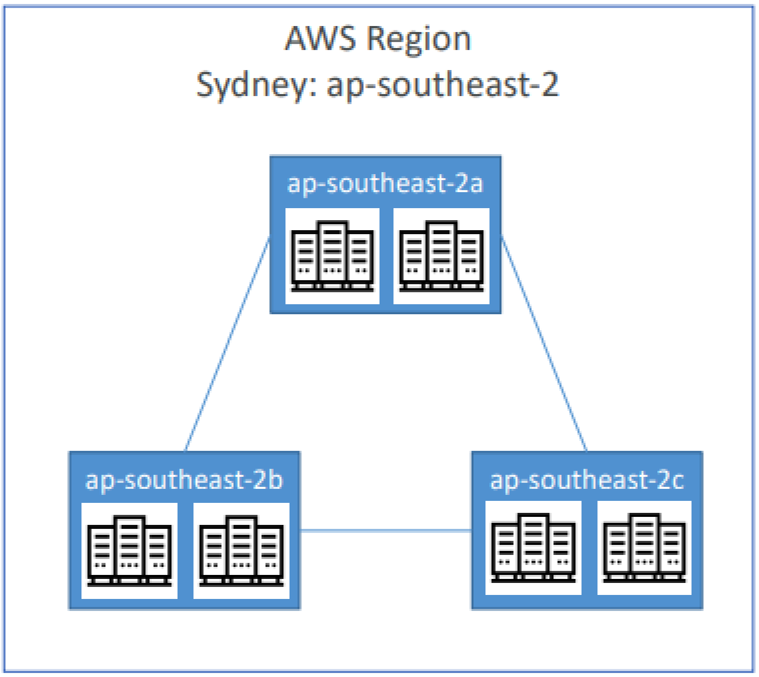


## AWS Availability Zones

- Each region has many availability zones (usually 3, min is 3, max is 6).

Example:

- ap-southeast-2a
- ap-southeast-2b
- ap-southeast-2c



- Each availability zone (AZ) is one or more discrete data centers with redundant power, networking, and connectivity
- They're separate from each other, so that they're isolated from disasters
- They're connected with high bandwidth, ultra-low latency networking

## Amazon EC2

- EC2 is one of the most popular of AWS' offering

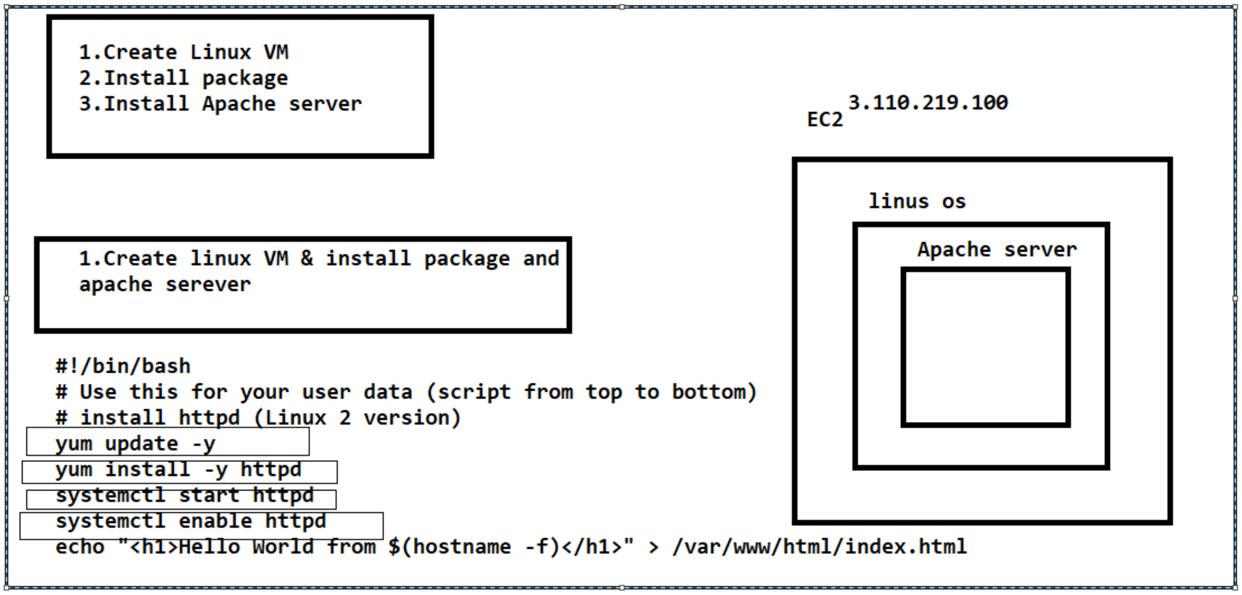
- EC2 = Elastic Compute Cloud = Infrastructure as a Service
- It mainly consists in the capability of :
- Renting virtual machines (EC2)
- Storing data on virtual drives (EBS)
- Distributing load across machines (ELB)
- Scaling the services using an auto-scaling group (ASG)
- Knowing EC2 is fundamental to understand how the Cloud works

## EC2 sizing & configuration options

- Operating System (OS): Linux, Windows or Mac OS
- How much compute power & cores (CPU)
- How much random-access memory (RAM)
- How much storage space:
- Network-attached (EBS & EFS)
- hardware (EC2 Instance Store)
- Network card: speed of the card, Public IP address
- Firewall rules: security group
- Bootstrap script (configure at first launch): EC2 User Data

Sample script :

```
#!/bin/bash
# Use this for your user data (script from top to bottom)
# install httpd (Linux 2 version)
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
echo "<h1>Hello World from $(hostname -f)</h1>" >
/var/www/html/index.html
```



1.Create Linux VM
2.Install package
3.Install Apache server

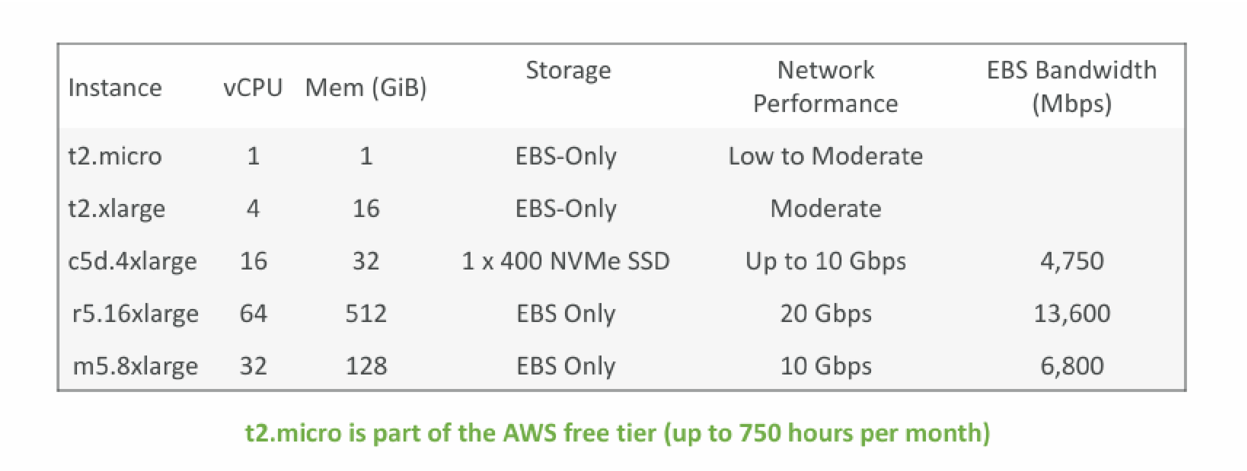
1.Create linux VM & install package and apache serever

```
#!/bin/bash
# Use this for your user data (script from top to bottom)
# install httpd (Linux 2 version)
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
echo "<h1>Hello World from $(hostname -f)</h1>" > /var/www/html/index.html
```

## Feature of EC2

- We can create or manage lifecycle of EC2 instance.
- Load Balancing & Auto scaling for multiple EC2 instance. EC2/ELB/EBS
- Attach storage (& network storage) to our EC2 instance.

# EC2 Instance Type



| Instance | vCPU | Mem (GiB) | Storage | Network Performance | EBS Bandwidth (Mbps) |
|---|---|---|---|---|---|
| t2.micro | 1 | 1 | EBS-Only | Low to Moderate | |
| t2.xlarge | 4 | 16 | EBS-Only | Moderate | |
| c5d.4xlarge | 16 | 32 | 1 x 400 NVMe SSD | Up to 10 Gbps | 4,750 |
| r5.16xlarge | 64 | 512 | EBS Only | 20 Gbps | 13,600 |
| m5.8xlarge | 32 | 128 | EBS Only | 10 Gbps | 6,800 |

**t2.micro is part of the AWS free tier (up to 750 hours per month)**

Optimized combination of compute(CPU),memory,disk. 270+ instance types across 40+ instance type.

t2.micro

- t- instance family
- 2- generation
- micro - size(nano< micro < small< medium < large < xlarge <...)

m5.2xlarge

- m : instance family
- 5 : generation
- 2Xlarge : size within instance class

General Purpose :

- Great for a diversity of workloads such as web servers or code repositories

- Balance between:
- Compute
- Memory
- Networking
  - In the course, we will be using the t2.micro which is a General Purpose

## Compute Optimised

- Batch processing workloads
- Media transcoding
- High performance web servers
- High performance computing (HPC)
- Scientific modeling & machine learning
- Dedicated gaming server

Eg: c6g/c6gn/c5/c5a/c4

## Memory Optimized

- High performance, relational/non-relational databases
- Distributed web scale cache stores
- In-memory databases optimized for BI (business intelligence)
- Applications performing real-time processing of big unstructured data

Eg : R6g/R5/R5b/R5n

## EC2 Instance Types – Storage Optimized

- Great for storage-intensive tasks that require high, sequential read and write access to large data sets on local storage
- Use cases:
- High frequency online transaction processing (OLTP) systems
- Relational & NoSQL databases
- Cache for in-memory databases (for example, Redis) • Data warehousing applications • Distributed file systems

->For storage of data that required sequential read & write access.

