Amazon Web Services - Cloud Computing

In 2006, **Amazon Web Services (AWS)** started to offer IT services to the market in the form of web services, which is nowadays known as **cloud computing**. With this cloud, we need not plan for servers and other IT infrastructure which takes up much of time in advance. Instead, these services can instantly spin up hundreds or thousands of servers in minutes and deliver results faster. We pay only for what we use with no up-front expenses and no long-term commitments, which makes AWS cost efficient.

Today, AWS provides a highly reliable, scalable, low-cost infrastructure platform in the cloud that powers multitude of businesses in 190 countries around the world.

What is Cloud Computing?

Cloud computing is an internet-based computing service in which large groups of remote servers are networked to allow centralized data storage, and online access to computer services or resources.

Using cloud computing, organizations can use shared computing and storage resources rather than building, operating, and improving infrastructure on their own.

Cloud computing is a model that enables the following features.

- Users can provision and release resources on-demand.
- Resources can be scaled up or down automatically, depending on the load.
- Resources are accessible over a network with proper security.
- Cloud service providers can enable a pay-as-you-go model, where customers are charged based on the type of resources and per usage.

Types of Clouds

There are three types of clouds − Public, Private, and Hybrid cloud.

Public Cloud

In public cloud, the third-party service providers make resources and services available to their customers via Internet. Customer’s data and related security is with the service providers’ owned infrastructure.

Private Cloud

A private cloud also provides almost similar features as public cloud, but the data and services are managed by the organization or by the third party only for the customer’s

organization. In this type of cloud, major control is over the infrastructure so security related issues are minimized.

Hybrid Cloud

A hybrid cloud is the combination of both private and public cloud. The decision to run on private or public cloud usually depends on various parameters like sensitivity of data and applications, industry certifications and required standards, regulations, etc.

Cloud Service Models

There are three types of service models in cloud − IaaS, PaaS, and SaaS.

IaaS

IaaS stands for **Infrastructure as a Service**. It provides users with the capability to provision processing, storage, and network connectivity on demand. Using this service model, the customers can develop their own applications on these resources.

PaaS

PaaS stands for **Platform as a Service**. Here, the service provider provides various services like databases, queues, workflow engines, e-mails, etc. to their customers. The customer can then use these components for building their own applications. The services, availability of resources and data backup are handled by the service provider that helps the customers to focus more on their application's functionality.

SaaS

SaaS stands for **Software as a Service**. As the name suggests, here the third-party providers provide end-user applications to their customers with some administrative capability at the application level, such as the ability to create and manage their users. Also some level of customizability is possible such as the customers can use their own corporate logos, colors, etc.

Advantages of Cloud Computing

Here is a list of some of the most important advantages that Cloud Computing has to offer −

- **Cost-Efficient** − Building our own servers and tools is time-consuming as well as expensive as we need to order, pay for, install, and configure expensive hardware, long before we need it. However, using cloud computing, we only pay for the amount we use and when we use the computing resources. In this manner, cloud computing is cost efficient.

- **Reliability** − A cloud computing platform provides much more managed, reliable and consistent service than an in-house IT infrastructure. It guarantees 24x7 and 365 days of service. If any of the server fails, then hosted applications and services can easily be transited to any of the available servers.
- **Unlimited Storage** − Cloud computing provides almost unlimited storage capacity, i.e., we need not worry about running out of storage space or increasing our current storage space availability. We can access as much or as little as we need.
- **Backup & Recovery** − Storing data in the cloud, backing it up and restoring the same is relatively easier than storing it on a physical device. The cloud service providers also have enough technology to recover our data, so there is the convenience of recovering our data anytime.
- **Easy Access to Information** − Once you register yourself in cloud, you can access your account from anywhere in the world provided there is internet connection at that point. There are various storage and security facilities that vary with the account type chosen.

Disadvantages of Cloud Computing

Although Cloud Computing provides a wonderful set of advantages, it has some drawbacks as well that often raise questions about its efficiency.

Security issues

Security is the major issue in cloud computing. The cloud service providers implement the best security standards and industry certifications, however, storing data and important files on external service providers always bears a risk.

AWS cloud infrastructure is designed to be the most flexible and secured cloud network. It provides scalable and highly reliable platform that enables customers to deploy applications and data quickly and securely.

Technical issues

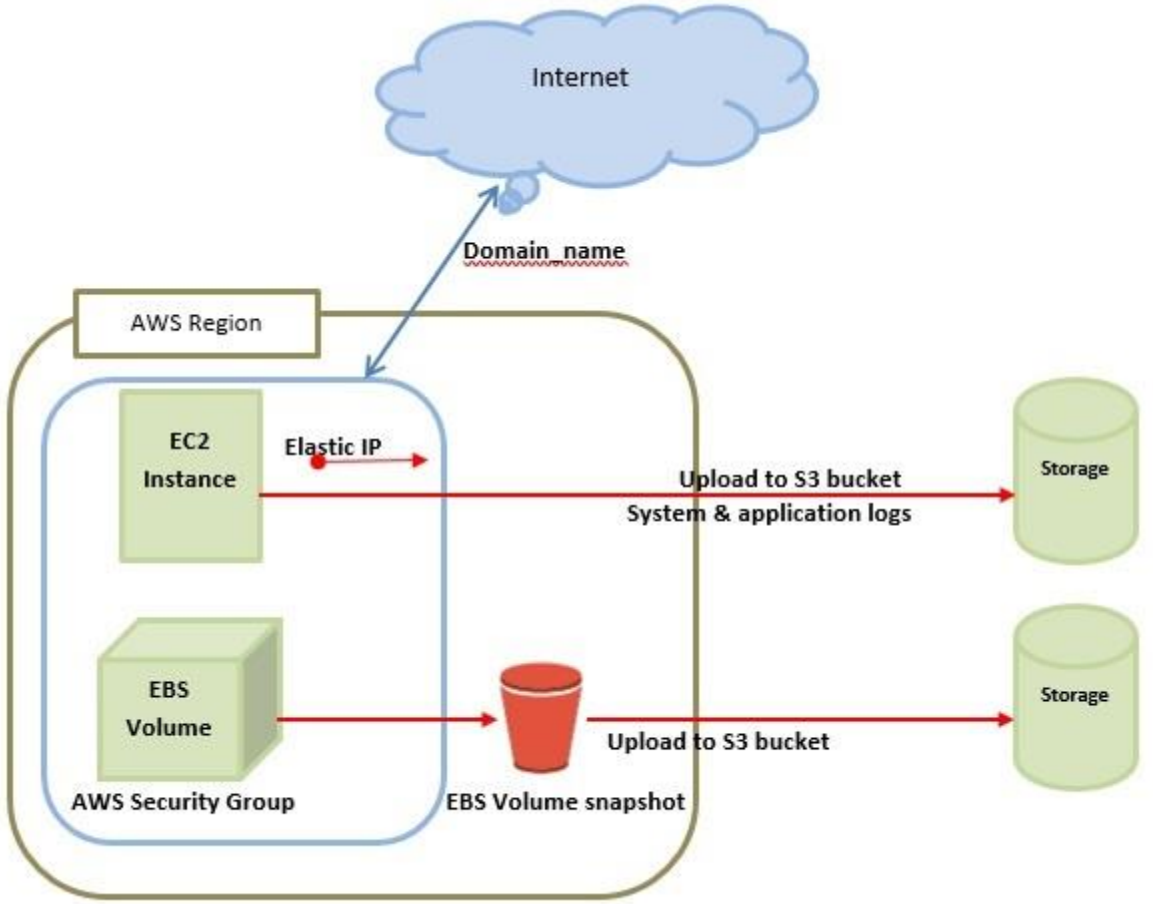
As cloud service providers offer services to number of clients each day, sometimes the system can have some serious issues leading to business processes temporarily being suspended. Additionally, if the internet connection is offline then we will not be able to access any of the applications, server, or data from the cloud.

Not easy to switch service providers

Cloud service providers promises vendors that the cloud will be flexible to use and integrate, however switching cloud services is not easy. Most organizations may find it difficult to host and integrate current cloud applications on another platform.

Amazon Web Services - Basic Architecture

This is the basic structure of **AWS EC2**, where **EC2** stands for Elastic Compute Cloud. EC2 allow users to use virtual machines of different configurations as per their requirement. It allows various configuration options, mapping of individual server, various pricing options, etc. We will discuss these in detail in AWS Products section. Following is the diagrammatic representation of the architecture.



Note − In the above diagram **S3** stands for Simple Storage Service. It allows the users to store and retrieve various types of data using API calls. It doesn't contain any computing element. We will discuss this topic in detail in AWS products section.

Load Balancing

Load balancing simply means to hardware or software load over web servers, that improver's the efficiency of the server as well as the application. Following is the diagrammatic representation of AWS architecture with load balancing.

Hardware load balancer is a very common network appliance used in traditional web application architectures.

AWS provides the Elastic Load Balancing service, it distributes the traffic to EC2 instances across multiple available sources, and dynamic addition and removal of Amazon EC2 hosts from the load-balancing rotation.

Elastic Load Balancing can dynamically grow and shrink the load-balancing capacity to adjust to traffic demands and also support sticky sessions to address more advanced routing needs.

Amazon Cloud-front

It is responsible for content delivery, i.e. used to deliver website. It may contain dynamic, static, and streaming content using a global network of edge locations. Requests for content at the user's end are automatically routed to the nearest edge location, which improves the performance.

Amazon Cloud-front is optimized to work with other Amazon Web Services, like Amazon S3 and Amazon EC2. It also works fine with any non-AWS origin server and stores the original files in a similar manner.

In Amazon Web Services, there are no contracts or monthly commitments. We pay only for as much or as little content as we deliver through the service.

Elastic Load Balancer

It is used to spread the traffic to web servers, which improves performance. AWS provides the Elastic Load Balancing service, in which traffic is distributed to EC2 instances over multiple available zones, and dynamic addition and removal of Amazon EC2 hosts from the load-balancing rotation.

Elastic Load Balancing can dynamically grow and shrink the load-balancing capacity as per the traffic conditions.

Security Management

Amazon’s Elastic Compute Cloud (EC2) provides a feature called security groups, which is similar to an inbound network firewall, in which we have to specify the protocols, ports, and source IP ranges that are allowed to reach your EC2 instances.

Each EC2 instance can be assigned one or more security groups, each of which routes the appropriate traffic to each instance. Security groups can be configured using specific subnets or IP addresses which limits access to EC2 instances.

Elastic Caches

Amazon Elastic Cache is a web service that manages the memory cache in the cloud. In memory management, cache has a very important role and helps to reduce the load on the services, improves the performance and scalability on the database tier by caching frequently used information.

Amazon RDS

Amazon RDS (Relational Database Service) provides a similar access as that of MySQL, Oracle, or Microsoft SQL Server database engine. The same queries, applications, and tools can be used with Amazon RDS.

It automatically patches the database software and manages backups as per the user's instruction. It also supports point-in-time recovery. There are no up-front investments required, and we pay only for the resources we use.

Hosting RDMS on EC2 Instances

Amazon RDS allows users to install RDBMS (Relational Database Management System) of your choice like MySQL, Oracle, SQL Server, DB2, etc. on an EC2 instance and can manage as required.

Amazon EC2 uses Amazon EBS (Elastic Block Storage) similar to network-attached storage. All data and logs running on EC2 instances should be placed on Amazon EBS volumes, which will be available even if the database host fails.

Amazon EBS volumes automatically provide redundancy within the availability zone, which increases the availability of simple disks. Further if the volume is not sufficient for our databases needs, volume can be added to increase the performance for our database.

Using Amazon RDS, the service provider manages the storage and we only focus on managing the data.

Storage & Backups

AWS cloud provides various options for storing, accessing, and backing up web application data and assets. The Amazon S3 (Simple Storage Service) provides a simple web-services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.

Amazon S3 stores data as objects within resources called **buckets**. The user can store as many objects as per requirement within the bucket, and can read, write and delete objects from the bucket.

Amazon EBS is effective for data that needs to be accessed as block storage and requires persistence beyond the life of the running instance, such as database partitions and application logs.

Amazon EBS volumes can be maximized up to 1 TB, and these volumes can be striped for larger volumes and increased performance. Provisioned IOPS volumes are designed to meet the needs of database workloads that are sensitive to storage performance and consistency.

Amazon EBS currently supports up to 1,000 IOPS per volume. We can stripe multiple volumes together to deliver thousands of IOPS per instance to an application.

Auto Scaling

The difference between AWS cloud architecture and the traditional hosting model is that AWS can dynamically scale the web application fleet on demand to handle changes in traffic.

In the traditional hosting model, traffic forecasting models are generally used to provision hosts ahead of projected traffic. In AWS, instances can be provisioned on the fly according to a set of triggers for scaling the fleet out and back in. Amazon Auto Scaling can create capacity groups of servers that can grow or shrink on demand.

Key Considerations for Web Hosting in AWS

Following are some of the key considerations for web hosting −

No physical network devices needed

In AWS, network devices like firewalls, routers, and load-balancers for AWS applications no longer reside on physical devices and are replaced with software solutions.

Multiple options are available to ensure quality software solutions. For load balancing choose Zeus, HAProxy, Nginx, Pound, etc. For establishing a VPN connection choose OpenVPN, OpenSwan, Vyatta, etc.

No security concerns

AWS provides a more secured model, in which every host is locked down. In Amazon EC2, security groups are designed for each type of host in the architecture, and a large variety of simple and tiered security models can be created to enable minimum access among hosts within your architecture as per requirement.

