****

**Cloud Computing**

* why,
* how to use,
* where we use

Shared Responsibility Mode:

<https://github.com/Vaibhav8855/AWS/blob/main/EC2.md>

Steps of migration:

<https://medium.com/@singhsaloni890/a-step-by-step-guide-to-successful-aws-data-migration-project-a-hands-on-journey-1753c13168bb>

**Cloud Computing:**Cloud computing provide on-demand services like Compute storage, security, CDN(Content Delivery Network), network via internet as pay as you go pricing

**Advantage of Cloud Computing:**

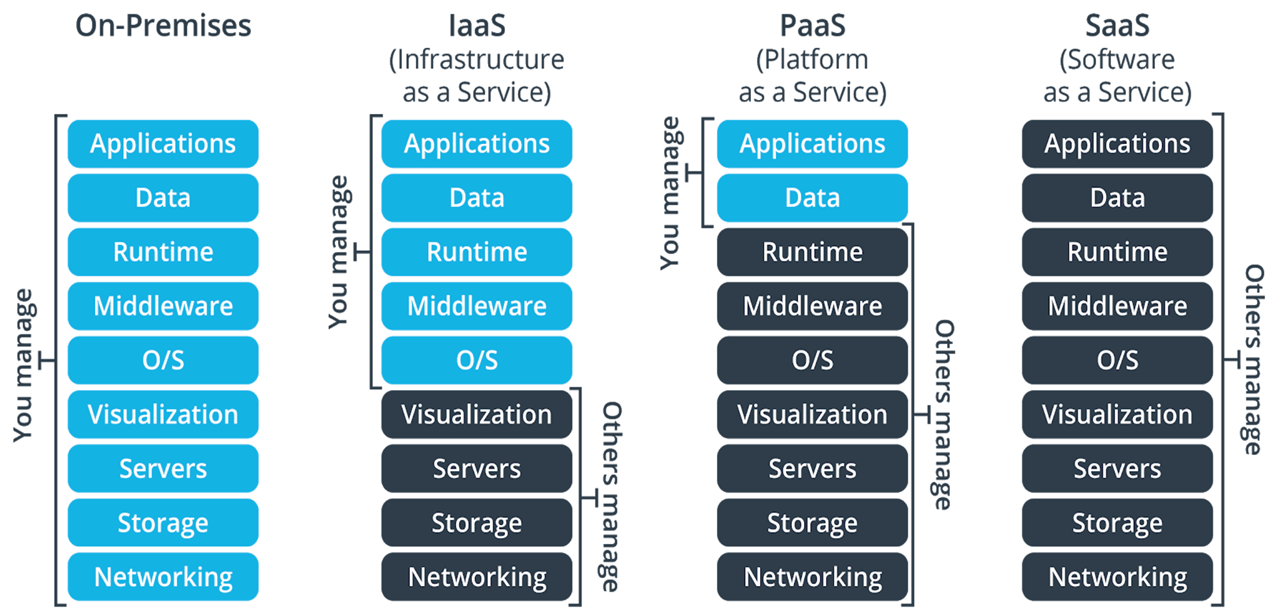
* 1. Cost Optimization: You only pay for what you use,
  2. Scalability: Easily scale up or down based on your needs without having to invest in new infrastructure.
  3. Fault tolerance: if one part of the system fails, other parts continue to function
  4. High Availability: Access your data and applications from anywhere with an internet connection,
  5. Secure:

**Disadvantage of Cloud Computing:**

* 1. Dependancies of Internet: You need a good internet connection to access cloud services.
  2. Lack of Controle:
  3. Security:While cloud providers work hard to keep your data safe, storing sensitive information online can still pose security risks.

**Cloud Computing Models in simple word?**

Cloud computing models are different ways to use and deliver cloud services. Here’s a simple overview:



1. **Infrastructure as a Service (IaaS)**: In AWS, Infrastructure as a Service (IaaS) means you can use virtual servers, storage, and networking without needing to own physical hardware. AWS provides these resources over the internet. You pay only for what you use, and you control how these resources are set up and used for your applications

Provides the basic building blocks, like virtual servers and storage, that you can use to build your own applications. Think of it as renting computer hardware.

1. **Platform as a Service (PaaS)**: Platform as a Service (PaaS) provides hardware and software infrastructure that you can use to develop and maintain applications. You can build, test, run, and scale applications faster and at a lower cost by using PaaS than on your on-premises infrastructure.
2. **Software as a Service (SaaS)**: Delivers fully functional applications over the internet. You use the software directly, like email or office tools, without needing to install or maintain anything.

**Types Of Cloud Computing:**

**1.Public Cloud:** Public cloud is open to all to store and access information via the Internet using the pay-per-usage Prizing.

In public cloud, computing resources are managed and operated by the Cloud Service Provider (CSP).

**2.Private Cloud**: Private cloud is also known as an **internal cloud** or **corporate cloud**. It is used by organizations to build and manage their own data centres internally or by the third party.

**3.Hybrid Cloud** is a combination of the public cloud and the private cloud. we can say: **Hybrid Cloud = Public Cloud + Private**

IAM:[AWS IAM | AWS Cheat Sheet (digitalcloud.training)](https://digitalcloud.training/aws-iam/#:~:text=An%20IAM%20user%20is%20an,access%20to%20the%20management%20console)

**why we use iam?** AWS Identity and Access Management (IAM) is used in AWS to securely control access to AWS services and resources.

**when we use iam ?**IAM (Identity and Access Management) in AWS is used whenever there is a need to manage access to AWS resources securely.

1. **User:Steps:**  <https://medium.com/@sam.xzo.developing/create-aws-iam-user-02ee9c65c877>

IAM user is an entity that represents a person or service.

We can assign an access key ID and secret access key for programmatic access to the AWS API, CLI, SDK,

Max 5000 users are created in iam

We can attach maximum 20 policies

1. **Roles**: In AWS IAM, roles give temporary permissions to trusted users or services for certain tasks.

Max 1000 roles are created in iam.

policies that can be attached to an IAM role to 10

1. **group**:Steps:<https://medium.com/@ayush.ranjan0503/creating-iam-users-and-groups-on-aws-a-step-by-step-guide-9e576e8a9e17>

Groups are collections of users and have policies attached to them.

Maximum 20 groups are created

1. **Policies:** policies are documents that define the permission and attachched these permissions to the user, group and roles.

There are two types of policies

1. **Aws managed policies 2. Costumer manages policies**

**1.identity based policy:** identity based policies are attached with user, group and roles

**2.Resource based policies:**Resource based policies are attached with resources like s3 bucket, Ec2 instance.

1. **Multi-Factor Authentication (MFA)**: IAM supports MFA, adding an extra layer of security to user accounts

Use MFA to increase the security of your AWS environment.

**IAM best practices:**

1. **Use Least Privilege:** Grant only the permissions needed to perform a task.
2. **Enable Multi-Factor Authentication (MFA):** Add an extra layer of security for all accounts.
3. **Use Groups to Assign Permissions:** Manage permissions through groups rather than individual users.
4. **Rotate Credentials Regularly:** Periodically change passwords and access keys.
5. **Use IAM Roles:** Assign roles to services and applications instead of using long-term credentials.
6. **Monitor and Audit IAM Activity:** Use AWS CloudTrail and IAM Access Analyzer to track changes and access.
7. **Enforce Strong Password Policies:** Implement complex password requirements and expiration policies.
8. **Limit Root Account Usage:** Use the root account only for tasks that require it, and secure it with MFA.