-> SQL DB and NO sql DB

-> Cashe for in-memory (Redis)

-> Distributed file system

Eg: D2/D3

## Security Groups

-> It is kind of network security in AWS.

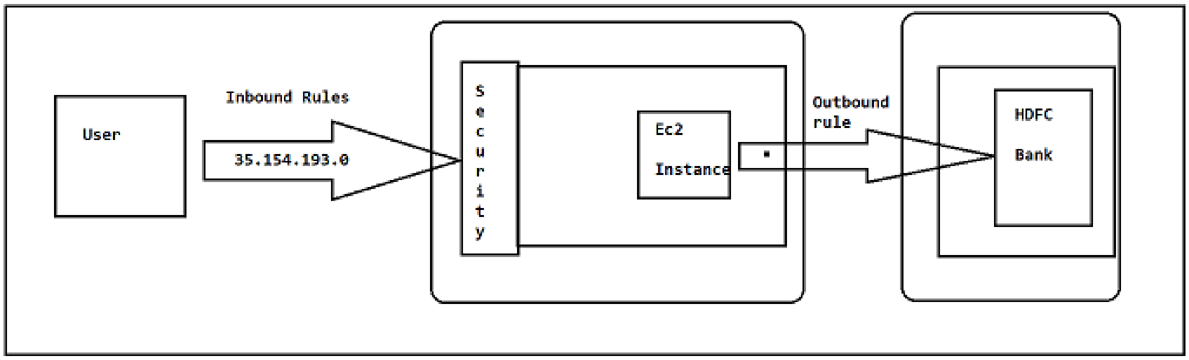
-> SG control traffic "in & Out" of our EC2 Instance.

-> SG contains only rules to allow or reject traffic.

-> It is acting as an firewall on EC2 instance.

-> SG is controlling

- Ports
- IP ranges - IPv4 to IPv6
- Control in bound rules (from outside to our Ec2 instance)
- control out bound rules (from Ec2 instance to other)



## Linux

Windows OS:

- It is provided by Microsoft company.
- It is paid s/w
- It is single user based .
- It can run multiple application.
- It is less secured
- It is giving beautiful UI

Linux OS:

- Linux is free & Open source.
- Anyone can take Linux OS source code and customize.
- Linux is multi user based OS.
- It is very secured.
- It is community based.
- First OS come into market in the year 1956.
- General motor la b implemented the OS for IBM
- In 1969 the first version of UNIX OS come into market by Ken Thomson
- Linus Torvolds , made the changes in existing OS anf then release the new one in the market

LINUS+UNIX => LI+NIX => Linux

Different flavors of Linux OS

RHEL -> Red Hat

CENTOS -> community

Ubuntu -> community

openuse-> Microsoft

## Linux Commands

PWD : Present Working directory

cd : change directory

## Ports to Know

SSH = 22 -> log into a linux instance

FTP = 21 ->File Transfer Protocol

->upload files into a file share

SFTP= 22 ->Secure File Transfer Protocol

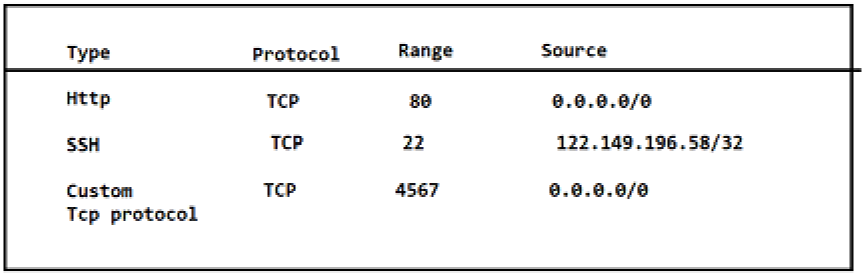
-> Upload the file using SSH

HTTP = 80 -> We want to access unsecured website

HTTPS = 443 ->We want to access secured website

RDP = 3389 ->(Remote Desktop Protocol)

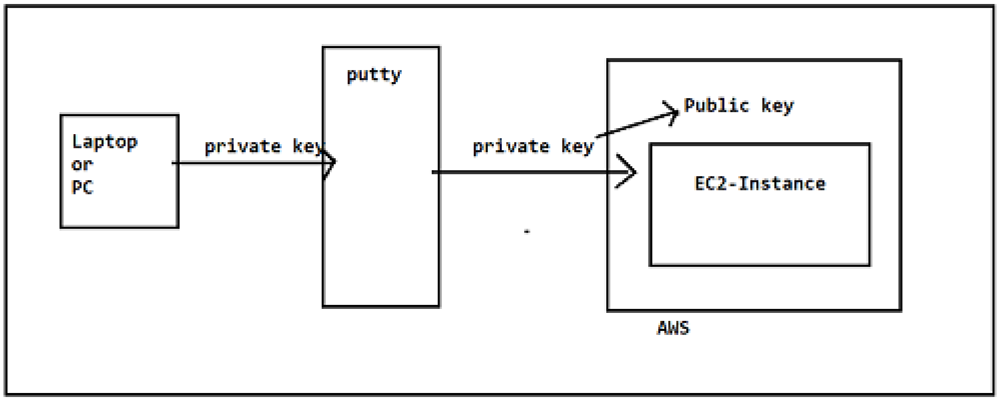
-> log into a windows instance



| Type | Protocol | Range | Source |
|---|---|---|---|
| Http | TCP | 80 | 0.0.0.0/0 |
| SSH | TCP | 22 | 122.149.196.58/32 |
| Custom<br>Tcp protocol | TCP | 4567 | 0.0.0.0/0 |

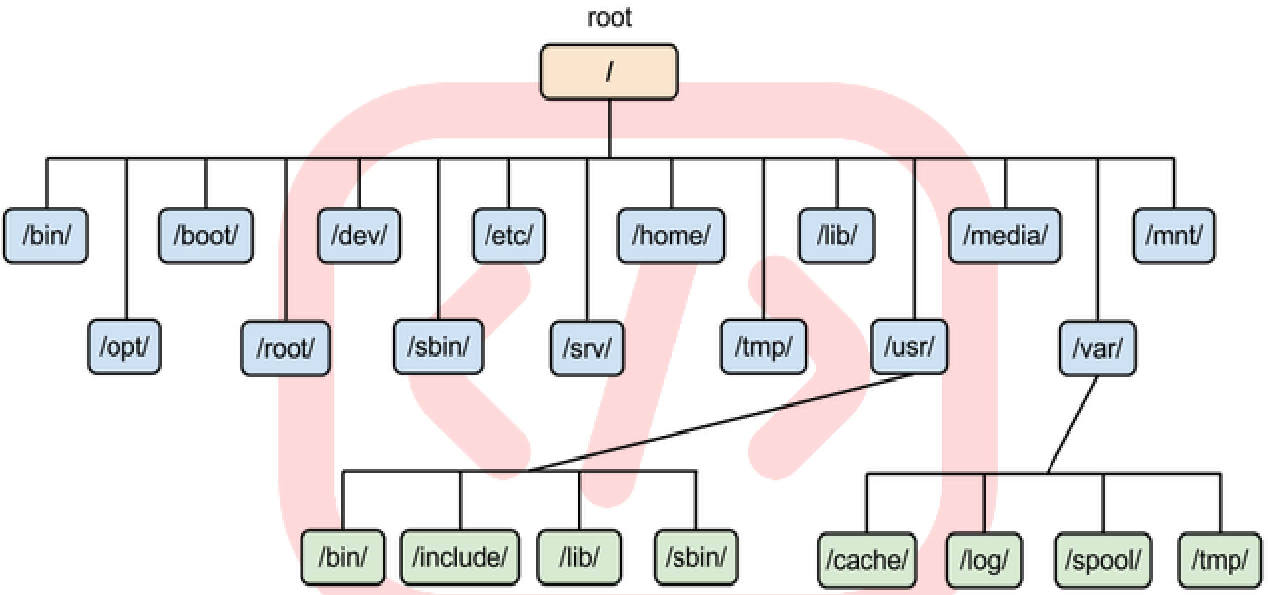
- clear : It is used to clear the console/terminal
- pwd : present working directory
- whoami: user details
- mkdir : It is used to create the directory/
- ls : List the directory.
- touch : It is used to create the text files.

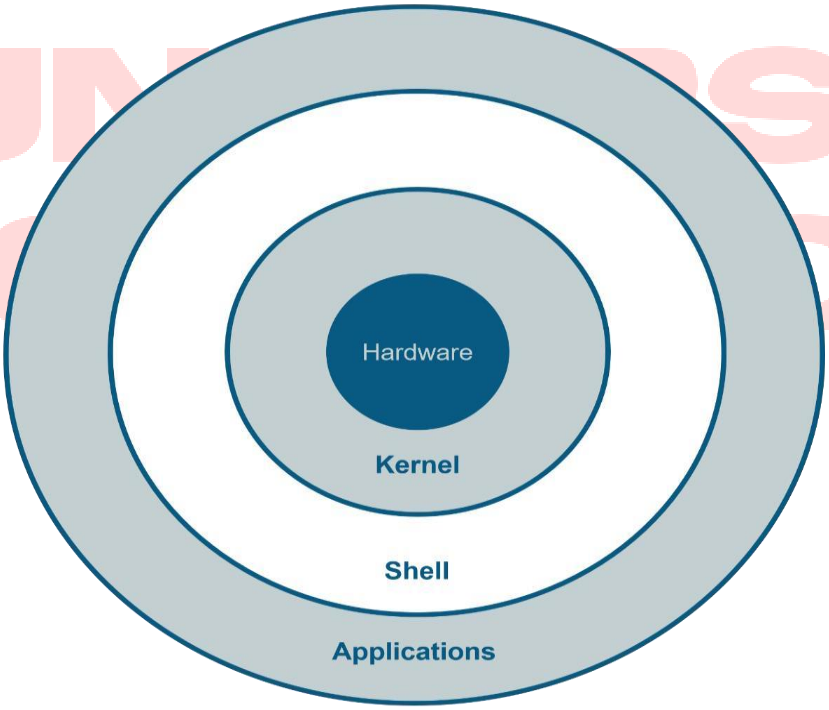
## What is the key-pair ?



