**AWS Cloud Practitioner**

# Introduction to the Cloud

## Client–Server Model (Fundamental Concept)

The **Client–Server model** is a way of **designing computer systems** where:

* One side (the **client**) makes a **request**, and
* The other side (the **server**) **responds** with the required data or service.

**In short:**  
 *Client asks, Server serves.*

### How It Works (Basic Flow):

**Client** → Sends a request (e.g., “Give me this file,” “Show this web page”)  
 **Server** → Processes the request and sends back a response

### Simple Analogy:

Imagine a **restaurant** 🍽️

* **Client:** Customer who places an order
* **Server:** Kitchen that cooks and serves the meal
* **Network:** Waiter who carries the order and delivers food

Customer asks → Waiter delivers → Kitchen cooks → Waiter brings back → Customer eats

That’s the **client–server model** in real life!

## Cloud Computing

**Cloud Computing** is the **on-demand delivery of IT resources** (like servers, storage, databases, and networking) **over the Internet**, with **pay-as-you-go pricing**.

**In short:**  
 *You rent computing resources instead of owning them.*

**Example:**

Instead of buying a physical computer or server:  
You use **AWS EC2** to get a virtual server on the cloud.  
You only pay for what you use — just like electricity or water

### Key Characteristics of Cloud Computing

| **Feature** | **Description** |
| --- | --- |
| **On-demand self-service** | You can create resources anytime, without needing human approval. |
| **Broad network access** | Access resources from anywhere via the internet. |
| **Resource pooling** | AWS shares resources across multiple customers securely. |
| **Rapid elasticity** | Easily scale resources up or down as needed. |
| **Measured service** | Pay only for what you use (pay-as-you-go). |

### Benefits of Cloud Computing

| **Benefit** | **Explanation** |
| --- | --- |
| **Cost-effective** | No upfront hardware costs. Pay only for usage. |
| **Scalable** | Quickly add or remove resources based on demand. |
| **Reliable** | AWS has multiple data centers (high availability). |
| **Global** | Access from anywhere in the world. |
| **Secure** | AWS offers strong security and compliance tools. |

## Cloud Deployment Models (Types)

Cloud deployment types describe **how the cloud is set up** and **who controls it**.

There are **3 main deployment types** you must remember for the exam

### 1. Public Cloud

Cloud resources are **owned and operated by a third-party provider** (like AWS, Azure, or Google Cloud) and shared by multiple customers.

**Example:**

Using **AWS EC2** or **S3** — you share AWS infrastructure with others securely.

**Features:**

* Owned by cloud provider (e.g., AWS)
* Shared across multiple customers (multi-tenant)
* Accessible over the Internet
* Pay-as-you-go pricing

**Example Use Case:**

Startups or businesses that don’t want to maintain their own servers.

**Exam Tip:** “AWS is an example of a **Public Cloud**.”

### 2. Private Cloud

Cloud infrastructure is used **by a single organization only** — not shared with others.  
It can be **on-premises** or hosted by a third party.

**Features:**

* Dedicated resources (single tenant)
* Greater control and customization
* Enhanced security and compliance
* Higher cost than public cloud

**Example:**

A bank or government agency running its own private cloud using **VMware** or **AWS Outposts**.

**Exam Tip:** “If data security or compliance is critical → choose **Private Cloud**.”

### 3. Hybrid Cloud

A **combination of Public and Private Clouds** that work together.  
You can move data or apps between them as needed.

**Features:**

* Best of both worlds: scalability + control
* Data or applications can move between environments
* Used for gradual migration to the cloud

**Example:**

A company keeps sensitive data in **Private Cloud** but uses **AWS** for analytics or backups.

**Exam Tip:** “Hybrid Cloud = Connects on-prem data center with AWS.”

### Multi-Cloud (Optional for deeper understanding)

Using **multiple public cloud providers** (e.g., AWS + Azure + Google Cloud) at once.

**Why Use:**

* Avoid vendor lock-in
* Use best features of each provider
* Add redundancy or backup

*(Note: Multi-cloud is not an official AWS deployment model, but good to know.)*

### Quick Summary Table

| **Deployment Type** | **Who Owns It** | **Shared?** | **Example** | **Use Case** |
| --- | --- | --- | --- | --- |
| **Public Cloud** | Cloud provider (e.g., AWS) | Yes | AWS EC2, S3 | Most businesses |
| **Private Cloud** | Single organization | No | VMware Cloud, AWS Outposts | Banks, Govt |
| **Hybrid Cloud** | Mix of both | Partial | On-prem + AWS | Cloud migration |
| **Multi-Cloud** | Multiple providers | Yes | AWS + Azure | Redundancy, flexibility |

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Cloud owned and managed by third-party provider” | **Public Cloud** |
| “Used exclusively by one organization” | **Private Cloud** |
| “Combination of public and private cloud” | **Hybrid Cloud** |
| “On-demand delivery of IT resources over the internet” | **Cloud Computing** |

## Benefits of AWS Cloud

AWS Cloud offers six key benefits that help businesses **save costs, improve speed, stay secure, and scale easily** without managing physical hardware.

### 1. Pay-as-you-go Pricing

**Meaning:**

You **only pay for what you use** — no upfront cost or long-term contract.

**Example:**

If you use an EC2 instance for 3 hours, you only pay for 3 hours — not for the full day or month.

**Why It Matters:**

* Reduces waste
* No need to guess future usage
* Ideal for startups or variable workloads

**Exam Tip:** “AWS pricing model is pay-as-you-go, just like electricity or mobile data.”

### 2. Scalability and Elasticity

**Meaning:**

You can **quickly add (scale up)** or **remove (scale down)** resources based on demand.

**Example:**

* During a sale, Amazon.com automatically adds more servers.
* After the sale ends, it reduces them to save cost.

**Why It Matters:**

* Always have the right capacity
* Saves money and ensures good performance

**Elasticity = Auto-adjusting resources automatically.**

### 3. Flexibility and Agility

**Meaning:**

AWS supports **many technologies, programming languages, operating systems**, and databases — so you can use what you prefer.

**Example:**

You can deploy apps using Windows, Linux, Python, Java, or Node.js — all on AWS.

**Why It Matters:**

* Easy to experiment, build, and deploy fast
* Reduces time to market for new ideas

**Agility = Move fast and innovate easily.**

### 4. Reliability

**Meaning:**

AWS provides **high availability**, **fault tolerance**, and **disaster recovery** across multiple Regions and Availability Zones.

**Example:**

If one data center goes down, AWS automatically routes traffic to another one.

**Why It Matters:**

* Your applications stay online
* AWS has a global, redundant infrastructure

**Exam Tip:** “Multiple Availability Zones = High Reliability.”

### 5. Security

**Meaning:**

AWS provides a **secure infrastructure** with strong encryption, compliance, and access control features.

**Example:**

Use IAM to control who can access your AWS resources, encrypt data in S3, and use AWS Shield for DDoS protection.

**Why It Matters:**

* Protects data and applications
* Meets industry standards (e.g., ISO, GDPR, HIPAA)

**Shared Responsibility Model:**  
AWS secures the cloud; **you secure what you put in it**.

### 6. Global Reach

**Meaning:**

AWS has **data centers (Regions and Edge Locations)** all around the world, so you can deploy applications close to your users.

**Example:**

Host your website in Mumbai Region for India, and in London Region for the UK — ensuring low latency.

**Why It Matters:**

* Better user experience
* Compliance with data location laws

**Exam Tip:** “Global infrastructure = Global Reach.”

**Quick Summary Table**

| **Benefit** | **Description** | **Simple Example** |
| --- | --- | --- |
| **1. Pay-as-you-go** | Pay only for what you use | Like electricity bill |
| **2. Scalability & Elasticity** | Auto increase/decrease resources | More servers during peak sale |
| **3. Flexibility & Agility** | Use any OS, language, or tool | Run Windows or Linux on AWS |
| **4. Reliability** | Redundant systems ensure uptime | Failover between regions |
| **5. Security** | Strong encryption & IAM | Protects your data |
| **6. Global Reach** | Deploy worldwide easily | Users get low-latency access |

**Simple Analogy:**

Think of AWS as a **smart utility service**:

| **Utility** | **AWS Equivalent** |
| --- | --- |
| Pay per unit used | Pay-as-you-go |
| Add more current when needed | Scalability |
| Works 24/7 reliably | Reliability |
| Works globally | Global reach |
| Secure grid | AWS security |
| Flexible usage | Flexibility |

## AWS Shared Responsibility Model

The **AWS Shared Responsibility Model** explains **who is responsible for security and compliance** in the cloud —  
**AWS** or **the customer** (you).

**In short:**  
 *AWS secures the cloud.*  
 *You (the customer) secure what you put in the cloud.*

**Why This Model Exists**

In traditional IT (on-premises), **you** manage everything — hardware, networking, data, security, etc.  
But in the **cloud**, AWS takes care of some responsibilities (like the physical data centers),  
and **you** handle the rest (like your data, configurations, and access controls).

### 1. AWS – “Security *of* the Cloud”

AWS is responsible for **protecting the infrastructure** that runs all the services offered in the AWS Cloud.

**AWS Handles:**

* Physical security of data centers
* Hardware, networking, and facility maintenance
* Global infrastructure (Regions, Availability Zones, Edge Locations)
* Underlying hypervisor, storage, compute, and networking layers
* Patching and securing managed services (like S3, DynamoDB)

**Example:**  
You don’t have to worry about who guards the AWS data center or replaces faulty hard drives — AWS does that.