**Steps for launch instance:**

When explaining this to an interviewer, you can summarize the steps like this:

1. Log in to the AWS Management Console.
2. Click "Launch Instance" to start the process.
3. Choose an Amazon Machine Image (AMI) that includes the operating system.
4. Select the instance type based on your requirements.
5. Configure instance details, such as network settings.
6. Add storage to the instance.
7. (Optional) Add tags to organize and identify the instance.
8. Set up a security group to define inbound and outbound traffic rules.
9. Choose or create a key pair for SSH access.
10. Review the configuration and launch the instance.

**EC2 instance :** [Amazon EC2 | AWS Cheat Sheet (digitalcloud.training)](https://digitalcloud.training/amazon-ec2/)

Steps:- <https://medium.com/@jeetanshu/a-step-by-step-guide-to-launching-an-ec2-instance-on-aws-530ee00ceb8>

**when use an EC2 instance ?** You use an EC2 instance in various scenarios, including:

1. **Web Hosting:** Hosting dynamic websites and web applications.
2. **Development and Testing:** Creating isolated environments for dev/test.
3. **Big Data Processing:** Running big data frameworks like Hadoop.
4. **Batch Processing:** Handling large-scale batch processing jobs.
5. **Machine Learning:** Training and deploying ML models.
6. **Disaster Recovery:** Setting up backup and disaster recovery solutions.
7. **Gaming:** Hosting multiplayer game servers.

**Region:**

* An AWS region is like a specific area or location where Amazon Web Services (AWS) has its data centers. Each region is like a separate zone that contains multiple data centers.
* The primary purpose of regions in AWS is to provide infrastructure redundancy, improve fault tolerance, and reduce latency for users in different parts of the world.

**Availability Zone:**

An AWS availability zone (AZ) is an isolated location within an AWS region. Each availability zone is essentially a separate data center facility with its own power, cooling, and networking infrastructure.

* + - 1. **ec2:** an ec2 is a virtual machine that represent the application

ec2 contain specific type of CPU, memory, storage.

With the help of AMI(Amazone Machine Image) we can create a ec2 instance.

20 instances per region.

* + - 1. **name and tag:** A name and tag are a Label or metadata which is assign to the ec2 instance.
      2. **Ami:** use preconfigured templates for your instances known as Amazon Machine Images (AMIs).

Each AMI includes the information needed to launch your EC2 instance like application, operating System, application server

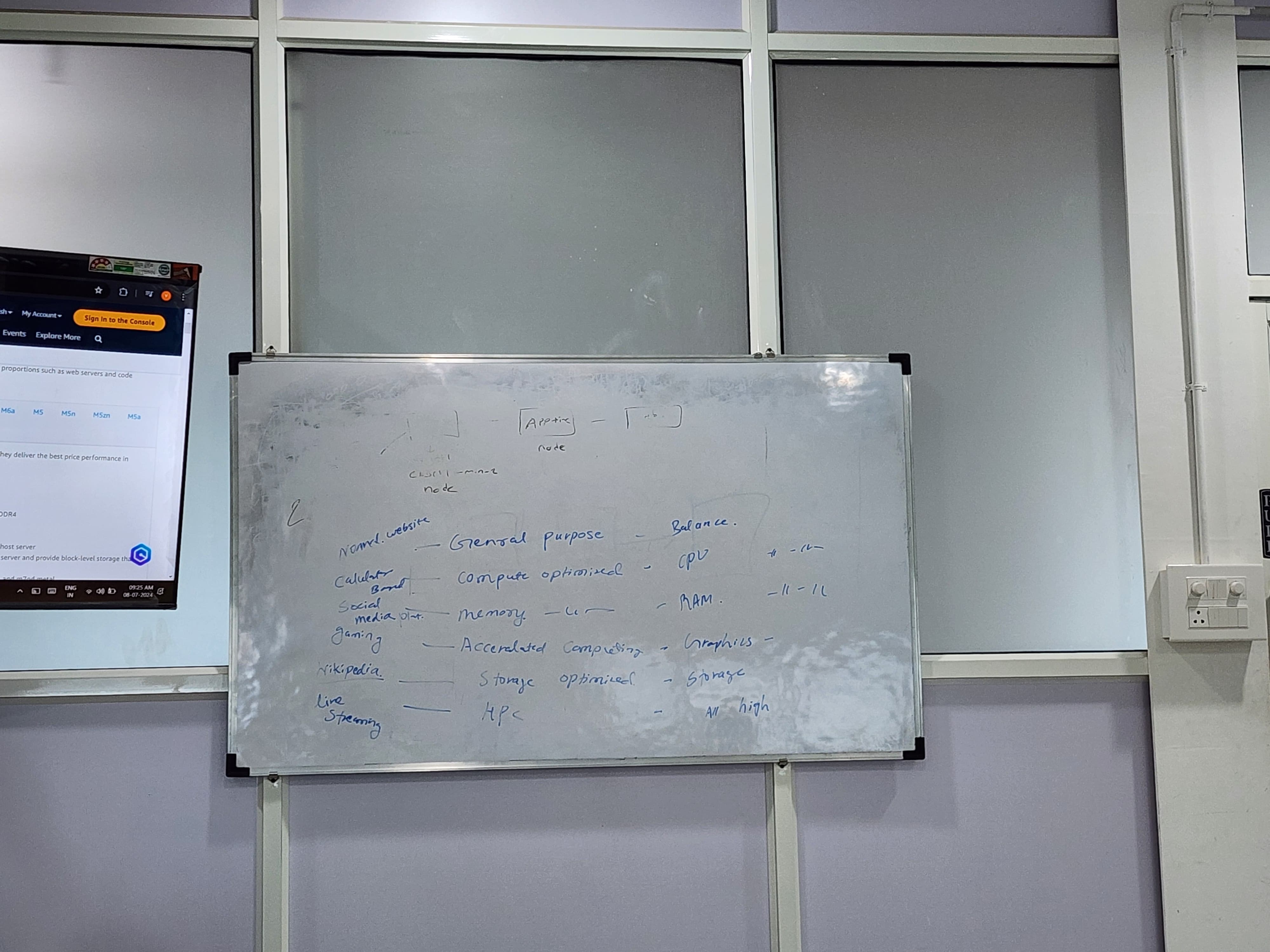
By default, AWS allows you to have up to **50 AMIs** per region

* + - 1. **instance type:**

1. **General Purpose:** Use General Purpose instances for balanced computing, memory, and storage
2. **Compute Optimized:** applications that need high processing power and fast performance, like complex calculations or data processing.
3. **Memory Optimized:** applications that require a lot of memory to handle large datasets
4. **Storage Optimized:** High I/O performance for large datasets.
5. **Accelerated Computing:** when you need extra processing power for tasks like machine learning or graphics

**Families of Instance**

1. **General purpose**: A1 | M1 | M2 | M3 | M4 | T1
2. **Compute optimized**: C1 | C3 | C4
3. **Memory optimized**: R3 | R4
4. **Storage optimized**: I2
5. **Accelerated computing**: G3



* + - 1. **key pairs**: Key pairs are used to securely connect to EC2 instances.