- Key pair consist basically Public Key & an Private Key.
- AWS stores the public key and we store the private key.
- These keys are used to connect EC2 instance securely.

# Linux File System & Architecture





-> All the files in Linux are of 3 types

1.Ordinary Files (-)

2.Directory Files (-d)

3.Device Files

4.linked files (l)

/ -> It is root directory of entire file system

/ & root : -> / means root

-> /root it represent root account user home directory

/bin/ : It is having binary files

/boot/ : It is having static files for boot loader

/dev/ : Device files

/etc/ :System configuration files

/home/ : Uder home directory

/lib/ : Shared libraries

/media/ : Removable media

/mnt/ : Mounted file system

/opt/ : Application software packages

/sbin/ : System binaries files

/srv/ : Site specific data of the system

/tmp/ : temporary files

/usr/:  binaries/libraries/documentation/source code

Linux commands

ls-l : It is giving details in alphabetical order

ls-lr: It is giving details in reverse alphabetical order

ls | more : It is giving files and directory names line by line

clear : It is clear the console/terminal

ls -a :  To see the hidden files with "." & ".."

ls -A :  To see the hidden files with "."

ls-i : It represent the address of the files attribute are store.

1.To create an directory "mkdir"

```
mkdir abc
```

2.To create multiple directory "mkdir"

```
mkdir aws linux java
```

3.To create one directory into another "mkdir"

```
mkdir  dir1/dir2
```

4.To remove empty directory

```
rmdir dir1  (It will remove only empty directories)
```

5.To remove non-empty directory

```
rmdir -r dir1  (It will remove only  non-empty directories)
```

6.To list out all files and directory "ls"

7.To check present working directory"pwd"

```
pwd
```

8.To check present user "whoami"

```
whoami
```

9.To add user

```
adduser user1
```

10.To add user

```
adduser user1
```

11.To create an empty file "touch"

```
touch file.txt
```

12.To view/write the data "cat"

```
cat >file1.txt
-Hello  world  (
-ctrl+D
```

13.To view the data "head".It will print first 10 lines of files

```
head file1.txt
```

14.To view the first n lines "head -n 3 file1.txt".It will print first 3 lines of files

```
    head -n 3 file1.txt
```

15.To view the first n lines "head -n 20 file1.txt".It will print first 20 lines of files

```
    head -n 20 file1.txt
```

16.To exclude the bottom n lines "head -n -20 file1.txt".It will exclude last 2 lines of files(use negative)

```
    head -n -2 file1.txt
```

**********************************************

UseCase1-Starts

**********************************************

In Logger files  contains details of the files
Issue is available in lgo files
Latest data always available in bottom of the files.

So to read the file from bottom to top we use "tail"

17.To view the bottom n lines "tail -n 3 file1.txt".It will print last 3 lines of files

```
    tail -n 3 file1.txt      |same O/p
    tail -n -3 file1.txt     |same O/p

    tail +3 file1.txt        |From second line to last line it will print
```

18.To view the live changes n lines "tail -f  file1.txt"

```
    tail -f  file1.txt
```

**********************************************

UseCase1-End

**********************************************

19.To copy the data from one file to another file "cp"

```
    cp file1.txt  file2.txt     =>it will copy data from file1 to file2
```

20.To copy the data of two files into third files `cat file1.txt  file2.tx > file3.txt`     =>it will copy data from file1 and file2 to file3

**21.To count file count data"wc"**

```
wc file1.txt
```

**22.To rename or move file count data"mv"**

```
mv file1.txt  file2
```

**23.Directory1(dir1) has 5 files to move all files to Driectory2(dir2)**

```
mv dir1/* dir2
```

## 24.Grep : Global Regular expression print

-Grep command is used to fine the pattern from the file.

-In log files exception occur we use ctrl+ f to search data in file.

-Same way in linux we use "grep" command to search  for text.

**********************************************

## UseCase2-Starts

**********************************************

example: file1.txt

content

```
=======
India is my country.
My Friend                <=======
You are my true friend.
```

command:

```
grep my file1.txt      |case senstive|
```

output

```
======
India is my country.
You are my true friend.
```

**********************************************

UseCase2-Ends

**********************************************

UseCase3-Starts

**********************************************

example: file2.txt
content
=======
India is my country.
Hello world
My Friend                 <=======
You are my true friend.

command:
grep -i my file1.txt      |not case senstive|
output
======
India is my country.
my love.
My love.                 <=======
You are my true friend.

**********************************************

UseCase3- Ends

**********************************************

**********************************************

UseCase4 -Starts

**********************************************

command : grep <word> *
It will look for the specific word in all files

**********************************************

UseCase4 -Ends

**********************************************

UseCase5 -Starts

********************************************ccommand : grep -c 'word' filename

It will print number of lines contain word text

********************************************

UseCase5 -Ends

********************************************

UseCase6 -Starts

********************************************

command : grep -l 'word' filename

It will print files name which contain 'word' text

********************************************

UseCase6 -Ends

********************************************

* To compare two files "cmp"

```
cmp file1.txt  file2.txt
```

* To edit the file "vi" command

- vi file.txt
  - press "i" for insert mode to make the changes
  - press 'esc' to get out of it
  - type :wq and click on enter to save .

To edit the file "vi" & "sed"command

UseCase6 –Starts

(To search the word & change the word from the file) "sed"

********************************************

sed(Stream Editor)

We can perform operation on fiule data without openeing the file of very first occurance of word command :sed  's/<old-word>/<new word>' file.txt

UseCase6 -Ends

**********************************************

UseCase7 -Starts

(To search the word & change the word from the file from specific line)  "sed"
sed(Stream Editor)
We can perform operation on fiule data without openeing the file of from second line

command :
sed  's/<old-word>/<new word>/<line number>' file.txt

**********************************************

UseCase7 -Ends

**********************************************

UseCase8 –Starts

(To search the word & change the word from the file from all line)  "sed"
sed(Stream Editor)
We can perform operation on file data without opening the file of from all line
command :
sed 's/<old-word>/<new word>/g' file.txt

UseCase8 -Ends

UseCase9 -Starts

(To delete the specific line from the file)  "sed"
sed(Stream Editor)

command :
sed 'nd' file.txt    |nth line will be delete/n is the line number|

UseCase9 -Ends

**********************************************

"man" command

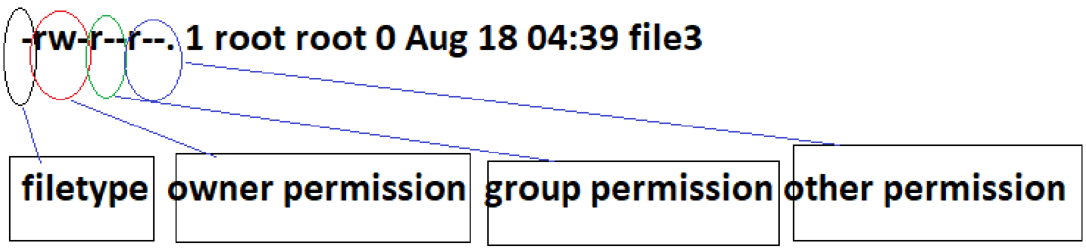
man locate ==>It will give documentation of locate commands

## FILE PERMISSION

- we will use the files to store the data
- In order to protect our data we need to secure our files by using files permission in linux.
- Linux is multi user OS
- Multiple use can connect to Linux machine at a time
- Lets an application running inside in a server which is installed in linux server.
- Our application reading config file from db config/mail config/security config.
- we should not allow everyone to allow config files for security reason.

## WORKING WITH FILES

- Normal file<br>
b Block file(Hardisk file)
c character file
d directory
l link file



```
-rw-r--r--. 1 root root 0 Aug 18 04:39 file3
```

filetype

owner permission

group permission

other permission

r=>Read
w=>write
x=>execute
-=>No permission

**To grant write permission to user**

command "chmod o+w <file name>"

o : means other
+:means add
w: means write

**To remove write permission to user**

command "chmod o-r <file name>"

o : means other
-: means remove permission
r: means read

**To remove  execute permission from group**

command "chmod g-x <file name>"

o : means other
-:means remove permission
w: means execute

Note: Similarly we can use for other combination. Check combination always

## Symbolically mode to modify the files permissions

->777 =>read/write/execute permision
example :"chmod 777 file1.txt"

Number      Permission
0---------->No Permission
1---------->Execute
2---------->Write
3---------->Execute &Write
4---------->Read
5---------->Read & Execute
6---------->Read & Write
7---------->Read,Write<Execute

## Changing Owner ship of User

**chown** can be used by root user only
change to root user

```
chown john:john <filename>
```

## Working With Account User

->Linux is multiple user based OS.
->We can create multiple accounts in linux.

## Archiving of Files or Directories

Zip the files
**gzip** -> Create a compressed file
**gunzip** -> unzip a file

## Working With Group User

->Group means collection of users
->Main purpose of group is to defined set of privileges for a given resource withing the group.

Note: Do we need to give 10 user each time out of 20 user ?

Create an group and defined the permission to avoid above issue.

