***NETWORKING AND DISTASTER MANAGEMENT***

A **computer network** is a system of interconnected devices that communicate with each other to share data, resources, and services. These devices can include computers, servers, routers, switches, and other networking hardware.

**Types of Computer Networks**

1. **LAN (Local Area Network)** – Covers a small area like a home, office, or campus.
2. **WAN (Wide Area Network)** – Covers large geographic areas, like the internet.
3. **MAN (Metropolitan Area Network)** – Connects multiple LANs within a city.
4. **PAN (Personal Area Network)** – Connects personal devices like smartphones and laptops.
5. **VPN (Virtual Private Network)** – Provides a secure connection over the internet.

**Key Components of a Network**

* **Nodes** (Devices like computers, servers)
* **Links** (Wired or wireless connections)
* **Protocols** (Rules for communication, like TCP/IP)
* **Switches & Routers** (Manage and direct network traffic)

**How does the Internet Work?**

The internet works by connecting millions of devices worldwide through a complex network of servers, routers, and communication protocols. Here's a simplified breakdown of how it functions:

**1. Data Transmission Using Packets**

* When you send a request (e.g., opening a website), the data is broken into small chunks called packets.
* Each packet contains source and destination addresses and follows the best available path to reach its destination.

**2. Role of IP Addresses and DNS**

* Every device on the internet has a unique IP address (e.g., 192.168.1.1 or 2001:db8::ff00:42:8329 for IPv6).
* Instead of remembering IPs, we use domain names (like google.com).
* The Domain Name System (DNS) translates domain names into IP addresses.

**3. Routing Through Networks**

* Packets travel through multiple routers and switches.
* Each router determines the best path for packets to take to their destination.

**4. Web Servers and HTTP/HTTPS**

* A web server (e.g., Google’s server) processes the request and sends back a response.
* Websites use HTTP/HTTPS protocols to ensure structured and secure communication.

**5. Internet Service Providers (ISP)**

* Your device connects to the internet via an ISP (like Airtel, Jio, or Comcast).
* ISPs maintain connections to backbone networks that link different regions.

**6. Security and Encryption**

* Data is encrypted (especially with HTTPS) to prevent interception.
* Firewalls, VPNs, and other security measures help protect data.

**OSI (Open Systems Interconnection) Model**

The OSI (Open Systems Interconnection) Model is a conceptual framework used to understand network communications in seven layers. It helps standardize networking protocols and troubleshooting.

**7 Layers of OSI Model:**

1. **Physical Layer** :Handles raw data transmission (bits) through cables, radio waves, or optical fibers.
   * Example: Ethernet cables, Wi-Fi signals, fiber optics.
2. **Data Link Layer** – Manages error detection and correction, and organizes data into frames for physical transmission.
   * Example: MAC addresses, Ethernet, switches.
3. **Network Layer** – Handles routing, addressing, and forwarding data packets across networks.
   * Example: IP addresses, routers.
4. **Transport Layer –** Ensures reliable or fast data delivery using protocols like TCP (reliable) and UDP (fast, connectionless).
   * Example: TCP, UDP.
5. **Session Layer –** Manages sessions and maintains connections between applications.
   * Example: Session establishment in APIs, remote login.
6. **Presentation Layer –** Translates, encrypts, or compresses data for the application layer.
   * Example: SSL/TLS encryption, data compression.
7. **Application Layer** – Provides user interaction with the network through applications.
   * Example: HTTP, FTP, SMTP, DNS.

The **TCP/IP model** (Transmission Control Protocol/Internet Protocol) is a conceptual framework used for network communications. It consists of **four layers**, each handling specific network functions.

**Layers of the TCP/IP Model:**

1. **Application Layer**
   * Handles **high-level protocols** like HTTP, FTP, SMTP, DNS, etc.
   * Provides an interface for applications to communicate over a network.
2. **Transport Layer**
   * Manages **end-to-end communication** and error handling.
   * Uses **TCP (reliable, connection-oriented)** and **UDP (fast, connectionless)** protocols.
3. **Internet Layer**
   * Responsible for **routing and addressing** packets across networks.
   * Uses **IP (Internet Protocol)** for addressing and **ICMP (Internet Control Message Protocol)** for error handling.
4. **Network Access Layer (Link Layer)**
   * Deals with **hardware-level communication** (Ethernet, Wi-Fi, etc.).
   * Includes **MAC addressing, framing, and physical transmission of data**.