Availability of data centers

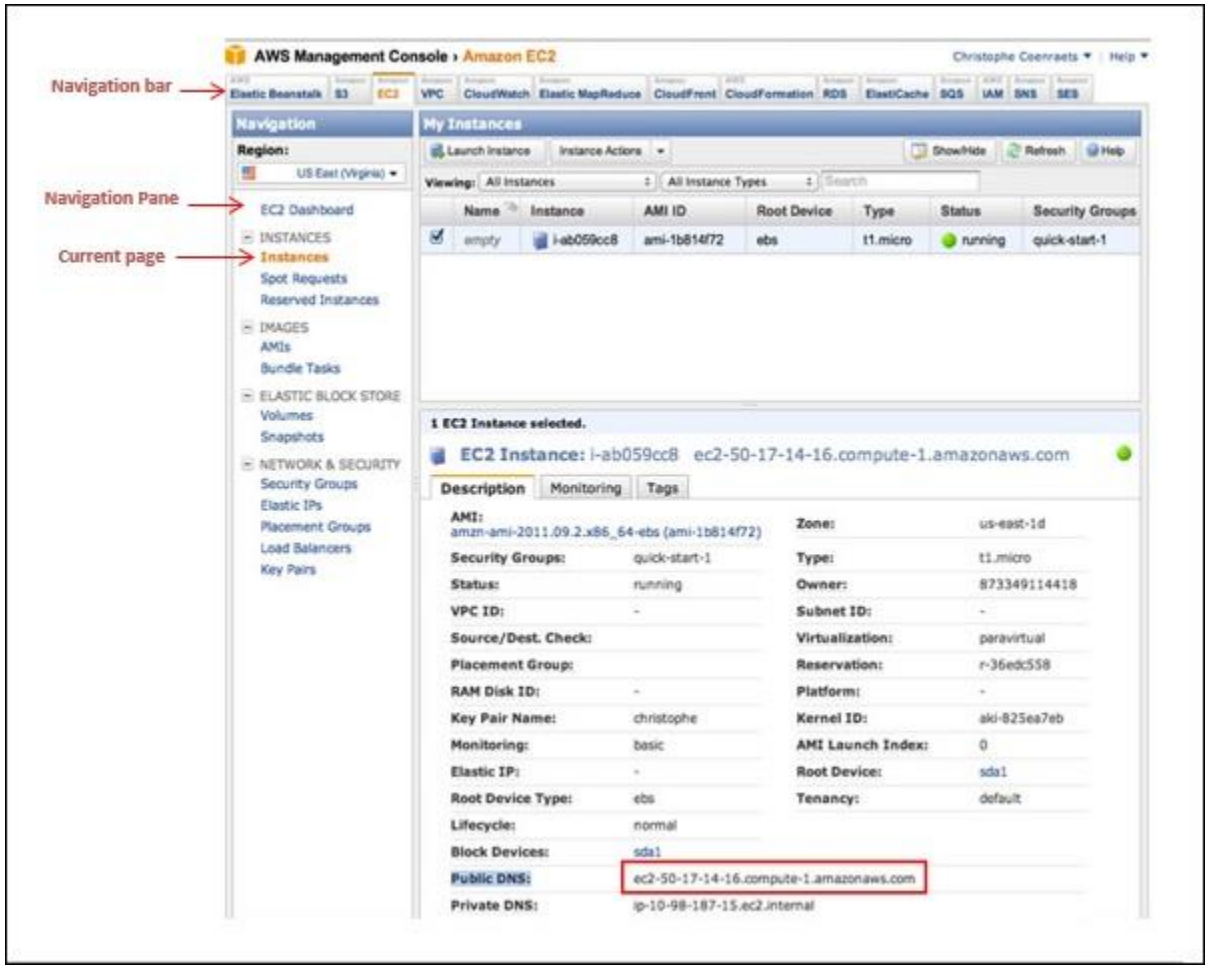
EC2 instances are easily available at most of the availability zones in AWS region and provides model for deploying your application across data centers for both high availability and reliability.

AWS - Management Console

AWS Management Console is a web application for managing Amazon Web Services. AWS Management Console consists of list of various services to choose from. It also provides all information related to our account like billing.

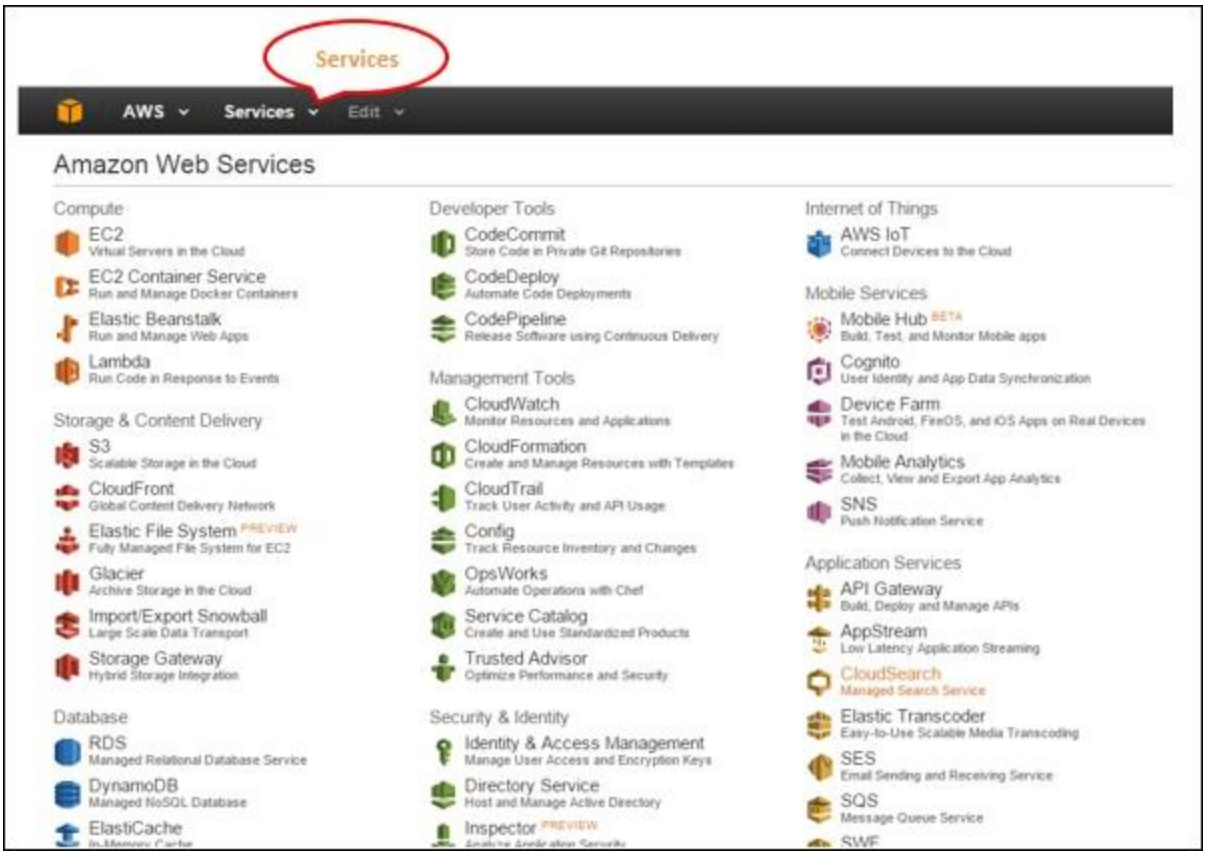
This console provides an inbuilt user interface to perform AWS tasks like working with Amazon S3 buckets, launching and connecting to Amazon EC2 instances, setting Amazon CloudWatch alarms, etc.

Following is the screenshot of AWS management console for Amazon EC2 service.



How to Access AWS?

Step 1 − Click on services. We get a list of various services.



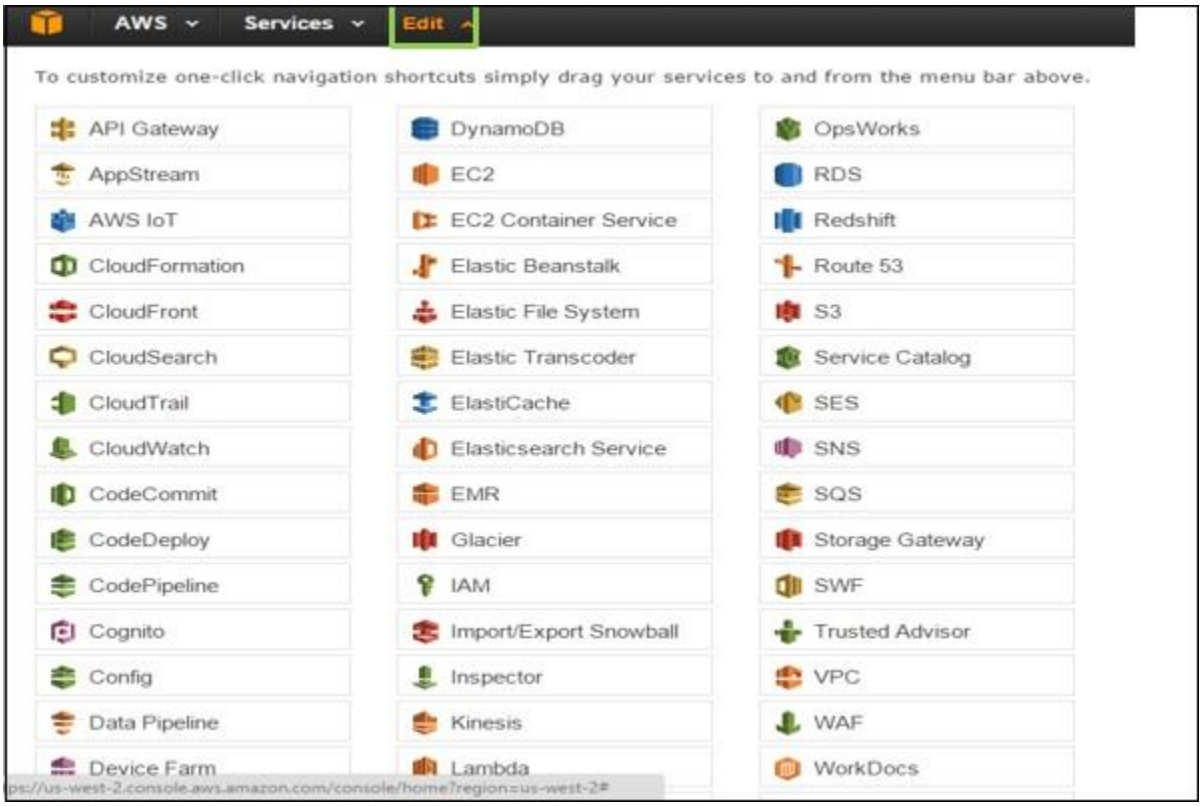
Step 2 − Select the choice from the list of categories and we get their sub-categories such as Computer and Database category is selected in the following screenshots.

Step 3 − Select the service of your choice and the console of that service will open.

Customizing the Dashboard

Creating Services Shortcuts

Click the Edit menu on the navigation bar and a list of services appears. We can create their shortcuts by simply dragging them from the menu bar to the navigation bar.



Adding Services Shortcuts

When we drag the service from the menu bar to the navigation bar, the shortcut will be created and added. We can also arrange them in any order. In the following screenshot we have created shortcut for S3, EMR and DynamoDB services.

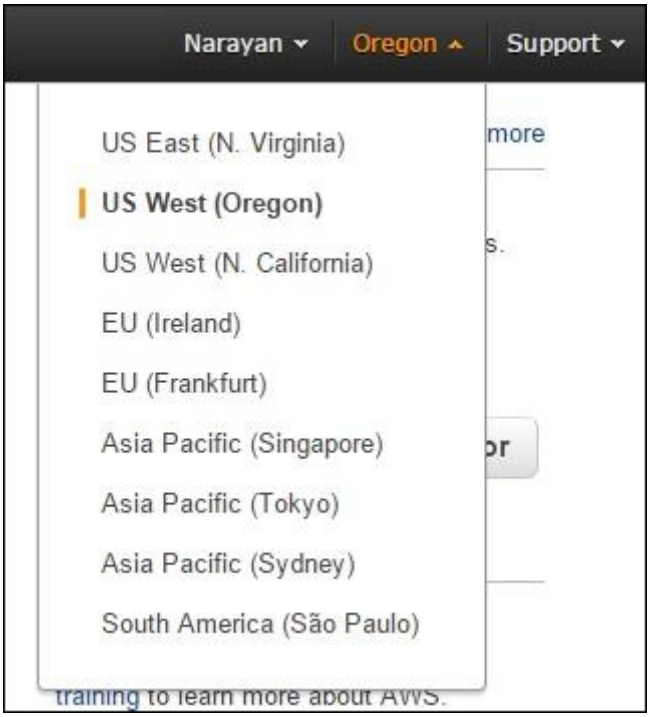
Deleting Services Shortcuts

To delete the shortcut, click the edit menu and drag the shortcut from the navigation bar to the service menu. The shortcut will be removed. In the following screenshot, we have removed the shortcut for EMR services.

Selecting a Region

Many of the services are region specific and we need to select a region so that resources can be managed. Some of the services do not require a region to be selected like AWS Identity and Access Management (IAM).