**Exam Tip:**

“AWS manages everything *below* the virtualization layer.”

### 2. Customer – “Security *in* the Cloud”

**Customer’s Responsibility:**

You (the customer) are responsible for **what you put into the cloud** and **how you configure it**.

**You Handle:**

* Managing your data (encryption, backup, classification)
* Setting access controls (IAM users, roles, MFA)
* Configuring network security (VPC security groups, NACLs, firewalls)
* Keeping OS and applications updated (for EC2, containers, etc.)
* Managing credentials and API keys
* Compliance with laws and regulations for your data

**Example:**  
If you make your S3 bucket public by mistake — that’s your responsibility, not AWS’s.

### 3. Shared Responsibilities

Some areas are **shared** — depending on the service type (IaaS, PaaS, SaaS).

For example, AWS might manage part of the configuration, but you manage how it’s used.

**Shared Areas:**

* Patching (AWS patches infrastructure, you patch OS/app)
* Monitoring and logging (AWS provides tools like CloudTrail, you enable and review them)
* Identity and access management (AWS gives IAM, you define permissions)
* Encryption (AWS offers KMS, you decide which data to encrypt)

### Responsibility Split Example

| **Category** | **AWS Responsibility** | **Customer Responsibility** |
| --- | --- | --- |
| **Physical security** | Guarding data centers | None |
| **Compute** | Hardware, hypervisor | OS, patches, apps |
| **Storage** | Infrastructure | Data protection, permissions |
| **Networking** | Physical network | VPC setup, firewall rules |
| **IAM** | Service itself | Creating users, roles, policies |
| **Monitoring** | CloudTrail, CloudWatch | Reviewing and acting on alerts |

### How It Differs by Service Type

| **Service Type** | **AWS Manages** | **Customer Manages** |
| --- | --- | --- |
| **IaaS (e.g., EC2)** | Infrastructure | OS, data, apps, IAM |
| **PaaS (e.g., RDS)** | Infra + database engine | Your data, users, backups |
| **SaaS (e.g., WorkMail)** | Everything except user data | Access control and data content |

### Simple Examples:

| **Scenario** | **Who’s Responsible** |
| --- | --- |
| A hacker breaks into an AWS data center | **AWS** |
| You accidentally expose an S3 bucket | **Customer** |
| AWS EC2 hypervisor failure | **AWS** |
| You forget to patch your EC2 OS | **Customer** |
| Enabling encryption for your database | **Shared (you configure, AWS provides tools)** |

### Quick Summary Table

| **Responsibility** | **AWS Handles** | **Customer Handles** |
| --- | --- | --- |
| Physical security |  | ❌ |
| Network infrastructure |  | ❌ |
| Virtualization layer |  | ❌ |
| OS & applications | ❌ |  |
| Data & encryption | ❌ |  |
| Access management | ❌ |  |
| Shared tasks (IAM, logging, patching) | Both | Both |

# Compute in the Cloud

## Amazon EC2 Instance Types

Amazon EC2 **instances** are **virtual servers** that run your applications in the AWS Cloud.  
Each instance type is designed for **different workloads** — CPU-heavy, memory-heavy, storage-heavy, etc.

**Simple idea:**  
Think of instance types like **different car models** — each one is built for a purpose (speed, storage, fuel efficiency, etc.).

### 1. General Purpose (A, T, M families)

**Balanced performance** — CPU, memory, and networking.  
 **Best for:** web servers, small databases, development environments.

*Exam Tip:* “Balanced performance” = General Purpose.

### 2. Compute Optimized (C family)

High-performance **CPU** for compute-heavy tasks.  
 **Best for:** scientific modeling, gaming servers, batch processing, high-performance web servers.

*Exam Tip:* “CPU-intensive workload” = Compute Optimized.

### 3. Memory Optimized (R, X, z families)

More **RAM** for memory-heavy applications.  
 **Best for:** large databases, in-memory caching, analytics.

*Exam Tip:* “Memory-heavy workload” = Memory Optimized.

### 4. Storage Optimized (I, D, H families)

Designed for **fast storage access** and **large data sets**.  
 **Best for:** big data analytics, data warehousing, high I/O operations.

*Exam Tip:* “High disk throughput or database storage” = Storage Optimized.

### 5. Accelerated Computing (P, G, Inf families)

Use **GPUs or specialized hardware** for high-performance tasks.  
 **Best for:** AI/ML, deep learning, 3D rendering, video processing.

**Examples:**

* **P4, P5** → Machine learning
* **G5** → Graphics and rendering
* **Inf1/Trn1** → AI inference and training

*Exam Tip:* “Machine learning or GPU workloads” = Accelerated Computing.

### EC2 Instance Naming Structure

<Family><Generation><Processor/SpecialOptions>.<Size>

**Example: c7gn.xlarge**

**c** – Compute-optimized

**7** – Seventh generation

**g** – Graviton-based ARM processor

**n** – Enhanced networking & EBS

**xlarge** – Mid-to-large instance size

| **Instance Family** | **Main Focus** | **Best For** |
| --- | --- | --- |
| **General Purpose** | Balanced CPU + Memory | Web servers, dev/test |
| **Compute Optimized** | CPU performance | Gaming, batch processing |
| **Memory Optimized** | RAM | Databases, caching |
| **Storage Optimized** | Disk speed & I/O | Big data, warehousing |
| **Accelerated Computing** | GPUs/AI | Machine learning, rendering |

## Ways to Access AWS

AWS gives you **three main ways** to create, manage, and interact with your cloud resources.

### 1. AWS Management Console

A **web-based graphical interface** that lets you manage AWS services using your browser.

**Key Features:**

* Easy-to-use **GUI** (no coding needed)
* Best for **beginners** and **manual setup**
* Provides **dashboards** and **visual monitoring tools (CloudWatch, Cost Explorer, etc.)**
* Access via: https://aws.amazon.com/console

**Exam Tip:** *If you see “clicking and managing services visually” → that’s AWS Management.*

### 2. AWS Command Line Interfac

⌨️   
A **command-line tool** that lets you manage AWS services using **text commands**.

**Key Features:**

* Run commands from a **terminal or command prompt**
* Useful for **automation and scripting**
* Same functionality as the Console but **faster** for repetitive tasks
* Must be **installed** and configured with credentials (aws configure)

**Example Command:**

aws s3 ls

→ Lists all S3 buckets in your account.

**Exam Tip:** *If you see “manage AWS using commands or scripts” → that’s AWS CLI.*

## Amazon EC2 Pricing

Amazon EC2 (Elastic Compute Cloud) pricing is based on **how you use** virtual servers (instances).  
You **only pay for the compute capacity you use**, and there are several pricing options to suit different needs and budgets.

### 1. On-Demand Instances

* Pay **per second or per hour** for compute capacity.
* **No upfront payment** or long-term commitment.
* Best for:
  + Short-term, unpredictable workloads
  + Testing or development environments

**Exam Tip:** *“Pay as you go” = On-Demand pricing.*

### 2. Reserved Instances (RI)

* Commit to using EC2 for **1 or 3 years** → get **up to 72% discount**.
* You reserve a specific instance type and region.
* Best for:
  + Steady, predictable workloads
  + Long-term use

**Types of Reserved Instances:**

1. **Standard RI** – Biggest discount, fixed instance type.
2. **Convertible RI** – Can change instance family or OS; lower discount.
3. **Scheduled RI** – Run during specific time windows.

### 3. Savings Plans

* Flexible pricing model similar to RIs but with **more flexibility**.
* Commit to a **specific spend per hour (e.g., $10/hour)** for 1 or 3 years.
* You can change instance types or regions freely.

**Exam Tip:** *Savings Plans = Flexibility + Discount.*

### 4. Spot Instances

* Use **unused EC2 capacity** at **up to 90% discount**.
* AWS can **terminate** your instance if capacity is needed elsewhere.
* Best for:
  + Fault-tolerant or flexible workloads (batch jobs, data analysis, etc.)

**Exam Tip:** *Cheapest option but not reliable for continuous workloads.*

### 5. Dedicated Hosts / Dedicated Instances

* **Physical servers** dedicated to one customer.
* Helps with **compliance** and **license management**.
* More expensive than shared instances.

| **Pricing Option** | **Commitment** | **Discount** | **Best For** | **Key Point** |
| --- | --- | --- | --- | --- |
| On-Demand | None | ❌ | Short-term / Testing | Pay per use |
| Reserved | 1–3 yrs | Up to 72% | Steady workloads | Pre-booked |
| Savings Plans | 1–3 yrs (spend-based) |  | Flexible long-term | Flexible |
| Spot | None | Up to 90% | Fault-tolerant tasks | Can be interrupted |
| Dedicated Host | Long-term | ❌ | Compliance / Licensing | Physical server |

## What is Load Balancing?

Load balancing is the process of **distributing incoming network traffic** across multiple servers (EC2 instances) to make sure:

* No single server gets overloaded
* Applications stay **available**, **scalable**, and **performant**

**Simple example:**  
Think of a restaurant — if one waiter gets too many customers, service slows down.  
Load balancing spreads customers (traffic) among multiple waiters (servers).

### What is Elastic Load Balancing (ELB)?

**Elastic Load Balancing (ELB)** is an AWS service that **automatically distributes traffic** across multiple EC2 instances, containers, or IP addresses in one or more Availability Zones.