**IP (Internet Protocol):** An IP (Internet Protocol) address is a unique identifier assigned to each device on a network. It allows devices to communicate with each other over the internet or a local network. An IP (Internet Protocol) address is a unique number assigned to each device on a network, allowing them to communicate with each other. It's like a device's "address" on the internet or local network.

Types of IP Addresses

1. **IPv4 (Internet Protocol Version 4)**
   * Format: 32-bit (e.g., 192.168.1.1)
   * Range: 0.0.0.0 to 255.255.255.255
   * Divided into network and host parts using a subnet mask.
2. **IPv6 (Internet Protocol Version 6)**
   * Format: 128-bit (e.g., 2001:db8::ff00:42:8329)
   * Supports a larger address space (3.4×10³⁸ addresses).
   * Uses hexadecimal notation.

IPv4 (Internet Protocol version 4) and IPv6 (Internet Protocol version 6) are two versions of the Internet Protocol used for identifying devices on a network.

**Private vs. Public IPs**

**1. Private IP Addresses**

* Used within a **local network (LAN)** (e.g., home, office, or enterprise networks).
* **Not routable on the internet** (cannot be used for direct internet access).
* Assigned by a **router or DHCP server** within the network.
* Requires **NAT (Network Address Translation)** to communicate with the internet.

**2. Public IP Addresses**

* Used for **internet communication** (assigned by ISPs).
* **Routable on the internet** (can be accessed globally).
* Each device with a public IP is **directly reachable** unless behind a firewall.

**3. Static IP**: 172.16.x.x Manually assigned, doesnʼt change. 192.168.x.x - 172.31.x.x . Often used for servers and devices that need a consistent address.

**4. Dynamic IP:** Automatically assigned by a DHCP (Dynamic Host Configuration Protocol) server. Changes periodically; commonly used for home devices.

A **subnet (subnetwork)** is a logically segmented portion of a larger network. Subnetting divides an IP network into smaller, more manageable parts, improving performance and security.

**Subnetting: Divides a network into smaller, more manageable segments.**

**Subnet Components**

1. **Network Address** – Identifies the subnet (e.g., 192.168.1.0/24).
2. **Subnet Mask** – Defines the size of the subnet (e.g., 255.255.255.0).
3. **Broadcast Address** – The last address in the subnet used for communication to all devices (192.168.1.255 for /24).
4. **Usable Host Range** – The range of valid IPs for devices (e.g., 192.168.1.1 - 192.168.1.254).

**Firewall**

* A **security device/software** that monitors and controls **incoming and outgoing traffic**.
* Protects networks from **unauthorized access, malware, and cyberattacks**.

**What is Loopback?**

Loopback address allows a device to communicate with itself. Itʼs often used for testing network software on the local machine.

**Key Points:**

1. 127.0.0.1 is commonly known as "localhost."
2. Any IP address in the 127.x.x.x range will loop back to the same device.

### **CIDR (Classless Inter-Domain Routing)**

* Used for **IP address allocation** and routing.
* Defines **IP ranges** using a combination of an IP address and a subnet mask (e.g., 192.168.1.0/24).
* Helps optimize **IP address usage** and reduce waste.
* Commonly used by **ISPs, cloud providers, and network admins**.
* Example: 10.0.0.0/8 covers all IPs from 10.0.0.0 to 10.255.255.255.

### **DNS (Domain Name System)**

* Translates **domain names** into **IP addresses**.
* Enables users to access websites using **human-readable names** instead of numbers.
* Uses **records** like A (IPv4), AAAA (IPv6), CNAME, MX (Mail Exchange), etc.
* Hierarchical structure with **root, top-level domains (TLDs), and subdomains**.
* Example: www.google.com resolves to 142.250.180.14.

**LDAP (Lightweight Directory Access Protocol):** LDAP (Lightweight Directory Access Protocol) is a protocol used to access and manage directory services, often for authentication and authorization. It is commonly used in enterprise environments to store user credentials, groups, and policies.

### **Key Features of LDAP:**

* Centralized Authentication: Manages user credentials in a single location.
* Hierarchical Structure: Uses a tree-like structure to organize users, groups, and resources.
* Lightweight Protocol: Optimized for fast read operations.
* Integration: Works with Active Directory, OpenLDAP, and various enterprise applications.

