**What is cloud?**

The cloud refers to a network of servers that are accessed over the internet to store, manage, and process data and applications, rather than on a local server or personal computer.

**Difference between public cloud and private cloud?**

Public Cloud: Services are provided over the internet by third-party providers like Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform. Resources are shared among multiple users, and users pay only for the resources they consume.

Private Cloud: Services are maintained on a private network, either on-premises or hosted by a third-party provider. Resources are dedicated to a single organization, providing greater control and customization but often with higher costs.

**Why is public cloud so popular?**

Public clouds offer scalability, flexibility, and cost-effectiveness. They eliminate the need for organizations to invest in and maintain their own infrastructure, allowing them to quickly scale resources up or down as needed, and only pay for what they use. Additionally, public clouds provide access to advanced technologies and global infrastructure.

**How is AWS better than others?**

AWS offers a wide range of services, global infrastructure, and a strong ecosystem of partners and developers. It provides highly reliable, scalable, and cost-effective solutions for businesses of all sizes. Additionally, AWS continues to innovate with new services and features, maintaining its position as a leader in the cloud computing market.

**What is AWS?**

aws or amazon web service is a secure cloud computing platform that provides computing power database networking content storage and much more the platform also works with a pay as you go pricing model which means you only pay for how much of the services

**(Or)**

AWS stands for Amazon Web Services, which is a cloud computing platform that provides a wide range of cloud-based services to businesses and individuals. AWS is a subsidiary of Amazon and is one of the largest providers of cloud computing services in the world.

AWS offers a variety of services including compute, storage, databases, networking, analytics, machine learning, security, and more. These services are designed to help businesses and individuals build and run their applications and infrastructure in the cloud, without having to worry about the underlying hardware or infrastructure.

Some of the popular AWS services include Amazon EC2 (Elastic Compute Cloud), Amazon S3 (Simple Storage Service), Amazon RDS (Relational Database Service), Amazon Lambda (serverless computing), Amazon DynamoDB (NoSQL database), and Amazon SQS (Simple Queue Service).

**Uses of AWS**

* A small manufacturing organization uses their expertise to expand their business by leaving their IT management to the AWS.
* A large enterprise spread across the globe can utilize the AWS to deliver the training to the distributed workforce.
* An architecture consulting company can use AWS to get the high-compute rendering of construction prototype.
* A media company can use the AWS to provide different types of content such as ebox or audio files to the worldwide files.

**Pay-As-You-Go**

Based on the concept of Pay-As-You-Go, AWS provides the services to the customers.

AWS provides services to customers when required without any prior commitment or upfront investment. Pay-As-You-Go enables the customers to procure services from AWS.

* Computing
* Programming models
* Database storage
* Networking

## Advantages and Features of AWS

### 1) Flexibility

### 2) Cost-effectiveness

### 3) Scalability/Elasticity

### 4) Security

5) Experienced

**Why We Use AWS:**

There are many reasons why businesses and individuals use AWS:

**Scalability:** AWS provides a scalable infrastructure that can easily accommodate changes in demand. This allows businesses to quickly and easily scale their resources up or down based on their needs.

**Cost-effectiveness:** AWS offers a pay-as-you-go pricing model, which means that businesses only pay for the resources they use. This can be more cost-effective than investing in and maintaining their own infrastructure.

**Security:** AWS has built-in security measures and protocols to help protect data and infrastructure. AWS also offers various compliance certifications, making it suitable for businesses with strict security requirements.

**Reliability:** AWS has a global network of data centers, which means that businesses can choose the location that is closest to their customers to ensure fast and reliable access to their applications and services.

**Flexibility:** AWS offers a wide range of services that can be used for a variety of purposes, from hosting websites and applications to running complex big data and machine learning workloads.

Overall, AWS provides a flexible, scalable, and cost-effective cloud infrastructure that can help businesses of all sizes to innovate and grow.

**Important Services in AWS :**

**Compute:** This includes the servers, virtual machines, containers, and serverless functions used to run the application. AWS provides a range of compute services, including EC2, Lambda, ECS, EKS, Batch, and Fargate.

**Storage:** This includes the data stores used to store and manage application data. AWS provides a range of storage services, including S3, EBS, EFS, and Glacier.

**Database**: This includes the relational and non-relational databases used to store and manage application data. AWS provides a range of database services, including RDS, DynamoDB, Neptune, ElastiCache, and Redshift.

**Networking**: This includes the virtual private cloud (VPC), subnets, routing tables, security groups, and network interfaces used to connect the various components of the application. AWS provides a range of networking services, including VPC, Direct Connect, Route 53, and Elastic Load Balancing.

**Security:** This includes the identity and access management (IAM), encryption, and security monitoring used to secure the application and data. AWS provides a range of security services, including IAM, Shield, KMS, and Certificate Manager.

**Management and Monitoring:** This includes the tools used to manage and monitor the various components of the application. AWS provides a range of management and monitoring services, including CloudFormation, CloudTrail, Config, and Systems Manager.

**Migration**

Overall, the architecture of an AWS application is designed to be scalable, reliable, secure, and cost-effective. AWS provides a range of services and tools that enable developers and organizations to build and operate complex applications and services in the cloud.

**1) Compute Services :**

**What Is An Instance?**

An instance is nothing but a virtual server for running applications on Amazon ec2.

It can also be understood like a tiny part of a larger computer a tiny part which has

its own Hardware network connection operating system. but it is actually virtual in nature.

An EC2 instance is a virtual server that runs on the AWS cloud infrastructure. These instances are available in various configurations, with different combinations of CPU, memory, storage, and networking capacity.

**What is an instance type in AWS?**

Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications.

There are five main instance types in AWS:

General Purpose: These instances balance computation, memory, and networking resources.