**Key Features:**

* **Automatic traffic distribution**
* **High availability** — spreads load across multiple AZs
* **Health checks** — sends traffic only to healthy targets
* **Scalability** — automatically adjusts to traffic changes
* **Integration** with Auto Scaling, EC2, ECS, etc.

**Exam Tip:** *ELB = AWS-managed load balancer that scales automatically.*

## Amazon SQS, SNS, and EventBridge

These are all **messaging and event services** that help AWS systems **communicate** with each other **without being directly connected**.

**Simple idea:**  
They are like different ways of **sending messages or alerts** between AWS services or applications.

### 1. Amazon SQS (Simple Queue Service)

Amazon SQS is a **message queue service** that lets one system **send messages** to another system **asynchronously** (not in real time).

It works like a **waiting line (queue)** — messages are stored until the receiver is ready to process them.

**Key Features:**

* **Fully managed** message queue
* **Decouples** components — sender and receiver don’t need to run at the same time
* **Highly scalable** and reliable
* **Messages are temporary** — deleted after being processed

**Types of Queues:**

| **Type** | **Description** | **Best For** |
| --- | --- | --- |
| **Standard Queue** | Unlimited throughput, messages can arrive out of order or be duplicated | General-purpose apps |
| **FIFO Queue (First-In-First-Out)** | Ensures order and no duplicates | Financial apps, ordered tasks |

**Exam Tip:**  
If the question says *“store and process messages later”* → **SQS**.

### 2. Amazon SNS (Simple Notification Service)

Amazon SNS is a **publish/subscribe (pub/sub)** messaging service.  
It sends **notifications to multiple subscribers at once**.

**Key Features:**

* **One-to-many** messaging (1 publisher → many subscribers)
* **Instant delivery** (real-time notifications)
* Supports delivery to:
  + **Email**
  + **SMS**
  + **Lambda functions**
  + **HTTP endpoints**
* **Fully managed and serverless**

**Example:**  
When a photo is uploaded to S3, SNS can **notify** multiple users or services instantly.

**Exam Tip:**  
If you see “send the same message to many receivers” → **SNS**.

### 3. Amazon EventBridge

Amazon EventBridge is a **serverless event bus** service that connects AWS services, your apps, and SaaS apps using **events**.

It reacts to **events happening in your AWS environment** and triggers actions automatically.

**Key Features:**

* **Event-driven** architecture
* Can receive events from AWS services, your apps, or external SaaS apps
* **Automates workflows** (e.g., when an EC2 instance stops → trigger a Lambda)
* **Integrates with hundreds of AWS services**

**Example:**  
When an EC2 instance changes state → EventBridge triggers a Lambda function to log it.

**Exam Tip:**  
If the question says “automate a response when something happens in AWS” → **EventBridge**.

| **Service** | **Type** | **Main Function** | **Message Flow** | **Example Use** |
| --- | --- | --- | --- | --- |
| **SQS** | Queue | Store messages temporarily | One-to-one | Order processing queue |
| **SNS** | Notification | Send alerts to many | One-to-many | Send notifications (SMS/email) |
| **EventBridge** | Event Bus | React to system events | Many-to-many | Trigger Lambda on AWS events |

# Exploring Compute Services

## Managed, Unmanaged, and Serverless Services

These terms describe **how much AWS manages for you** versus **how much you manage yourself**.

### Unmanaged Services

Unmanaged services are where **you manage almost everything** — AWS only provides the basic infrastructure (like virtual machines).

You handle installation, configuration, scaling, security patches, etc.

**Key Features:**

* You manage: OS, runtime, scaling, backups, and updates
* More **control**, but more **work**
* Common in traditional IT setups

**Examples:**

* **Amazon EC2** (you manage what runs inside it)
* **EBS volumes** (you manage data and usage)

**Exam Tip:**  
If AWS only provides the infrastructure → **Unmanaged Service**.

### Managed Services

Managed services are where **AWS handles most of the operations**, like setup, maintenance, scaling, and availability — you just use the service.

You still configure and use it, but **AWS manages the backend**.

**Key Features:**

* Less operational work
* Automatic updates, scaling, and fault tolerance
* Focus on using the service, not maintaining it

**Examples:**

* **Amazon RDS** – AWS manages database patching, backups, failover
* **Amazon ECS** – AWS manages container orchestration
* **AWS Elastic Load Balancer** – AWS manages traffic distribution

**Exam Tip:**  
If AWS manages infrastructure and maintenance → **Managed Service**.

### Serverless Services

Serverless services mean **you don’t manage any servers** — AWS handles *everything* (servers, scaling, availability).  
You only focus on your **code or business logic**, and **pay only when your code runs**.

**Key Features:**

* **No server management** at all
* **Auto-scaling** and **pay-per-use**
* **Highly available** and **event-driven**
* Best for **modern, event-based applications**

**Examples:**

* **AWS Lambda** – run code without servers
* **Amazon S3** – store and retrieve data without managing storage servers
* **Amazon DynamoDB** – fully serverless NoSQL database
* **EventBridge**, **SNS**, **SQS** – serverless messaging/event services

**Exam Tip:**  
If the question says *“no servers to manage, pay only when used”* → **Serverless**.

**Quick Comparison Table**

| **Type** | **Who Manages Most** | **You Manage** | **Example Services** | **Billing Model** |
| --- | --- | --- | --- | --- |
| **Unmanaged** | You | OS, updates, scaling | EC2, EBS | Pay for instance uptime |
| **Managed** | AWS (partly) | Configurations | RDS, ECS, ELB | Pay for provisioned resources |
| **Serverless** | AWS (fully) | Your code / logic | Lambda, S3, DynamoDB | Pay per request or execution |

## AWS Lambda

**AWS Lambda** is a **serverless compute service** that lets you **run code without managing servers**.  
You simply upload your code, and AWS automatically runs it **when triggered** by an event.

**In short:**  
 *You write the code — AWS handles the rest (servers, scaling, uptime).*

### How It Works

1. You **upload your code** (in Python, Node.js, Java, etc.).
2. You **set a trigger** (like an S3 upload, API call, or event).
3. AWS Lambda **runs your code automatically** whenever that trigger happens.
4. You **pay only for the time your code runs** — not for idle time.

### Key Features

* **No servers to manage** (completely serverless)
* **Event-driven** (runs when triggered by AWS events like S3, DynamoDB, or API Gateway)
* **Auto-scaling** (runs as many instances as needed automatically)
* **Pay-per-use** (charged for execution time and number of requests)
* **Supports multiple languages** (Python, Node.js, Java, Go, C#, etc.)
* **Integrates easily** with other AWS services (S3, DynamoDB, SNS, EventBridge, etc.)

### Common Triggers for Lambda

| **AWS Service** | **Event That Triggers Lambda** | **Example Use** |
| --- | --- | --- |
| **S3** | File upload | Process an image or file after upload |
| **DynamoDB** | Table update | React to data changes |
| **API Gateway** | API request | Build serverless web APIs |
| **SNS / SQS** | Message received | Send notification or process message |
| **EventBridge** | AWS system events | Automate responses to changes |

## Containers and Orchestration on AWS

### What are Containers?

A **container** is a lightweight, portable way to package an application **and everything it needs** (code, runtime, libraries) so it runs the same everywhere.

**Simple idea:**  
Think of a container like a **sealed lunchbox** — everything your app needs is inside, so it works consistently anywhere.

**Why Use Containers?**

* Fast to start and stop
* Portable (run the same app on laptop, server, or cloud)
* Efficient — share the same OS kernel
* Easier to scale and deploy microservices

**Exam Tip:**  
Containers = fast, lightweight, and portable applications.

### Orchestration

Orchestration means **automatically managing** many containers — deciding **where they run**, **when to start/stop**, and **how to scale** them.

AWS provides multiple orchestration tools for containers

### AWS Container Services Overview

| **Service** | **Full Name** | **Purpose** | **Who Manages Infrastructure** | **Best For** |
| --- | --- | --- | --- | --- |
| **ECS** | Elastic Container Service | Run containers easily on AWS | AWS manages orchestration | AWS-native workloads |
| **EKS** | Elastic Kubernetes Service | Run containers using Kubernetes | AWS manages control plane | Kubernetes users |
| **ECR** | Elastic Container Registry | Store & manage container images | AWS | Secure image storage |
| **Fargate** | AWS Fargate | Run containers *without managing servers* | Fully managed by AWS | Serverless container execution |

Let’s go over each in short

### Amazon ECS (Elastic Container Service)

**What It Is:**

A **fully managed container orchestration** service by AWS.  
It helps you **run and scale Docker containers** on AWS easily.

**Use When:**

* You want to use containers but **don’t need Kubernetes**.
* You prefer an **AWS-managed solution** (simple and integrated).

**Why ECS:**

* Deep integration with AWS (CloudWatch, IAM, ALB, etc.)
* Easier to set up than Kubernetes
* Works with **EC2 (you manage servers)** or **Fargate (serverless)**

**Exam Tip:** *ECS = AWS’s own container orchestrator.*

### Amazon EKS (Elastic Kubernetes Service)

**What It Is:**

A **managed Kubernetes service** on AWS.  
You get **all Kubernetes features**, but AWS manages the control plane (the brain of Kubernetes).

**Use When:**

* You already use **Kubernetes** or want portability across clouds.
* You need more **customization and control**.