### **Common LDAP Operations:**

1. Bind: Authenticate a user.
2. Search: Query the directory for users or objects.
3. Add: Create new entries.
4. Modify: Update existing entries.
5. Delete: Remove entries.

### **Use Cases:**

* Single Sign-On (SSO)
* User and Group Management
* Access Control for Applications

**DHCP (Dynamic Host Configuration Protocol):** DHCP (Dynamic Host Configuration Protocol) is a network protocol used to automatically assign IP addresses and other network configuration parameters (such as subnet mask, gateway, and DNS) to devices in a network.

### How DHCP Works:

1. Discovery: A client device sends a DHCP request to find a DHCP server.
2. Offer: The DHCP server responds with an available IP address and configuration details.
3. Request: The client accepts the offer by requesting the IP assignment.
4. Acknowledgment: The DHCP server confirms the lease, and the client starts using the assigned IP.

### Key Features:

* Automatic IP Assignment: Reduces manual configuration errors.
* IP Lease Time: Ensures dynamic allocation and reuse of IPs.
* Subnet Mask, Gateway, and DNS Configuration: Simplifies network management.

### Use Cases:

* Enterprise and home networks
* Large-scale cloud and data center environments
* Wireless networks

### **1. Switching (Layer 2 - Data Link Layer)**

Switching occurs within a **Local Area Network (LAN)** and is responsible for forwarding data frames based on **MAC addresses**.

### **2. Routing (Layer 3 - Network Layer)**

Routing enables communication **between different networks** by forwarding packets based on **IP addresses**.

#### **Types of Routing:**

1. **Static Routing**
   * Manually configured routes.
   * Used in small networks.
2. **Dynamic Routing**
   * Routes are learned and updated automatically.
   * Used in larger networks.
   * Protocols: **RIP, OSPF, EIGRP, BGP**
3. **Default Routing**
   * Used when no specific route matches.
   * Common in stub networks.

**Disaster Recovery on AWS**

Disaster Recovery (DR) on AWS ensures business continuity by providing strategies and services to recover applications and data in case of outages, cyberattacks, natural disasters, or human errors. AWS offers multiple disaster recovery solutions based on Recovery Time Objective (RTO) and Recovery Point Objective (RPO).

### ***RPO (Recovery Point Objective) & RTO (Recovery Time Objective) in AWS Disaster Recovery***

#### **1. RPO (Recovery Point Objective)**

* Definition: The maximum acceptable amount of data loss measured in time.
* Question it answers: “How much data can we afford to lose?”
* Example:
  + If the RPO is 5 minutes, backups or replication must occur at least every 5 minutes.
  + If a failure happens at 12:10 PM and the last backup was at 12:05 PM, data loss from 12:05 PM to 12:10 PM is acceptable.

#### **2. RTO (Recovery Time Objective)**

* Definition: The maximum acceptable downtime for restoring services after a disaster.
* Question it answers: “How quickly must we recover?”
* Example:
  + If the RTO is 30 minutes, the system should be back online within 30 minutes of a failure.
  + If a system fails at 12:10 PM, full recovery should be completed by 12:40 PM.

## 

## ***Disaster Recovery Strategies on AWS***

AWS offers four primary disaster recovery (DR) strategies, each with different levels of **cost, complexity, and recovery speed**. The choice depends on **Recovery Time Objective (RTO)** and **Recovery Point Objective (RPO)** requirements.

### **1. Backup & Restore (Low Cost, High RTO & RPO)**

* **Description:** Data is periodically backed up and restored when needed.
* **RTO:** Hours
* **RPO:** Hours
* **Use Case:** Cost-effective for non-critical workloads (e.g., archives, infrequent access).
* **AWS Services Used:**
  + **AWS Backup** – Centralized backup management.
  + **Amazon S3 & Glacier** – Long-term, low-cost storage.
  + **Amazon RDS Snapshots** – Database backups.
  + **AWS Lambda & S3 Lifecycle Rules** – Automate backups and retention policies.

🔹 **Example:** A company stores daily backups of customer transactions on Amazon S3 Glacier. In case of failure, data is restored within a few hours.