A key pair in AWS is like a password you use to securely connect to an Amazon EC2 instance.

A key pair consists of a public key that AWS stores, and a private key file that user store.

* + - 1. **security group:** It performs the function of a virtual firewall, managing the inbound and outbound traffic for one or more Amazon EC2 instances.

With the help of ip address, port number we can controle the inbound and outbound traffic.

* + - 1. **volumes:**
      2. **User Data** is used to set up the instance when it starts, User Data is a script or set of commands that you can provide when launching an EC2 instance
      3. **Metadata is information about the EC2 instance itself that can be accessed from within the instance.Metadata** provides information about the instance itself.

**Prizing option in AWS:**

AWS offers several pricing options to help manage costs based on usage patterns. Here are the main options:

1. **On-Demand**: Pay for computing power by the hour or second without long-term commitments, perfect for unpredictable needs.
2. **Reserved Instances**: Commit to using AWS services for a one- or three-year term and receive a significant discount compared to on-demand pricing. Good for predictable workloads.

There are three types of Reserved instance:

**Standard**: Cheapest option for long-term, regular use.

**Convertible**: Allows changes in instance types or regions while still saving money.

**Scheduled**: Reserve capacity for specific times each week, like only during certain hours

1. **Spot Instances**: Purchase unused EC2 capacity at a lower price, but the availability can vary. Suitable for flexible and cost-sensitive applications.
2. **Savings Plans**: Commit to a certain amount of usage over a one- or three-year term in exchange for lower rates. Offers flexibility with the instance types and regions.
3. **Dedicated Hosts**: Pay for a physical server dedicated to your use, offering more control over the underlying hardware and licensing.
4. **Dedicated Instances**: EC2 instances that run on hardware dedicated to you, but within a shared physical server..

**OR**

**pricing options:**

* **On-demand instance**: unpredictable workload, spiky
* **spot** **instances**: 90% discount, predictable work, interruptible
* **reserved instances**: 1-3 Y 75% discount no upfront partial upfront
* **saving plans:** 1-3 Y 75% discount no option
* **dedicated host**: isolated hardware for organizations
* **dedicated instances**: it is used for users

**Instance storage:- There are two types of storage:**

1.Instances Store Backed EC2

2 .EBS(Elastic Block Storage)/Volume/Hard Disk

**1.Instances Store Backed EC2:**Basically the virtual hard drive on the host allocated to this EC2 instance.

• Limit to 10GB per device

• Temporary data storage

• The EC2 instance can’t be stopped, can only be rebooted or terminated. Terminate will delete data.

**2 .EBS(Elastic Block Storage)/Volume/Hard Disk:-**

* + EBS volume behaves like RAW, unformatted, external block storage devices that you can attached to your EC2 instance.
  + Per AWS account up to 5000 EBS volumes can be created.
  + EBS volumes are block storage devices suitable for database style data that requires frequent reads and writes.
  + EBS volumes are attached to your EC2 instances through the AWS network, like virtual hard drive.
  + Both EBS volumes and EC2 instances must be in the same AZ.
  + EBS is used for booting purpose
  + It is virtual disk

**Why we use EBS in AWS?**

We use EBS in AWS to provide reliable, scalable storage for data that can be attached to EC2 instances.

**Types of EBS:-**

**1. SSD(solid state drive) backed volume:-There are two types**

**1.general purpose:**

1. gp3: iops: 16,000 (64 KiB I/O)

Durability: 99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)

throughput per volume: 1000 MiB/s

2. gp2: iops: 16,000 (16 KiB I/O)

Durability: 99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)

throughput per volume: 250 MiB/s 1

**2.provissional IOPS(input output speed per second) SSD :**

1. io1:

* + - * + Durability: 99.999% durability (0.001% annual failure rate)
        + IOPS: 64,000 (16 KiB I/O)
        + throughput per volume: 1,000 MiB/s 2
        + Amazon EBS Multi-attach Not supported

2. io2:

* + - * + IOPS: 256,000 (16 KiB I/O) 5
        + Durability: 99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
        + throughput per volume: 4,000 MiB/s
        + Amazon EBS Multi-attach supported
  1. **Hdd**:-

1. Throughput optimized HDD:

* st1: 99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
* IOPS : 500
* Throughput: 500 MiB/s
* Amazon EBS Multi-attach Not supported

1. Cold HDD:

* Sc1: 99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
* IOPS : 250
* Throughput: 250 MiB/s
* Amazon EBS Multi-attach Not supported

Use case : Big Data

* 1. **Magnetic standard**:- operational speed is low so we don’t use these type of EBS

Input and outputs per second 40-200

Througput (speed/performance) 40–90 MiB/s

|  |  |  |
| --- | --- | --- |
| **Feature** | **HDD (Hard Disk Drive)** | **SSD (Solid State Drive)** |
| 1. **Technology** | Magnetic storage with spinning disks | Flash memory with no moving parts |
| 1. **Performance** | Slower read/write speeds (80-160 MB/s) | Faster read/write speeds (200-3500+ MB/s) |
| 1. **Durability** | More susceptible to physical damage | More durable, resistant to physical shock |
| 1. **Reliability** | Limited by mechanical wear and tear | Longer lifespan, limited by write cycles |
| 1. **Power Consumption** | Higher due to mechanical parts | Lower, more energy-efficient |
| 1. **Noise** | Generates noise from moving parts | Silent operation |
| 1. **Size and Weight** | Larger and heavier | Smaller and lighter |
| 1. **Cost** | Cheaper per gigabyte | More expensive per gigabyte |
| 1. **Application** | Suitable for bulk storage | Ideal for OS, applications, and fast access |

**Snapshot:**

* + - snapshot is nothing but a backup or A snapshot is a copy of your data from an Amazon EBS.
    - Per account up to 10,000 EBS snapshots can be created.

**NIC:**

* + A NIC, or Network Interface Card, is a hardware component that connects a computer to a network.
  + With the help of NIC we communicate with other devices.
  + When we create or launch the instance then NIC will be automatic created.