**Why EKS:**

* Compatible with open-source Kubernetes
* Integrates with AWS networking, security, and scaling tools
* You can run workloads **on EC2 or Fargate**

**Exam Tip:** *EKS = Managed Kubernetes on AWS.*

### Amazon ECR (Elastic Container Registry)

**What It Is:**

A **fully managed container image registry** where you **store, share, and manage Docker images** securely.

**Use When:**

* You need a **secure, private place** to store container images.
* You want **integration** with ECS, EKS, or Fargate.

**Why ECR:**

* AWS handles storage, encryption, and permissions
* Works seamlessly with ECS & EKS

**Exam Tip:** *ECR = Storage for your container images.*

### AWS Fargate

**What It Is:**

A **serverless compute engine for containers** — runs containers **without you managing servers or clusters**.

**Use When:**

* You want to run containers **without EC2 instances**
* You prefer **fully managed, pay-per-use** container execution

**Why Fargate:**

* No need to manage EC2 instances
* Automatically scales based on demand
* Works with **ECS and EKS**

**Exam Tip:** *Fargate = Serverless containers.*

**How They Work Together**

You build and push your image to **ECR**  
 You choose to run it using **ECS** or **EKS**  
 You can choose **Fargate** (serverless) or **EC2** (you manage servers) to host it

**Quick Summary Table**

| **Service** | **Type** | **Manages Servers?** | **Use When** | **Example Use Case** |
| --- | --- | --- | --- | --- |
| **ECS** | Container Orchestrator | Optional (Fargate = no, EC2 = yes) | You want AWS-managed orchestration | Run web app containers |
| **EKS** | Kubernetes Orchestrator | Optional (Fargate = no, EC2 = yes) | You use Kubernetes | Multi-cloud |
| **ECR** | Container Image Registry | N/A | You need to store container images | Store Docker app images |
| **Fargate** | Compute Engine (Serverless) | ❌ No servers to manage | You want simplicity | Run containers serverlessly |

**Putting It All Together (Workflow)**

1. **Create the container (lunchbox)** → your app is ready.
2. **Store it in ECR (pantry)** → ready for execution.
3. **Choose ECS or EKS (restaurant manager or expert chef)** → orchestrate containers.
4. **Optional: Fargate (automatic cooking machine)** → run containers **serverlessly** without managing servers.

**Key Takeaways for Exam:**

* Containers = portable apps
* ECR = store them
* ECS = AWS manages orchestration
* EKS = Kubernetes orchestration (more control)
* Fargate = serverless execution

## Elastic Beanstalk, AWS Batch, and Amazon Lightsail

### AWS Elastic Beanstalk

AWS Elastic Beanstalk is a **Platform as a Service (PaaS)** that lets you **deploy and manage web applications** quickly **without worrying about the infrastructure**.

You just **upload your code**, and Beanstalk automatically:

* Provisions EC2 instances
* Sets up load balancing & auto scaling
* Deploys your app
* Monitors health

**In short:**  
 “Upload your app, AWS handles everything else.”

**Key Features:**

* Supports popular languages (Python, Java, Node.js, .NET, PHP, Go, etc.)
* Automatically handles **scaling**, **load balancing**, and **monitoring**
* You **still have access** to the underlying EC2 if you want control
* Perfect for **web developers** who just want to focus on code

**Simple Example:**

You built a web app in **Python**.  
→ Upload your .zip file to Elastic Beanstalk.  
→ AWS automatically launches EC2s, configures an ELB, deploys your code, and manages scaling.

**Best For:**

* Quickly deploying web applications
* Developers who don’t want to manage infrastructure manually

**Exam Tip:**

If the question says *“deploy web applications easily without managing servers”* → **Elastic Beanstalk**

### AWS Batch

AWS Batch is a **fully managed batch processing service** that runs **large numbers of compute jobs** efficiently — without you having to manage servers.

**Simple idea:**  
It’s like a **task scheduler** — you give it a list of jobs, and it runs them **when compute resources are available**.

**Key Features:**

* Automatically provisions **EC2 or Spot Instances**
* Handles **job scheduling and scaling**
* Supports **thousands of batch jobs** at once
* Pay only for compute used

**Simple Example:**

You need to **process 10,000 images or videos** overnight.  
Instead of starting EC2 manually, you submit all jobs to **AWS Batch** → it runs them in parallel, scales resources automatically, and shuts them down after completion.

**Best For:**

* Data processing
* Simulations
* Scientific or financial calculations
* Video rendering

**Exam Tip:**

If the question says *“run many jobs that can run in parallel or scheduled batches”* → **AWS Batch**

### Amazon Lightsail

Amazon Lightsail is the **simplest way** to launch a **virtual private server (VPS)** on AWS.  
It’s designed for **small businesses, students, or simple projects** that don’t need full AWS complexity.

**Simple idea:**  
It’s like a **preconfigured mini cloud** — click once to get a ready-to-use server.

**Key Features:**

* Pre-built templates for **WordPress, LAMP, Node.js**, etc.
* Includes **compute, storage, database, and networking** in a simple bundle
* **Fixed monthly pricing** (easy to understand bills)
* Simple **management console**

**Simple Example:**

You want to host a **small website or WordPress blog**.  
→ Instead of using EC2 + S3 + RDS separately, you use **Lightsail** → click “Create instance,” select WordPress, and your site is online in minutes.

**Best For:**

* Small businesses
* Websites, blogs, or development projects
* People who want AWS simplicity + predictable pricing

**Exam Tip:**

If the question says *“simple VPS hosting for small projects with fixed pricing”* → **Amazon Lightsail**

### Quick Comparison Table

| **Service** | **Type** | **What It Does** | **Best For** | **AWS Handles** |
| --- | --- | --- | --- | --- |
| **Elastic Beanstalk** | PaaS (Platform as a Service) | Deploy web apps easily | Web apps | Infra, scaling, monitoring |
| **AWS Batch** | Batch Processing | Run multiple compute jobs automatically | Data, analytics, simulations | Scheduling, scaling |
| **Amazon Lightsail** | Simplified VPS | Launch simple servers easily | Small projects, websites | Setup, pricing simplification |

### Real-Life Analogy

| **Service** | **Analogy** | **Meaning** |
| --- | --- | --- |
| **Elastic Beanstalk** | “Chef who cooks from your recipe” | You bring the code, AWS cooks and serves it |
| **AWS Batch** | “Factory manager running many machines at once” | Automatically processes large job batches |
| **Lightsail** | “Microwave-ready meal” | One-click ready setup, no complexity |

## Amazon Outposts

**Amazon Outposts** brings **AWS infrastructure and services** **to your own on-premises data center**.  
It’s like **running AWS inside your building** — same hardware, same services, same tools.

**In short:**  
 *It’s AWS, but physically located in your own data center.*

**Why It Exists (The Need):**

Some companies need to:

* Keep data **on-premises** (for security, compliance, or low-latency reasons)
* Still want to use **AWS services and management tools**

So AWS sends and installs **Outposts racks (hardware)** at your site — giving you a **true hybrid cloud** setup.

**How It Works (Simple Flow):**

1. AWS **ships Outposts hardware** (servers, storage, networking) to your data center.
2. It’s **connected securely** to the nearest **AWS Region**.
3. You can run **AWS services locally** (like EC2, EBS, RDS) on that Outpost.
4. AWS manages and updates it remotely — **you use it like AWS cloud**, but it runs **physically at your site**.

**Example:**

A hospital or bank needs to keep data **within their building** for compliance,  
but still wants to use AWS compute and storage.  
→ They install **AWS Outposts** in their data center.  
→ They can run EC2, EBS, or RDS **on-premises** using AWS APIs.  
→ Data stays local; management stays AWS-style.

**Key Features:**

* **AWS hardware installed locally**
* **Fully managed by AWS** (updates, monitoring, support)
* **Consistent AWS experience** (same APIs, console, tools)
* 🕸️ **Connected to AWS Region** (for hybrid operations)
* **Low latency** for applications needing local processing

**Use Cases:**

| **Use Case** | **Why Outposts Helps** |
| --- | --- |
| **Healthcare / Finance** | Data residency or compliance rules |
| **Manufacturing / IoT** | Local processing with low latency |
| **Edge computing** | Applications near end-users |
| **Migration to cloud** | Gradual move from on-prem to AWS |

**Pricing (Basic Idea):**

* You pay for **Outposts capacity** (like EC2 + EBS)
* Plus **installation and support**
* It’s billed through your AWS account

# Going Global

## Components of AWS Global Infrastructure

**Simple**

The **AWS Global Infrastructure** is the **foundation that AWS uses to deliver cloud services** reliably and securely **all around the world**.

It’s made up of **Regions**, **Availability Zones (AZs)**, **Edge Locations**, and a few more components.

**Main Components**

Let’s understand each one **step by step**

### 1. AWS Regions

A **Region** is a **geographical area** that contains **two or more Availability Zones (data centers)**.

Each Region operates **independently** — meaning your data stays in that region unless you move it.

**Key Points:**

* Each region is **isolated** for fault tolerance and data control.
* You **choose a region** to host your resources (like EC2, S3, RDS).
* Regions are named like:
  + us-east-1 → (N. Virginia, USA)
  + ap-south-1 → (Mumbai, India)

**Example:**

If your customers are mostly in India, you’d choose the **Asia Pacific (Mumbai)** region to reduce latency.

### 2. Availability Zones (AZs)

An **Availability Zone** is **one or more data centers** in a Region, each with **independent power, cooling, and networking**.

All AZs within a Region are connected with **high-speed, low-latency links**.