### **2. Pilot Light (Minimal Resources, Faster Recovery)**

* **Description:** A minimal version of the application runs in a secondary AWS region. Full capacity is activated when needed.
* **RTO:** Minutes to Hours
* **RPO:** Minutes
* **Use Case:** Suitable for applications that need faster recovery but don't require full-time resources.
* **AWS Services Used:**
  + **Amazon EC2 (Minimal Instances)** – Core infrastructure running at low capacity.
  + **Amazon RDS Read Replicas** – Ensures database readiness.
  + **S3 Cross-Region Replication (CRR)** – Continuous backup of critical data.
  + **AWS CloudFormation** – Automates full environment scaling.

🔹 **Example:** An e-commerce platform keeps a small version of its web app running in a standby AWS region. If the primary region fails, instances scale up and Route 53 directs traffic to the standby region.

### **3. Warm Standby (Higher Availability, Lower Downtime)**

* **Description:** A scaled-down but fully functional version of the system runs in a secondary AWS region.
* **RTO:** Minutes
* **RPO:** Seconds to Minutes
* **Use Case:** For business-critical applications requiring quick failover.
* **AWS Services Used:**
  + **Amazon EC2 (Running at Reduced Capacity)** – Minimal servers, ready to scale.
  + **Amazon RDS Multi-AZ or Read Replicas** – High availability databases.
  + **AWS Auto Scaling** – Automatically increases capacity when needed.
  + **Amazon Route 53 Failover Routing** – Redirects traffic when failure occurs.
  + **AWS Elastic Disaster Recovery (DRS)** – Continuous data replication.

🔹 **Example:** A media streaming service has a warm standby setup where a smaller instance of its API server runs in a different region. If the primary region fails, Auto Scaling scales up the standby environment within minutes.

### **4. Multi-Site (Active-Active) – Highest Availability**

* **Description:** Fully operational systems run in multiple AWS regions, handling traffic simultaneously.
* **RTO:** Near Zero
* **RPO:** Near Zero
* **Use Case:** Mission-critical applications requiring zero downtime (e.g., financial transactions, online gaming).
* **AWS Services Used:**
  + **Amazon Route 53 Latency-Based or Geolocation Routing** – Distributes traffic across multiple active sites.
  + **Amazon Aurora Global Database** – Low-latency, globally distributed database.
  + **Amazon DynamoDB Global Tables** – Real-time data synchronization across regions.
  + **Amazon EC2 Auto Scaling** – Ensures sufficient capacity at all times.
  + **AWS Global Accelerator** – Optimizes traffic routing between regions.

🔹 **Example:** A stock trading platform operates active servers in **US-East** and **EU-West** regions. Traffic is dynamically routed based on user location, ensuring zero downtime.

**AWS Services for Disaster Recovery**

AWS provides several services to build a robust DR plan:

#### **a. Backup & Data Protection**

* **AWS Backup** – Centralized backup service for EBS, RDS, DynamoDB, S3, and FSx.
* **Amazon S3 Versioning & Replication** – Stores multiple versions of objects and replicates them across regions.
* **EBS Snapshots** – Used for backing up EC2 instance volumes.

#### **b. Replication & Failover**

* **AWS Elastic Disaster Recovery (DRS)** – Continuous replication of workloads for quick recovery.
* **AWS Database Migration Service (DMS)** – Replicates and migrates databases with minimal downtime.
* **Amazon RDS Multi-AZ & Read Replicas** – Provides automated failover for databases.

#### **c. High Availability & Redundancy**

* **AWS Auto Scaling** – Automatically scales applications in response to demand.
* **Elastic Load Balancer (ELB)** – Distributes traffic across multiple instances for high availability.
* **Route 53 DNS Failover** – Redirects traffic to healthy endpoints in case of failure.

#### **d. Infrastructure as Code (IaC)**

* **AWS CloudFormation & Terraform** – Automates infrastructure provisioning and recovery.
* **AWS Systems Manager** – Centralized management and automation of resources.

#### **e. Monitoring & Alerts**

* **Amazon CloudWatch** – Monitors logs, metrics, and sets alarms for system health.
* **AWS Trusted Advisor** – Recommends best practices for security, cost, and resilience.