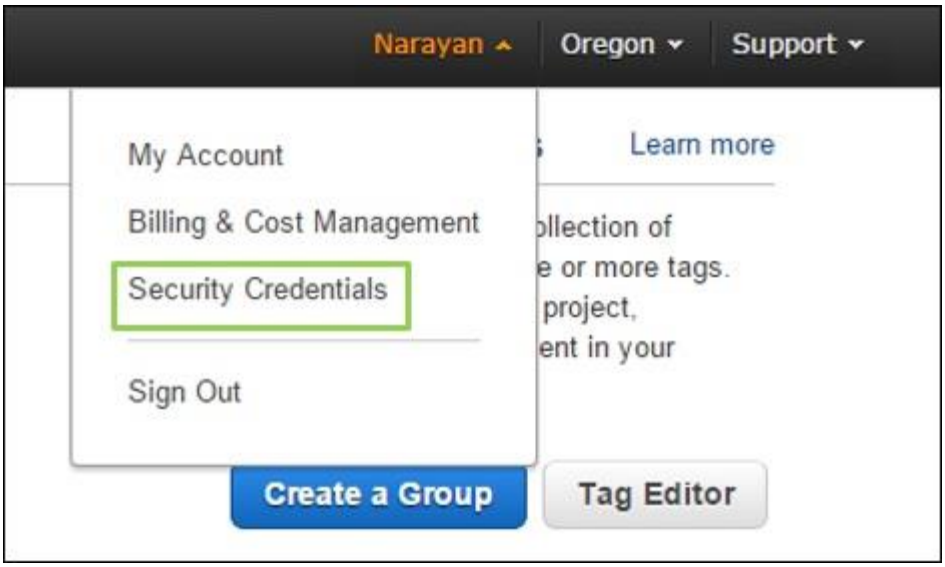
To select a region, first we need to select a service. Click the Oregon menu (on the left side of the console) and then select a region



Changing the Password

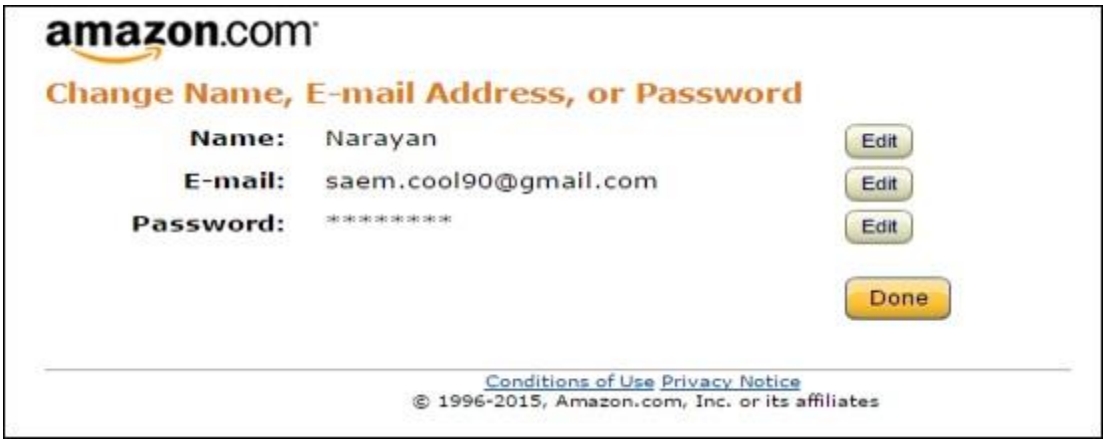
We can change password of our AWS account. To change the password, following are the steps.

Step 1 − Click the account name on the left side of the navigation bar.



Step 2 − Choose Security Credentials and a new page will open having various options. Select the password option to change the password and follow the instructions.

Step 3 − After signing-in, a page opens again having certain options to change the password and follow the instructions.

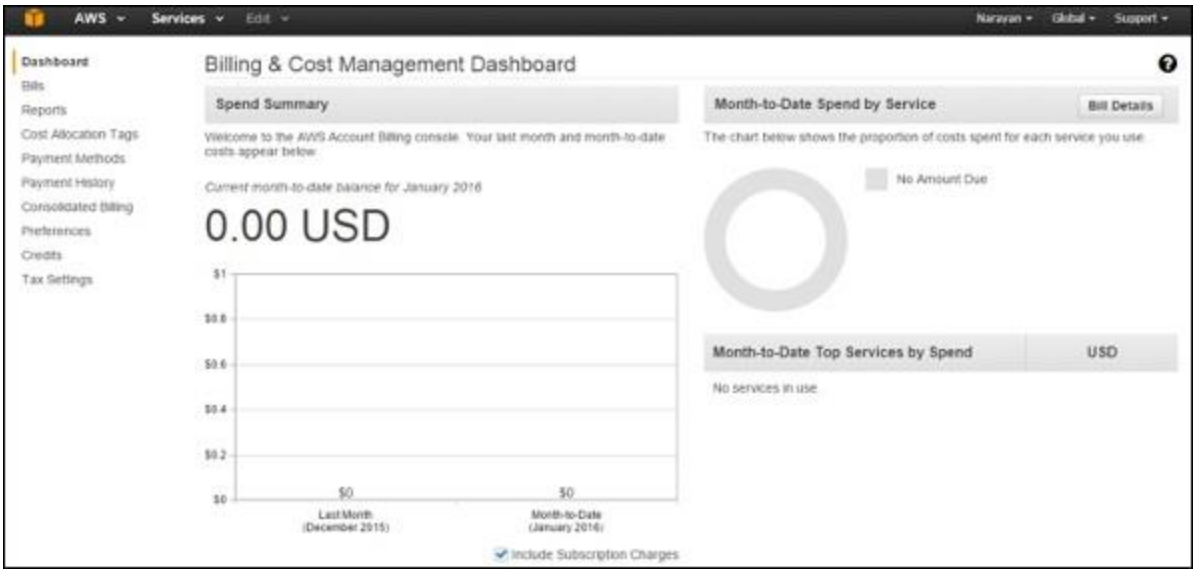


When successful, we will receive a confirmation message.

Know Your Billing Information

Click the account name in the navigation bar and select the 'Billing & Cost Management' option.

Now a new page will open having all the information related to money section. Using this service, we can pay AWS bills, monitor our usage and budget estimation.



Amazon Web Services - Account

How to Use AWS Account?

Following are the steps to access AWS services −

- Create an AWS account.
- Sign-up for AWS services.
- Create your password and access your account credentials.
- Activate your services in credits section.

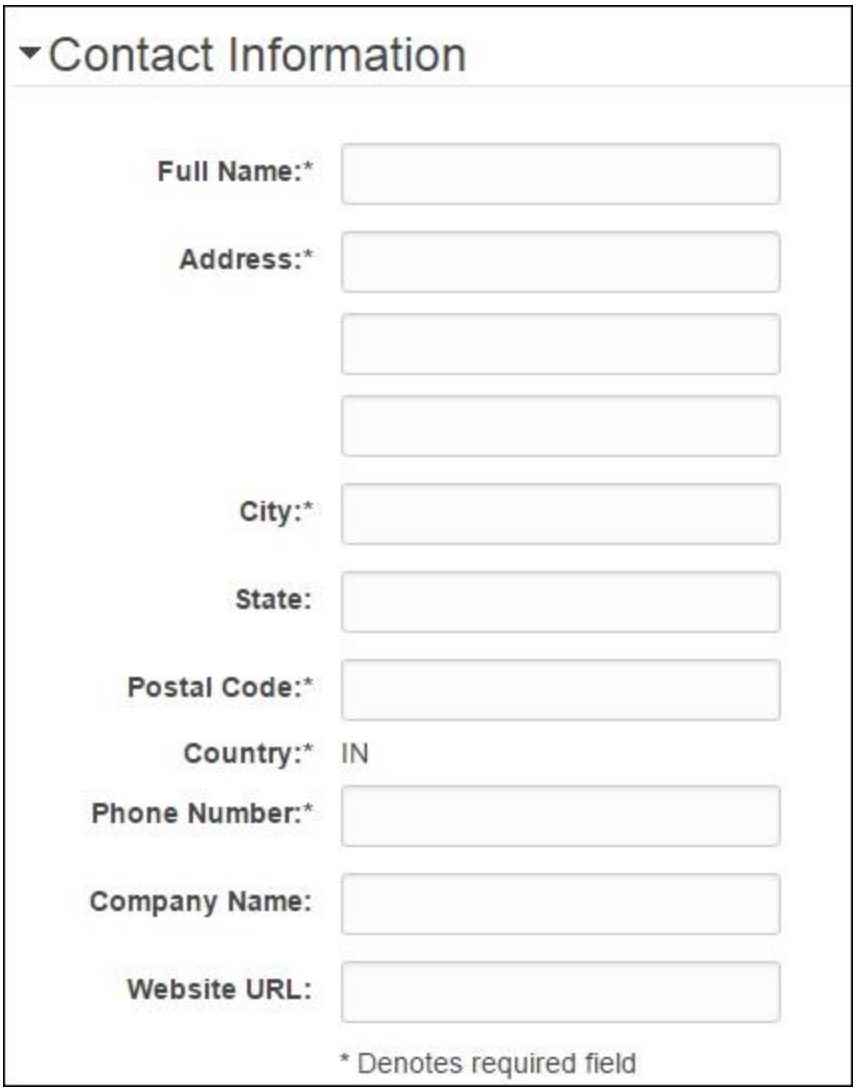
Create an AWS Account

Amazon provides a fully functional free account for one year for users to use and learn the different components of AWS. You get access to AWS services like EC2, S3, DynamoDB, etc. for free. However, there are certain limitations based on the resources consumed.

Step 1 − To create an AWS account, open this link https://aws.amazon.com and sign-up for new account and enter the required details.

If we already have an account, then we can sign-in using the existing AWS password.

Step 2 − After providing an email-address, complete this form. Amazon uses this information for billing, invoicing and identifying the account. After creating the account, sign-up for the services needed.



Step 3 − To sign-up for the services, enter the payment information. Amazon executes a minimal amount transaction against the card on the file to check that it is valid. This charge varies with the region.

Step 4 − Next, is the identity verification. Amazon does a call back to verify the provided contact number.

Step 5 − Choose a support plan. Subscribe to one of the plans like Basic, Developer, Business, or Enterprise. The basic plan costs nothing and has limited resources, which is good to get familiar with AWS.

Step 6 − The final step is confirmation. Click the link to login again and it redirects to AWS management console.

Now the account is created and can be used to avail AWS services.

AWS Account Identifiers

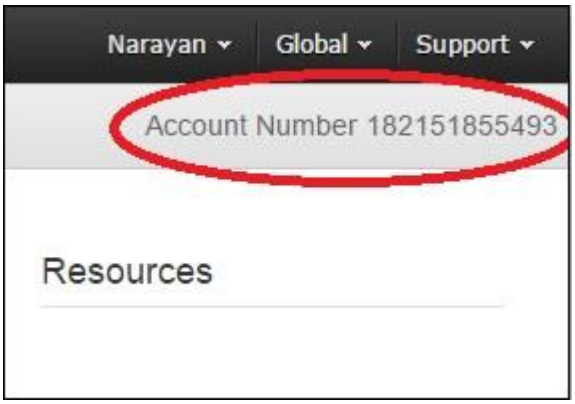
AWS assigns two unique IDs to each AWS account.

- An AWS account ID
- A conical user ID

AWS Account ID

It is a 12-digit number like 123456789000 and is used to construct Amazon Resource Names (ARN). This ID helps to distinguish our resources from resources in other AWS accounts.

To know the AWS account number, click Support on the upper right side of the navigation bar in AWS management console as shown in the following screenshot.



Conical String User ID

It is a long string of alphanumeric characters like 1234abcdef1234. This ID is used in Amazon S3 bucket policy for cross-account access, i.e. to access resources in another AWS account.

Account Alias

Account alias is the URL for your sign-in page and contains the account ID by default. We can customize this URL with the company name and even overwrite the previous one.

How to Create/Delete Your Own AWS Account Alias?

Step 1 − Sign in to the AWS management console and open the IAM console using the following link https://console.aws.amazon.com/iam/

Step 2 − Select the customize link and create an alias of choice.

Step 3 − To delete the alias, click the customize link, then click the Yes, Delete button. This deletes the alias and it reverts to the Account ID.

Multi Factor Authentication

Multi Factor Authentication (MFA) provides additional security by authenticating the users to enter a unique authentication code from an approved authentication device or SMS text message when they access AWS websites or services. If the MFA code is correct, then only the user can access AWS services or else not.

Requirements

To use MFA services, the user has to assign a device (hardware or virtual) to IAM user or AWS root account. Each MFA device assigned to the user must be unique, i.e. the user cannot enter a code from another user's device to authenticate.

How to Enable MFA Device?

Step 1 − Open the following link, https:// console.aws.amazon.com/iam/

Step 2 − On the web page, choose users from the navigation pane on the right side to view the list of user name.

Step 3 − Scroll down to security credentials and choose MFA. Click activate MFA.

Step 4 − Follow the instructions and the MFA device will get activated with the account.

There are 3 ways to enable a MFA device −

SMS MFA Device

In this method, MFA requires us to configure the IAM user with the phone number of the user's SMS-compatible mobile device. When the user signs in, AWS sends a six-digit code by SMS text message to the user's mobile device. The user is required to enter the same code on a second web page during sign-in to authenticate the right user. This SMS-based MFA cannot be used with AWS root account.

Hardware MFA Device

In this method, MFA requires us to assign an MFA device (hardware) to the IAM user or the AWS root account. The device generates a six-digit numeric code based upon a time synchronized one-time password algorithm. The user has to enter the same code from the device on a second web page during sign-in to authenticate the right user.