**Key Points:**

* Each Region has **at least 2 or more AZs**.
* AZs are labeled like ap-south-1a, ap-south-1b, etc.
* Used for **high availability** — if one AZ fails, others continue.
* You can **distribute resources across AZs** for disaster recovery.

**Example:**

If you deploy two EC2 instances — one in ap-south-1a and one in ap-south-1b — your app stays up even if one AZ fails.

### 3. Edge Locations

**Edge Locations** are **AWS data centers used by CloudFront (CDN)** to **cache content closer to users** for faster delivery.

**Key Points:**

* Used for **Content Delivery Network (CDN)** via **Amazon CloudFront**.
* Improve **latency** for end-users.
* There are **hundreds of Edge Locations** globally.
* Mainly used for **web content**, **video streaming**, or **API acceleration**.

**Example:**

A user in Delhi requests your website hosted in the US.  
→ CloudFront serves it from the **nearest Edge Location in India** → faster load time.

### 4. Regional Edge Caches

Larger caching layers between AWS Regions and Edge Locations.

They help **store less-frequently accessed content** to improve performance.

**Think of it as:**  
 A **bigger warehouse** that supports multiple smaller delivery points (Edge Locations).

### 5. Local Zones

A **Local Zone** brings AWS compute, storage, and database services **closer to large population centers** for **ultra-low latency**.

**Key Points:**

* Extension of an AWS Region but **physically closer** to users.
* Great for **gaming, media, video editing, or real-time apps**.

**Example:**

If your main AWS Region is in Mumbai,  
→ A **Local Zone in Delhi** can serve users in North India faster.

### 6. AWS Wavelength

AWS Wavelength brings **AWS services to telecom (5G) networks** to provide **ultra-low latency** for mobile and IoT applications.

**Example:**

Used for real-time apps like **autonomous vehicles** or **AR/VR**, where milliseconds matter.

**Exam Tip:**

“Wavelength = AWS inside 5G networks for ultra-low latency.”

### 7. AWS Outposts (Recap)

Outposts bring **AWS hardware and services into your own data center**.

**Exam Tip:**

“Outposts = AWS on-premises (hybrid cloud).”

### Quick Summary Table

| **Component** | **Description** | **Example** | **Key Purpose** |
| --- | --- | --- | --- |
| **Region** | Geographical area with multiple AZs | ap-south-1 (Mumbai) | Choose where to host your resources |
| **Availability Zone** | One or more data centers in a region | ap-south-1a | High availability & fault tolerance |
| **Edge Location** | Cache data close to users | CloudFront location in Delhi | Low latency content delivery |
| **Regional Edge Cache** | Larger cache between region and edge | Supports multiple edges | Improves caching performance |
| **Local Zone** | AWS extension near users | Local Zone in Chennai | Ultra-low latency compute |
| **Wavelength Zone** | AWS inside telecom 5G networks | 5G edge location | Ultra-low latency for mobile/IoT |
| **Outposts** | AWS on-premises hardware | Bank’s local data center | Hybrid cloud setup |

### Key Factors to Consider (Easy to Remember: “L-C-C-G-C-R”)

| **#** | **Factor** | **Meaning** | **Why It Matters** |
| --- | --- | --- | --- |
| 1 | **Latency and Proximity to Users** | Choose the Region **closest to your customers or users**. | Closer = faster response time, lower latency. |
| 2 | **Compliance and Data Residency** | Some businesses (banks, healthcare, government) must **store data within specific countries**. | Choose a Region that **meets legal or regulatory rules**. |
| 3 | **Cost (Pricing Differences)** | AWS prices **vary by Region**. | You can save money by choosing a Region with lower pricing (if latency or laws allow). |
| 4 | **Service Availability** | Not all AWS services are available in every Region. | Check if your required services (like Outposts, Wavelength, etc.) exist in that Region. |
| 5 | **Geopolitical Stability** | Consider Regions that are **politically stable** and have **reliable infrastructure**. | Ensures long-term reliability and business continuity. |
| 6 | **Resiliency and Disaster Recovery** | Some businesses deploy in **multiple Regions** for backup. | Improves fault tolerance and availability in case one Region fails. |

## Amazon CloudFront

**Amazon CloudFront** is a **Content Delivery Network (CDN)** service from AWS that **delivers data, videos, applications, and APIs to users quickly and securely** — using a **global network of Edge Locations**.

**In short:**  
 *It stores (caches) copies of your content in many locations worldwide so users get it faster — from the nearest location.*

### How It Works (Simple Flow)

Your **main content** (like a website, image, or video) is stored in an **origin server** (like **S3 bucket**, **EC2**, or an **on-prem server**).  
 When a user requests that content:

* CloudFront checks the **nearest Edge Location**.
* If cached there → it delivers instantly
* If not cached → it fetches from the origin once and stores it for future requests.

**Result:** Faster performance, lower latency, less load on your main server.

### Pricing (Simplified for Exam)

You pay for:

1. **Data transfer out** (from Edge to users)
2. **Number of requests**

No upfront cost — pay only for what you use.  
 Often **cheaper** than sending all traffic directly from your Region.

### Key Benefits of CloudFront

| **Benefit** | **Explanation** | **Example / Analogy** |
| --- | --- | --- |
| **Low Latency & High Speed** | Content served from the **nearest edge location** | Users in India access content from Mumbai edge instead of US server |
| **Scalability** | Automatically handles **high traffic spikes** | Streaming video to millions of users during a live event |
| **Global Reach** | Edge locations in **hundreds of cities worldwide** | Users worldwide get fast access to your content |
| **Security** | Integrates with **AWS Shield, AWS WAF, and HTTPS** | Protects your website from DDoS and malicious attacks |
| **Cost Efficiency** | Reduces load on origin servers, saving bandwidth | Less data transferred from S3 → lower costs |
| **Customizable Content Delivery** | Can set caching rules, headers, cookies, and query strings | Serve different content to different users or devices |

## AWS CloudFormation

**AWS CloudFormation** is a service that lets you **create and manage AWS resources automatically** using **templates** (written in JSON or YAML).

**In short:**  
 *You describe what you want (in a template), and CloudFormation builds it for you.*

### Why It’s Used (The Problem It Solves):

Without CloudFormation:

* You’d have to **manually click** through the AWS console to create EC2, S3, VPCs, etc.
* It’s **slow**, **error-prone**, and **hard to repeat** across environments (like dev/test/prod).

With CloudFormation:

* You **define everything as code**.
* AWS automatically **creates, updates, or deletes** all your resources in the **correct order**.

**Result:** Fast, consistent, and automated infrastructure setup.

### How It Works (Simple Steps):

1. You write a **template** (in JSON or YAML) describing your resources — e.g. EC2, S3, IAM Role.
2. You upload it to **CloudFormation**.
3. CloudFormation reads it and creates a **stack** — a group of AWS resources.
4. AWS automatically builds and configures everything in the right sequence.

### Example:

You want:

* 1 EC2 instance
* 1 Security Group
* 1 S3 bucket

Instead of creating each manually, you define them in one **CloudFormation template**.  
CloudFormation automatically:  
 Launches the EC2  
 Applies the Security Group  
 Creates the S3 bucket

All **in minutes**, with no manual steps!

**Use Cases:**

| **Use Case** | **Description** |
| --- | --- |
| **Deploy full environments** | Launch complete apps with all needed AWS resources. |
| **Automate infrastructure setup** | Build new servers, databases, and networks easily. |
| **Consistent testing environments** | Reuse templates for dev/test/prod. |
| **Disaster recovery** | Quickly rebuild infrastructure from a saved template. |

### Pricing (Easy for Exam):

**No extra cost** for CloudFormation itself!  
💰 You only pay for the **AWS resources** it creates (like EC2, S3, etc.).

# Networking

## AWS Virtual Private Cloud (VPC)

A **Virtual Private Cloud (VPC)** is your **own isolated section of the AWS cloud** where you can **launch AWS resources (like EC2 instances) in a virtual network** that you define.

**In short:**  
 *VPC = Your private “slice” of AWS where you control networking.*

**What a VPC Does:**

1. Lets you define your **IP address range** (like your network in a company).
2. Allows creation of **subnets** to organize resources.
3. Provides **security controls**:
   * **Security Groups** (like virtual firewalls for EC2)
   * **Network ACLs** (like firewall for subnets)
4. Enables **routing and Internet access** via **route tables and gateways**.
5. Supports **private connectivity** (VPN, Direct Connect) or **public Internet access**.

### Subnet

A **subnet** is a **smaller network within a VPC**.  
It lets you **divide your VPC’s IP range into segments** for better organization and security.

**In short:**  
 *Subnet = A room in your VPC building.*

**What a Subnet Does:**

1. Organizes resources logically (e.g., web servers in one subnet, databases in another).
2. Controls **Internet access** using **route tables** and **gateways**.
3. Improves **security and fault isolation**.
4. Can be **public or private**, depending on whether it has Internet access.

### Public vs Private Subnets

#### 1. Public Subnet

A **public subnet** is a subnet **connected to the Internet** via an **Internet Gateway (IGW)**.  
Resources here can **send and receive traffic from the Internet**.

**Features:**

* Has a **route to Internet Gateway**
* Usually hosts **web servers, load balancers, NAT gateways**
* Accessible from the Internet

**Example:**

* Your **website EC2 instances** in a public subnet can be accessed by users worldwide.