Compute-Optimized

Memory-Optimized

Storage-Optimized

Accelerated Computing

General Purpose Instances (e.g., t3, m5): Suitable for a wide range of applications including web servers, development environments, and small to medium databases.

Compute Optimized Instances (e.g., c5): Designed for compute-bound applications that require high performance processing, such as batch processing, high-performance computing (HPC), and scientific modeling.

Memory Optimized Instances (e.g., r5): Ideal for memory-intensive applications such as in-memory databases, real-time big data analytics, and high-performance databases.

Storage Optimized Instances (e.g., i3): Optimized for applications that require high I/O performance and low-latency storage, such as NoSQL databases, data warehousing, and Elasticsearch.

Accelerated Computing Instances (e.g., p3, g4): Equipped with specialized hardware accelerators like GPUs and FPGAs, suitable for tasks such as machine learning, graphics rendering, and video encoding

**Key Pair:**

In Amazon Web Services (AWS), a key pair is a set of security credentials that consists of a public key and a private key. Key pairs are used to authenticate and securely connect to EC2 instances, and they are a standard way to secure communication between instances and clients.

An EC2 key pair consists of a public key and a private key. You use the public key to encrypt data and the private key to decrypt it. EC2 key pairs are used for secure SSH access to EC2 instances.

**What is the relation between the Availability Zone and Region?**

AWS regions are separate geographical areas, like the US-West 1 (North California) and Asia South (Mumbai). On the other hand, availability zones are the areas that are present inside the regions. These are generally isolated zones that can replicate themselves whenever required.

**(or)**

An AWS Availability Zone is a physical location where an Amazon data center is located. On the other hand, an AWS Region is a collection or group of Availability Zones or Data Centers.

**Steps to launch the instances :**

1. Sign in to the AWS Management Console.
2. Navigate to the EC2 service.
3. Click the "Launch Instance" button to start the instance creation process.
4. Add any required tags to the instance.
5. Choose an Amazon Machine Image (AMI) that includes the operating system and other software that you want to run on your instance.
6. Choose an instance type based on your performance and capacity requirements.
7. Create or select an existing key pair to securely log in to your instance
8. Configure the instance details, such as the number of instances to launch, network settings, and storage options.
9. Configure the security groups that control inbound and outbound traffic to the instance.
10. Review your instance settings and click the "Launch" button to start the instance.

Once the instance is launched, you can connect to it and start using it to run your applications or perform other tasks. You can also stop or terminate the instance at any time, and you will only be billed for the time that the instance is running

**i) What Is EC2?**

Amazon Elastic Compute Cloud (EC2) is a web service provided by AWS that allows users to rent virtual servers (instances) on which they can run their own applications. EC2 instances provide scalable computing capacity in the cloud, making it easy to scale up or down as needed.

With EC2, users can launch instances of various types and sizes depending on their specific needs. EC2 instances can run a variety of operating systems, including Linux, Windows, and Unix, and can be customized to meet specific performance and security requirements.

EC2 instances are billed on a per-hour or per-second basis, depending on the pricing model selected by the user. EC2 offers several pricing options, including On-Demand instances, Reserved instances, and Spot instances.

Overall, EC2 is a powerful and flexible service that provides scalable and cost-effective computing capacity in the cloud. It is widely used by organizations of all sizes to run their applications and services in a highly scalable and secure environment.

**ii) Features in EC2 :**

Virtual Computing environments

Preconfigured templates for your instances

Various configurations of memory, cpu, storage and networking capacity

Secure login information

Metadata known as Tags

**Why is EC2 used?**

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster.

**Is EC2 a server?**

An Amazon EC2 instance is a virtual server in Amazon's Elastic Compute Cloud (EC2) for running applications on the Amazon Web Services (AWS) infrastructure.

**Is EC2 a virtual machine?**

Amazon Elastic Compute Cloud (EC2) is the Amazon Web Service you use to create and run virtual machines in the cloud (we call these virtual machines 'instances')

## iii) EC2 Pricing Options

AWS offers several pricing models for its services, each designed to meet different needs and usage patterns. The following are some of the pricing models offered by AWS:

**On-Demand:** With this model, users pay for compute capacity by the hour or second, without any long-term commitments or upfront payments. This pricing model is best suited for applications with unpredictable workloads or short-term usage patterns.

**Reserved Instances:** With this model, users can make a one-time, upfront payment for a specific instance type and receive a discount on the hourly charge for that instance. This pricing model is best suited for applications with steady-state usage patterns or applications that require reserved capacity.

You commit to using specific instance types in a specific region for a one- or three-year term, which offers a significant discount compared to On-Demand pricing. It is suitable for predictable workloads.

**Spot Instances:** With this model, users can bid for unused Amazon EC2 capacity and receive a discount on the hourly charge for that capacity. This pricing model is best suited for applications with flexible start and end times or applications that can handle interruptions in their compute capacity.

**Dedicated Hosts:** With this model, users can rent an entire physical server that is dedicated to their use. This pricing model is best suited for applications with strict regulatory or compliance requirements.

The best pricing model for an application or workload on AWS depends on its usage patterns, workload characteristics, and budget. For example, if an application has unpredictable usage patterns, the On-Demand pricing model may be the best option, whereas if an application has a predictable usage pattern, the Reserved Instances or Savings Plans pricing models may be more cost-effective. It's important to carefully analyze your application's usage patterns and requirements to choose the best pricing model for your needs.

**iv) AMI [Amazon Machine Image]**

* 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.