Here’s a high-level AWS Disaster Recovery (DR) architecture diagram illustrating a **Warm Standby** setup with key AWS services:

🟢 **Primary AWS Region (Active)**

* EC2 Instances (Minimal running capacity)
* RDS Multi-AZ Deployment
* S3 with Cross-Region Replication
* Auto Scaling Group (Partial capacity)
* Route 53 (Primary)

🔴 **Secondary AWS Region (Standby)**

* EC2 Instances (Stopped or Minimal)
* RDS Read Replica
* S3 (Replicated data)
* Auto Scaling (Triggers full capacity when needed)
* Route 53 (Failover routing to secondary region)

✅ **DR Process Flow**:

1. Data replicated continuously via **S3 Replication, AWS Backup, and RDS Read Replicas**.
2. AWS Route 53 detects failure and redirects traffic to the standby region.
3. Auto Scaling triggers additional instances to scale up the standby environment.
4. AWS Elastic Disaster Recovery (DRS) facilitates rapid recovery.

**Amazon Aurora :** Amazon Aurora is a managed relational database service designed for high performance, scalability, and availability. It is compatible with MySQL and PostgreSQL, **making it easier to migrate existing applications.**

### **Key Features:**

✅ High Performance – Up to 5x faster than MySQL and 3x faster than PostgreSQL  
 ✅ Scalability – Automatically scales storage up to 128 TiB  
 ✅ High Availability – Replicates data across multiple Availability Zones (AZs)  
 ✅ Serverless Option – Aurora Serverless scales automatically based on demand  
 ✅ Security – Supports encryption at rest and in transit using AWS KMS  
 ✅ Automatic Backups – Continuous backups to Amazon S3 with point-in-time recovery

### **Use Cases**

📌 SaaS applications requiring high availability  
📌 E-commerce platforms handling large transactions  
📌 Gaming applications needing low-latency databases **📌** Enterprise applications with global distribution

### **AWS Database Migration Service (DMS)**

AWS **DMS** helps migrate databases to AWS with minimal downtime. It supports **homogeneous** (e.g., MySQL → MySQL) and **heterogeneous** (e.g., Oracle → PostgreSQL) migrations.

### **Key Features**

✅ **Minimal Downtime** – Continuous data replication with Change Data Capture (**CDC**)  
 ✅ **Supports Multiple Databases** – MySQL, PostgreSQL, Oracle, SQL Server, MariaDB, MongoDB, etc.  
 ✅ **Fully Managed** – AWS handles **provisioning, monitoring, and scaling** ✅ **Schema Conversion** – Works with **AWS Schema Conversion Tool (SCT)** for heterogeneous migrations  
 ✅ **Secure & Reliable** – Supports **SSL encryption**, automatic failover, and IAM roles

### **Common Use Cases**

📌 **Database Migration** – Move from on-premise or another cloud provider to AWS  
 📌 **Database Replication** – Keep databases in sync for disaster recovery  
 📌 **Data Consolidation** – Merge multiple databases into a data warehouse (e.g., Amazon Redshift)

### ***Homogeneous vs. Heterogeneous Database Migration***

#### **1️. Homogeneous Migration**

✅ **Source & target databases are the same engine** (e.g., MySQL → MySQL, PostgreSQL → PostgreSQL)  
✅ **Schema structure, data types, and features remain the same**✅ **Easier & faster migration** because no schema conversion is needed  
✅ **Tools:** AWS **DMS** (Database Migration Service)

🔹 **Example:** Migrating an on-premises **MySQL** database to **Amazon RDS for MySQL**

#### **2️. Heterogeneous Migration**

✅ **Source & target databases are different engines** (e.g., Oracle → PostgreSQL, SQL Server → MySQL)  
✅ **Schema conversion is required** (data types, functions, queries may differ)  
✅ **More complex & time-consuming**✅ **Tools:** AWS **Schema Conversion Tool (SCT)** + **DMS**

🔹 **Example:** Migrating from **Oracle** to **Amazon Aurora PostgreSQL**

**AWS DMS Supported Source & Target Databases:**

### **✅ Supported Source Databases**

These databases can be migrated to AWS:

🔹 Relational Databases

* Amazon RDS (MySQL, PostgreSQL, MariaDB, Oracle, SQL Server)
* Amazon Aurora (MySQL, PostgreSQL)
* On-Premises (MySQL, PostgreSQL, Oracle, SQL Server, MariaDB, SAP ASE)
* Other Cloud Providers (Self-hosted DBs on Azure, GCP, etc.)