Virtual MFA Device

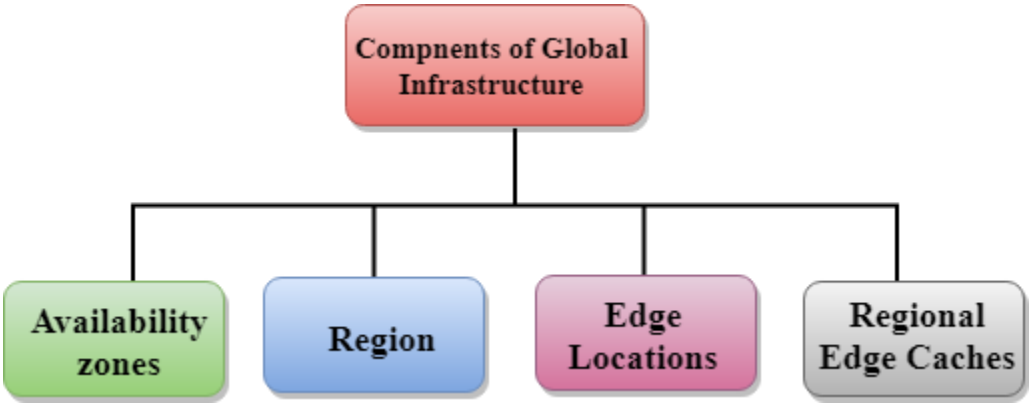
In this method, MFA requires us to assign an MFA device (virtual) to the IAM user or the AWS root account. A virtual device is a software application (mobile app) running on a mobile device that emulates a physical device. The device generates a six-digit numeric code based upon a time-synchronized one-time password algorithm. The user has to enter the same code from the device on a second web page during sign-in to authenticate the right user.

AWS Global Infrastructure

- AWS is a cloud computing platform which is globally available.
- Global infrastructure is a region around the world in which AWS is based. Global infrastructure is a bunch of high-level IT services which is shown below:
- AWS is available in 19 regions, and 57 availability zones in December 2018 and 5 more regions 15 more availability zones for 2019.

The following are the components that make up the AWS infrastructure:

- Availability Zones
- Region
- Edge locations
- Regional Edge Caches

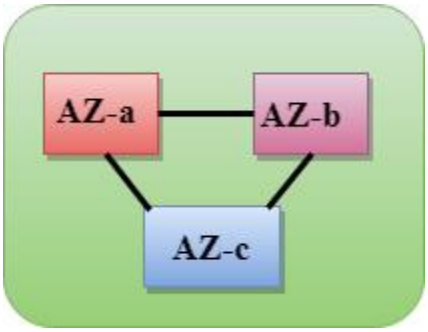


Availability zone as a Data Center

- An availability zone is a facility that can be somewhere in a country or in a city. Inside this facility, i.e., Data Centre, we can have multiple servers, switches, load balancing, firewalls. The things which interact with the cloud sits inside the data centers.
- An availability zone can be a several data centers, but if they are close together, they are counted as 1 availability zone.

Region

- A region is a geographical area. Each region consists of 2 more availability zones.
- A region is a collection of data centers which are completely isolated from other regions.
- A region consists of more than two availability zones connected to each other through links.



- Availability zones are connected through redundant and isolated metro fibers.

Edge Locations

- Edge locations are the endpoints for AWS used for caching content.
- Edge locations consist of CloudFront, Amazon's Content Delivery Network (CDN).
- Edge locations are more than regions. Currently, there are over 150 edge locations.
- Edge location is not a region but a small location that AWS have. It is used for caching the content.
- Edge locations are mainly located in most of the major cities to distribute the content to end users with reduced latency.
- For example, some user accesses your website from Singapore; then this request would be redirected to the edge location closest to Singapore where cached data can be read.

Regional Edge Cache

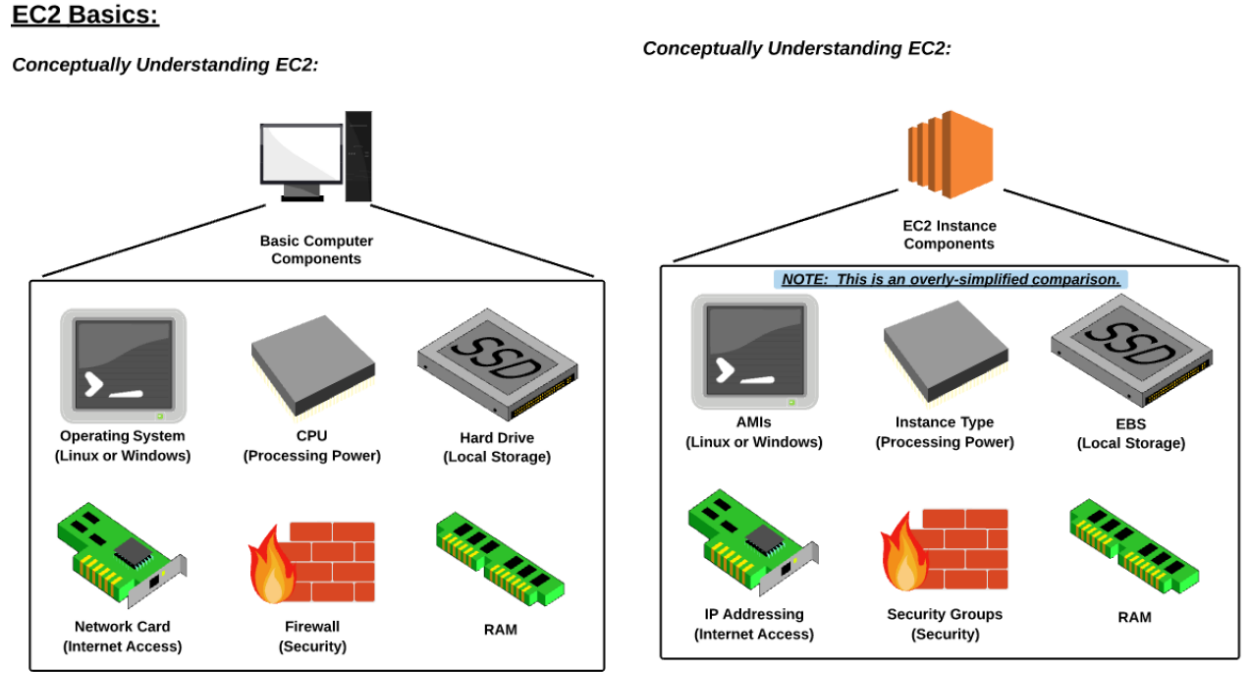
- AWS announced a new type of edge location in November 2016, known as a Regional Edge Cache.
- Regional Edge cache lies between CloudFront Origin servers and the edge locations.
- A regional edge cache has a large cache than an individual edge location.

AWS - Elastic Compute Cloud

Amazon EC2 (Elastic Compute Cloud) is a web service interface that provides resizable compute capacity in the AWS cloud. It is designed for developers to have complete control over web-scaling and computing resources.

EC2 instances can be resized and the number of instances scaled up or down as per our requirement. These instances can be launched in one or more geographical locations or regions, and **Availability Zones (AZs)**. Each region comprises of several AZs at distinct locations, connected by low latency networks in the same region.

EC2 Basics:



EC2 Components

In AWS EC2, the users must be aware about the EC2 components, their operating systems support, security measures, pricing structures, etc.

Operating System Support

Amazon EC2 supports multiple OS in which we need to pay additional licensing fees like: Red Hat Enterprise, SUSE Enterprise and Oracle Enterprise Linux, UNIX, Windows

Server, etc. These OS needs to be implemented in conjunction with Amazon Virtual Private Cloud (VPC).

Security

Users have complete control over the visibility of their AWS account. In AWS EC2, the security systems allow create groups and place running instances into it as per the requirement. You can specify the groups with which other groups may communicate, as well as the groups with which IP subnets on the Internet may talk.

Pricing

AWS offers a variety of pricing options, depending on the type of resources, types of applications and database. It allows the users to configure their resources and compute the charges accordingly.

Fault tolerance

Amazon EC2 allows the users to access its resources to design fault-tolerant applications. EC2 also comprises geographic regions and isolated locations known as availability zones for fault tolerance and stability. It doesn't share the exact locations of regional data centers for security reasons.

When the users launch an instance, they must select an AMI that's in the same region where the instance will run. Instances are distributed across multiple availability zones to provide continuous services in failures, and Elastic IP (EIPs) addresses are used to quickly map failed instance addresses to concurrent running instances in other zones to avoid delay in services.

Migration

This service allows the users to move existing applications into EC2. It costs $80.00 per storage device and $2.49 per hour for data loading. This service suits those users having large amount of data to move.

Features of EC2

Here is a list of some of the prominent features of EC2 −

- **Reliable** − Amazon EC2 offers a highly reliable environment where replacement of instances is rapidly possible. Service Level Agreement commitment is 99.9% availability for each Amazon EC2 region.
- **Designed for Amazon Web Services** − Amazon EC2 works fine with Amazon services like Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon SQS.

It provides a complete solution for computing, query processing, and storage across a wide range of applications.

- **Secure** − Amazon EC2 works in Amazon Virtual Private Cloud to provide a secure and robust network to resources.
- **Flexible Tools** − Amazon EC2 provides the tools for developers and system administrators to build failure applications and isolate themselves from common failure situations.
- **Inexpensive** − Amazon EC2 wants us to pay only for the resources that we use. It includes multiple purchase plans such as On-Demand Instances, Reserved Instances, Spot Instances, etc. which we can choose as per our requirement.

EC2 instances Types come in the following categories:

- **General Purpose**: The most popular; used for web servers, development environments, etc.

  A1

  M5

  T3 and T3a
- Compute Optimized: Good for compute-intensive applications such as some scientific modeling or high-performance web servers.

  C5
- Memory Optimized: Used for anything that needs memory-intensive applications, such as real-time big data analytics, or running Hadoop or Spark.

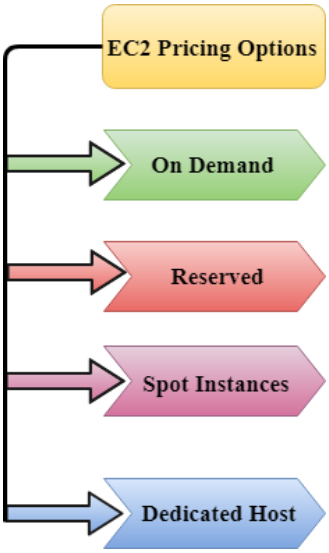
  R5 & R5a

  X1 & X1e
- Accelerated Computing: Include additional hardware (GPUs, FPGAs) to provide massive amounts of parallel processing for tasks such as graphics processing.

  P3, G3, F1
- Storage Optimized: Ideal for tasks that require huge amounts of storage, specifically with sequential read-writes, such as log processing.

  H1 & D2

EC2 Pricing Options



On Demand

- It allows you to pay a fixed rate by the hour or even by the second with no commitment.
- Linux instance is by the second and windows instance is by the hour.
- On Demand is perfect for the users who want low cost and flexibility of Amazon EC2 without any up-front investment or long-term commitment.
- It is suitable for the applications with short term, spiky or unpredictable workloads that cannot be interrupted.
- It is useful for the applications that have been developed or tested on Amazon EC2 for the first time.
- On Demand instance is recommended when you are not sure which instance type is required for your performance needs.

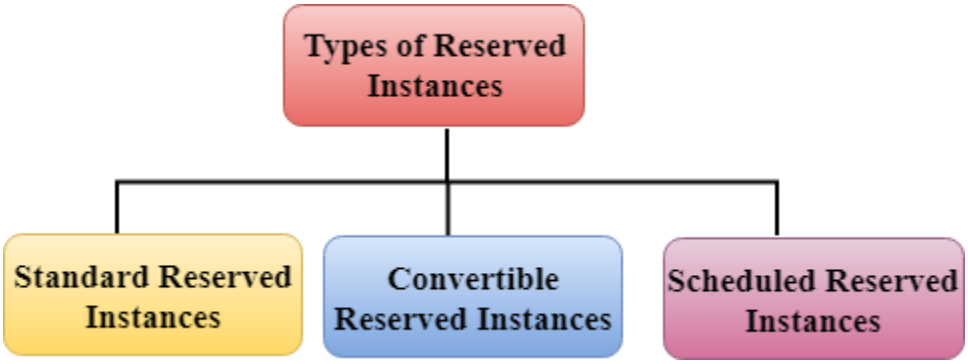
Reserved

- It is a way of making a reservation with Amazon or we can say that we make a contract with Amazon. The contract can be for 1 or 3 years in length.
- In a Reserved instance, you are making a contract means you are paying some upfront, so it gives you a significant discount on the hourly charge for an instance.

- It is useful for applications with steady state or predictable usage.
- It is used for those applications that require reserved capacity.
- Users can make up-front payments to reduce their total computing costs. For example, if you pay all your upfronts and you do 3 years contract, then only you can get a maximum discount, and if you do not pay all upfronts and do one year contract then you will not be able to get as much discount as you can get If you do 3 year contract and pay all the upfronts.

Types of Reserved Instances:

- Standard Reserved Instances
- Convertible Reserved Instances
- Scheduled Reserved Instances



Standard Reserved Instances

- It provides a discount of up to 75% off on demand. For example, you are paying all up-fronts for 3 year contract.
- It is useful when your Application is at the steady-state.