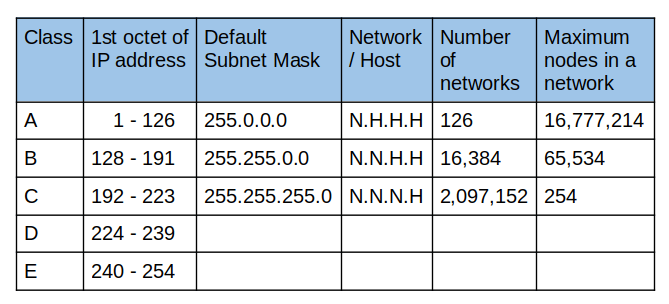
**EFS:Why we use EFS in AWS?**

We use EFS in AWS for scalable, shared file storage that is accessible by multiple EC2 instances simultaneously.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | | **Amazon EFS (Elastic File System)** | | **Amazon EBS (Elastic Block Store)** |
| * Type | | Network File System | | Block Storage |
| * Use Case | | Shared file storage | | Dedicated storage for single instances |
| * Accessibility | | Multiple EC2 instances | | One EC2 instance at a time |
| * Performance | | Scales with data stored | | Various types: General Purpose, IOPS |
| * Durability | | Multi-AZ redundancy | | Single AZ redundancy |
| * Protocol | | NFS | | Direct block storage |
| * Pricing | | Pay for storage used | | Pay for provisioned storage and IOPS |
| * Typical Uses | | Shared storage for web apps, CMS | | Databases, boot volumes, app storage |
| * Automatic Scaling | | Yes | | No |
| * Backup | | Managed automatically | | Manual snapshots to S3 |
|  | |  | |  |
| **PORT NAME** | **PORT NUMBER** | | **USE-CASE** | | |
| SSH | 22 | | SSH typically uses port 22 for secure connections. | | |
| SMTP | 25 | | SMTP typically uses port 25 for sending emails. | | |
| DNS | 53 | | DNS typically uses port 53 for domain name resolution. | | |
| HTTP | 80 | | HTTP typically uses port 80 for web traffic. | | |
| POP3 | 110 | | POP3 typically uses port 110 for retrieving emails. | | |
| IMAP | 143 | | IMAP typically uses port 143 for retrieving emails. | | |
| LDAP | 389 | | LDAP typically uses port 389 for directory services. AND manage directory information. | | |
| HTTPS | 443 | | HTTPS uses port 443 for secure web traffic. | | |
| SMB | 445 | | SMB uses port 445 for file sharing and network services. | | |
| SMTPS | 465 | | SMTPS uses port 465 for secure email sending. | | |
| IMAPS | 993 | | IMAPS uses port 993 for secure email retrieval. | | |
| POP3S | 995 | | POP3S uses port 995 for secure email retrieval. | | |
| MSSQL | 1433 | | MSSQL uses port 1433 for database connections. | | |
| RDS | 1521 | | Oracle RDS uses port 1521 for database connections. | | |
| NFS | 2049 | | NFS uses port 2049 for file sharing over a network. | | |
| MYSQL | 3306 | | MySQL and Aurora use port 3306 for database connections. | | |
| RDP | 3389 | | RDP uses port 3389 for remote desktop connections. | | |
| REDSHIFT | 5439 | | Redshift uses port 5439 for database connections. | | |
| POSTGRESQL | 5432 | | PostgreSQL uses port 5432 for database connections. | | |
| WINRM-HTTP | 5985 | | WinRM (Windows Remote Management) over HTTP uses port 5985. | | |
| WINRM-HTTPS | 5986 | | WinRM over HTTPS uses port 5986 for secure remote management. | | |

**What is ip address and how can be used:** The unique identifying number assigned to every device connected to the internet.

**The range of ip addresses are:**



**DIffrence between ip v4 and ip v6:wsa**

|  |  |  |
| --- | --- | --- |
| **Feature** | **IPv4** | **IPv6** |
| **Octate** | 4 octate | 16 Octate |
| **Address Length** | 32 bits | 128 bits |
| **Address Format** | 192.168.1.1 | 2001:0db8:85a3:0000:0000:8a2e:0370:7334 |
| **Header Complexity** | Simple and shorter | More complex but more efficient |
| **Security** | Less Security | More secure than ipv4 |
| **Configuration** | Manual (static) or automatic (DHCP) | Auto-configuration (stateless address auto-configuration) |
| **Compatibility** | Not compatible with IPv6 | Can coexist with IPv4 in a dual-stack setup |
| **Usage** | Widely used | Slowly being adopted as replacement |

**Difference between static and dynamic ip address?**

| **Feature** | **Static IP Address** | **Dynamic IP Address** |
| --- | --- | --- |
| **Address Stability** | Fixed, does not change | Changes periodically |
| **Assignment** | Manually configured | Automatically assigned by DHCP |
| **Reliability** | Highly reliable for consistent access | Less reliable for consistent access |
| **Common Uses** | Servers, websites, remote access | General home use, typical devices |
| **Ease of Setup** | More complex to set up | Simple, plug-and-play |
| **Cost** | May cost more, often requires a business plan | Typically included in basic service plans |

**Difference between private and public ip address?**

| **Feature** | **Private IP Address** | **Public IP Address** |
| --- | --- | --- |
| **Usage** | Within private networks | On the internet |
| **Scope** | Internal communication only | Global communication |
| **Address Range** | 10.0.0.0 to 10.255.255.255 | Unique and varies |
|  | 172.16.0.0 to 172.31.255.255 | Assigned by ISPs |
|  | 192.168.0.0 to 192.168.255.255 |  |
| **Reachability** | Not reachable from the internet | Reachable from anywhere on the internet |
| **Examples** | Home/office devices (e.g., 192.168.1.1) | Websites, online services (e.g., 203.0.113.1) |
| **Cost** | Free, managed locally | May incur cost, provided by ISP |

**What is CIDR:**  CIDR is a way to allocate and manage IP addresses more efficiently.

CIDR provide a range of ip address

CIDR block size can be between /16 to /28

**VPC(Virtual Private Cloud):**

**Why we use VPC in aws?**

We use a VPC in AWS to create a private, secure network for managing and isolating(separate) our cloud resources.

**Or**

Vpc is secure isolated environment to launch the resources and services.

* A default VPC is automatically created for each AWS account
* We can create 5 vpc in one region.
* A VPC (Virtual Private Cloud) is a private, isolated network in the cloud where you can launch and manage your resources like servers, databases, and applications

**Components of VPC**

**IGW:** Establish the connection between vpc and internet.

With the help of igw we can connect internet to the resources or services which is work in VPC.

An Internet Gateway is a VPC component that allows communication between instances in your VPC and the internet.

If your subnet is associated with a route to the Internet, then it is a public subnet.

You cannot have multiple Internet Gateways in a VPC. only attach 1 IGW to a VPC at a time.

**Subnet mask :** A subnet mask divides an IP address into network and host parts to manage IP addresses.

**Subnet:**  it convert large ip address into the small.

There are 3 types of subnet private, public,vpn only.

In vpc we can create 200 subnet.

Subnets are subdivisions of a VPC's IP address range within which you place your AWS resources.

**The first 4 and last 1 IP addresses in a subnet are reserved.**

1. 10.0.0.0: Network address.
2. 10.0.0.1: Reserved by AWS for the VPC route.
3. 10.0.0.2: Reserved by AWS.
4. 10.0.0.3: Reserved by AWS for future use.
5. 10.0.0.255: Network broadcast address (broadcast not supported).

**Route Table:**Route table determine how traffic should flow within your VPC.

The VPC router connects different AZs together and connects the VPC to the Internet Gateway.

Up to 200 route tables per VPC.

Up to 50 route entries per route table.

Can assign one route table to multiple subnets.

**NAT Gateway:** A NAT gateways used to connect public subnet to private subnet

Steps:- <https://www.datanextsolutions.com/blog/using-nat-gateways-in-aws/>

**NACL:** A security feature in your VPC that controls inbound and outbound traffic to and from your subnets.

With NACLs you can have permit and deny rules

Network ACLs have a list of rules that are checked one by one, starting from the lowest number, and stop checking if a deny rule is matched.