* Create group

==========

```
  sudo groupadd <group-name>
```

* Delete group

==========

```
  sudo groupdel <group-name>
```

* Add user to group

====================

```
  sudo usermod -aG <group-name> <user-name>
```

* Remove user to group

====================

```
  sudo gpasswd -d  <user-name> <group-name>
```

* Add see existing  group

=========================

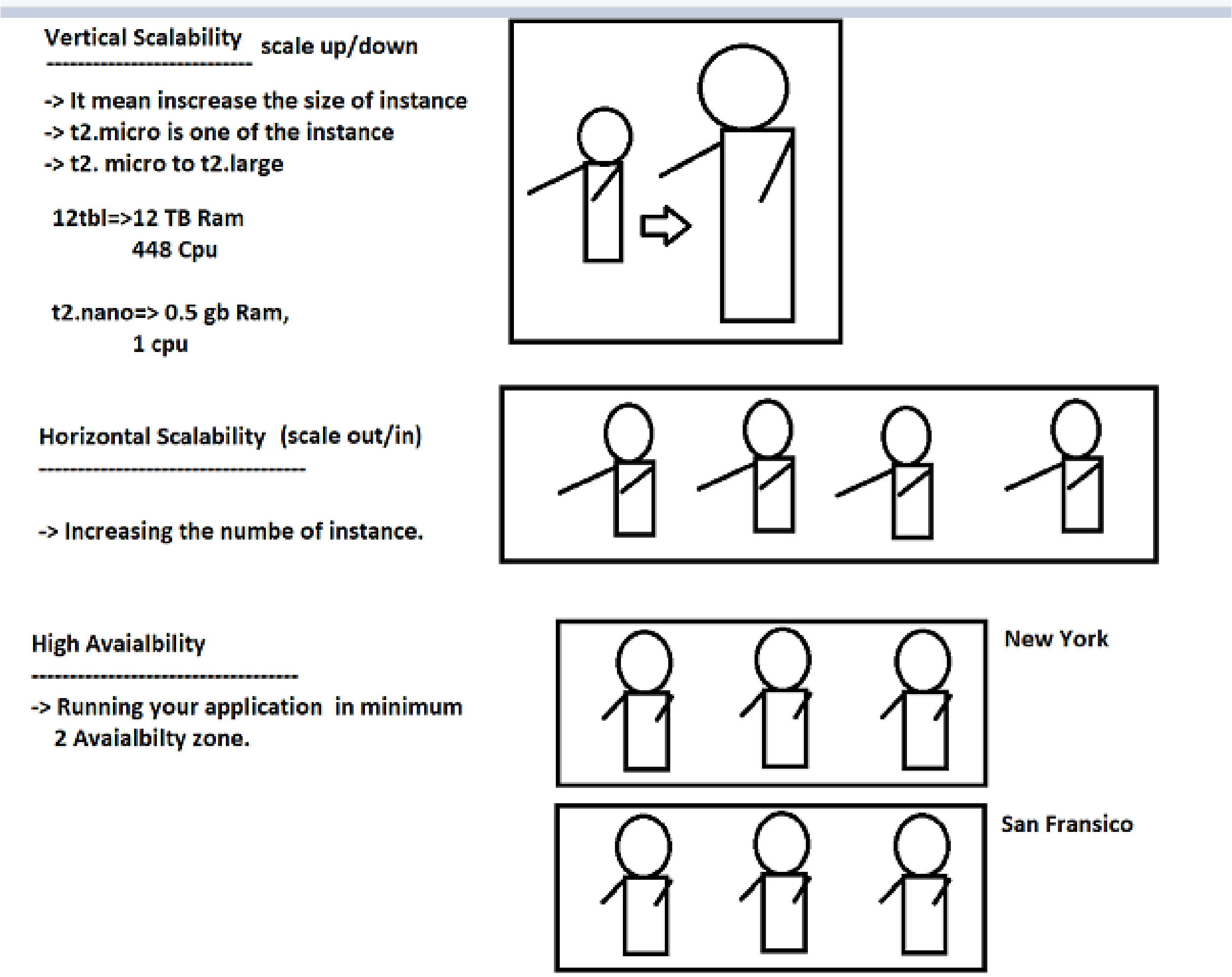
```
  "cat /etc/group"
```

* To see existing  group belong to user

=========================

```
  "id <usernames>"
```

# High Availability & Scalability



Vertical Scalability  scale up/down
---------------------------

-> It mean inscrease the size of instance
-> t2.micro is one of the instance
-> t2. micro to t2.large

12tbl=>12 TB Ram
448 Cpu

t2.nano=> 0.5 gb Ram,
1 cpu

Horizontal Scalability  (scale out/in)
-----------------------------------

-> Increasing the numbe of instance.

High Avaialbility
-----------------------------------

-> Running your application  in minimum
2 Avaialbilty zone.

- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
- Vertical Scalability
- Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability

## Vertical Scalability

- Vertically scalability means increasing the size of the instance
- For example, your application runs on a t2.micro
- Scaling that application vertically means running it on a t2.large
- Vertical scalability is very common for non distributed systems, such as a database.
- RDS, ElastiCache are services that can scale vertically.

- There’s usually a limit to how much you can vertically scale (hardware limit

## Horizontal Scalability

- Horizontal Scalability means increasing the number of instances / systems for your application
- Horizontal scaling implies distributed systems.
- This is very common for web applications / modern applications
- It’s easy to horizontally scale thanks the cloud offerings such as Amazon EC2

## High Availability

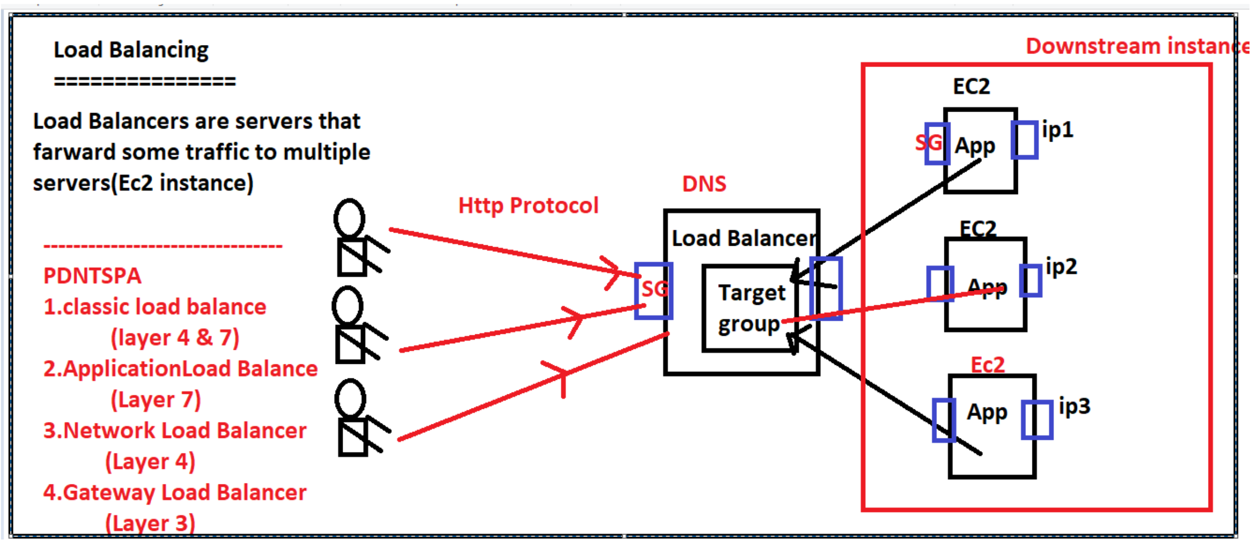
- High Availability usually goes hand in hand with horizontal scaling
- High availability means running your application / system in at least 2 data centers (== Availability Zones)
- The goal of high availability is to survive a data center loss
- The high availability can be passive (for RDS Multi AZ for example)
- The high availability

## What is load balancing?

- Load Balances are servers that forward traffic to multiple servers (e.g., EC2 instances) downstream

## Why use a load balancer

- Spread load across multiple downstream instances
- Expose a single point of access (DNS) to your application
- Seamlessly handle failures of downstream instances
- Do regular health checks to your instances
- Provide SSL termination (HTTPS) for your websites
- Enforce stickiness with cookies
- High availability across zones
- Separate public traffic from private traffic



Load Balancing
===============

Load Balancers are servers that farward some traffic to multiple servers(Ec2 instance)

------------------------------

PDNTSPA

1.classic load balance (layer 4 & 7)

2.ApplicationLoad Balance (Layer 7)

3.Network Load Balancer (Layer 4)

4.Gateway Load Balancer (Layer 3)

## Why use an Elastic Load Balancer?

- An Elastic Load Balancer is a managed load balancer
  - AWS guarantees that it will be working
  - AWS takes care of upgrades, maintenance, high availability
  - AWS provides only a few configuration knobs
- It costs less to setup your own load balancer but it will be a lot more effort on your end
- It is integrated with many AWS offerings / services
  - EC2, EC2 Auto Scaling Groups, Amazon ECS
- AWS Certificate Manager (ACM), CloudWatch
- Route 53, AWS WAF, AWS Global Accelerator

## Types of load balancer on AWS

- AWS has 4 kinds of managed Load Balancers
- Classic Load Balancer (v1 - old generation) – 2009 – CLB
- HTTP, HTTPS, TCP, SSL (secure TCP)
- Application Load Balancer (v2 - new generation) – 2016 – ALB
- HTTP, HTTPS, WebSocket
  - Network Load Balancer (v2 - new generation) – 2017 – NLB
- TCP, TLS (secure TCP), UDP
  - Gateway Load Balancer – 2020 – GWLB
- Operates at layer 3 (Network layer) – IP Protocol

## Application Load Balancer

- Application load balancers is Layer 7 (HTTP)
- Load balancing to multiple HTTP applications across machines (target groups)
- Load balancing to multiple applications on the same machine (ex: containers)
- Support for HTTP/2 and WebSocket
- Support redirects (from HTTP to HTTPS for example)

## Application Load Balancer

- Routing tables to different target groups:
- Routing based on path in URL (example.com/users & example.com/posts)
- Routing based on hostname in URL (one.example.com & other.example.com)
- Routing based on Query String, Headers (example.com/users?id=123&order=false)
- ALB are a great fit for micro services & container-based application (example: Docker & Amazon ECS)
- Has a port mapping feature to redirect to a dynamic port in ECS
- In comparison, we'd need multiple Classic Load Balancer per application