**What the AMI includes :**

An AMi includes the Storage (Elastic Block storage)

A block device mapping that specifies the volumes to attach the instance

Lauch the permissions to control the aws instances

**Launch permissions -**

Public - The owner grants permissions to launch

Implicit - The owner has an implicit has an AMi

Explicit - To lauch the permissions for the organizations, specific aws or Organizational units

**Step by step to Launch the AMI :**

Choose an AMI

Instance Type

Choose Key pair

Network settings

Storage

Tags

Review and Launch

We can able to Launch the AMI :

To create the Image using the instance we can able to create the Image

**why do we need ami in aws?**

In Amazon Web Services (AWS), an Amazon Machine Image (AMI) is a pre-configured virtual machine image that is used to create new instances. Essentially, an AMI is a template that contains the necessary information to launch an instance, including the operating system, application server, and any other software that you need to run your application.

**v) Security Groups:**

In Amazon Web Services (AWS), a security group is a virtual firewall that controls the inbound and outbound traffic for one or more Amazon Elastic Compute Cloud (EC2) instances. A security group acts as a set of rules that define which traffic is allowed to access the instances based on protocols, ports, and source or destination IP addresses.

Each security group in AWS has its own set of rules, and you can apply one or more security groups to an EC2 instance. When an instance is launched, it is automatically associated with one or more security groups, and traffic is allowed or blocked based on the rules defined in the associated security groups.

**vi) Elastic Ip Address:**

In Amazon Web Services (AWS), an Elastic IP address (EIP) is a static, public IPv4 address that you can allocate to your AWS account and assign to your EC2 instances or other AWS resources. Elastic IP addresses are designed to provide a way to mask the failure of an instance or software by quickly remapping the address to another instance in your account.

The main difference between IPv4 and IPv6 is the address size of IP addresses. The IPv4 is a 32-bit address and it is in numeric values, whereas IPv6 is a 128-bit hexadecimal address. And it is in alpha numeric values. IPv6 provides a large address space, and it contains a simple header as compared to IPv4.

Imagine you have a computer (a virtual one on the internet) that runs a website, and people access that website using its address, just like your home has a street address.

The problem is that this computer's address can change when you turn it off and on. So, every time it changes, people wouldn't know where to find your website.

To solve this, you can get something called an Elastic IP address, which is like a permanent address for your computer on the internet. Even if you turn off your computer and then turn it back on, it keeps the same address.

**vii) Load Balancing:**

**\*) What is a load balancer in AWS?**

Load balancing is the method of distributing network traffic equally across a pool of resources that support an application

(OR)

Load balancing in AWS refers to the practice of distributing incoming network traffic across multiple servers or instances to improve application availability, scalability, and reliability.

AWS offers several load balancing options that can be used to distribute traffic across EC2 instances or containers, including:

**Elastic Load Balancing (ELB)** - A managed load balancer service that automatically scales up or down based on traffic demands.

**Application Load Balancer (ALB)** - A layer 7 load balancer that distributes incoming traffic based on application-level content, such as URL paths and HTTP headers.

**Network Load Balancer (NLB)** - A layer 4 load balancer that distributes incoming traffic based on transport-level protocols, such as TCP and UDP.

**\*) What is the purpose of a load balancer?**

Load balancers improve application availability and responsiveness and prevent server overload.

**\*) What are the layers of load balancer?**

Load balancers are generally grouped into two categories: Layer 4 and Layer 7. Layer 4 load balancers act upon data found in network and transport layer protocols (IP, TCP, FTP, UDP). Layer 7 load balancers distribute requests based upon data found in application layer protocols such as HTTP.

**\*) What is difference between ELB and ALB?**

ELB only allows routing via a single port, while ALB supports distribution through multiple ports and lambda functions

**viii) Traget Groups**

In AWS, a target group is a logical group of resources, such as EC2 instances, IP addresses, or Lambda functions, that are registered with a load balancer and receive traffic from the load balancer.

When you create a load balancer, you must specify one or more target groups to receive traffic from the load balancer. The target group determines the routing algorithm that the load balancer uses to distribute traffic among the registered resources.

For example, if you create an Application Load Balancer and define a target group of EC2 instances, the load balancer will distribute incoming requests among the registered EC2 instances based on rules that you specify, such as round robin or least connections.

**ix) What is Launch Template?**

A launch template is similar to a [launch configuration](https://docs.aws.amazon.com/autoscaling/ec2/userguide/launch-configurations.html), in that it specifies instance configuration information. It includes the ID of the Amazon Machine Image (AMI), the instance type, a key pair, security groups, and other parameters used to launch EC2 instances. However, defining a launch template instead of a launch configuration allows you to have multiple versions of a launch template.

**What is Launch Configuration?**

A launch configuration is a set of instructions that specifies how to launch and configure instances in a particular AWS Auto Scaling group. It defines the AMI (Amazon Machine Image) to use, the instance type, and other launch parameters such as the security groups, key pairs, and block device mappings.

Once you create a launch configuration, you can associate it with an Auto Scaling group. This group can then automatically launch instances based on the specified launch configuration in response to changes in demand, such as increased traffic or higher usage.

**what is the use of launch configuration in aws?**

A launch configuration is a template that an EC2 Auto Scaling group uses to launch EC2 instances.

**What is the purpose of launch template in AWS?**

A launch template is similar to a launch configuration, in that it specifies instance configuration information. It includes the ID of the Amazon Machine Image (AMI), the instance type, a key pair, security groups, and other parameters used to launch EC2 instances.

**What is the difference between launch template and launch configuration in AWS**

launch configurations are used with Auto Scaling Groups. While launch templates are used when you launch an instance using the aws EC2 console, an AWS SDK, or a command line tool. Launch templates enable you to store the parameters (AMI, instance type, security groups, and key pairs etc.)

**What is Auto Scaling in AWS EC2?**

Amazon EC2 Auto Scaling helps you maintain application availability and lets you automatically add or remove EC2 instances using scaling policies that you define.