Convertible Reserved Instances

- It provides a discount of up to 54% off on demand.
- It provides the feature that has the capability to change the attributes of RI as long as the exchange results in the creation of Reserved Instances of equal or greater value.
- Like Standard Reserved Instances, it is also useful for the steady state applications.

Scheduled Reserved Instances

- Scheduled Reserved Instances are available to launch within the specified time window you reserve.
- It allows you to match your capacity reservation to a predictable recurring schedule that only requires a fraction of a day, a week, or a month.

Spot Instances

- It allows you to bid for a price whatever price that you want for instance capacity, and providing better savings if your applications have flexible start and end times.
- Spot Instances are useful for those applications that have flexible start and end times.
- It is useful for those applications that are feasible at very low compute prices.
- It is useful for those users who have an urgent need for large amounts of additional computing capacity.
- EC2 Spot Instances provide less discounts as compared to On Demand prices.
- Spot Instances are used to optimize your costs on the AWS cloud and scale your application's throughput up to 10X.
- EC2 Spot Instances will continue to exist until you terminate these instances.

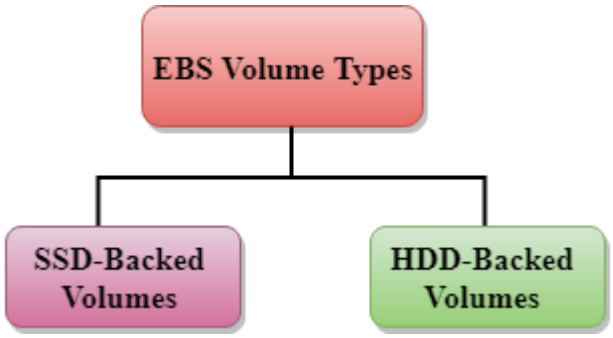
Dedicated Hosts

- A dedicated host is a physical server with EC2 instance capacity which is fully dedicated to your use.
- The physical EC2 server is the dedicated host that can help you to reduce costs by allowing you to use your existing server-bound software licenses. For example, Vmware, Oracle, SQL Server depending on the licenses that you can bring over to AWS and then they can use the Dedicated host.
- Dedicated hosts are used to address compliance requirements and reduces host by allowing to use your existing server-bound server licenses.
- It can be purchased as a Reservation for up to 70% off On-Demand price.

What is EBS?

- EBS stands for **Elastic Block Store**.
- EC2 is a virtual server in a cloud while EBS is a virtual disk in a cloud.
- Amazon EBS allows you to create storage volumes and attach them to the EC2 instances.
- Once the storage volume is created, you can create a file system on the top of these volumes, and then you can run a database, store the files, applications or you can even use them as a block device in some other way.
- Amazon EBS volumes are placed in a specific availability zone, and they are automatically replicated to protect you from the failure of a single component.
- EBS volume does not exist on one disk, it spreads across the Availability Zone. EBS volume is a disk which is attached to an EC2 instance.
- EBS volume attached to the EC2 instance where windows or Linux is installed known as Root device of volume.

EBS Volume Types

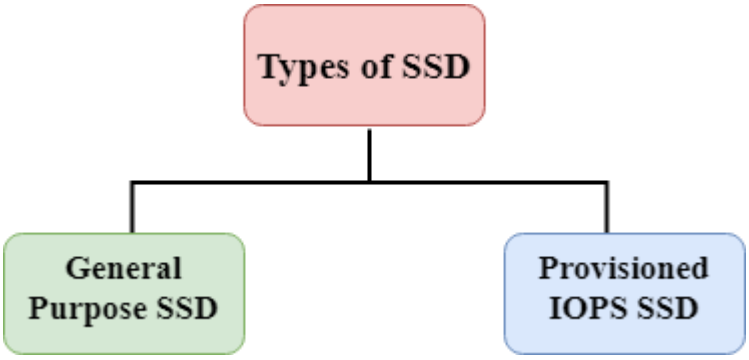


Amazon EBS provides two types of volume that differ in performance characteristics and price. EBS Volume types fall into two parts:

- SSD-backed volumes
- HDD-backed volumes

SSD

- SSD stands for solid-state Drives.
- In June 2014, SSD storage was introduced.
- It is a general purpose storage.
- It supports up to 4000 IOPS which is quite very high.
- SSD storage is very high performing, but it is quite expensive as compared to HDD (Hard Disk Drive) storage.
- SSD volume types are optimized for transactional workloads such as frequent read/write operations with small I/O size, where the performance attribute is IOPS.



SSD is further classified into two parts:

- General Purpose SSD
- Provisioned IOPS SSD

General Purpose SSD

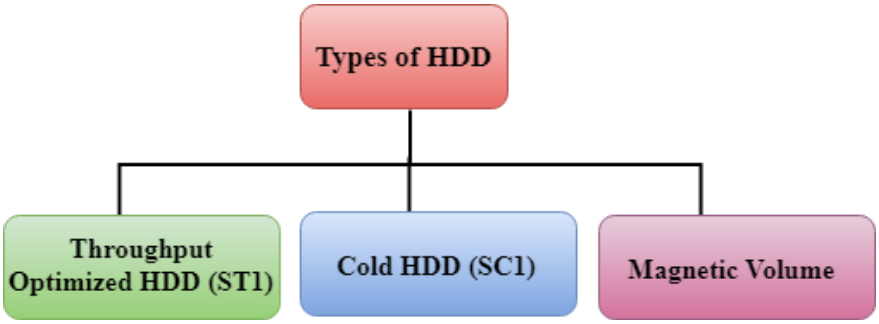
- General Purpose SSD is also sometimes referred to as a GP2.
- It is a General purpose SSD volume that balances both price and performance.
- You can get a ratio of 3 IOPS per GB with up to 10,000 IOPS and the ability to burst up to 3000 IOPS for an extended period of time for volumes at 3334 GiB and above. For example, if you get less than 10,000 IOPS, then GP2 is preferable as it gives you the best performance and price.
-

Provisioned IOPS SSD

- It is also referred to as IO1.
- It is mainly used for high-performance applications such as intense applications, relational databases.
- It is designed for I/O intensive applications such as large relational or NOSQL databases.
- It is used when you require more than 10,000 IOPS.

HDD

- It stands for Hard Disk Drive.
- HDD based storage was introduced in 2008.
- The size of the HDD based storage could be between 1 GB to 1TB.
- It can support up to 100 IOPS which is very low.



Throughput Optimized HDD (st1)

- It is also referred to as ST1.
- Throughput Optimized HDD is a low-cost HDD designed for those applications that require higher throughput up to 500 MB/s.
- It is useful for those applications that require the data to be frequently accessed.
- It is used for Big data, Data warehouses, Log processing, etc.
- It cannot be a boot volume, so it contains some additional volume. For example, if we have Windows server installed in a C: drive, then C drive cannot be a Throughput Optimized Hard disk, D: drive or some other drive could be a Throughput Optimized Hard disk.

- The size of the Throughput Hard disk can be 500 GiB to 16 TiB.
- It supports up to 500 IOPS.

Cold HDD (sc1)

- It is also known as SC1.
- It is the lowest cost storage designed for the applications where the workloads are infrequently accessed.
- It is useful when data is rarely accessed.
- It is mainly used for a File server.
- It cannot be a boot volume.
- The size of the Cold Hard disk can be 500 GiB to 16 TiB.
- It supports up to 250 IOPS.

Magnetic Volume

- It is the lowest cost storage per gigabyte of all EBS volume types.
- It is ideal for the applications where the data is accessed infrequently
- It is useful for applications where the lowest storage cost is important.
- Magnetic volume is the only hard disk which is bootable. Therefore, we can say that it can be used as a boot volume.

Practical:

How to Create a Volume

How to Attach to running instance

How to Detach the Volume

How to Delete the Volume

Step1: Create an EC2 instance

Step2: Create an another Volume

Create an another Volume in same Availability zone (matched to instance AZ) and attach it to above created instance

Step3: Mount an EBS volume to EC2 Linux

- Now, login to your ec2 instance and list the available disks using the following command.

  lsblk

  The above command will list the disk you attached to your instance.

- Check if the volume has any data using the following command.

  sudo file -s /dev/xvdf

  If the above command output shows "/dev/xvdf: data", it means your volume is empty.

- Format the volume to the ext4 filesystem using the following command.

mkfs -t ext4 /dev/xvdf

Alternatively, you can also use the xfs format. You have to use either ext4 or xfs.

sudo mkfs -t xfs /dev/xvdf

- Create a directory of your choice to mount our new ext4 volume. I am using the name "newvolume". You can name it something meaningful to you.

  sudo mkdir /newvolume

- Mount the volume to "newvolume" directory using the following command.

  sudo mount /dev/xvdf /newvolume/

- cd into newvolume directory and check the disk space to validate the volume mount.

  cd /newvolume

  df -h .

  The above command should show the free space in the newvolume directory.

Step4: Automount EBS Volume on Reboot

By default on every reboot, the EBS volumes other than root volume will get unmounted. To enable automount, you need to make an entry in the /etc/fstab file.

Follow the steps given below to automount the EBS volume to ec2 instance.

- Back up the /etc/fstab file.

  sudo cp /etc/fstab /etc/fstab.bak

- Open /etc/fstab file and make an entry in the following format.

  device_name mount_point file_system_type fs_mntops fs_freq fs_passno

  For example,

  /dev/xvdf /newvolume ext4 defaults,nofail 0 0

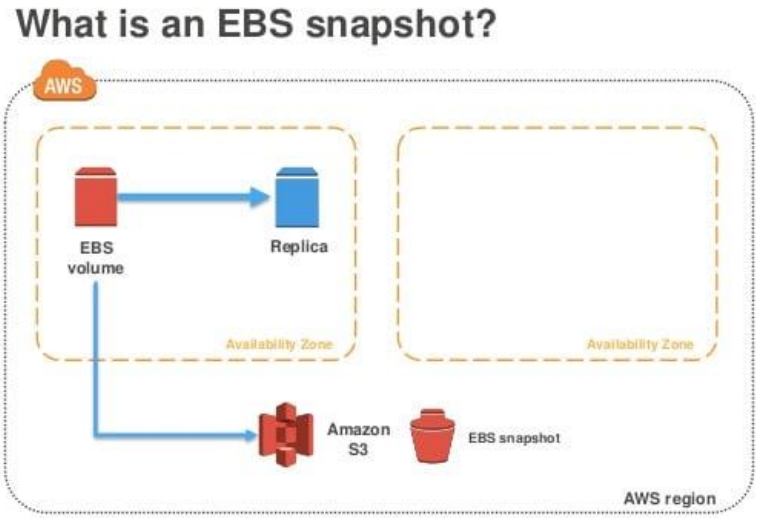
- Execute the following command to check id the fstab file has any error.

  sudo mount -a

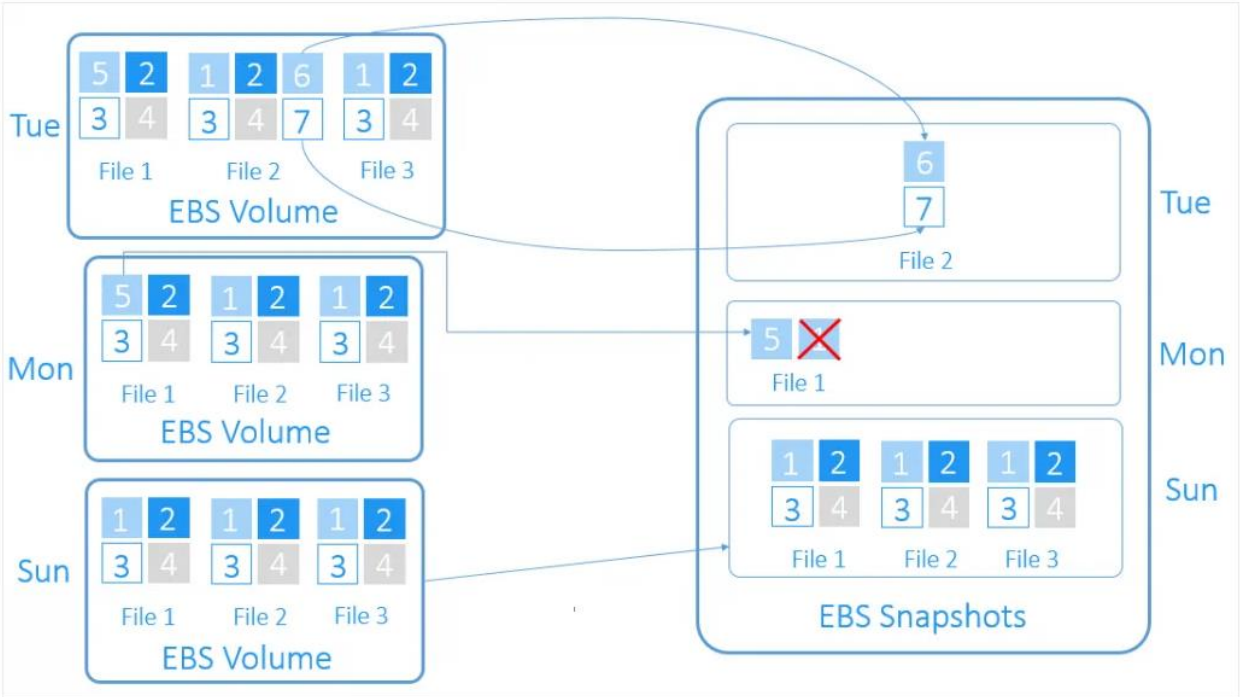
  If the above command shows no error, it means your fstab entry is good.