🔹 NoSQL & Document Databases

* MongoDB (self-hosted & Atlas)
* Amazon DocumentDB

🔹 Data Warehouses

* Amazon Redshift
* Oracle Data Warehouse
* Teradata
* Microsoft Azure SQL Database

### **🎯 Supported Target Databases**

These databases can be the destination for migration:

🔹 Relational Databases

* Amazon RDS (MySQL, PostgreSQL, MariaDB, Oracle, SQL Server)
* Amazon Aurora (MySQL, PostgreSQL)
* On-Premises Databases (MySQL, PostgreSQL, Oracle, SQL Server)

🔹 NoSQL & Document Databases

* Amazon DynamoDB
* Amazon DocumentDB

🔹 Data Warehouses

* Amazon Redshift
* Amazon S3 (for data lake architectures)

**AWS Schema Conversion Tool (SCT)**

AWS **Schema Conversion Tool (SCT)** helps convert **database schemas, stored procedures, and functions** when migrating databases from one engine to another (heterogeneous migration).

### **🔹 Key Features**

✅ **Schema & Code Conversion** – Converts **database objects** (tables, indexes, views, stored procedures)  
✅ **Heterogeneous Database Migration** – Supports **Oracle → PostgreSQL**, **SQL Server → MySQL**, etc.  
✅ **Assessment Reports** – Identifies **incompatible objects** & suggests fixes  
✅ **Supports Data Warehouses** – Converts **Oracle, Teradata** schemas to **Amazon Redshift**✅ **Works with AWS DMS** – AWS SCT **converts schemas**, and **DMS moves the data**✅ **Supports On-Prem & Cloud** – Works with **on-prem, AWS RDS, Aurora, Redshift, and DynamoDB**

### **🔄 Supported Migrations**

🔹 **Relational Databases**

* **Or**acle → PostgreSQL, MySQL, Aurora
* SQL Server → MySQL, PostgreSQL, Aurora
* DB2 → PostgreSQL, MySQL

### **🚀 How AWS SCT Works?**

1️. **Connect** to source & target databases  
2️. **Analyse** schema & generate an assessment report  
3️. **Convert** schema and stored procedures  
4️. **Apply** converted schema to the target database  
5️. **Use AWS DMS** to migrate data

### **🛠️ When to Use AWS SCT?**

✅ When migrating **heterogeneous databases** (Oracle → PostgreSQL)  
 ✅ When moving from **on-prem** to **AWS RDS/Aurora/Redshift** ✅ When converting **data warehouse schemas** to Redshift  
 ✅ When assessing **schema incompatibilities before migration**

### **AWS Backup Vault Lock**

AWS Backup Vault Lock is a security feature in AWS Backup that prevents deletion or modification of backups stored in a backup vault for a defined retention period. It helps with data protection, compliance, and ransomware defense.

### **🔹 Key Features**

✅ Immutability – Prevents deletion, modification, or expiration changes  
✅ WORM (Write Once, Read Many) – Ensures backups cannot be altered  
✅ Two Lock Modes

* Governance Mode – Allows authorized IAM roles to modify lock settings
* Compliance Mode – No one (even root user) can delete or modify the lock settings  
   ✅ Supports AWS Organizations – Enforces backup policies across accounts  
   ✅ Works with Multiple AWS Services – Protects backups for EC2, RDS, DynamoDB, EFS, FSx, etc.

### **🚀 How to Enable Backup Vault Lock?**

1️. Go to AWS Backup → Backup Vaults  
2️. Select a backup vault  
3️. Click Edit Vault Lock  
4️. Choose Governance Mode or Compliance Mode  
5️. Set Retention Period (min & max retention duration)  
6️. Click Save – Lock is now active!

### **📌 When to Use AWS Backup Vault Lock?**

✔️ Ransomware Protection – Prevents accidental or malicious deletion  
✔️ Regulatory Compliance – Helps meet financial & healthcare data retention rules  
✔️ Disaster Recovery – Ensures backups remain secure for critical workloads

🔹 ⚠️ Important: In Compliance Mode, once locked, even AWS support cannot override or delete backups before the retention period expires.

### **WORM (Write Once, Read Many) in AWS:**

**WORM (Write Once, Read Many)** is a **data protection model** where data is written **once** and cannot be **modified or deleted** for a specified retention period. This ensures **data integrity, security, and compliance**.

### **🔹 WORM Modes in AWS**

1️. **Governance Mode** (Flexible)

* Allows **certain IAM users/roles** to override retention settings
* Used when **some administrative control is needed**

2️. **Compliance Mode** (Strict)

* **No one (even root user) can delete or modify** data before the retention period expires
* Used for **financial, healthcare, and regulatory compliance**