**(Or)**

Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application. You create collections of EC2 instances, called Auto Scaling groups. You can specify the minimum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes below this size. You can specify the maximum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes above this size. If you specify the desired capacity, either when you create the group or at any time thereafter, Amazon EC2 Auto Scaling ensures that your group has this many instances. If you specify scaling policies, then Amazon EC2 Auto Scaling can launch or terminate instances as demand on your application increases or decreases.

**Example**

 1 instance - traffic - 100 people - got the 150 - downtime (it will occur the issues)

Auto scaling - Monitor your application and it adjusts

Max - 100 - It will create the another volume , or another service -

Traffic - 100 - 50 - 60 people - another instance

1 instance we have created 100 - MAX - it is created by the auto scaling 20, 30, 60 people - 40 people

100 people - 10 left - 10

Min - 70  Max Traffic - 100  - auto scaling -

**Step by step Process :**

We have to launch the instance

Setup the load balancers and we have to connect to the instance

Create the Target groups and connect to the load balancers and in security groups

Launch the configuration

Create AMI

Setup the Auto scaling and in it we have to setup the desired capacity

**ii) Lambda:**

 Amazon released Lambda in 2015. You do not have to take care of managing Data centre, managing infrastructure as a service, managing platform as a service or container. You need to upload the code and Amazon will do everything for you.

**What is Lambda?**

* Lambda is used to encapsulate Data centres, Hardware, Assembly code/Protocols, high-level languages, operating systems, AWS APIs.
* Lambda is a compute service where you can upload your code and create the Lambda function.
* Lambda takes care of provisioning and managing the servers used to run the code.
* While using Lambda, you don't have to worry about scaling, patching, operating systems, etc.

AWS Lambda is a serverless computing platform provided by Amazon Web Services (AWS) that allows developers to run code in response to events and triggers, without the need for provisioning or managing servers.

With AWS Lambda, developers can write code in a variety of programming languages, including Python, Node.js, Java, C#, and Go. They can then upload their code to the platform and specify the events or triggers that should trigger the execution of that code. AWS Lambda will automatically provision and manage the necessary infrastructure to run the code, and scale it automatically based on the volume of requests.

**iii) What is Amazon Lightsail?**

Amazon Lightsail is a virtual private server (VPS) provider and is the easiest way to get started with AWS for developers, small businesses, students, and other users who need a solution to build and host their applications on cloud. Lightsail provides developers compute, storage, and networking capacity and capabilities to deploy and manage websites and web applications in the cloud. Lightsail includes everything you need to launch your project quickly – virtual machines, containers, databases, CDN, load balancers, DNS management etc. – for a low, predictable monthly price.

**iv) What is AWS Elastic Beanstalk**

AWS Elastic Beanstalk is a fully managed service that makes it easy to deploy and run web applications and services developed with popular programming languages such as Java, Python, .NET, Ruby, Node.js, Go, and PHP.

Elastic Beanstalk provides developers with a platform to upload their code, and then automatically handles the deployment, scaling, and management of the application on a fully managed infrastructure. This allows developers to focus on building their applications, without the need to worry about the underlying infrastructure, such as server configuration and management.

Elastic Beanstalk also provides additional features such as load balancing, auto scaling, monitoring, and logging, which help ensure that applications are scalable, reliable, and highly available.

In summary, AWS Elastic Beanstalk simplifies the process of deploying and managing web applications by providing a fully managed platform for developers, which enables them to focus on writing code and delivering business value, rather than worrying about the underlying infrastructure.

**2) Storage :**

# **i) EBS[ Elastic Block Store**]

# **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

We have to attach to instance - Volume and instance has to be in the same availability zone.

Volume – partitions or segments of a physical disk

Snapshot - it is the backup of the volume

After the volume we can able to create the volume again

Using the snapshot we can able to create the volume.

For a single instance also you have to attach the multiple volumes

**Benefit of volume**  : Data availability

Data persistence

Data security

Snapshots

Flexibility

**To mount the volume and to unmount the Steps :**

Launch the instance

Create the volume ( Volume and instance should be in the same availability zone)

Create snapshot

Attach the volume to the instance

To findout the disk - fdisk -l

Create partition space - fdisk device name (/dev/xvdf)

Format partition space - mkfs partition space

Create the directory - mkdir

Mount the partition space - mount partition name directory (/dev/xvdf march)

To verify the mount - df -h

Mountpoint - Mountpoint directoryname

To unmount we have to use - umount directory name

To verify - df –h

**ii) Amazon S3 (Simple Storage Service)**

**What is AWS S3 Used for?**

Amazon s3 is a global Service. s3 was launched in 2006. Another name of s3 is **SQS [SIMPLE QUEUE SERVICE]**

Amazon S3 (Simple Storage Service) is a highly scalable, secure, and durable object storage service provided by Amazon Web Services (AWS). S3 is designed to store and retrieve any amount of data, at any time, from anywhere on the web.

S3 provides a simple web services interface that can be used to store and retrieve data, and also supports several advanced features,

1) Providing High Durability

2) Flexible

3) Availability

4) Cost Efficient

5) Scalable

6) Secure

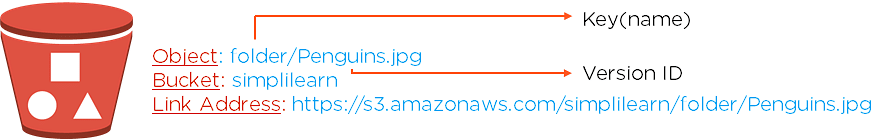
* S3 is a safe place to store the files.
* It is Object-based storage, i.e., you can store the images, word files, pdf files, etc.
* The files which are stored in S3 can be from 0 Bytes to 5 TB.
* It has unlimited storage means that you can store the data as much you want.
* Files are stored in Bucket. A bucket is like a folder available in S3 that stores the files.
* S3 is a universal namespace, i.e., the names must be unique globally. Bucket contains a DNS address. Therefore, the bucket must contain a unique name to generate a unique DNS address.