#### 2. Private Subnet

A **private subnet** **cannot communicate directly with the Internet**.  
Resources here are **isolated** and only communicate with internal resources or via NAT.

**Features:**

* No direct route to Internet Gateway
* Hosts **databases, application servers, internal services**
* Can access Internet indirectly via **NAT Gateway** if needed

**Example:**

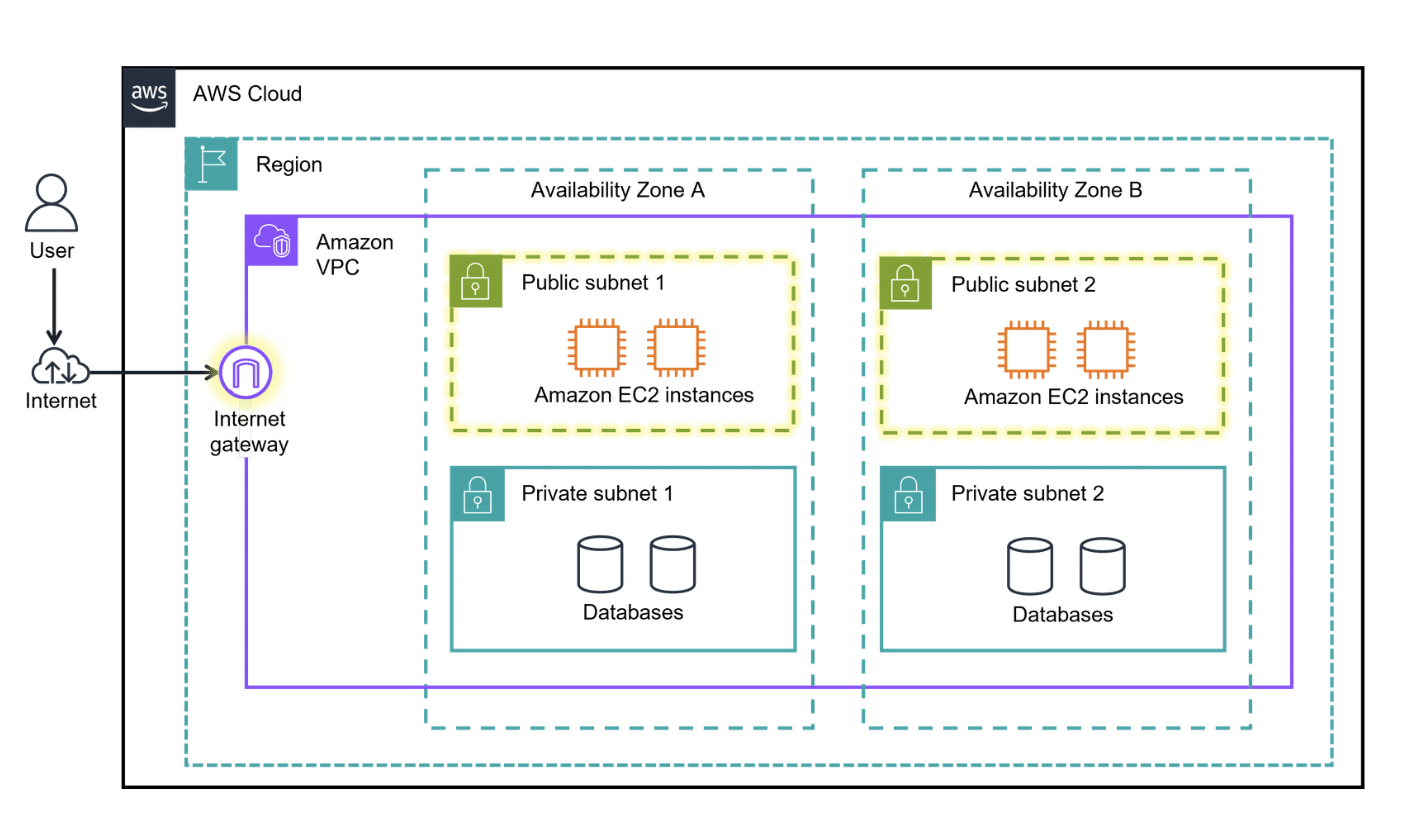
* Your **RDS database** sits in a private subnet — not exposed to the Internet, but accessible by your web servers in the public subnet.

**Comparison Table:**

| **Feature** | **Public Subnet** | **Private Subnet** |
| --- | --- | --- |
| Internet Access | Yes, direct via IGW | No direct access |
| Use Case | Web servers, Load balancers | Databases, internal apps |
| Security | More exposed, needs firewall | More secure, internal only |
| Route Table | Route to Internet Gateway | Route to NAT or local only |

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “A virtual network isolated in AWS” | **VPC** |
| “Divides VPC into smaller networks” | **Subnet** |
| “Hosts public-facing web servers” | **Public Subnet** |
| “Hosts internal databases not exposed to Internet” | **Private Subnet** |
| “Needs Internet Gateway for Internet access” | **Public Subnet** |
| “Access Internet via NAT Gateway” | **Private Subnet** |



## Organizing AWS Cloud Resources

### Virtual Private Gateway (VGW)

A **Virtual Private Gateway** is a **logical VPN concentrator on the AWS side** of a VPN connection.

**In short:**  
 *It’s the gateway that lets your VPC connect securely to your on-premises network.*

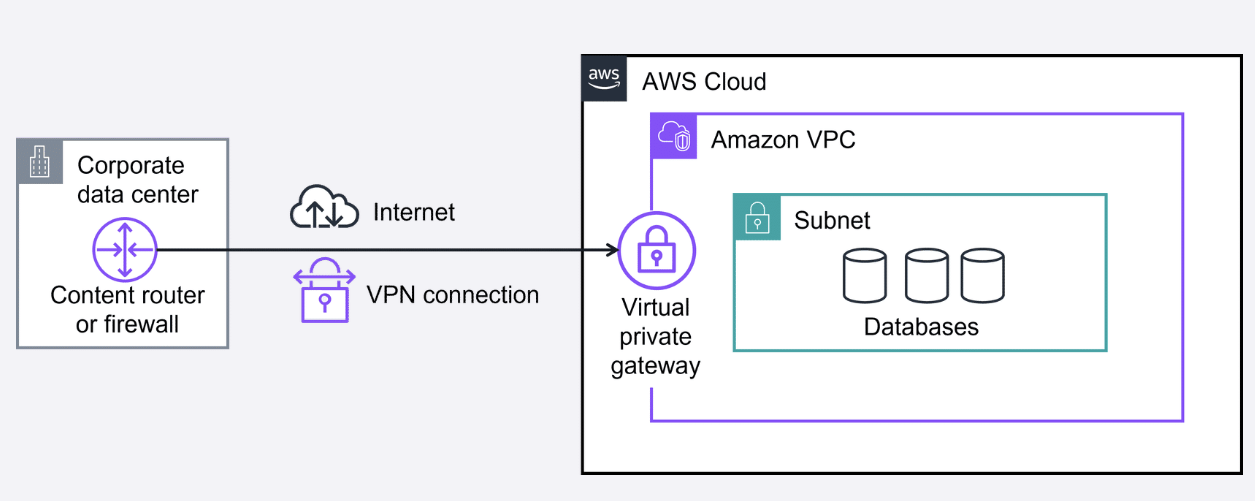
**What It Does:**

1. Connects a **VPC to an on-premises network** via **IPSec VPN**.
2. Supports **AWS Direct Connect** for private connections.
3. Allows secure, encrypted communication between your **AWS cloud and your office data center**.

**Example:**

* Your company has a **data center in Mumbai**.
* You create a **VPN connection** from your on-premises router to a **Virtual Private Gateway** in your AWS VPC.
* This allows internal apps in AWS to communicate securely with your local servers.

Think of it like a **secure bridge between your private office network and AWS VPC**.



### Core Components of a VPC

A **VPC** is made up of several building blocks:

| **Component** | **What It Does** |
| --- | --- |
| **Subnets** | Divide the VPC IP range into smaller networks (public/private). |
| **Route Tables** | Define how traffic flows between subnets, Internet, and VPN. |
| **Internet Gateway (IGW)** | Enables Internet access for resources in the VPC. |
| **Virtual Private Gateway (VGW)** | Enables VPN or Direct Connect to on-prem networks. |
| **Security Groups** | Virtual firewalls for EC2 instances; controls inbound/outbound traffic. |
| **Network ACLs (NACLs)** | Additional subnet-level firewall for traffic control. |
| **Elastic IPs** | Static public IPs for Internet access. |
| **Endpoints** | Private connection to AWS services without Internet. |

### Internet Gateway (IGW)

An **Internet Gateway** is a **horizontally scaled, redundant, and highly available VPC component** that **allows communication between resources in your VPC and the Internet**.