**VPC Peering:** VPC Peering is a way to connect two Virtual Private Clouds (VPCs) in AWS so they can communicate with each other as if they were on the same network. VPC Peering is a way to connect two Virtual Private Clouds (VPCs) in AWS so they can communicate with each other as if they were on the same network.

**VPC Endpoint**: A VPC endpoint is a way to connect your VPC directly to AWS services without using the internet.

There are two main types:

Steps:- <https://medium.com/@jadhav.swatissj99/vpc-endpoints-12ef54859e4f>

1. **Interface Endpoints**: Connects to AWS services using private IPs in your VPC, so traffic stays within the AWS network.
2. **Gateway Endpoints**: Specifically for S3 and DynamoDB, routes traffic directly to these services using a gateway, keeping it within the AWS network.

**Interface Endpoints vs. Gateway Endpoints**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Interface Endpoints** | **Gateway Endpoints** |
| **Purpose** | Connect to multiple AWS services privately | Connect to S3 and DynamoDB privately |
| **Technology** | AWS PrivateLink | VPC endpoints |
| **Flexibility** | Highly flexible, can connect to various services | Less flexible, limited to S3 and DynamoDB |
| **Complexity** | More complex to set up | Simpler to set up |
| **Cost** | Hourly charges + data transfer | No direct charges, only data transfer |

**Learn about Transitive Gatewoy:**

* + A Transit Gateway is a service provided by AWS (Amazon Web Services) that helps manage and connect multiple Virtual Private Clouds (VPCs) and on-premises(physical location or something located) networks.
  + It acts as a central hub that allows different networks (VPCs) to communicate with each other efficiently(well) and securely.

**Difference Between NACL and Security group:**

|  |  |
| --- | --- |
| **Security Group** | **NACL** |
| 1. Security group is the firewall of EC2 Instances. | Network ACL is the firewall of the VPC Subnets. |
| 1. Security Groups operate at Instance (Network Interface) level. | Network ACLs at the subnet level. Applies automatically to all instances deployed in the associated subnet. |
| 1. Security groups are stateful. | Network ACLs are stateless. |
| 1. Security group supports allow rules only (everything else is denied implicitly) | Network ACL supports allow and deny rules. |
| 1. Instance can have multiple Security groups. | Subnet can have only one NACL. |

**Public Subnet vs. Private Subnet**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Public Subnet** | **Private Subnet** |
| 1. **Internet Access** | Direct access to internet | No direct internet access |
| 1. **IP Address** | Public IP address | Private IP address |
| 1. **Security** | Less secure (exposed to internet) | More secure (isolated from internet) |
| 1. **Use Cases** | Web servers, load balancers | Databases, application servers |
| 1. **Additional Components** | Internet Gateway | NAT Gateway or VPN |

**Auto scalling :**

* AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady(fix), predictable performance at the lowest possible cost.
* dynamically adjust compute capacity based on demand."
* We increase the resources and decrease the resources on the demand
* **It is used for increase copasity.**
* Default copasity to launch instance.
* You create collections of EC2 instances, called Auto Scaling groups.
* Auto Scaling is a region-specific service.
* Auto Scaling works with ELB, CloudWatch and CloudTrail.
* Therw are two types of autoscaling.

**1.Vertical Scalling**: in vertical scalling we increase the size of server.

**2.Hrizontal Scalling**: In horizontal scalling we add or extend the new servers .

**OR**

**Auto Scaling:**

* Automatically adjusts the number of instances based on predefined metrics (CPU utilization, network traffic, etc.).
* Ensures sufficient resources are available to handle increased load.

**Benefits of Auto Scaling:**

* **Cost Optimization:** Pay only for the resources you need.
* **Improved Performance:** Handle traffic spikes without performance degradation.
* **High Availability:** Distribute your application across multiple instances.
* **Scalability:** Easily handle growing user base or seasonal fluctuations.

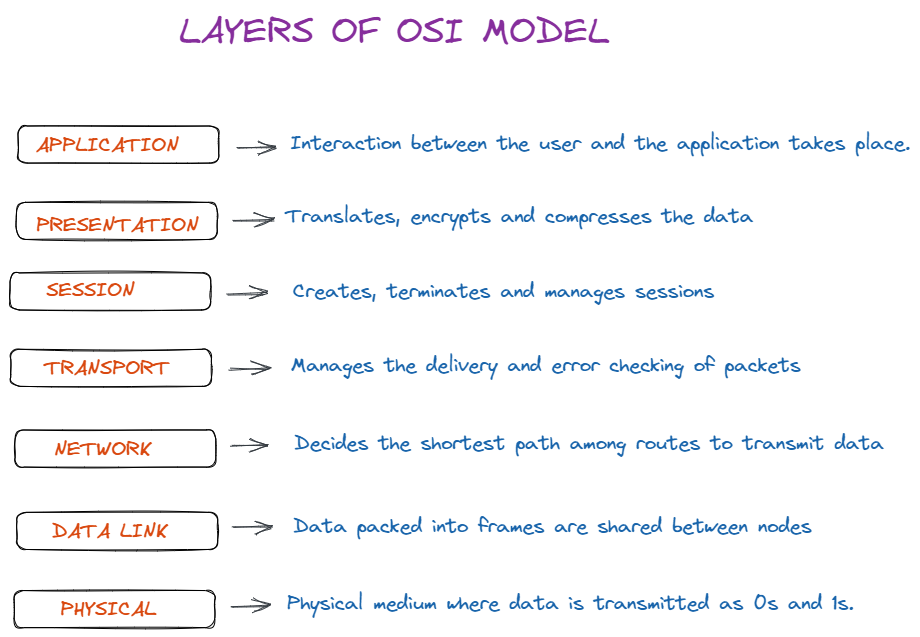
**Features of auto scalling:**

1. **Health Monitoring**: It checks if your instances are working well. If not, it replaces them automatically to keep things running smoothly.
2. **Custom Health Checks**: You can set your own health checks specific to your app. If an instance fails, it's replaced automatically.
3. **Availability Zones**: Distributes your instances across different locations to avoid issues if one location fails.
4. **Instance Variety**: Allows using different types of instances and purchase options (like Spot or On-Demand) to save costs.
5. **Spot Instance Replacement**: Automatically replaces Spot Instances if they are interrupted or at risk of interruption.
6. **Load Balancing**: Distributes traffic evenly across your instances and updates the load balancer when instances are added or removed.
7. **Scalability**: Adjusts the number of instances based on demand. You can also manually change the number if needed.
8. **Instance Refresh**: Updates instances gradually when you change your AMI or configuration. You can test updates on a small number of instances first.
9. **Lifecycle Hooks**: Allows you to perform custom actions when instances are launched or terminated.
10. **Stateful Workloads**: Supports maintaining state and protects important instances from being terminated too soon.

**What is OSI Model:**

The OSI model, created in 1984 by [ISO](https://www.geeksforgeeks.org/iso-9000-certification-in-software-engineering), is a reference framework that explains the process of transmitting data between computers. It is divided into seven layers that work together to carry out specialised network functions, allowing for a more systematic approach to networking.

OSI stands for Open Systems Interconnection, where open stands to say non-proprietary. It is a 7-layer architecture with each layer having specific functionality to perform. All these 7 layers work collaboratively to transmit the data from one person to another across the globe. The OSI reference model was developed by ISO – ‘International Organization for Standardization‘, in the year 1984.