  Now, on every reboot, the extra EBS volumes will get mounted automatically.

EBS Snapshot



An EBS snapshot is a point-in-time copy of your Amazon EBS volume, which is lazily copied to Amazon Simple Storage Service (Amazon S3). EBS snapshots are incremental copies of data. This means that only unique blocks of EBS volume data that have changed since the last EBS snapshot are stored in the next EBS snapshot. This is how incremental copies of data are created in Amazon AWS EBS Snapshot.



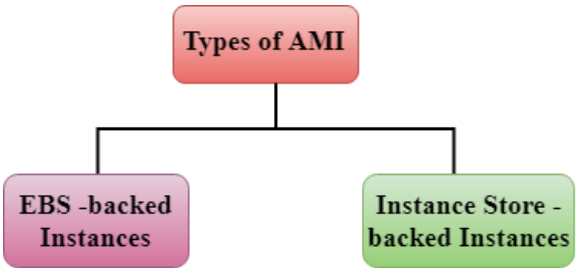
AMI

- An AMI stands for **Amazon Machine Images**.
- An AMI is a virtual image used to create a virtual machine within an EC2 instance.
- You can also create multiple instances using single AMI when you need instances with the same configuration.
- You can also create multiple instances using different AMI when you need instances with a different configuration.
- It also provides a template for the root volume of an instance.

AMI Lifecycle

- First, you need to create and register an AMI.
- You can use an AMI to launch EC2 instances.
- You can also copy an AMI to some different region.
- When AMI is no longer required, then you can also deregister it.

AMI Types



AMI is divided into two categories:

- EBS - backed Instances
- Instance Store - backed Instances

EBS - backed Instances

- EBS is nothing but a volume that provides you persistent storage.
- When you run an EC2 instance that provides you temporary storage, if you delete an EC2 instance then the data stored in the EC2 instance will also be deleted. To make a data persistent, Amazon provides an EBS Volume. If you launch an EC2 instance and want to make some data persistent, then you need to attach an instance with the EBS Volume so that your data would be available even on deleting an EC2 instance.
- When you launch an EC2 instance, it will always have a root device as an EBS Volume which makes the data persistent. Therefore, we can say that when we delete an EC2 instance, then the data is available in a root device.
- In EBS - backed instances, you will be charged or billed for the storage of static data such as operating systems files, etc.

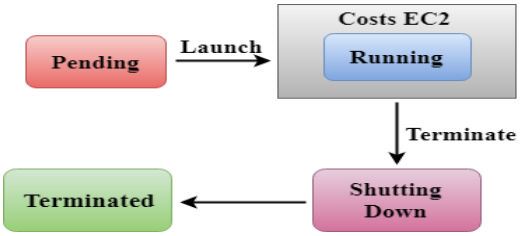
  The cost of adding the EBS Volume to an EC2 instance is minimal.

Instance Store - backed Instances

- In Instance-Store, an instance consists of storage approx 1 TB or 2 TB which is temporary storage. As soon as the instance is terminated, all the data will be lost. For example, if you launch an instance, and deploy the database in it. If you delete an instance, then all the data will be lost and this becomes the challenge. In such a scenario, you can add an additional EBS Volume that also stores the data, so even if you delete an instance, your data would not be lost.
- In this case, EBS Volume is not a root volume. It's an additional volume that you attach to your EC2 instance manually.

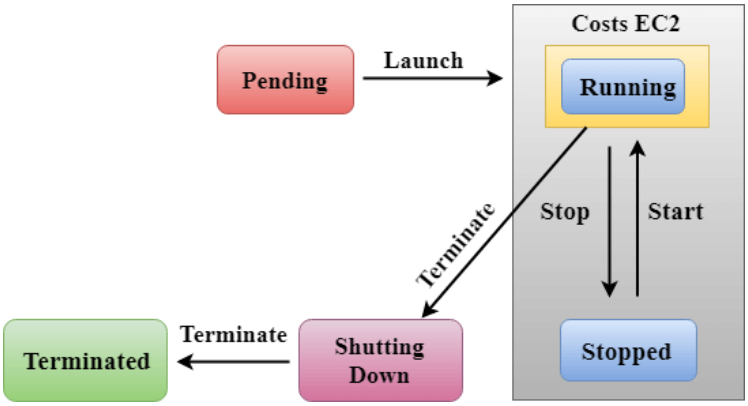
Why EBS - backed instance is more popular than Instance Store - backed instance?

Instance Store - backed instances



In **Instance Store - backed instance**, if you launch an instance, it would be in a pending state. After pending state, an instance comes in a running state then it would be in a shutting down state. Amazon would charge you only when it is in a running state. When you terminate an instance, Amazon would not charge you any cost. For example, if you want to run an instance for 4 hours a day and it would cost you 10 cents per hour. In instance store, my instance would be running 24 hrs a day as it has no stopped state. Therefore, it would cost 72 dollars a month.

- **EBS - backed Instances**



In EBS - backed instances, an instance can be either in a running state or in a stopped state. In this case, Amazon would cost you only for a running state, not for a stopped state. For example, if you want to run an instance for 4 hours a day and it would cost you 10 cents per hour. In EBS - backed instance, an instance will run for 4 hours as it has stopped state as well. I take a 100 GB volume that would cost you 5 dollars. The running cost of an instance would be 12 dollars in a month. Therefore, the total cost taken by this instance is volume cost plus running cost which is equal to 17 dollars.

Difference b/w Instance store & EBS - backed instance

Characteristics	EBS-backed instance	Instance Store-backed instance
Lifecycle	It supports stopping as well as restarting of an instance by saving the state to EBS volume.	In this case, an instance cannot be stopped. It can be either in a running or terminated state.
Data Persistence	Data persists in EBS volume. If an instance is terminated, no data would be lost.	Data does not persist so when instance is terminated, data would be lost.
Boot time	It takes less than 1 min.	It usually takes less than 5 min.
Size limit	1 TB	10 - 16 TB
AMI creation	AMI is very easily created by using a single command.	To create an AMI, it requires installation and AMI tools.
Expensive	It is less expensive.	It is more expensive as compared to Instance Store-backed instance.

Practical:

Create an EC2 instance and install httpd package

Create an AMI with EC2 instance.

Try to create three servers with AMI.

AWS CLI

What is the AWS CLI?

The AWS Command Line Interface (CLI) is for managing your AWS services from a terminal session on your own client, allowing you to control and configure multiple AWS services and implement a level of automation.

Install PIP:

curl -O https://bootstrap.pypa.io/get-pip.py

python3 get-pip.py --user

ls -a ~

export PATH=~/.local/bin:$PATH

source ~/.bash_profile

pip3 --version

Install and update the AWS CLI version 1 using pip

$ pip3 install awscli --upgrade --user

For a specific version of the AWS CLI, append a less-than symbol < and the version number to the filename. For this example the filename for version 1.16.312 would be <1.16.312 resulting in the following command:

$ aws --version

aws-cli/1.22.7 Python/3.8.8 Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.13

Configure AWS CLI using the CONFIGURE command

Access Key and Secret Access Key

To connect with a user (root or IAM), we require login credentials. In AWS, we call it as access key (user) and secret access key (password). It is required to connect AWS using the way such as CLI programmatically.

Go to My Security Credentials after login to the AWS console.

Execute the following command in a command prompt to configure AWS CLI in your environment.

>aws configure

Enter the Access Key, Secret access key, default region name, and default output format. Here I do not specify a default region, so we need to specify it explicitly while querying AWS services. We have specified the JSON format for output.

Usage

The AWS Command Line Interface User Guide walks you through installing and configuring the tool. After that, you can begin making calls to your AWS services from the command line.

$ aws ec2 describe-instances

$ aws ec2 start-instances --instance-ids i-1348636c

$ aws ec2 describe-volumes

$ aws ec2 describe-vpcs

Elastic Load Balancing

A *load balancer* serves as the single point of contact for clients. Clients send requests to the load balancer, and the load balancer sends them to targets, such as EC2 instances, in one or more Availability Zones.

load balancer performs the following functions:

- Distributes client requests or network load efficiently across multiple servers
- Ensures high availability and reliability by sending requests only to servers that are online
- Provides the flexibility to add or subtract servers as demand dictates

Elastic Load Balancing

Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Elastic Load Balancing offers three types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant.

AWS ELB comes in three versions which perform different tasks.

- The Version 1 provides detailed instructions for using *Classic Load Balancers*. Released on 2009
- 2nd version provides detailed instructions for using *Application Load Balancers*. Released on 2016
- 3rd provides detailed instructions for using *Network Load Balancers*. Released on 2017
- 4th – Gateway Load balancer . Works on both Network and Transport layers

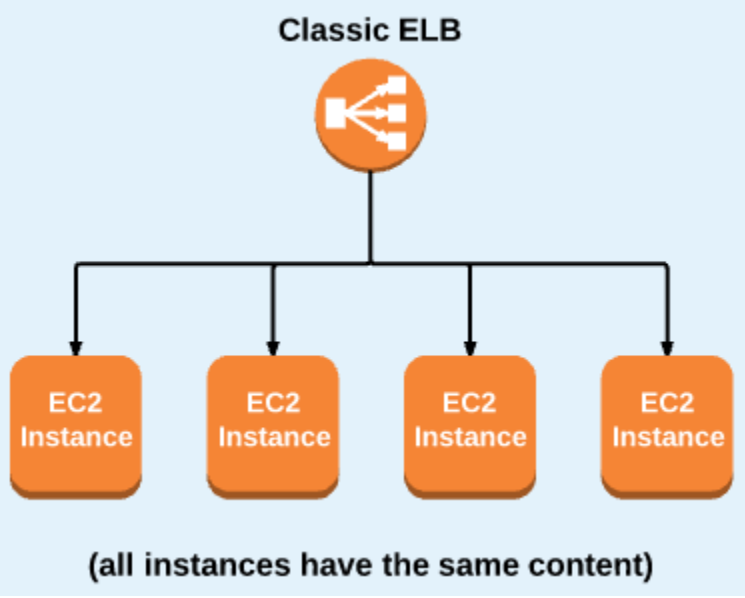
Advantages of AWS ELB

- **Highly Available**
    - ELB distributes traffic evenly among all the targets, for example multiple EC2 instances.
    - ELB has an SLA of 99.99%
- **Flexible**
    - ELB let's you route traffic with the application's IP address, this allows you launch multiple applications in a single instance.

- **Highly secure**
    - You can implement robust security features using Amazon VPC with Amazon ELB
- **Elastically scalable**
    - ELB can handle sudden spikes in traffic and can handle millions of requests per second. Whenever there is a traffic increase, auto scaling feature will be enabled and also load balancing rules will be used to provide the website users an seamless performance
- **Hybrid load balancing**
    - You can use the same Amazon load balancer to balance across applications on your on-premises set up and you AWS infrastructure. Now, it will be very easy to migrate your application from on-premise to AWS cloud.
- **Robust monitoring and auditing**
    - Applications and their performance can be monitored and maintained. You can also use CloudWatch metrics and logs to analyze our applications data, traffic, and working.

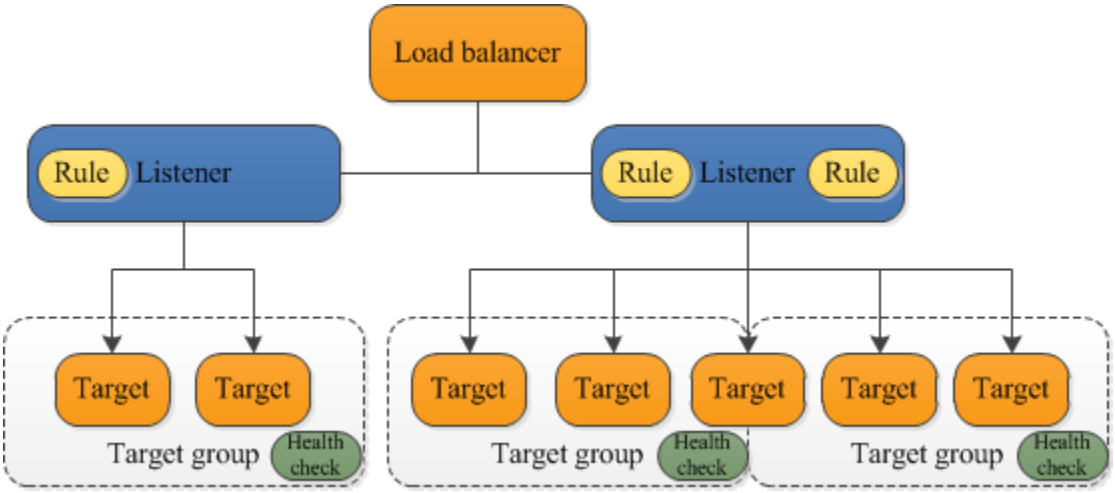
Classic Load Balancer

- Classic Load balancer in AWS is used on EC2-classic instances. This is the previous generation's load balancer and also it doesn't allow host-based or path based routing.



Application Load Balancer

Application Load Balancer is best suited for load balancing of HTTP and HTTPS traffic and provides advanced request routing targeted at the delivery of modern application architectures, including microservices and containers. Operating at the individual request level (Layer 7), Application Load Balancer routes traffic to targets within Amazon Virtual Private Cloud (Amazon VPC) based on the content of the request.



A target group tells **a load balancer where to direct traffic to : EC2 instances, fixed IP addresses**; or AWS Lambda functions, amongst others.