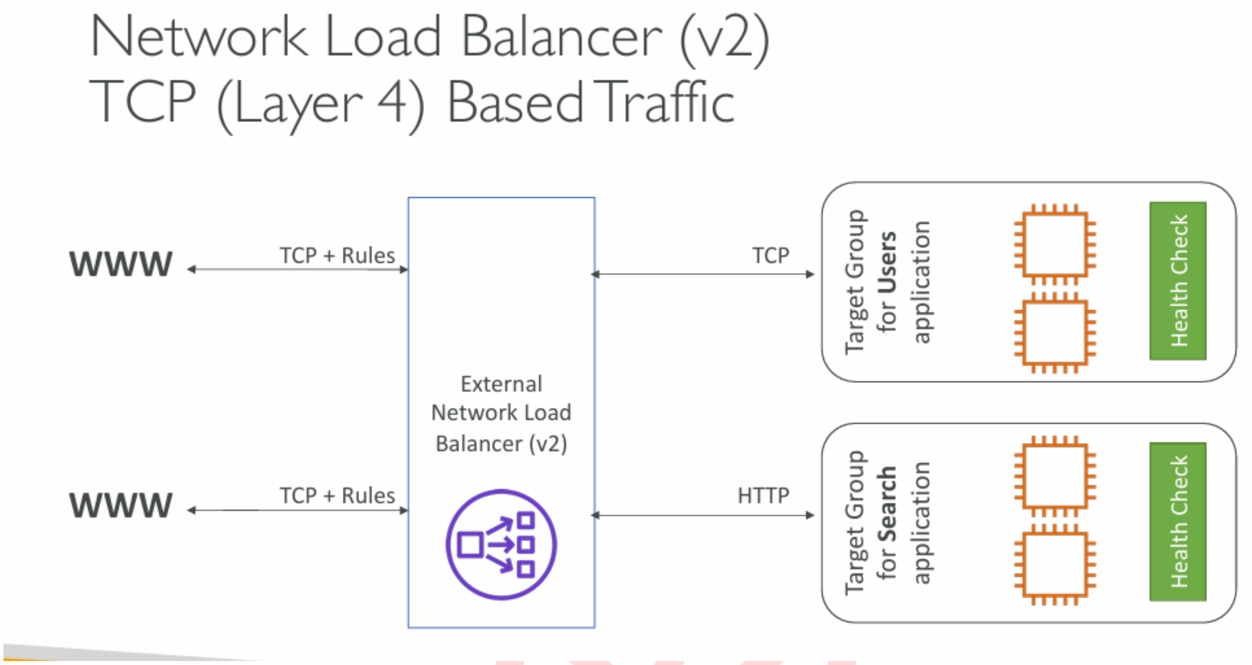
## Application Load Balancer  Target Groups

- EC2 instances (can be managed by an Auto Scaling Group) – HTTP
- ECS tasks (managed by ECS itself) – HTTP
- Lambda functions – HTTP request is translated into a JSON event
- IP Addresses – must be private IPs
- ALB can route to multiple target groups
- Health checks are at the target group level

## Network Load Balancer

- Network load balancers (Layer 4) allow to:
- Forward TCP & UDP traffic to your instances
- Handle millions of request per seconds
- Ultra-low latency
- NLB has one static IP per AZ , and supports assigning Elastic IP (helpful for whitelisting specific IP)

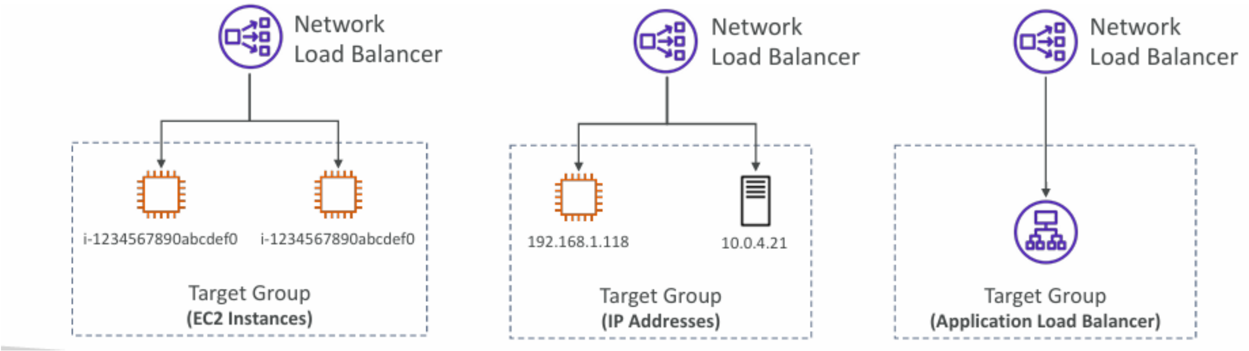
- NLB are used for extreme performance, TCP or UDP traffic
- Not included in the AWS free tier



Network Load Balancer (v2)
TCP (Layer 4) Based Traffic

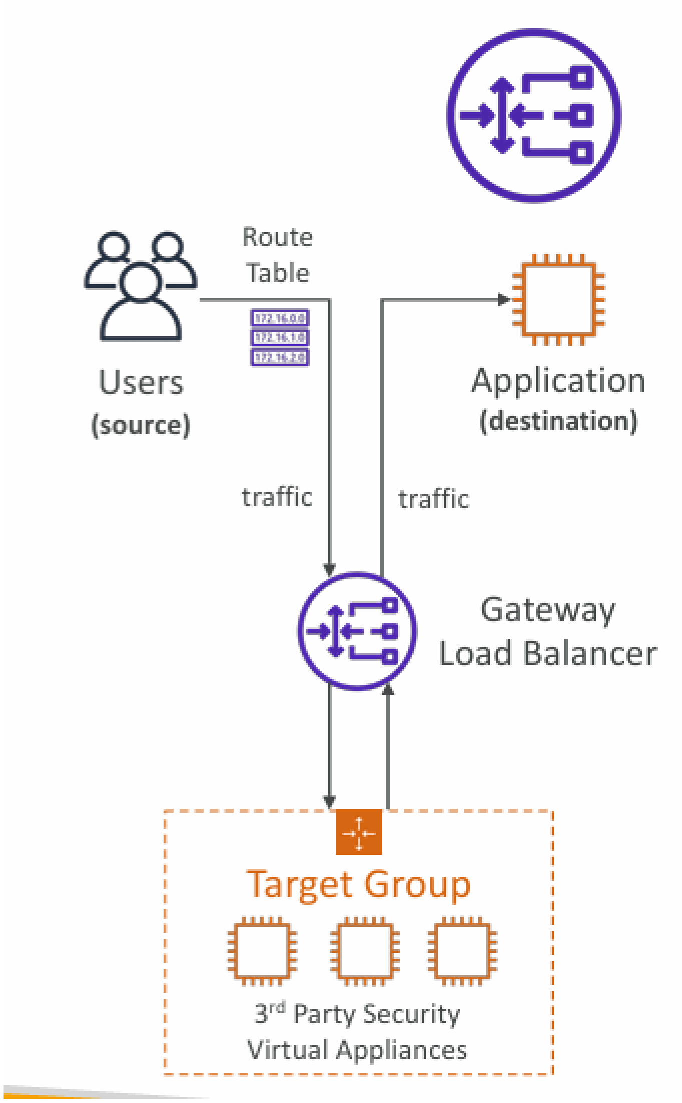
## Network Load Balancer – Target Groups

- EC2 instances
- IP Addresses – must be private IPs
- Application Load Balancer
- Health Checks support the TCP, HTTP and HTTPS Protocols



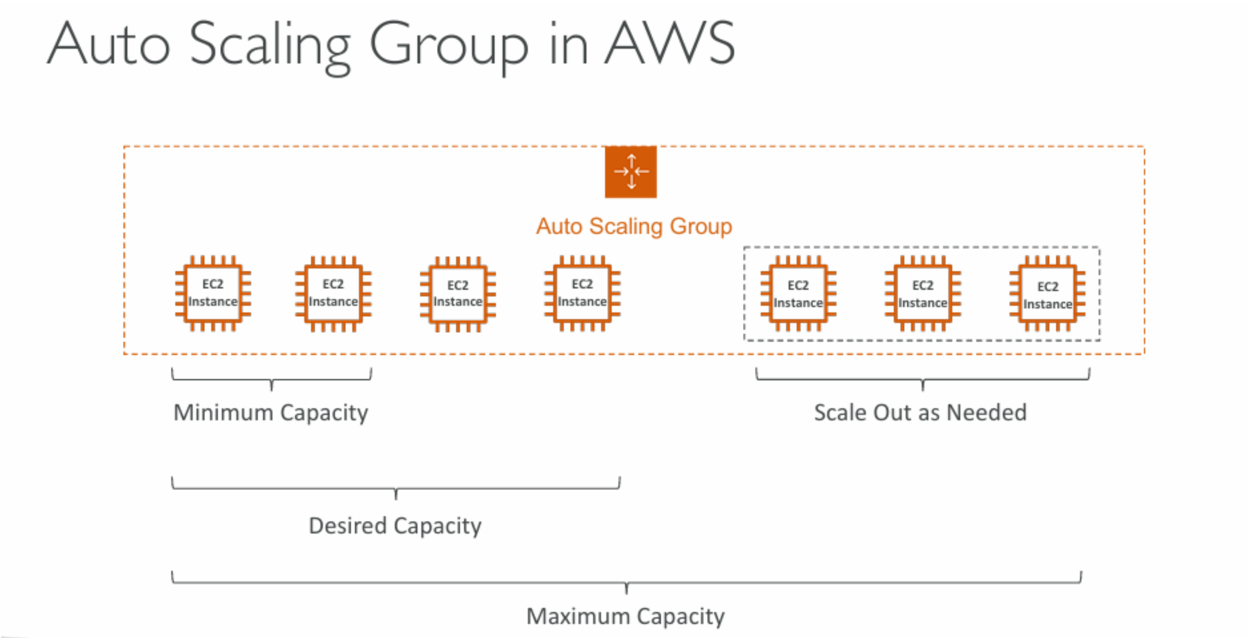
Gateway Load Balancer

- Deploy, scale, and manage a fleet of 3rd party network virtual appliances in AWS
- Example: Firewalls, Intrusion Detection and Prevention Systems, Deep Packet Inspection Systems, payload manipulation, …
- Operates at Layer 3 (Network Layer) – IP Packets
- Combines the following functions:
- Transparent Network Gateway – single entry/exit for all traffic
- Load Balancer – distributes traffic to your virtual appliances
- Uses the GENEVE protocol on port 6081