**Data flows through the OSI model in a step-by-step process:**

* **Application Layer:**Applications create the data.
* **Presentation Layer:** Data is formatted and encrypted.
* **Session Layer:** Connections are established and managed.
* **Transport Layer:** Data is broken into segments for reliable delivery.
* **Network Layer**: Segments are packaged into packets and routed.
* **Data Link Layer:**Packets are framed and sent to the next device.
* **Physical Layer:**Frames are converted into bits and transmitted physically.

**Load Balancer:**

* A load balancer is a system that distributes incoming network traffic across multiple servers to ensure no single server becomes overwhelmed.
* This helps in improving the availability and reliability of applications by distributing the load, thus preventing server overload and ensuring better performance.
* Load balancers can operate at various layers of the OSI model, such as Layer 4 (transport layer) and Layer 7 (application layer).
* **Types of Load Balancer in aws:**

1.Application Load Balancer:7 th layer, http and https,

Target types Works with IP, instance, and lambda target types.

2.Network Load Balancer: 4 th layer, tcp and udp ,

Target types Works with IP, instance, and lambda target types.

3.Gateway Load Balancer: 3rd and 7 th layer,ip\based routing,

Works with IP and instance target types.

**Target groups:-**

* Target groups are a logical grouping of targets and are used with ALB, NLB, and GLB.
* Targets are the endpoints and can be EC2 instances, ECS containers, IP addresses, Lambda functions, and other load balancers.
* Target groups can exist independently from the ELB.
* A single target can be in multiple target groups.
* Only one protocol and one port can be defined per target group.
* You cannot use public IP addresses as targets.
* You cannot use instance IDs and IP address targets within the same target group.
* A target group can only be associated with one load balancer.

**S3:-**

* Amazon S3 is object storage built to store and retrieve any amount of data from anywhere on the Internet.
* It’s a simple storage service that offers an extremely durable, highly available, and infinitely scalable data storage infrastructure at very low costs.
* Amazon S3 is a distributed architecture and objects are redundantly stored on multiple devices across multiple facilities (AZs) in an Amazon S3 region.
* Amazon S3 is a simple key-based object store.
* Files can be from 0 bytes to 5TB.

**Buckets:-**

* Files are stored in buckets
* A bucket can be viewed as a container for objects.
* A bucket is a flat container of objects.
* It does not provide a hierarchy of objects.
* You can use an object key name (prefix) to mimic folders.
* 100 buckets per account by default.
* You can store unlimited objects in your buckets.
* You can create folders in your buckets (only available through the Console).
* You cannot create nested buckets.

**Storage Classes:**

There are six S3 storage classes.

* S3 Standard (durable, immediately available, frequently accessed).
* S3 Intelligent-Tiering (automatically moves data to the most cost-effective tier).
* S3 Standard-IA (durable, immediately available, infrequently accessed).
* S3 One Zone-IA (lower cost for infrequently accessed data with less resilience).
* S3 Glacier (archived data, retrieval times in minutes or hours).
* S3 Glacier Deep Archive (lowest cost storage class for long term retention).

**RDS(Relational database Service):**

* Amazon RDS is a fully managed service that makes it easy to deploy, manage, and scale relational databases in the cloud.
* RDS handles routine database tasks such as backups, patching, scaling, and replication
* Amazon RDS supports the following database engines:

1. Amazon Aurora.
2. MySQL.
3. MariaDB.
4. Oracle.
5. SQL Server.
6. PostgreSQL.

**Key Features of RDS:**

1. Automatic Backups:
2. High Availability:
3. Scaling:
4. Security:
5. Monitoring:

**Cloudfront in aws:**

* Amazon CloudFront is a fast content delivery network (CDN) service provided by AWS.
* It securely delivers data, videos, applications, and APIs to users globally with low latency and high transfer speeds.
* CloudFront caches content at edge locations around the world, bringing the content closer to users and improving the performance of web applications.

**Key Features of CloudFront:**

1. **Global Network of Edge Locations**:.
2. **Content Caching**:
3. **Dynamic Content Delivery**:
4. **Security Features**:
   * **SSL/TLS Encryption**: CloudFront supports HTTPS, ensuring secure communication between users and CloudFront.
   * **AWS Shield**: Integrated with AWS Shield Standard for DDoS protection at no additional cost.
   * **Access Control**: Supports access control features like signed URLs and signed cookies to restrict access to content.
   * **Origin Access Identity (OAI)**: Used to restrict access to S3 content, ensuring that only CloudFront can serve the content.
5. **Lambda@Edge**:
6. **Support for Multiple Origins**:
7. **Real-Time Monitoring**:
8. **Cost Efficiency**:

**Brief intro and usecases and console overview of these services**

**1.aws ACM**

**2.AWS shield**

**3.aws WAF 4.**

**4.aws kms**

**5.aws guard duty**

**1.AWS Certificate Manager (ACM)** is a service that simplifies the process of obtaining, managing, and deploying SSL/TLS certificates for your AWS-based websites and applications. It handles the complexities of creating, storing, and renewing these certificates, saving you time and effort

**SSL and TLS Certificates: A Brief Overview:**

* **SSL** (Secure Sockets Layer) and **TLS** (Transport Layer Security) are cryptographic protocols used to secure data transmission over the internet.
* While SSL is the older version, TLS is its successor and the preferred standard today.
* SSL/TLS certificates are the backbone of secure online communication, safeguarding user data and building trust between websites and their visitors.

**How They Work:** SSL/TLS certificates are digital documents that verify the identity of a website and enable encrypted communication between a website and a user's browser.

**Importance of SSL/TLS Certificates:**

SSL/TLS certificates are essential for online security for several reasons:

* **Protecting Sensitive Data:** Prevents data breaches by encrypting sensitive information.
* **Building Trust:** Users are more likely to trust websites with SSL/TLS certificates.
* **Improving SEO:** Search engines often give preference to websites with HTTPS (SSL/TLS enabled).
* **Compliance:** Many industries have regulations requiring SSL/TLS for data protection.

**2.What is the work of WAF:**

AWS WAF acts as a security shield for your web applications, helping you protect against a wide range of threats.

**AWS WAF: Web Application Firewall**

**AWS WAF** (Web Application Firewall) is a security service that helps protect your web applications from common web exploits and

bots that can affect availability, compromise security, or consume excessive resources.

**How does it work?**

* **Inspection:** AWS WAF inspects incoming HTTP(S) requests to your web applications.
* **Rule Evaluation:** It evaluates these requests against a set of rules defined by you.
* **Action:** Based on the rule evaluation, AWS WAF takes actions like allowing, blocking, or counting the requests.

**3.AWS Shield: A Shield Against DDoS Attacks:**

**AWS Shield** is a managed DDoS protection service that safeguards your applications running on AWS.

AWS Shield acts as a robust defense mechanism against DDoS attacks, safeguarding your applications and ensuring business continuity.

**How it works:**

* **Constant Monitoring:** AWS Shield continuously monitors your application traffic for signs of a DDoS attack.
* **Automatic Mitigation:** When a DDoS attack is detected, AWS Shield automatically implements mitigation techniques to protect your application.
* **Minimal Impact:** These mitigations are designed to minimize disruption to legitimate traffic.