A listener is **a process that checks for connection requests, using the protocol and port that you configure**. The rules that you define for a listener determine how the load balancer routes requests to the targets in one or more target groups.

Application Load Balancer

Host-based Routing using ALB

If you have two websites, sriman.com and dashboard.sriman.com. Both of the websites are hosted in different EC2 instance and you want to distribute incoming traffic between them to make them highly available.

Normally, we would create two AWS load balancers using CLB, but using ALB it is possible with one and also your money is saved. Instead of paying for 2 ELBs, only pay for a single ELB.

Path-based Routing using ALB

In this type of routing, the websites URL path will be hosted on different EC2 instances. For example, consider sriman.com and sriman.com/tutorial and these URL paths are hosted on different EC2 instances. Now, if you want to route traffic between these two URLs then you can use a path-based routing method. ALB can be used to solve this problem too, you can use traffic routing according to the path feature by using just one ALB.

Practical:

Create ALB:

Deploy 2 instances with httpd

#!/bin/bash

yum update -y

yum install -y httpd.x86_64

systemctl start httpd.service

systemctl enable httpd.service

echo “Hello World from 1a instance” > /var/www/html/index.html

Create target group with 2 instances

Create LB balancer with target

For Path based routhing

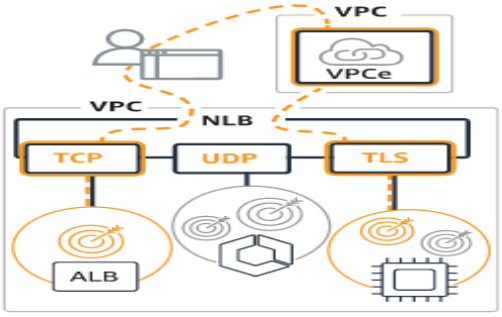
Create another server and another TG

Edit /var/www/html/ with clothes folder

Network Load Balancer

Network Load Balancer is best suited for load balancing of Transmission Control Protocol (TCP), User Datagram Protocol (UDP) and Transport Layer Security (TLS) traffic where extreme performance is required. Operating at the connection level (Layer 4), Network Load Balancer routes traffic to targets within Amazon Virtual Private Cloud (Amazon VPC) and is capable of handling millions of requests per second while maintaining ultra-low

latencies. Network Load Balancer is also optimized to handle sudden and volatile traffic patterns.



Creating Network Load balancer

1. Deploy 3 Servers with Nginx Installed in diffrent availibility zones.
2. Login to each server and add extra line to index.html as WebServer1,2,3.
3. Configure Target Group and add only WebServer 1 & 2 only.
4. Deploy Load balancer and add the target group.
5. Check the load balance of requests are happening between the servers 1 & 2.
6. Add the third Server and check reqeusts are gng to Server 3 or not.
7. Delete Server 1 & 2 and check servers should go to only Server-3.

EC2 user data code:

#!/bin/bash

sudo yum update -y

sudo amazon-linux-extras install nginx1 -y

echo Webserver1 from us-east-1a > /usr/share/nginx/html/index.html

systemctl start nginx

systemctl enable nginx

Load balancer state

A load balancer can be in one of the following states:

provisioning

The load balancer is being set up.

active

The load balancer is fully set up and ready to route traffic.

failed

The load balancer could not be set up.

Difference between NLB and ALB:

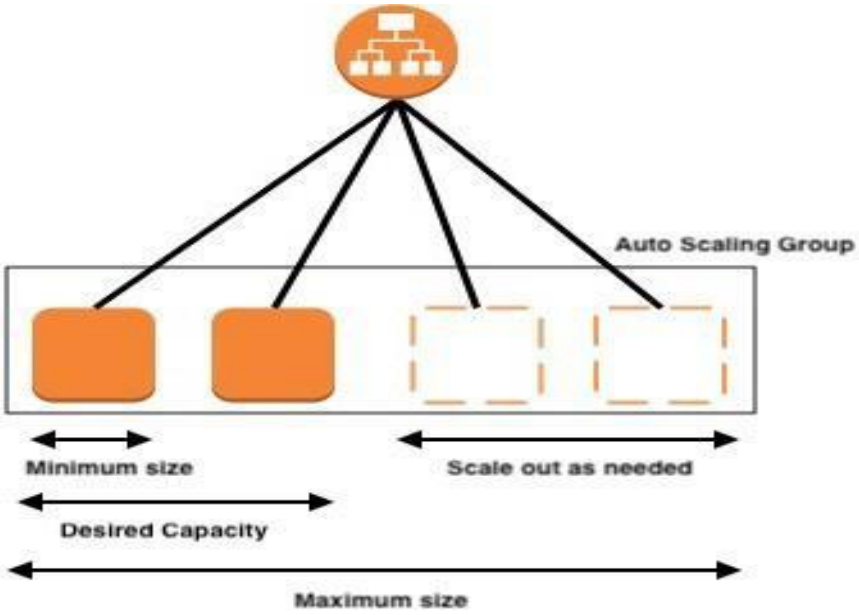
```
Network load balancer:
n millions of requests
single type of application
OSI 4th place
TCP, websockets
VPC only

Application load balancer:

Path based routing
multiple applications
OSI model 7th place
HTTP, HTTPS
```

Amazon AutoScaling:

Amazon Auto-scaling is the service offered by the AWS which allows us to increase or decrease the capacity of the application on demand depending on the limits we set using Cloudwatch. Auto-scaling not only helps in dealing with the servers by dynamically scaling EC2 capacity up or down but also helps in maintaining availability. It automatically adds instances when the traffic increases or removes the instance when not enough traffic is coming depending on the conditions defined by us.



Features:

Better fault tolerance:

Amazon EC2 auto scaling is capable of noticing if something is odd with an EC2 instance and disable the same as soon as it finds something odd. One can also use Amazon EC2 auto-scaling with multiple availability zones so if 1 availability zone is unavailable, it can launch the instance in another zone and will compensate for the loss.

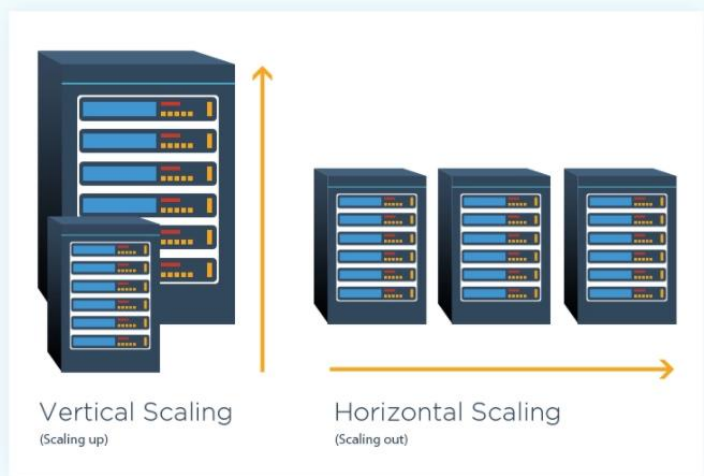
Better availability:

This is one of the main features that it makes the application always capable to handle the amount of traffic dogging in. It makes EC2 instance ready to handle the intensity.

Better cost management:

As AWS offers to pay as you go pricing, Autoscaling further manages the cost by eliminating the instance that is not currently used and re-launch instances when in need so that no instance is wasted.

Horizontal Vs. Vertical Scaling



Horizontal scaling classied into 2 types again:

- Use **Dynamic Scaling** to add and remove capacity for resources to maintain resource utilization at the specified target value.
- Use **Predictive Scaling** to forecast your future load demands by analyzing your historical records for a metric. It also allows you to schedule scaling actions that proactively add and remove resource capacity to reflect the load forecast, and control maximum capacity behavior. Only available for EC2 Auto Scaling groups.

Key components:

Groups	Your EC2 instances are organized into *groups* so that they are treated as a logical unit for scaling and management. When you create a group, you can specify its minimum, maximum, and desired number of EC2 instances.
Launch configurations	Your group uses a *launch configuration* as a template for its EC2 instances. When you create a launch configuration, you can specify information such as the AMI ID, instance type, key pair, security groups, and block device mapping for your instances.
Scaling options	How to scale your Auto Scaling groups.

Practical Auto Scaling

Create below things first

Create EC2 instance with httpd

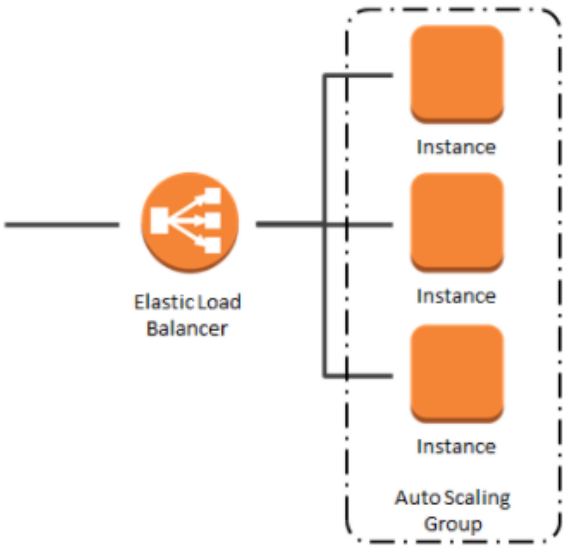
Create An AMI with above instance

Create Launch Configuration with above create AMI

Create ASG with above create LC.

Connect to Primary ASG instance and run **yes>/dev/null** command

Practical Auto Scaling With ELB



Create below things first

Create EC2 instance with httpd

Create An AMI with above instance

Create Target Group without registering any instance

Crate ALB and attach TG

Create Launch Configuration with above create AMI

Create ASG with above create LC.

Connect to Primary ASG instance and run **yes>/dev/null** command

VPC

A virtual private cloud (VPC) is a virtual network that closely resembles a traditional network that you'd operate in your own data centre, with the benefits of using the scalable infrastructure of Amazon Web Services (AWS).

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined.

In AWS the VPC cosists of the following components:-

Subnet: A segment of a VPC's where you can place groups to isolated resources.

Internet Gateway: VPC side of a connection to utilize public Internet.

NAT Gateway: A highly available, managed Network Address Translation (NAT) service for your resources in a private subnet to access the Internet.

Virtual private gateway: The Amazon VPC side of a VPN connection for secure transactions.

Peering Connection: To route traffic via private IP addresses between two peered VPCs.

VPC Endpoints: Enables private connectivity for your service in AWS without using an Internet Gateway, VPN, Network Address Translation (NAT) devices, or firewall proxies.

Egress-only Internet Gateway: A stateful gateway that provides egress only access for IPv6 traffic from the VPC to the Internet.

VPC security:

Amazon VPC provides three features that you can use to increase and monitor the security for your VPC:

Security groups: Act as a firewall for associated Amazon EC2 instances, controlling both inbound and outbound traffic at the instance level

Network access control lists (ACLs): Act as a firewall for associated subnets, controlling both inbound and outbound traffic at the subnet level

Flow logs: Capture information about the IP traffic going to and from network interfaces in your VPC

Security groups:

EC2 security groups are, essentially, a network firewall and they control incoming and outgoing traffic for EC2 instances.

Amazon EC2 security group rules

There are two sets of rules for an Amazon EC2 security group: inbound and outbound. Inbound rules define the incoming traffic the security group allows. Outbound rules define the traffic permitted to leave the compute resource associated with the security group.

Inbound means incoming traffic coming to your EC2 instances. For that you have to add inbound rule. For web server generally we use port 80.

Outbound means outgoing traffic from your EC2 instances. To connect internet or any browser you have to add outbound rule.