**S3 Storage Actually Works?**

**Buckets & Objects**

* **Buckets**
  + A bucket is a container used for storing the objects(files) in the cloud.
  + Every object is incorporated in a bucket.
  + For example, if the object named photos/tree.jpg is stored in the treeimage bucket, then it can be addressed by using the URL <http://treeimage.s3.amazonaws.com/photos/tree.jpg>.

An object consists of data, key (assigned name), and metadata. A bucket is used to store objects When data is added to a bucket, Amazon S3 creates a unique version ID and allocates it to the object.



* **Objects**
  + Objects are the entities which are stored in an S3 bucket.
  + An object consists of object data and metadata where metadata is a set of name-value pair that describes the data.
  + An object consists of some default metadata such as date last modified, and standard HTTP metadata, such as Content type. Custom metadata can also be specified at the time of storing an object.
  + It is uniquely identified within a bucket by key and version ID.
* **Key**
  + A key is a unique identifier for an object.
  + Every object in a bucket is associated with one key.
  + An object can be uniquely identified by using a combination of bucket name, the key, and optionally version ID.
  + For example, in the URL http://jtp.s3.amazonaws.com/2019-01-31/Amazons3.wsdl where "jtp" is the bucket name, and key is "2019-01-31/Amazons3.wsdl"
* **Regions**
  + You can choose a geographical region in which you want to store the buckets that you have created.
  + A region is chosen in such a way that it optimizes the latency, minimize costs or address regulatory requirements.
  + Objects will not leave the region unless you explicitly transfer the objects to another region.

**S3 Storage Classes:**

**1) S3 Standard :** This is the default storage class and provides high durability, availability, and performance for frequently accessed data. It is suitable for storing data that requires real-time access and low latency. [ **In case if there is some data that need to be fetch right away** ]

**2) S3 Standard-Infrequent Access (S3 Standard-IA):** This storage class is designed for data that is accessed less frequently but requires rapid access when needed. It provides a lower storage cost than S3 Standard but has a retrieval fee.

**3) S3 Glacier:** This storage class is designed for long-term data archival and provides the lowest storage cost among all S3 storage classes. It has a minimum storage duration of 90 days and a retrieval fee.

**4) S3 Intelligent-Tiering:** This storage class is designed to optimize costs by automatically moving data between two access tiers based on changing access patterns. It provides the same level of durability and availability as S3 Standard.

**5) S3 One Zone-Infrequent Access (S3 One Zone-IA**): This storage class is designed for infrequently accessed data that can be recreated if lost. It offers lower storage prices than S3 Standard-IA, but stores data in a single availability zone, making it less resilient to zone-level failures.

**6) S3 Glacier Deep Archive:** This storage class is designed for archiving data that is rarely accessed and has a retrieval time of 12 hours or more. It offers the lowest storage prices of all the storage classes in Amazon S3.

# **Versioning:**

**Versioning** is a means of keeping the multiple forms of an object in the same S3 bucket. Versioning can be used to retrieve, preserve and restore every version of an object in S3 bucket.

For example, bucket consists of two objects with the same key but with different version ID's such as photo.jpg (version ID is 11) and photo.jpg (version ID is 12).

Versioning-enabled buckets allow you to recover the objects from the deletion or overwrite. It serves two purposes:

* If you delete an object, instead of deleting the object permanently, it creates a delete marker which becomes a current version of an object.
* If you overwrite an object, it creates a new version of the object and also restores the previous version of the object.

# **Cross Region Replication**

* Cross Region Replication is a feature that replicates the data from one bucket to another bucket which could be in a different region.
* It provides asynchronous copying of objects across buckets. Suppose X is a source bucket and Y is a destination bucket. If X wants to copy its objects to Y bucket, then the objects are not copied immediately.

**Some points to be remembered for Cross Region Replication**

* **Create two buckets:** Create two buckets within AWS Management Console, where one bucket is a source bucket, and other is a destination bucket.
* **Enable versioning:** Cross Region Replication can be implemented only when the versioning of both the buckets is enabled.
* **Amazon S3 encrypts the data in transit across AWS regions using SSL:** It also provides security when data traverse across the different regions.
* **Already uploaded objects will not be replicated:** If any kind of data already exists in the bucket, then that data will not be replicated when you perform the cross region replication

**Lifecycle Management**

**Lifecycle Management** is used so that objects are stored cost-effectively throughout their lifecycle. A **lifecycle configuration** is a set of rules that define the actions applied by S3 to a group of objects.

**The lifecycle defines two types of actions:**

**Transition actions:** When you define the transition to another storage class. For example, you choose to transit the objects to Standard IA storage class 30 days after you have created them or archive the objects to the Glacier storage class 60 days after you have created them.

**Expiration actions:** You need to define when objects expire, the Amazon S3 deletes the expired object on your behalf.

Suppose business generates a lot of data in the form of test files, images, audios or videos and the data is relevant for 30 days only. After that, you might want to transition from standard to standard IA as storage cost is lower. After 60 days, you might want to transit to Glacier storage class for the longtime archival. Perhaps you want to expire the object after 60 days completely, so Amazon has a service known as Lifecycle Management, and this service exist within S3 bucket.

**iii) Amazon S3 Glacier:**

Amazon S3 Glacier (S3 Glacier) is a secure and durable service for low-cost data archiving and long-term backup. With S3 Glacier, you can store your data cost effectively for months, years, or even decades.