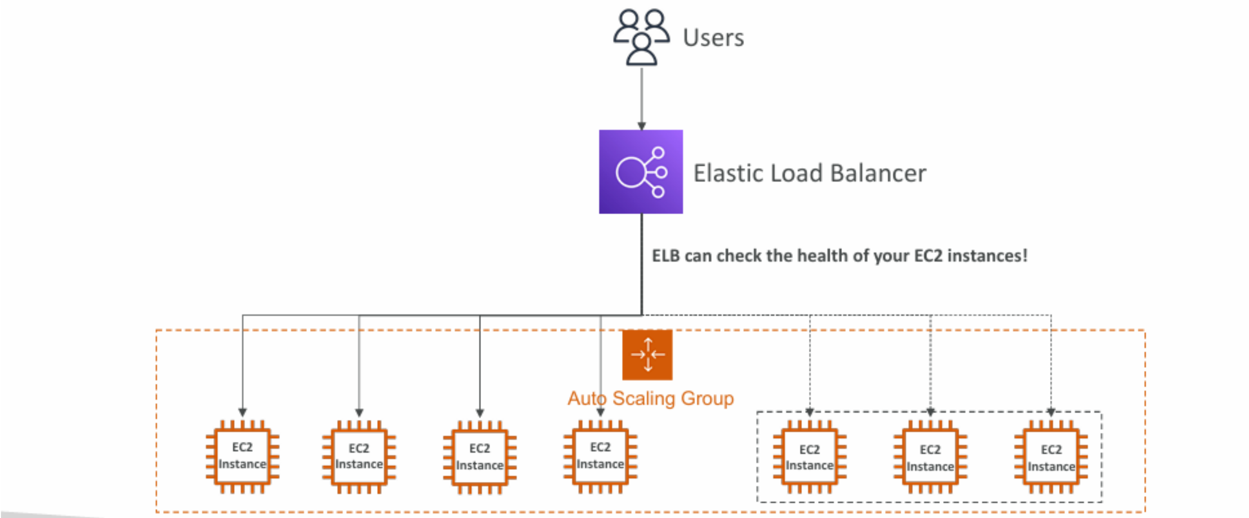
## What’s an Auto Scaling Group?

- In real-life, the load on your websites and application can change
- In the cloud, you can create and get rid of servers very quickly
- The goal of an Auto Scaling Group (ASG) is to:
- Scale out (add EC2 instances) to match an increased load
- Scale in (remove EC2 instances) to match a decreased load
- Ensure we have a minimum and a maximum number of EC2 instances running
- Automatically register new instances to a load balancer
- Re-create an EC2 instance in case a previous one is terminated (ex: if unhealthy)
- ASG are free (you only pay for the underlying EC2 instances)



Auto Scaling Group in AWS

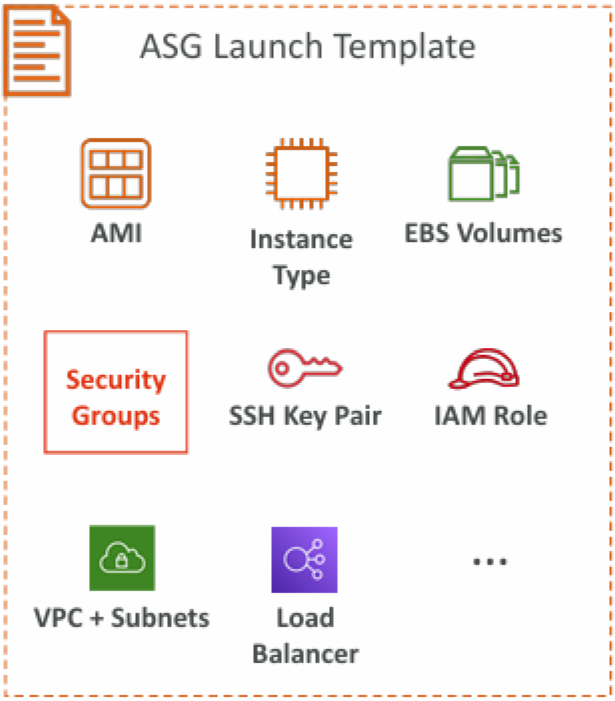
# Auto Scaling Group in AWS With Load Balancer



ELB can check the health of your EC2 instances!

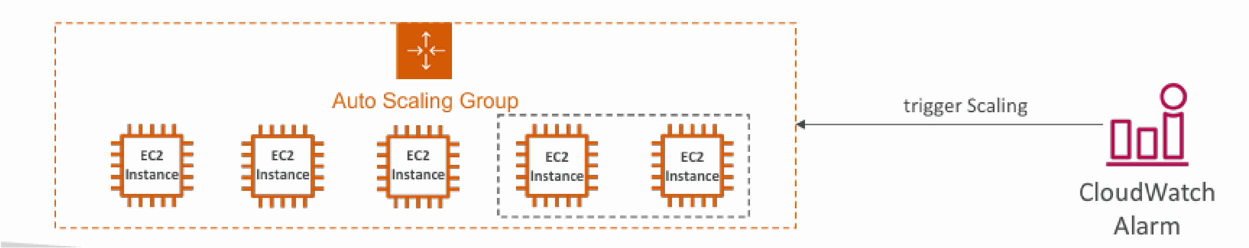
## Auto Scaling Group Attributes

- A Launch Template (older “Launch Configurations” are deprecated)
- AMI + Instance Type
- EC2 User Data
- EBS Volumes
- Security Groups
- SSH Key Pair
- IAM Roles for your EC2 Instances
- Network + Subnets Information
- Load Balancer Information
- Min Size / Max Size / Initial Capacity
- Scaling Policies



## Auto Scaling - CloudWatch Alarms & Scaling

- It is possible to scale an ASG based on CloudWatch alarms
  - An alarm monitors a metric (such as Average CPU, or a custom metric)
- Metrics such as Average CPU are computed for the overall ASG instances
- Based on the alarm:
  - We can create scale-out policies (increase the number of instances)
- We can create scale-in policies (decrease the number of instances)



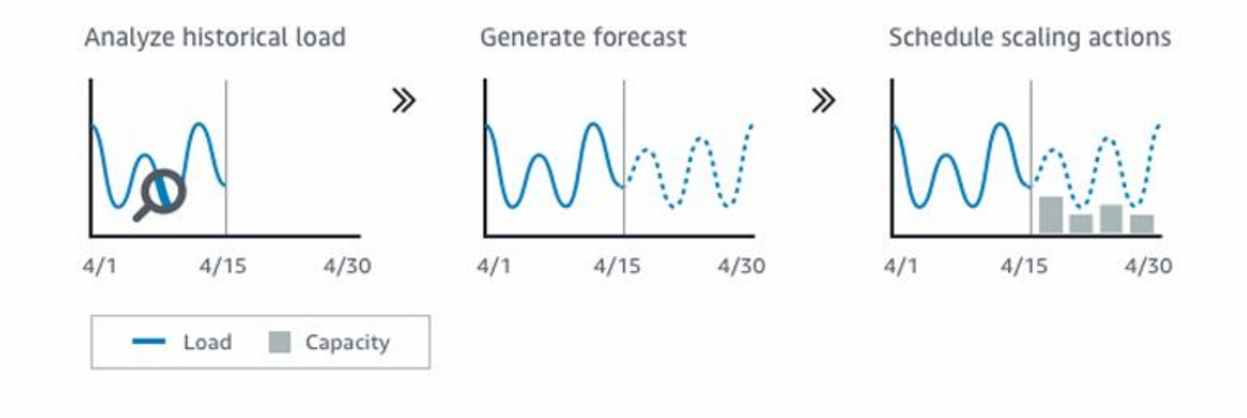
## Auto Scaling Groups – Scaling Policies

- Dynamic Scaling
- Target Tracking Scaling
- Simple to set-up
- Example: I want the average ASG CPU to stay at around 40%
- Simple / Step Scaling

- When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
- When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
- Scheduled Scaling • Anticipate a scaling based on known usage patterns
- Example: increase the min capacity to 10 at 5 pm on Fridays