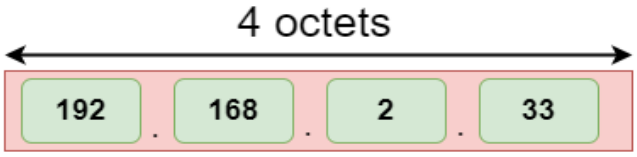
IPV4 Vs IPV6

Example: 127.255.255.255

Example:
2001:0db8:85a3:0000:0000:8a2e:0370:7334

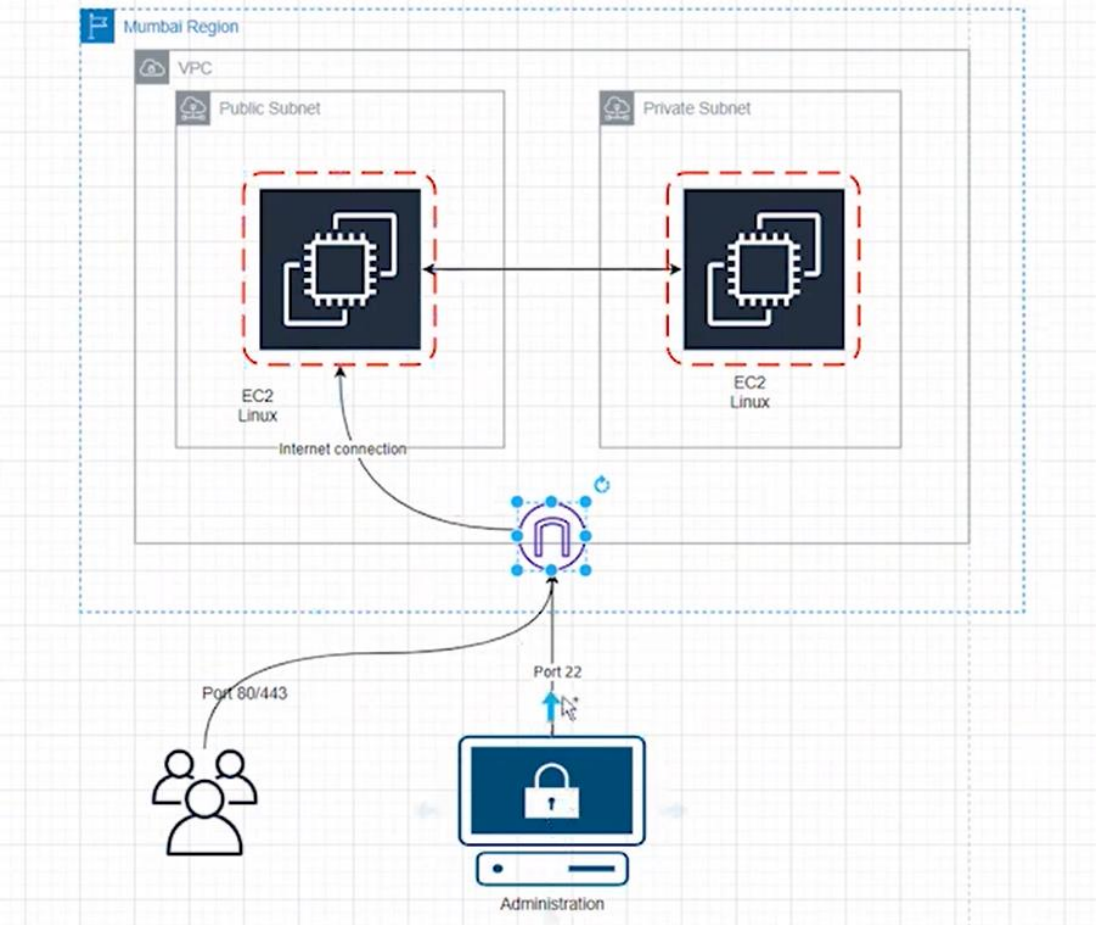
Address format

The address format of IPv4:



The address format of IPv6:

VPC architecture:



Practical:

Create VPC names as - Myvpc

Create private and public subnets

Create Internet gateway and attach VPC to it.

Launch one server in Public subnet and another one in private subnet (Create rule in security group enable at VPC level).

Rename -> existing route table of myvpc as PublicRT

Create new route table rename it as Private RT

Associate public subnet to public RT

Associate Private subnet to Private RT

Create rule 0.0.0.0/0 to Public RT at IGW

Try to connect to public subnet EC2 and Private subnet EC2

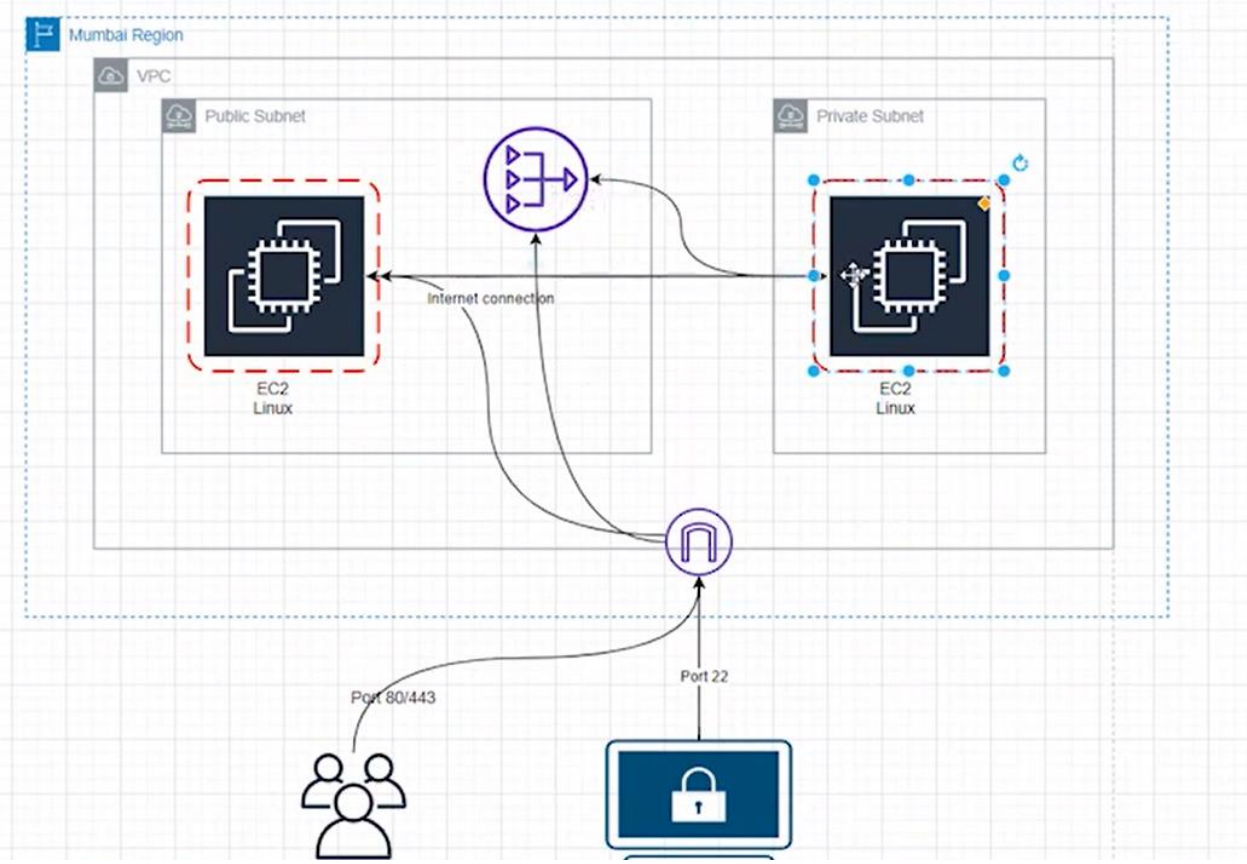
VPC with NAT Gateway

With the above practical internet support for Private subnet EC2 not enabled. We need to attach NAT Gateway.

Follow below steps to enable internet for private subnet EC2 server:

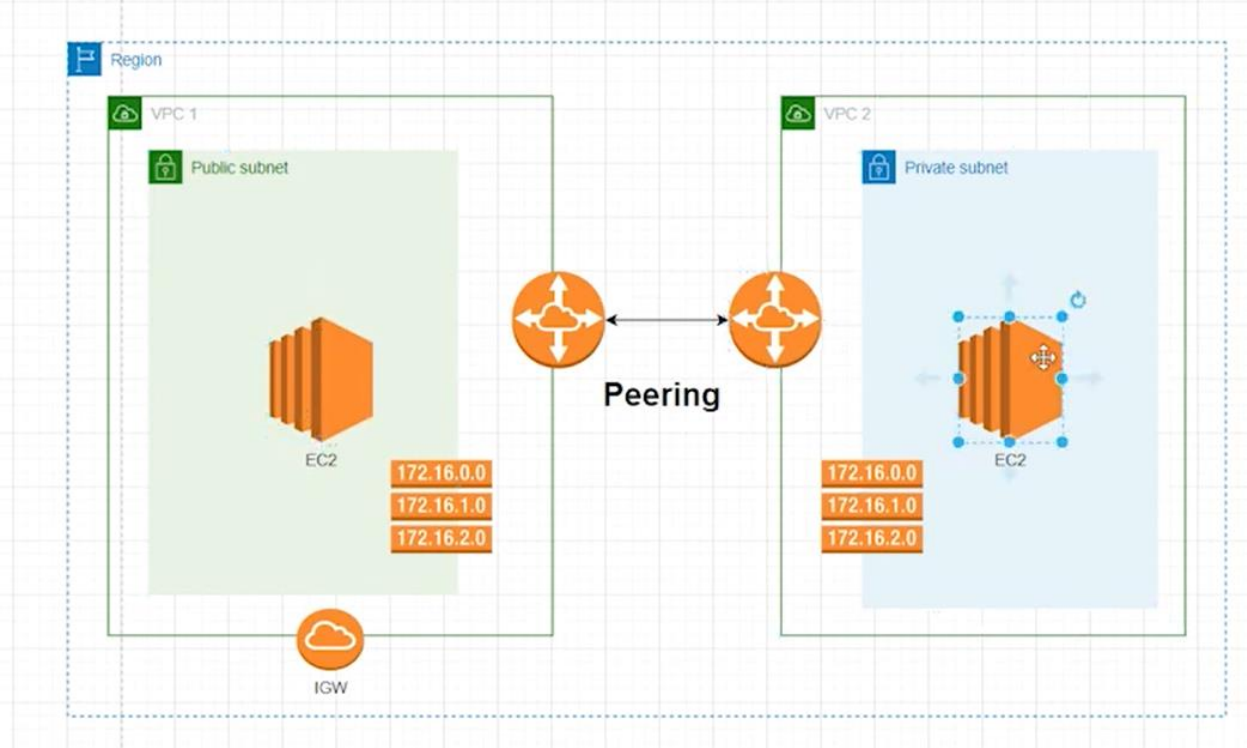
Create a NAT gateway within Public subnet.

And create a rule to Private RT with NAT gateway.



VPC Peering

A **VPC peering** connection is a networking connection between two VPCs that enables you to route traffic between them privately.



Amazon Route 53

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. It is designed for developers and corporates to route the end users to Internet applications by translating human readable names like www.mydomain.com, into the numeric IP addresses like 192.0.2.1 that computers use to connect to each other.

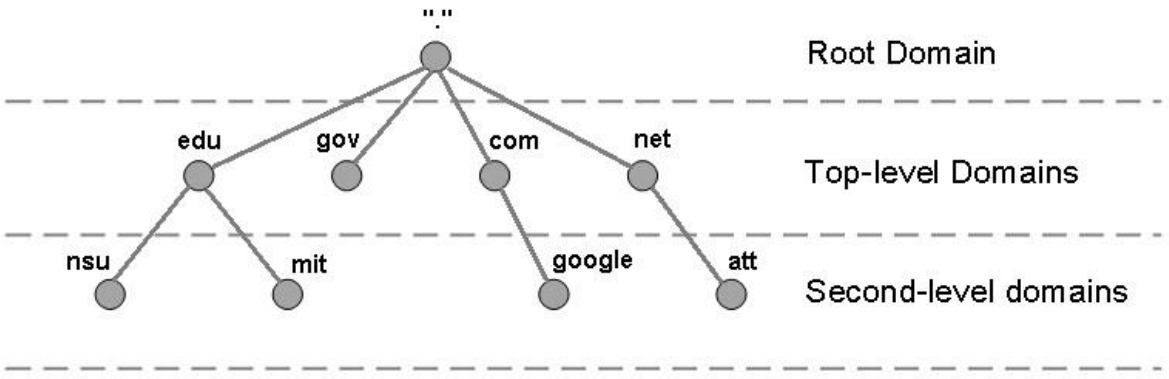
Features of Route 53

- Easy to register your domain − We can purchase all level of domains like .com, .net, .org, etc. directly from Route 53.
- **Highly reliable** − Route 53 is built using AWS infrastructure. Its distributed nature towards DNS servers help to ensure a consistent ability to route applications of end users.
- **Scalable** − Route 53 is designed in such a way that it automatically handles large volume queries without the user's interaction.
- **Can be used with other AWS Services** − Route 53 also works with other AWS services. It can be used to map domain names to our Amazon EC2 instances, Amazon S3 buckets, Amazon and other AWS resources.
- **Easy to use** − It is easy to sign-up, easy to configure DNS settings, and provides quick response to DNS queries.
- Health Check: Route 53 monitors the health of the application. If an outage is detected, then it automatically redirects the users to a healthy resource.
- **Cost-Effective** − Pay only for the domain service and the number of queries that the service answers for each domain.
- **Secure** − By integrating Route 53 with AWS (IAM), there is complete control over every user within the AWS account, such as deciding which user can access which part of Route 53.
- DNS Types: 10 Top DNS Record Types
- DNS servers create a DNS record to provide important information about a domain or hostname, particularly its current IP address. The most common DNS record types are:
- Address Mapping record (A Record)—also known as a DNS host record, stores a hostname and its corresponding IPv4 address.
- IP Version 6 Address record (AAAA Record)—stores a hostname and its corresponding IPv6 address.
- Canonical Name record (CNAME Record)—can be used to alias a hostname to another hostname. When a DNS client requests a record that contains a CNAME, which points to another hostname, the DNS resolution process is repeated with the new hostname.
- Mail exchanger record (MX Record)—specifies an SMTP email server for the domain, used to route outgoing emails to an email server.

- Name Server records (NS Record)—specifies that a DNS Zone, such as “example.com” is delegated to a specific Authoritative Name Server, and provides the address of the name server.
- Reverse-lookup Pointer records (PTR Record)—allows a DNS resolver to provide an IP address and receive a hostname (reverse DNS lookup).
- Start of Authority (SOA Record)—this record appears at the beginning of a DNS zone file, and indicates the Authoritative Name Server for the current DNS zone, contact details for the domain administrator, domain serial number, and information on how frequently DNS information for this zone should be refreshed.

Root Domain?

The root domain (example.com) is the overarching structure that contains the subdomains (blog.example.com), and every folder (/SEO/article) that belongs to a website.



Practical:

Purchase Domain from Godaddy

Lunch an EC2 instance with httpd package

Create a hosted zone in Route53

Added the hosted zone NS records to Godaddy

Creating A record with the instance public IP in AWS hosted zone.

Created CName with the other name