**DDoS Attacks and AWS Protection:**

**DDoS** (Distributed Denial of Service) attacks are malicious attempts to disrupt the normal traffic of a targeted server, service, or network by overwhelming it with a flood of internet traffic.

**How DDoS Attacks Work:**

DDoS attacks typically involve a large number of compromised computers (botnets) flooding a target with traffic, making it difficult for legitimate users to access the service.

**4.aws kms:**

* 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.
* AWS Key Management Store (KMS) is a managed service that enables you to easily encrypt your data.
* AWS KMS provides a highly available key storage, management, and auditing solution for you to encrypt data within your own applications and control the encryption of stored data across AWS services.
* AWS KMS allows you to centrally manage and securely store your keys. These are known as AWS KMS keys (formerly known as customer master keys (CMKs).
* The KMS key includes metadata, such as the key ID, creation date, description, and key state.
* A KMS key can encrypt data up to 4KB in size.

**aws guard duty:**

Amazon GuardDuty is a threat detection service that monitors for malicious activity and anomalous behavior to protect AWS accounts, workloads, and data.

**Interview Quection:**

What is aws organization?

What is your day to day task and journey?

What are the services which you work?

Mutiregion deployment of rds?

**What is RDS?**

**What are databases?**

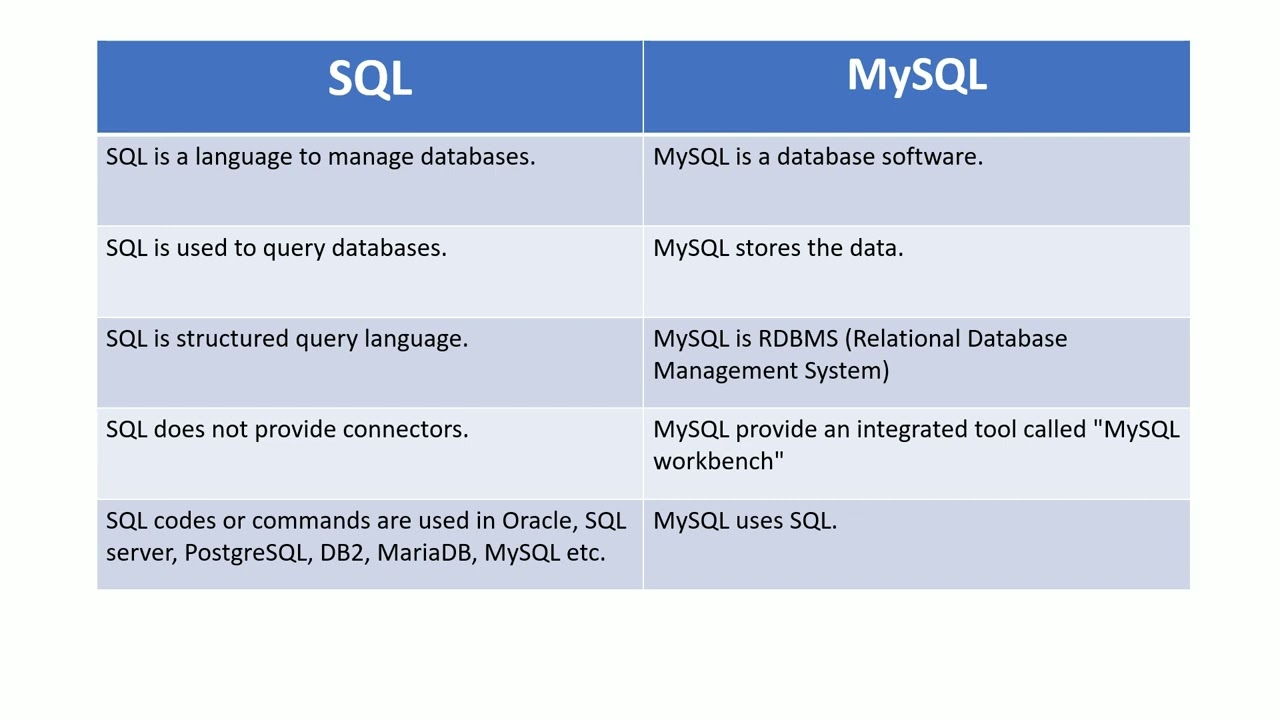
Every application needs a place to store data from users, devices, and the application itself. Databases are important backend systems that are used to store, manage, update, and analyze data for all types of applications, from small back-office systems to mobile and consumer web applications with global scale.

**How to access database from ec2 instance?**

**different aws services of databases:**

1. **Amazon Aurora**
2. **Amazon DynamoDB**
3. Amazon ElastiCache
4. Amazon Neptune
5. Amazon DocumentDB
6. **Amazon Redshift**
7. AWS Database Migration Service
8. **PostgreSQL**
9. Ledger databases
10. Document
11. Graph
12. **MySQL**
13. **Oracle Database**
14. **SQL Server**
15. In-memory
16. Key-value
17. MariaDB
18. NoSQL
19. Time-series
20. **Amazon RDS**
21. Amazon Timestream
22. Amazon Quantum ledger Database
23. Amazon Keyspaces
24. Amazon QLDB

**sql vs my sql vs nosql**



**what are the difference web servers and their working dierectories?**

**What are the different database server:-**

**Different Database Servers**

Database servers are software applications that manage and store data. They provide a structured way to organize, retrieve, and update information efficiently. Here are some of the most popular database servers:

**Relational Database Management Systems (RDBMS)**

* **MySQL:** A widely used, open-source RDBMS known for its performance, reliability, and ease of use.
* **PostgreSQL:** Another popular open-source RDBMS with advanced features like full-text search, JSON support, and spatial data types.
* **Oracle Database:** A commercial RDBMS from Oracle Corporation, known for its scalability, reliability, and security.
* **Microsoft SQL Server:** A commercial RDBMS from Microsoft, often used with Windows-based applications.
* **IBM DB2:** A commercial RDBMS from IBM, known for its performance and scalability.

**NoSQL Databases**

* **MongoDB:** A popular document-oriented NoSQL database, known for its flexibility and scalability.
* **Redis:** An in-memory data structure store, often used for caching, session management, and real-time analytics.
* **Cassandra:** A distributed, wide-column NoSQL database, known for its fault tolerance and scalability.
* **Couchbase:** A document-oriented NoSQL database that combines the flexibility of MongoDB with the performance of Redis.

**Other Database Types**

* **Graph Databases:** Designed to store and query data that is interconnected in a graph-like structure, often used for social networks and recommendation systems.
* **Time Series Databases:** Optimized for storing and querying time-stamped data, commonly used for IoT applications and financial data.
* **Key-Value Stores:** Simple databases that store data as key-value pairs, often used for caching and session management.

**Choosing the right database server** depends on various factors such as the type of data you need to store, the required performance, scalability, and your specific use case.