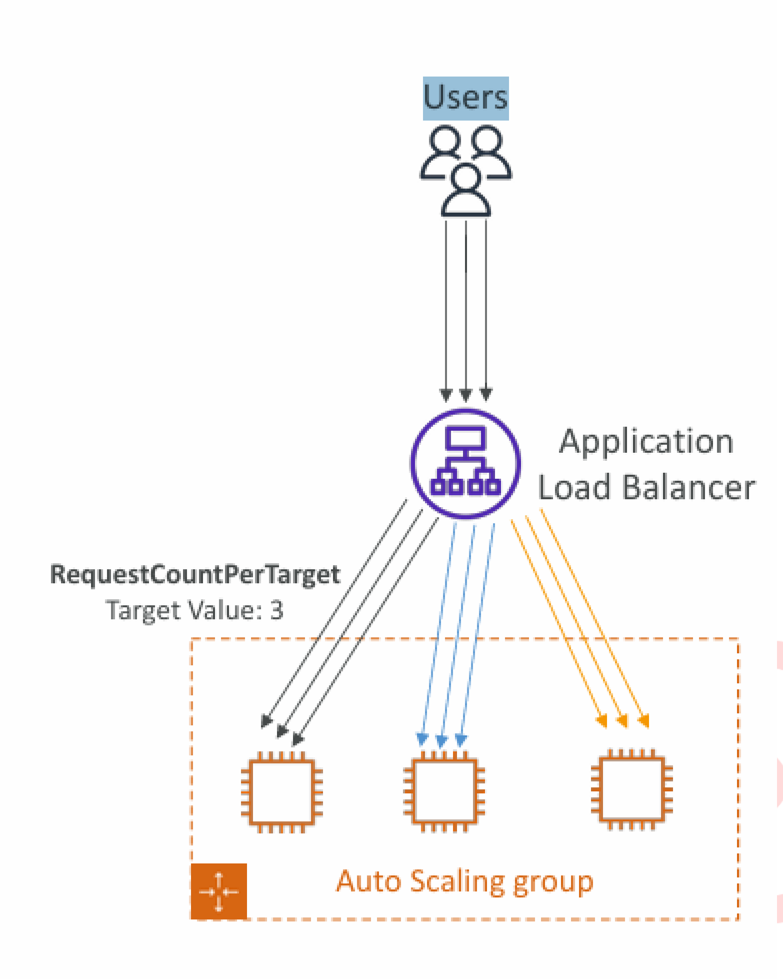
Auto Scaling Groups – Scaling Policies

- Predictive scaling: continuously forecast load and schedule scaling ahead



Good metrics to scale on

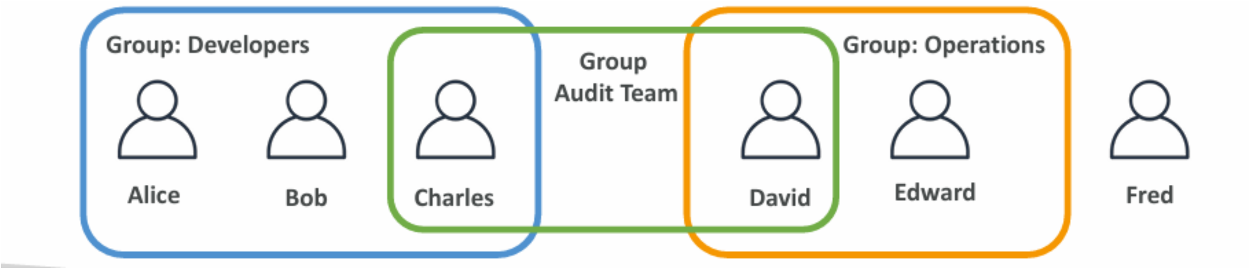
- CPU Utilization: Average CPU utilization across your instances
- RequestCountPerTarget: to make sure the number of requests per EC2 instances is stable
- Average Network In / Out (if you’re application is network bound)
- Any custom metric (that you push using CloudWatch)



# AWS Identity and Access Management (AWS IAM)

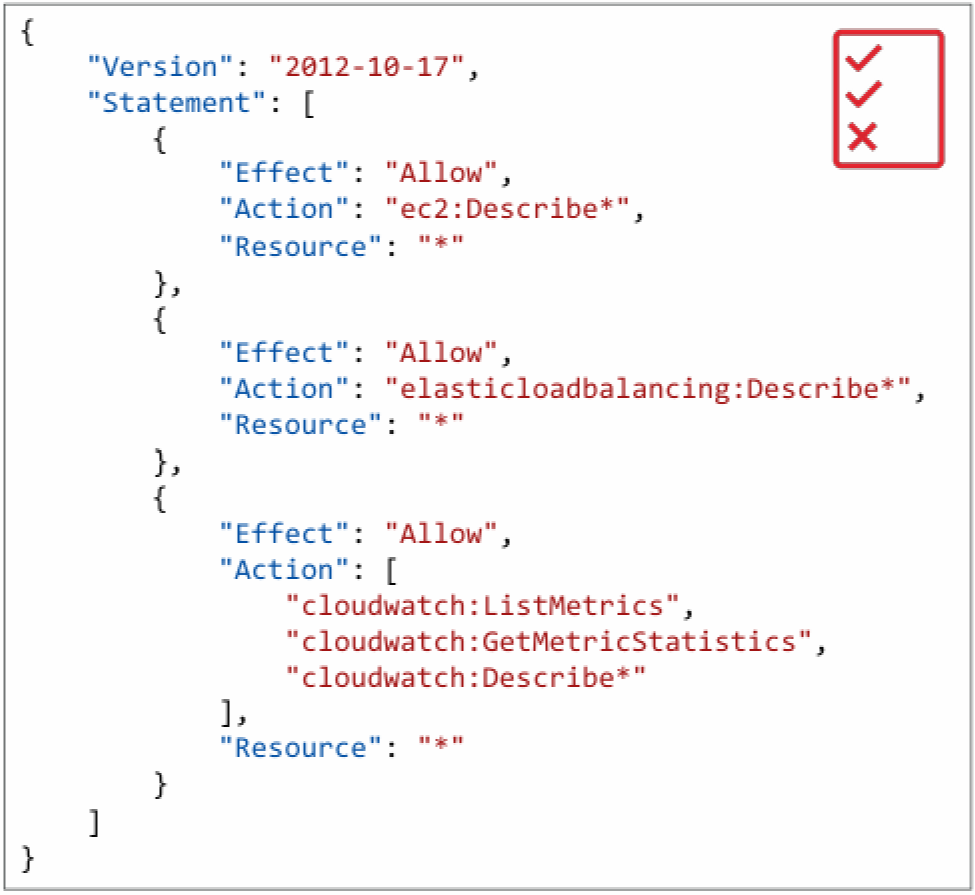
## IAM: Users & Group

- IAM = Identity and Access Management, Global service
- Root account created by default, shouldn’t be used or shared
- Users are people within your organization, and can be grouped
- Groups only contain users, not other groups
- Users don’t have to belong to a group, and user can belong to multiple groups

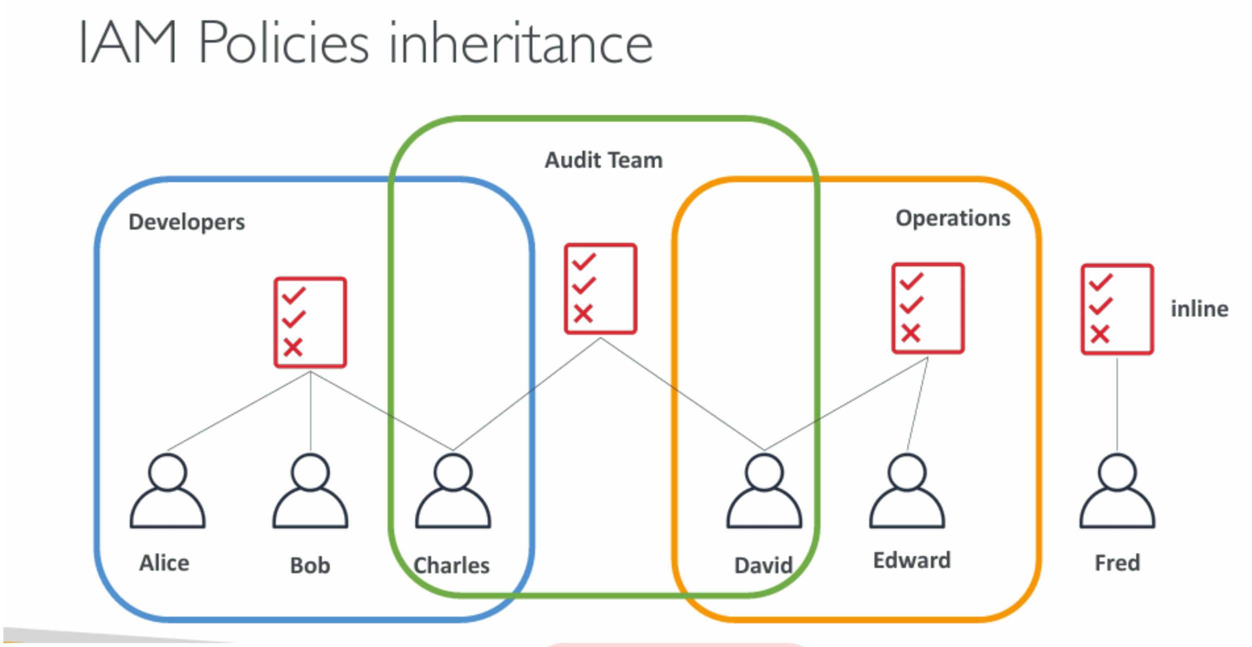


## IAM: Permissions

- Users or Groups can be assigned JSON documents called policies
- These policies define the permissions of the users
- In AWS you apply the least privilege principle: don’t give more permissions than a user needs