S3 Glacier provides three retrieval options, ranging from Expedited, Standard, and Bulk, depending on the amount of time required to retrieve the data. Expedited retrieval allows users to retrieve data in 1-5 minutes, while Standard and Bulk retrievals can take 3-5 hours and 5-12 hours, respectively.

Overall, S3 Glacier provides a cost-effective, durable, and secure archiving solution that is ideal for organizations that need to retain data for long periods of time, but do not need immediate access to it.

**iv) What is Amazon Elastic File System?**

Amazon Elastic File System (Amazon EFS) provides a simple serverless, fully elastic file storage so that you can share file data without provisioning or managing storage capacity and performance.

It can be used with aws cloud services or on premises resources and it is build to scale and demand without distributing your applications

**When to use AWS ELASTIC FILE SYSTEM?**

Amazon supports file object and block level storage

**1) for file storage we can use EFS**

**2) for object storage we can use AWS S3**

**3) for block level storage we can use AWS EBS**

EFS can use for application which required shared file system that can be accessed through multiple system at the same time. It can be used by application that access the data using a standard files system interface provided through the operating system they can take the advantage of scalability and reliability of storage in the cloud without writing a new code for adjusting applications

Object storage: it can be use by application that do not require for a file system structure and designed to work with objects storage. Those steps applications can be used amazon S3 which provides scalable cost efficient storage to store any type of data in his native format

Block storage : data broken into blocks and stored as a separate piece . Each block has a unique identifier. In AWS elastic block storage for broad range of cloud such as enterprises application, containerized applications, data analysis, etc.

**Difference between EBS and EFS?**

EFS: Thousands of instances for on premises servers multiple availability zones can concurrently access the file systems

EBS: only a single ec2 instance a single availability zones access the data

**v) Storage Gateway:**

An AWS Storage Gateway is like a bridge between your on-premises (local) data and the cloud. It makes it easier to store, access, and manage your data in Amazon Web Services (AWS) from your own servers or data centers.

Imagine it as a translator or connector that allows your local data to communicate with AWS services like Amazon S3, EBS, or Glacier.

**3) Security Services :**

**i) Identity Access Management:[Authenticating Authorization Management]**

**When you are granting permissions to users to access any services on AWS then we use IAM users whereas if we are granting permissions to any service to access another service in AWS then we use IAM roles**

In AWS (Amazon Web Services), IAM stands for Identity and Access Management. It is a service that helps you manage access to AWS resources by creating and managing AWS users and groups, and by setting permissions for them.

With IAM, you can create individual users, each with their own set of credentials (such as a username and password), and define what actions they can perform on AWS resources. You can also organize these users into groups, and apply permissions to the groups instead of individual users.

IAM also allows you to set up roles, which are sets of permissions that can be assumed by an AWS service or an external entity. This allows you to delegate access to AWS resources to services or applications without sharing credentials.

**Policies:**

There are two types of policies

1)aws managed policies->aws policy

2)customer managed policies -> create your own policy

# **What is a Role?**

In AWS IAM, a role is a set of permissions that define what actions an entity can perform on AWS resources. A role can be assigned to an AWS service or an AWS user, allowing them to assume the permissions of the role temporarily, without the need for the user or service to have their own credentials.

For example, if you have an EC2 instance that needs to access an S3 bucket, instead of creating an AWS user with the necessary permissions, you can create a role that grants access to the S3 bucket, and assign that role to the EC2 instance. When the instance needs to access the bucket, it assumes the role and is granted the necessary permissions.

**when do i use iam?**

You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

**ii) Guard Duty:**

Amazon GuardDuty is a threat detection service offered by AWS. It is a managed threat detection service that continuously monitors your AWS accounts and workloads for malicious or unauthorized behavior.

GuardDuty uses a combination of machine learning, anomaly detection, and integrated threat intelligence to identify potential security threats, such as unauthorized access, cryptojacking, and data exfiltration. It analyzes various sources of data, including AWS CloudTrail event logs, Amazon VPC Flow Logs, and DNS logs, to detect potential security issues.

When GuardDuty identifies a potential security threat, it generates a security finding, which includes details about the threat, such as the type of attack, the affected resources, and recommended actions for remediation. You can view these findings in the GuardDuty console or configure it to send them to Amazon SNS or AWS Lambda for automated remediation.

**Management Services**

**i) Cloud Trail**

**Why do we need CloudTrail?**

The Various capabilities of cloud trail are essential to simplify operation analysis and troubleshooting it directly contributes in achieving business goals faster

(Or)

CloudTrail enables auditing, security monitoring, and operational troubleshooting by tracking user activity and API usage.

**What is CloudTrail?**

AWS CloudTrail is a service provided by Amazon Web Services (AWS) that provides a record of all AWS API calls made in an account. It enables users to log, monitor, and retain account activity related to actions performed on AWS resources.

With CloudTrail, you can get visibility into account activity by recording API calls for services such as Amazon S3, Amazon EC2, and AWS Identity and Access Management (IAM). CloudTrail logs include details about who made the API call, the source IP address of the request, when the API call was made, and other metadata about the request.

(OR)

CloudTrail is a Management Service provided by AWS which enables governance, compliance, operational auditing as well as risk auditing of your AWS Account

Ex:[ so that basically means that it is a service that lets the user monitor user activity and the host activity about whatever is going on in the account it might even be about api activity and that is why cloudtrail gives the user a fair overview of what is going on in the account ]

**What is the use of CloudWatch and CloudTrail?**

CloudWatch is a monitoring service for AWS resources and applications. CloudTrail is a web service that records API activity in your AWS account. CloudWatch monitors applications and infrastructure performance in the AWS environment. CloudTrail monitors actions in the AWS environment.

**iv) Amazon inspector**

**What does Amazon inspector do?**

Amazon Inspector is an automated vulnerability management service that continually scans AWS workloads for software vulnerabilities and unintended network exposure.