**In short:**  
 *IGW = Door to the Internet for your VPC.*

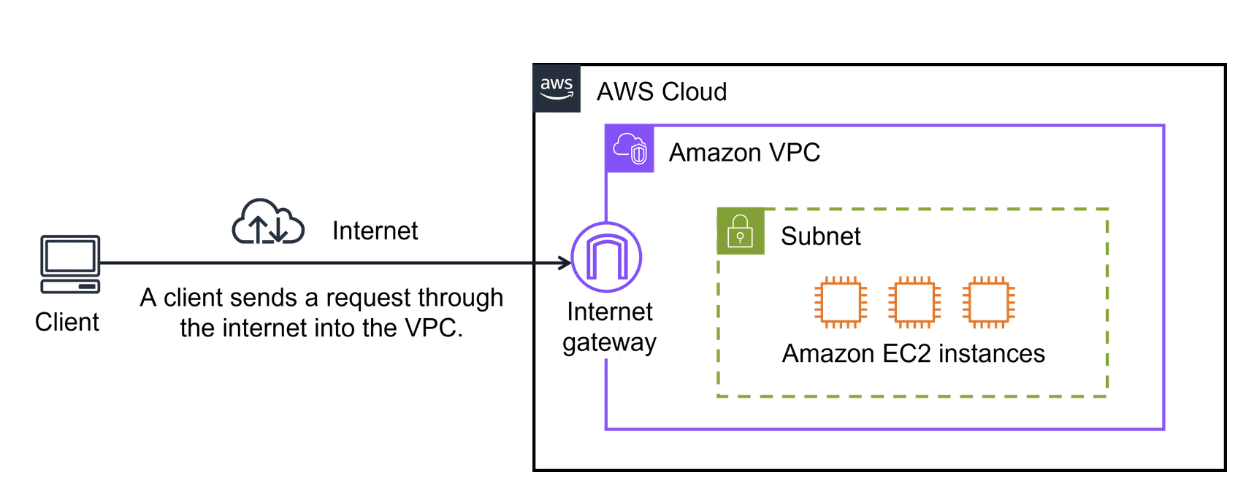
**What It Does:**

1. Provides **public Internet access** for resources in **public subnets**.
2. Supports **IPv4 and IPv6 traffic**.
3. Works with **route tables** to define which subnets are public.
4. Enables **two-way communication** (inbound & outbound) with the Internet.

**Example:**

* You have an **EC2 web server** in a public subnet.
* You attach an **Internet Gateway** and update the **route table** → your web server becomes accessible globally.

Think of IGW as the **main entrance/exit** to your cloud “city.”

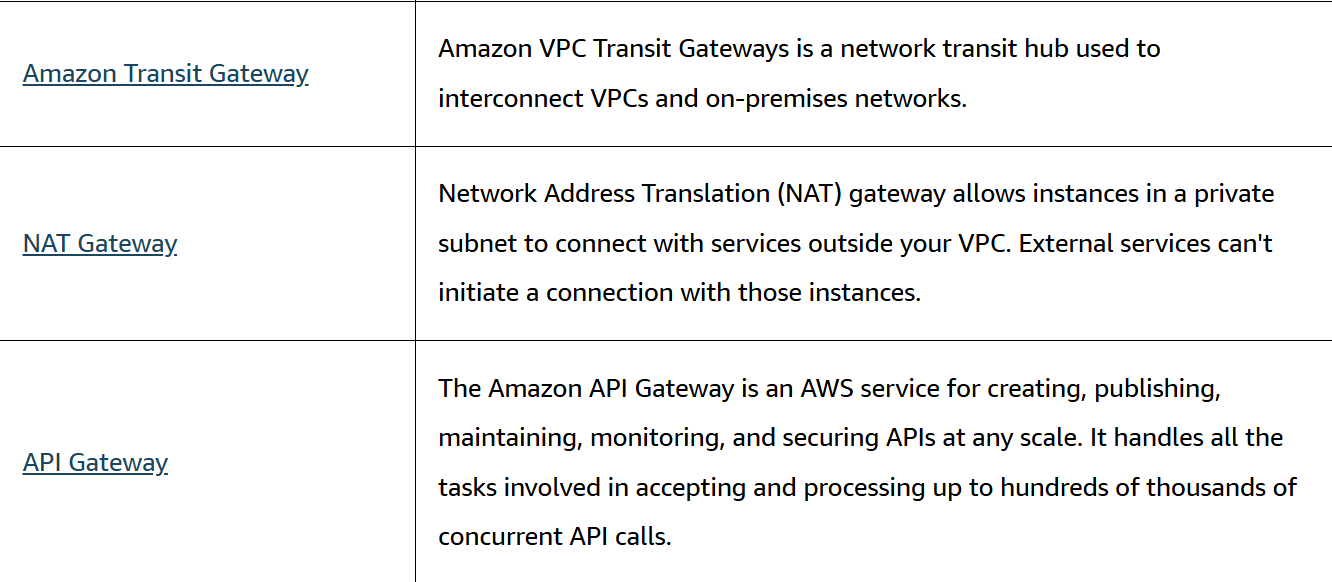


### VGW vs IGW (Quick Comparison)

| **Feature** | **Virtual Private Gateway** | **Internet Gateway** |
| --- | --- | --- |
| **Purpose** | Connect VPC to on-premises network (VPN/Direct Connect) | Connect VPC to Internet |
| **Traffic Type** | Private, encrypted | Public, Internet traffic |
| **Use Case** | Corporate office → VPC connection | Web servers accessible to Internet |
| **Route Table** | Routes for VPN | Routes for Internet |

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Allows secure connection between on-prem and VPC” | **Virtual Private Gateway** |
| “Provides public Internet access to a VPC” | **Internet Gateway** |
| “Core component that divides a VPC into smaller networks” | **Subnet** |
| “Controls traffic at instance level” | **Security Group** |
| “Controls traffic at subnet level” | **Network ACL (NACL)** |



## More Ways to Connect to the AWS Cloud

### 1. AWS Client VPN

**AWS Client VPN** is a **managed VPN service** that allows **individual clients (users) to securely access AWS resources or on-premises networks** over the Internet.

**In short:**  
 *It’s for remote users to connect securely to AWS.*

**When to Use:**

* Employees need **remote access to AWS VPC resources** (EC2, RDS, S3).
* Securely connect **laptops or mobile devices** to AWS without exposing servers publicly.
* Works with **OpenVPN client or any compatible VPN client**.

**Example:**

* Your developer works from home and needs to access **internal EC2 instances** in a private subnet.
* Using **Client VPN**, they connect over an encrypted tunnel and access AWS securely.

Think of it as a **secure VPN for individual users**.

### 2. AWS Site-to-Site VPN

**AWS Site-to-Site VPN** is a **VPN connection between your on-premises network (corporate office) and your AWS VPC** over the Internet.

**In short:**  
 *It’s for connecting entire networks securely, not just individual clients.*

**When to Use:**

* Your office network needs **secure, encrypted connection to AWS**.
* Enables **AWS hybrid cloud setups**.
* Works with **customer gateways (on-premises VPN devices)**.

**Example:**

* Your company has a **data center in Mumbai**.
* Connect it to **AWS VPC** using Site-to-Site VPN.
* All office computers can access AWS services securely.

Think of it as a **private bridge between two networks**.

### 3. AWS PrivateLink

**AWS PrivateLink** allows you to **privately access AWS services or your own VPC services** **without using the public Internet**.

**In short:**  
 *It keeps traffic* ***inside the AWS network*** *for private access.*

**When to Use:**

* You want to **access AWS services privately** (like S3, API Gateway, or custom VPC endpoints) without going over the Internet.
* Useful for **enhanced security and compliance**.

**Example:**

* Your EC2 instance in a private subnet needs to access **S3 or an internal API**.
* Use **PrivateLink** → traffic stays in AWS network, never touches the public Internet.

Think of it as a **private tunnel inside AWS**.

### 4. AWS Direct Connect

**AWS Direct Connect** is a **dedicated, private network connection** between your on-premises data center and AWS.

**In short:**  
 *It provides a* ***high-speed, low-latency, private connection*** *to AWS.*

**When to Use:**

* Large-scale or high-bandwidth workloads need **consistent network performance**.
* You want to **bypass the public Internet** for better security and speed.
* Hybrid cloud setups where latency and reliability are critical.

**Example:**

* Your company runs a **data-heavy application** in AWS.
* Use **Direct Connect** → 1 Gbps or 10 Gbps link directly to AWS, ensuring low latency and secure connectivity.

Think of it as a **private dedicated highway to AWS**.

### Comparison Table (Exam-Friendly)

| **Service** | **What It Connects** | **Use Case** | **Internet Required?** | **Key Feature** |
| --- | --- | --- | --- | --- |
| **Client VPN** | Individual users → VPC | Remote employees | Yes | Secure remote access |
| **Site-to-Site VPN** | Office network → VPC | Hybrid cloud | Yes | Secure network-to-network connection |
| **PrivateLink** | VPC → AWS service or another VPC | Private service access | No | Keeps traffic inside AWS network |
| **Direct Connect** | On-prem → AWS | High-bandwidth, low-latency | No | Dedicated private connection |

### Memory Tips / Analogy:

* **Client VPN** = Your laptop connects to AWS from home
* **Site-to-Site VPN** = Your office network connects to AWS
* **PrivateLink** = Private hallway inside AWS between VPCs or services
* **Direct Connect** = Private, dedicated highway from office to AWS

## Subnets, Security Groups, and Network Access Control Lists

### How Network Traffic Works in a VPC

In a **VPC**, network traffic flows according to **subnets, route tables, and security rules**.

**In short:**  
 *Traffic goes in/out through route tables, controlled by security groups and network ACLs.*

**Key Points:**

1. **Subnets:**
   * Divide VPC into smaller networks.
   * Public subnets route traffic to **Internet Gateway (IGW)**.
   * Private subnets route traffic to **NAT Gateway** for Internet access.
2. **Route Tables:**
   * Define **where network traffic should go** (e.g., Internet, VPN, other subnets).
3. **Traffic Flow:**
   * **Inbound traffic:** Comes into subnet, filtered by **NACLs → Security Groups → instance**
   * **Outbound traffic:** Leaves instance, filtered by **Security Groups → NACLs → route**
4. **Internet & VPN Connections:**
   * IGW for public Internet
   * VGW for Site-to-Site VPN
   * PrivateLink for private access

Think of it like a **city traffic system**: roads (routes), traffic lights (ACLs), and security gates (security groups).