**Common Network Protocols and Their Ports**

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Description** | **Default Ports** |
| **HTTP (Hypertext Transfer Protocol)** | **Used for transferring web pages and other content** | **80** |
| **HTTPS (Hypertext Transfer Protocol Secure)** | **A secure version of HTTP that uses encryption** | **443** |
| **FTP (File Transfer Protocol)** | **Used for transferring files between computers** | **20 (data), 21 (control)** |
| **SMTP (Simple Mail Transfer Protocol)** | **Used for sending emails** | **25** |
| **POP3 (Post Office Protocol version 3)** | **Used for retrieving emails from a server** | **110** |
| **IMAP (Internet Message Access Protocol)** | **Used for accessing and managing emails on a server** | **143** |
| **SSH (Secure Shell)** | **Used for secure remote access to computers** | **22** |
| **Telnet (Terminal Network)** | **A simple network protocol for remote access to computers (insecure)** | **23** |
| **DNS (Domain Name System)** | **Resolves domain names into IP addresses** | **53** |
| **DHCP (Dynamic Host Configuration Protocol)** | **Assigns IP addresses to devices on a network** | **67 (server), 68 (client)** |
| **NTP (Network Time Protocol)** | **Synchronizes clocks on networked computers** | **123** |
| **SMB (Server Message Block)** | **Used for file sharing and printing on a network** | **139(TCP),445 (TCP/UDP)** |
| **FTP (File Transfer Protocol)** | **Used for transferring files between computers** | **20 (data), 21 (control)** |
| **SMTP (Simple Mail Transfer Protocol)** | **Used for sending emails** | **25** |
| **POP3 (Post Office Protocol version 3)** | **Used for retrieving emails from a server** | **110** |
| **IMAP (Internet Message Access Protocol)** | **Used for accessing and managing emails on a server** | **143** |
| **SSH (Secure Shell)** | **Used for secure remote access to computers** | **22** |
| **Telnet (Terminal Network)** | **A simple network protocol for remote access to computers (insecure)** | **23** |
| **DNS (Domain Name System)** | **Resolves domain names into IP addresses** | **53** |
| **DHCP (Dynamic Host Configuration Protocol)** | **Assigns IP addresses to devices on a network** | **67 (server), 68 (client)** |
| **NTP (Network Time Protocol)** | **Synchronizes clocks on networked computers** | **123** |
| **SMB (Server Message Block)** | **Used for file sharing and printing on a network** | **139 (TCP), 445 (TCP/UDP)** |

**Popular Web Servers:- Commercial Web Servers**

**Web Server:**

1.Apache:

We use the Apache server to host websites and web applications.

/var/www/html (default)

2.Nginx:

We use an Nginx server for high-performance web serving, reverse proxying, and load balancing.

/usr/share/nginx/html3.Microsoft IIS:

3.Microsoft IIS (Internet Information Services) server is used to host and manage web applications and websites on Windows servers.

C:\inetpub\wwwroot (default)

**Database Server:**

1.MySQL: We use a MySQL server to manage and store relational databases for applications

/var/lib/mysql (default)

2.PostgreSQL:PostgreSQL is used when you need a robust, open-source relational database system with advanced features, scalability, and strong compliance with SQL standards.

/var/lib/postgresql (default)

3.Oracle Database:\* /u01/app/oracle (common location)

4.Microsoft SQL Server:\* C:\Program Files\Microsoft SQL Server\MSSQL15.SQLSERVER\MSSQL\DATA (default)

5.MongoDB:\* /var/lib/mongodb (default)

6.Redis:\* /var/lib/redis (default)

**Application Server:**

1. Tomcat: We use Tomcat Server to deploy and manage Java-based web applications.

/usr/share/tomcat

1. JBoss: JBoss is an open-source Java EE application server used for deploying and running Java-based applications

/usr/share/jboss-as/

1. Node.js: Node.js is used for building scalable and efficient network applications, particularly those that handle real-time data and I/O operations.

C:\inetpub\wwwroot (default)

Object storage and block storage

Classes

**What is PM2 in AWS?**

PM2 is a daemon process manager that will help you manage and keep your application online. Getting started with PM2 is straightforward, it is installable via NPM.

**Node Version Manager (nvm):**

Node Version Manager (nvm) is a command-line tool that allows users to install, uninstall, switch, and manage multiple versions of Node.js on their system.

**What is SNS used for AWS?**

Amazon Simple Notification Service (Amazon SNS) is a web service that makes it easy to set up, operate, and send notifications from the cloud.

**Interview Quections of VPC**

**VPC:**

1. What is Amazon Virtual Private Cloud (Amazon VPC), and why is it important in AWS networking?

ANSWER:-

Amazon Virtual Private Cloud (Amazon VPC) is a service that lets you create your own private network within AWS. It’s important because it allows you to control how your AWS resources (like EC2 instances) are connected and accessed, ensuring they are secure and isolated from others. This is crucial for protecting your data and managing traffic between your resources.

1. What is the primary difference between a public subnet and a private subnet in a VPC?
2. How do you connect a VPC to an on-premises data center, and what are the options available for this connection?
3. Explain the purpose of Amazon VPC peering and its use cases.
4. What is the significance of route tables in a VPC, and how do you control traffic routing between subnets?
5. What are VPC Endpoints, and how do they enhance security and reduce data transfer costs for certain AWS services?
6. Explain the use of a Bastion Host (Jump Host) in a VPC for secure remote access to instances.
7. What is Direct Connect, and how does it provide dedicated network connectivity between an on-premises data center and an AWS VPC?
8. Describe the concept of VPC Flow Logs and their benefits for network monitoring and troubleshooting.
9. What is AWS Transit Gateway, and how does it simplify network connectivity and management in complex VPC architectures?
10. Explain the use of AWS PrivateLink for securely accessing AWS services over private connections within a VPC.
11. What are some best practices for designing VPC architectures that are highly available, fault-tolerant, and scalable?
12. Give examples of scenarios where you would use VPC peering, VPC endpoints, or Direct Connect to enhance network connectivity.
13. Discuss strategies for managing and optimizing VPC resources, including IP address allocation, subnet sizing, and route table design.
14. What are the considerations when setting up VPCs in a multi-region or global configuration for disaster recovery or load balancing?

**Interview Quections for iam?**

**Whole AWS**

**Cloud Services:**

* **IAM:** Managed who can access AWS resources and what they can do.
* **Storage:** Used S3 for storing files, EBS for storing data on virtual hard drives, and EFS for shared storage across multiple servers**.**
* **Compute:** Worked with EC2 to run virtual servers, set up load balancers to distribute traffic, and used Autoscaling to adjust the number of servers based on demand.

**Networking:**

* **VPC:** Created isolated networks in the cloud with public and private subnets.
* **NAT Gateway:** Allowed private servers to access the internet without being exposed.
* **CDN:** Used CloudFront to deliver content globally at high speed.
* **Route 53:** Managed DNS to direct traffic to the right AWS resources.

**Monitoring:**

* **CloudWatch:** Tracked and logged what’s happening with AWS resources, setting up alerts for any issues.
* **Container Insights:** Monitored and troubleshooted applications running in containers.

**Serverless Computing:**

* **AWS Lambda:** Built and deployed applications without needing to manage servers, which automatically scale as needed.

**Database Management:**

* **RDS:** Set up and managed scalable, secure relational databases.
* **MongoDB:** Managed NoSQL databases for handling large amounts of flexible, unstructured data.

**Netflix deployment architecture and platform in devops**