**v) Certificate Manager**

AWS Certificate Manager (ACM) is a service provided by Amazon Web Services (AWS) that makes it easy to provision, manage, and deploy SSL/TLS certificates for use with AWS services and other internal resources.

ACM provides a simple, fully-managed solution for obtaining SSL/TLS certificates to secure your web applications and websites. You can use ACM to request public certificates for use with AWS resources such as Elastic Load Balancers, CloudFront distributions, and API Gateway endpoints. You can also use ACM to generate private certificates for internal use within your organization.

**vi) AWS WAF**

AWS WAF (Web Application Firewall) is a managed security service provided by Amazon Web Services (AWS) that helps protect web applications from common web exploits and attacks such as SQL injection, cross-site scripting (XSS), and more.

WAF provides a firewall that allows you to inspect and control inbound and outbound traffic to your web applications. You can create rules that allow, block, or monitor traffic based on various criteria such as IP addresses, HTTP headers, or the content of the request. WAF can also integrate with Amazon CloudFront and Application Load Balancers to protect your web applications running on AWS.

WAF also provides a set of pre-configured rules that you can use to protect your applications immediately. These rules are designed to protect against common web exploits and attacks. Additionally, you can create your own custom rules based on your specific application and security requirements.

WAF provides real-time monitoring of traffic and provides detailed logs and metrics to help you understand and respond to threats quickly. It also integrates with other AWS services such as AWS CloudFormation, AWS Lambda, and Amazon SNS to automate security workflows and enable you to respond to threats automatically.

**What is the difference between AWS WAF and AWS Shield?**

The difference between them is that AWS WAF (Web Application Firewall) provides protection on the application layer and AWS Shield protects the infrastructure layers of the OSI model.

**vii) AWS Key Management Service KMS:**

**What is AWS Key Management Service KMS used for?**

Overview. AWS Key Management Service (KMS) gives you centralized control over the cryptographic keys used to protect your data. The service is integrated with other AWS services making it easier to encrypt data you store in these services and control access to the keys that decrypt it.

**Where are KMS keys used?**

AWS KMS keys (KMS keys) are the primary resource in AWS KMS. You can use a KMS key to encrypt, decrypt, and re-encrypt data. It can also generate data keys that you can use outside of AWS KMS.

**4) Networks :**

**i) Amazon Virtual Private Cloud (VPC)**

VPC stands for Virtual Private Cloud, and it is a service provided by Amazon Web Services (AWS) that allows users to create a private, isolated virtual network in the AWS cloud.

With VPC, users can define a private IP address range, create subnets, and configure routing tables, network gateways, and security settings. Users can also choose to connect their VPC to their on-premises network or other VPCs in the same or different AWS regions.

**Why should I use Amazon VPC?**

Amazon VPC enables you to build a virtual network in the AWS cloud - no VPNs, hardware, or physical datacenters required. You can define your own network space, and control how your network and the Amazon EC2 resources inside your network are exposed to the Internet

**What is the use of subnet in VPC AWS?**

A subnet is a range of IP addresses in your VPC. You launch AWS resources, such as Amazon EC2 instances, into your subnets. You can connect a subnet to the internet, other VPCs, and your own data centers, and route traffic to and from your subnets using route tables.

**what is route table in vpc**

In Amazon Web Services (AWS) Virtual Private Cloud (VPC), a route table is a set of rules that determine how network traffic is directed within a VPC or between a VPC and other networks.

Every VPC has a default route table, which is created automatically when the VPC is created. You can also create additional custom route tables and associate them with specific subnets in the VPC.

**What is internet gateway in VPC?**

An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between your VPC and the internet. It supports IPv4 and IPv6 traffic.

**What is NAT and internet gateway?**

A NAT gateway is a Network Address Translation (NAT) service. You can use a NAT gateway so that instances in a private subnet can connect to services outside your VPC but external services cannot initiate a connection with those instances.

AWS Direct Connect, and AWS Elastic Load Balancing

**Amazon Route 53:**

Route 53 is a web service that is a highly available and scalable Domain Name System (DNS.)

Let’s understand what is Amazon Route 53 in technical terms. AWS Route 53 lets developers and organizations route end users to their web applications in a very reliable and cost-effective manner. It is a Domain Name System (DNS) that translates domain names into IP addresses to direct traffic to your website. In simple terms, it converts World Wide Web addresses like www.example.com to IP addresses like 10.20.30.40

**What is the use of Route 53?**

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. Route 53 connects user requests to internet applications running on AWS or on-premises.

**What are the three functions of Route main 53?**

Doamin Registration, Dns Routing, Health Checks

**What is health check protocol?**

HealthCheckProtocol. The protocol the load balancer uses when performing health checks on targets. The possible protocols are HTTP, HTTPS, and TCP. The default is the TCP protocol. If the target type is alb , the supported health check protocols are HTTP and HTTPS.

**What is CIDR in VPC?**

CIDR is an acronym that stands for Classless Inter-Domain Routing. In simpler terms, a CIDR block is an IP address range. A VPC can accommodate two CIDR blocks, one for IPv4 and another for IPv6, but for the sake of simplicity I am going to limit my discussion to IPv4.

**VPC (Virtual Private Cloud):**

A Virtual Private Cloud (VPC) is a virtual network dedicated to your AWS account. It enables you to launch AWS resources, like EC2 instances, in a logically isolated section of the AWS Cloud. Think of it as your own virtual data center in the cloud.

**Example:** Imagine you're building a house. The land your house sits on represents the VPC. It's your own space, separate from your neighbors' properties. Within this space, you can build rooms (subnets) and connect them with roads (routes) as you see fit.