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```



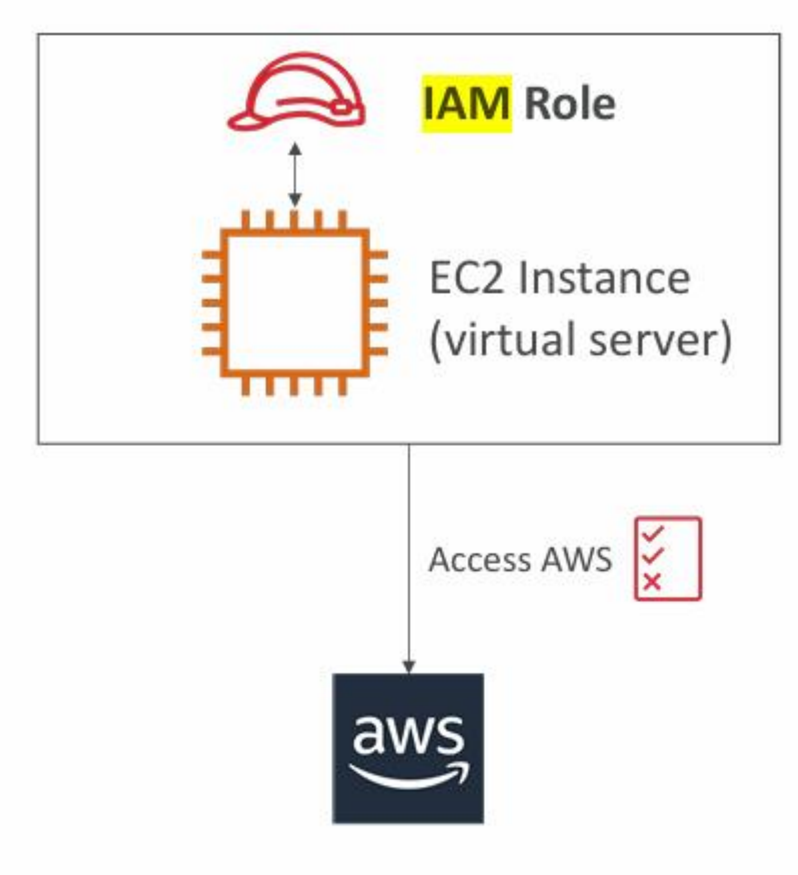
IAM Policies inheritance

## IAM – Password Policy

- Strong passwords = higher security for your account
- In AWS, you can setup a password policy:
- Set a minimum password length
- Require specific character types:
- including uppercase letters
- lowercase letters
- numbers
- non-alphanumeric characters
- Allow all IAM users to change their own passwords
- Require users to change their password after some time (password expiration)
- Prevent password re-use

## IAM Roles for Services

- Some AWS service will need to perform actions on your behalf
- To do so, we will assign permissions to AWS services with IAM Roles
- Common roles:
- EC2 Instance Roles



- Lambda Function Roles
- Roles for Cloud Formation

## S3 is used for storage purpose

- S3 is object based storage
- we can store flat files in s3.
- we can upload, download and access files from s3.
- we can not execute s3 files.
- we can not install any software in s3.
- S3 provide unlimited storage. It can be scale infinitely.
- we can attach s3 objects in Ec2.
- S3 supports static web hosting .
- S3 is serverless.
- In s3 we will store data in buckets.

- Buckets contain objects (object is nothing but files)
- key is called as name of the object.
- s3 is global service but buckets are regional specific.
- s3 buckets name should be unique.
- Always try to create an bucket name with your company name or project name with some details.
- We can not create one bucket inside another bucket.
- we can create multiple buckets in multiple region.
- Maximum we can create 100 buckets in s3 (soft limit)
- By default buckets are private.but if required we can make it public.
- Every buckets will have its own url/endpoints
- S3 follow WORM model(Write once Read Many)
- S3 is scalable,Highly available,Durable and secured services.
- In one bucket we can store objects

  Min size = 0 bytes  
  Max size = 5 TB  
  if we are uplaoding the size more than this  
  then it should be multi-part upload

S3 Use case:

-------------

- Backup and storage.
- Disaster recovery.
- Hybrid cloud storage
- Application Hosting
- Media Hosting
- Big Data Analaytics
- static websites

## Usecase

Nasdaq : It is an company who store its data for 7 years into s3 glacier.

Sysco : Who runs analytics on its data and gets insight of biz.

## S3-Buckets

- It allows people to store objects(files)
- Buckets are globally unique name(across all regions)
- Buckets are define at region level
- S3 look like global service, but buckets are created in regions

## Naming convention

- No uppercare
- No Underscore
- We can use ip
- must start with lowercase or number
- must not start with prefix "xn--"
- Most not end with suffix "-s3alias"

## S3- Objects

Objects have keys

The key full path (prefix + Object name)

s3://my-bucket/file1.txt

s3://my-bucket/my-folder/file2.txt

There is no concept of directories inside buckets.

Amazon S3 is a global service. NOT associated with a region.

HOWEVER a bucket is created in a specific AWS region

Objects are stored in buckets

Bucket names are globally unique

Bucket names are used as part of object URLs => Can contain ONLY lower case letters, numbers, hyphens and periods.

Unlimited objects in a bucket

Each object is identified by a key value pair

Key is unique in a bucket

Max object size is 5 TB

(Remember) No hierarchy of buckets, sub-buckets or folders

## Amazon S3 Versioning

Protects against accidental deletion

Versioning is optional and is enabled at bucket level

You can turn on versioning on a non versioned bucket

All old objects will have a version of null

You cannot turn off versioning on a versioned bucket

You can only suspend versioning

## Amazon S3 Static Website Hosting

Use S3 to host a static website using a bucket

Step 1 : Upload website content

Step 2 : Enable Static website hosting

Step 3 : Disable "Block public access"

Step 4 : Configure "Bucket policy" to enable public read access

## Policy in form of JSON

```
{
 "Id": "Policy1741658136418",
 "Version": "2012-10-17",
 "Statement": [
  {
    "Sid": "Stmt1741658131539",
    "Action": [
      "s3:GetObject"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:s3:::my-demo-vk-version-v2/*",
    "Principal": "*"
  }
 ]
}
```

## Amazon S3- Replication

- CRR : Cross Region Replication
- SRR : Same-Region Replication
- First we need to enable Versioning in source and destination bucket.
- Buckets can be in differnt AWS accounts.
- Copying of data is asynschronous

## S3 Storage classes

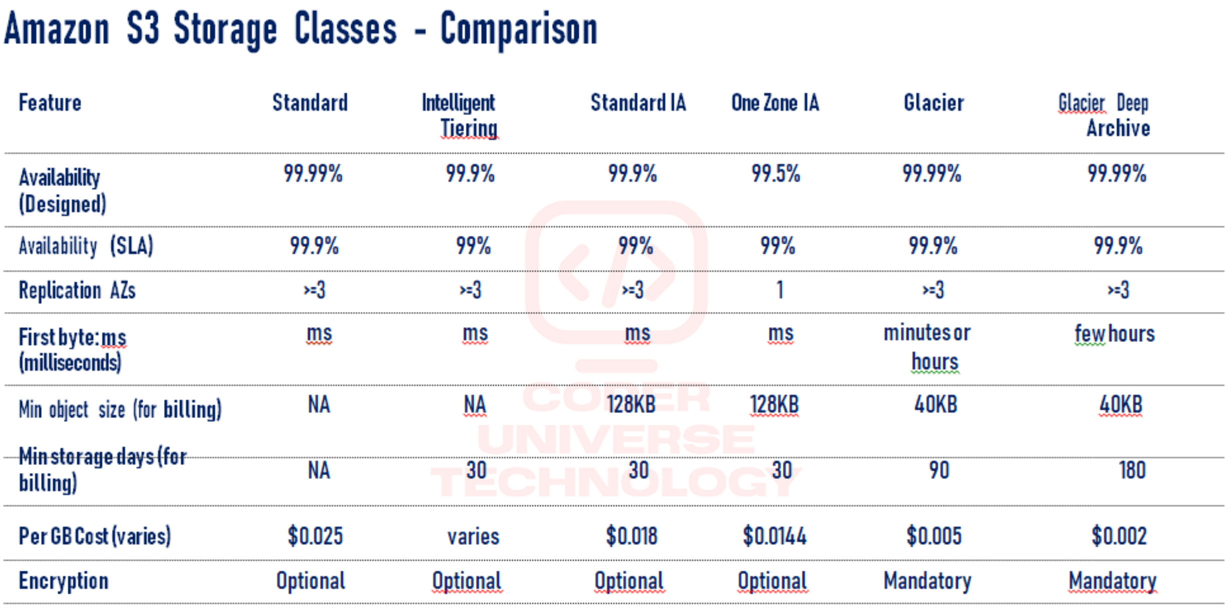
- S3 standard - General purpose
- S3 Standard - Infrequent Access
- S3 One Zone- Infrequent access
- S3 Glacier Instant retrieval
- s3 Glacier Flexible Retrieval
- S3 Glacier Deep Archive
- S3 Intelligent tiering

Note: we can move data between class manually or using s3 Lifecyle configurations

## S3 Durability and Availability

**Durability** 99.99999999999 % Durable If we store 1 crore objects then there may be chances loss of 1 object.

**Availability** : It is very high available. It depends on storage class.



## Amazon S3 Storage Classes - Comparison

| Feature | Standard | Intelligent Tiering | Standard IA | One Zone IA | Glacier | Glacier Deep Archive |
|---|---|---|---|---|---|---|
| Availability (Designed) | 99.99% | 99.9% | 99.9% | 99.5% | 99.99% | 99.99% |
| Availability (SLA) | 99.9% | 99% | 99% | 99% | 99.9% | 99.9% |
| Replication AZs | >=3 | >=3 | >=3 | 1 | >=3 | >=3 |
| First byte: ms (milliseconds) | ms | ms | ms | ms | minutes or hours | few hours |
| Min object size (for billing) | NA | NA | 128KB | 128KB | 40KB | 40KB |
| Min storage days (for billing) | NA | 30 | 30 | 30 | 90 | 180 |
| Per GB Cost (varies) | $0.025 | varies | $0.018 | $0.0144 | $0.005 | $0.002 |
| Encryption | Optional | Optional | Optional | Optional | Mandatory | Mandatory |

## Amazon S3 Replication

S3 Bucket
S3 Bucket

- Same Region and Multiple Region
- Replicate objects between buckets in same or different regions
- Could be cross account
- Can be configured at bucket level, a shared prefix level, or an object level using S3 object tags
- Access to destination bucket is provided using IAM Policy
- Versioning should be enabled on BOTH source and destination
- ONLY new objects are replicated (Explicitly copy existing objects)

- (Advantage) Reduces latency and helps you meet regulations
- (USECASE) Object replication between dev & test environments