**Subnets:**

Subnets are subdivisions of your VPC. They allow you to segment your network and organize your resources. Each subnet resides in one Availability Zone and can contain multiple instances.

**Example:** Going back to the house analogy, think of each room in your house as a subnet. You might have a living room (public subnet) and a bedroom (private subnet). Each serves a different purpose but exists within your overall property (VPC).

**Route Tables:**

Route tables determine where network traffic is directed within your VPC. They contain a set of rules, called routes, that specify the destination of traffic leaving the subnet.

**Example:** Suppose you want to go to the grocery store (destination). The route you take (route table) might differ based on whether you're walking (local route) or driving (Internet route). Route tables decide which path your traffic follows.

**Internet Gateway:**

An Internet Gateway (IGW) is a component that allows communication between your VPC and the internet. It enables resources within your VPC to connect to the internet and vice versa.

Example: Picture an IGW as the gateway to your neighborhood. It's the bridge that connects your house (VPC) to the outside world (internet). Without it, you'd be confined to your own property.

**NAT Gateway:**

A Network Address Translation (NAT) Gateway allows instances in a private subnet to initiate outbound traffic to the internet while preventing inbound traffic from reaching them. It provides internet access to resources that don't have public IP addresses.

Example: Think of a NAT Gateway as a receptionist at a gated community. Residents (instances in private subnets) can leave the community (access the internet) through the gate (NAT Gateway), but visitors (inbound traffic) can't enter without invitation.

**Security Groups:**

Security Groups act as virtual firewalls for your instances. They control inbound and outbound traffic at the instance level, allowing you to specify which traffic is allowed to reach your instances.

**Example:** Imagine security guards at the entrance of each room in your house. They check who is allowed to enter (inbound traffic) and who can leave (outbound traffic) based on predefined rules (security group settings).

**Network Access Control Lists (NACLs):**

NACLs are stateless, rule-based network traffic filters that control traffic in and out of subnets. Unlike security groups, which operate at the instance level, NACLs operate at the subnet level.

**Example:** Think of NACLs as checkpoints at the entrance of your neighborhood. They inspect every person or vehicle entering or leaving, following specific rules (NACL rules) set by the community management.

**Route 53:**

Route 53 is AWS's scalable domain name system (DNS) web service. It translates human-readable domain names into IP addresses and vice versa, enabling users to access websites and web applications using domain names.

Example: When you type a website address (e.g., www.example.com) into your browser, Route 53 functions like a digital phonebook, finding the corresponding IP address (e.g., 192.0.2.1) and directing your browser to the correct web server.

**5) Database :**

RDS

Dynamodb

Elastic cache

Redshift

**AWS CodeCommit:**

AWS CodeCommit is a fully managed source control service that hosts secure Git-based repositories. It allows teams to collaborate on code in a secure and highly scalable environment. With CodeCommit, you can store anything from code to binaries, and it integrates seamlessly with other AWS services like CodePipeline and CodeBuild.

**Example:** Imagine you're working on a software project with a team. You can use CodeCommit to store all your project's source code securely in the cloud. Each team member can clone the repository, make changes, and push them back, facilitating collaboration and version control.

**Advantages:**

Fully managed service by AWS, eliminating the need for infrastructure management.

Highly scalable and secure, with built-in encryption and access controls.

Integrates seamlessly with other AWS DevOps tools.

**Disadvantages:**

Limited to Git repositories only.

Pricing can be higher compared to self-hosted Git solutions for large projects with many users.

**AWS CodePipeline:**

AWS CodePipeline is a continuous integration and continuous delivery (CI/CD) service that automates the build, test, and deployment phases of your release process. It allows you to create automated workflows, called pipelines, to orchestrate the different stages of your software delivery process.

**Example:** Let's say you have a web application. With CodePipeline, you can set up a pipeline that triggers whenever you push new code to your CodeCommit repository. The pipeline can then automatically build the code using CodeBuild, run tests to ensure quality, and deploy the application to AWS infrastructure using CodeDeploy.

**Advantages:**

Automates the entire software release process, improving efficiency and reducing errors.

Supports integration with third-party tools and services.

Provides visualization of the pipeline, making it easy to track the progress of releases.

**Disadvantages:**

Can be complex to set up and configure initially, especially for beginners.

Limited customization options compared to self-hosted CI/CD solutions.

**AWS CodeBuild:**

AWS CodeBuild is a fully managed build service that compiles source code, runs tests, and produces software packages that are ready for deployment. It allows you to offload the build process to AWS, freeing up your local resources and providing scalability and reliability.

**Example**: In your CodePipeline, you can use CodeBuild as one of the stages to compile your source code, run unit tests, and generate artifacts such as executable files or Docker containers.

**Advantages:**

Fully managed service, eliminating the need to provision and manage build servers.

Provides preconfigured build environments or allows you to create custom build environments.

Scales automatically based on workload demands.

**Disadvantages:**

Build times can vary depending on the complexity of the project and available resources.

Limited support for certain programming languages and frameworks.

**AWS CodeDeploy:**

AWS CodeDeploy is a deployment service that automates the process of deploying applications to a variety of compute services, including Amazon EC2 instances, AWS Lambda functions, and on-premises servers. It allows you to deploy your application code reliably and consistently across different environments.

**Example:** Suppose you have a microservices-based architecture deployed on Amazon EC2 instances. With CodeDeploy, you can automate the deployment of new versions of your services, ensuring zero downtime and rollback capabilities if any issues arise.

**Advantages:**

Automates the deployment process, reducing the risk of human error.

Supports blue/green deployments for zero-downtime releases.

Provides detailed deployment logs and monitoring capabilities.

**Disadvantages:**

Requires some initial setup and configuration to integrate with your application.

Limited support for certain deployment scenarios, such as serverless deployments.