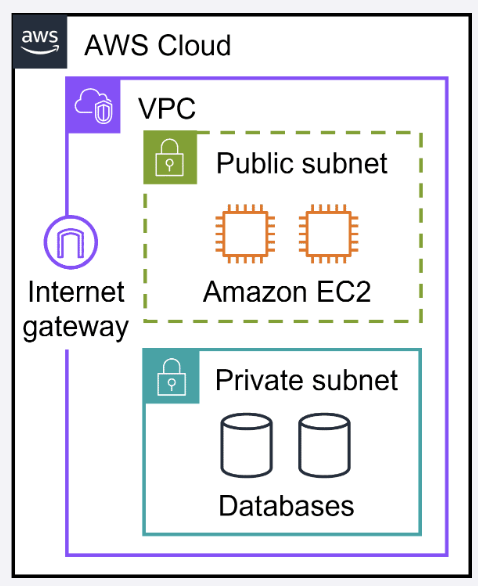
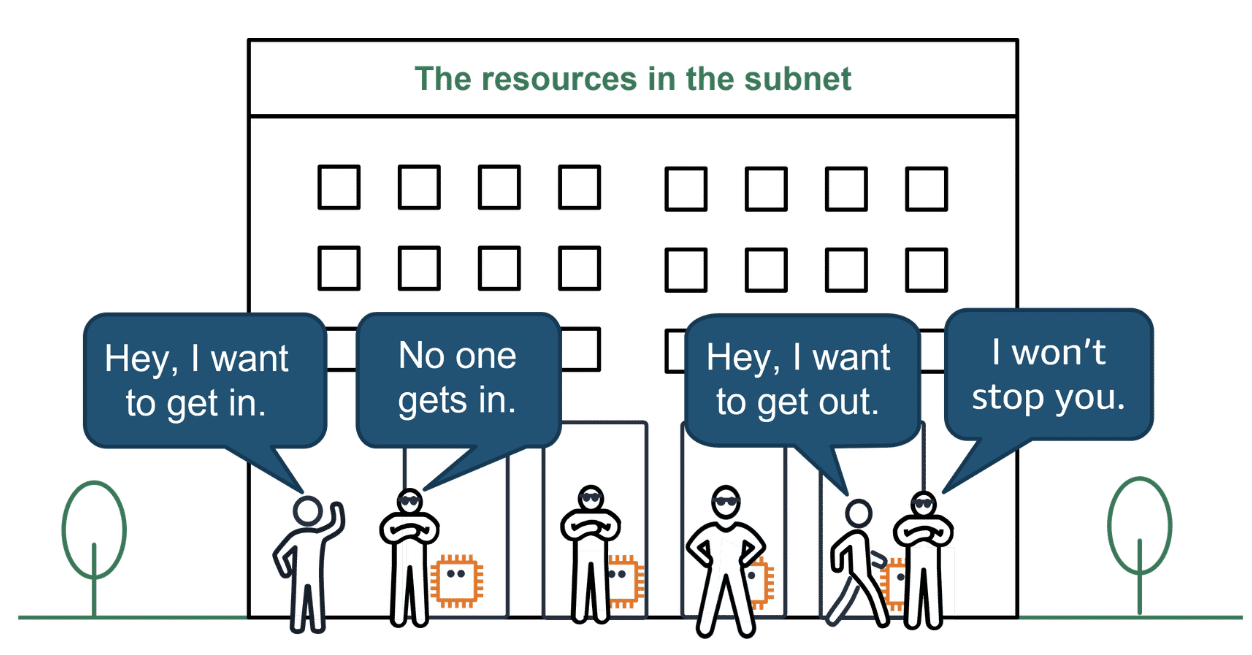
### Subnets

A **subnet** is a section of a VPC in which you can group resources based on security or operational needs. Subnets can be public or private.

**Public subnets**contain resources that need to be accessible by the public, such as an online store’s website.

**Private subnets** contain resources that should be accessible only through your private network, such as a database that contains customers’ personal information and order histories.

In a VPC, you can define rules to allow resources in different subnets to communicate with each other. For example, you might have an application that uses Amazon EC2 instances in a public subnet communicating with databases that are located in a private subnet.

### Security Groups (Stateful)

A **security group** is a **virtual firewall for your EC2 instance** that controls **inbound and outbound traffic**.

**Stateful:**

* If you **allow inbound traffic**, the **response traffic is automatically allowed**, no need to create a separate rule for outgoing traffic.

**Key Points:**

* Associated **with EC2 instances** (or other resources like RDS).
* Controls **IP, protocol (TCP/UDP), and port**.
* Default **deny all inbound**, **allow all outbound**.
* Stateful = automatic response traffic allowed.

Think of it as **a personal security guard at the instance door**.

### Network ACLs (Stateless)

**Network Access Control Lists (NACLs)** are **firewalls at the subnet level** that control **inbound and outbound traffic**.

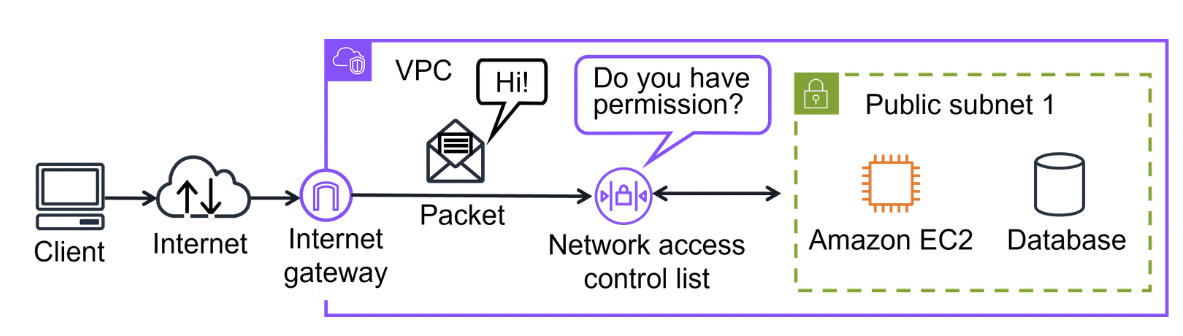
**Stateless:**

* If you **allow inbound traffic**, you **must also explicitly allow outbound traffic**.
* Traffic rules **apply to the entire subnet**, not individual instances.

**Key Points:**

* Associated with **subnets**, not instances.
* Evaluates rules **in order** (numbered rules).
* Default NACL **allows all inbound and outbound traffic**.
* Stateless = responses must have explicit rules.

Think of it as **a security gate for the entire neighborhood (subnet)**.



### Responsibility According to AWS Shared Responsibility Model

| **Component** | **Responsibility** | **Notes** |
| --- | --- | --- |
| **Security Groups** | Customer | You define which instance traffic is allowed |
| **Network ACLs** | Customer | You configure subnet-level traffic rules |
| **AWS** | Provides the infrastructure and network security (VPC itself) | Protects the underlying network, hypervisors, physical security |

**Quick Comparison Table: Security Groups vs NACLs**

| **Feature** | **Security Group** | **Network ACL** |
| --- | --- | --- |
| **Level** | Instance | Subnet |
| **Stateful / Stateless** | Stateful | Stateless |
| **Inbound & Outbound** | Automatically allows return traffic | Must allow return traffic explicitly |
| **Default Rule** | Deny all inbound, allow all outbound | Allow all inbound/outbound |
| **Use Case** | Control access per instance | Control traffic for entire subnet |

### Simple Analogy

| **Component** | **Analogy** |
| --- | --- |
| Security Group | Security guard at your house (instance) |
| NACL | Security guard at the neighborhood gate (subnet) |
| Route Table | City map showing where traffic should go |
| IGW/VGW | Main highway / bridge for entering/exiting city |

## Global Networking

### Domain Name System (DNS)

The **Domain Name System (DNS)** is like the **phonebook of the Internet**.  
It **translates human-readable domain names** (like www.amazon.com) into **IP addresses** (like 176.32.103.205) that computers use to communicate.

**In short:**  
 *DNS = Converts names → IP addresses so devices can find each other.*

**What DNS Does:**

1. **Name Resolution:**
   * Converts **domain names** into **IP addresses** so computers and servers can communicate.
2. **Routing Traffic:**
   * Directs user requests to the **correct server** based on the domain name.
3. **Load Balancing (Optional):**
   * Can distribute traffic among multiple servers for **high availability**.
4. **Ease of Use:**
   * Humans remember names; machines need numbers. DNS bridges this gap.

Think of it as a **GPS for the Internet** — it tells your computer exactly where to go.

### Amazon Route 53

**Amazon Route 53** is a **highly available and scalable DNS web service** that **routes Internet traffic to your AWS resources or external endpoints**.

**In short:**  
 *Route 53 = AWS’s smart DNS service that directs traffic reliably.*

**Key Benefits of Route 53**

| **Benefit** | **Explanation** | **Example / Analogy** |
| --- | --- | --- |
| **High Availability** | Built with **redundant global servers** for continuous operation | Your website stays reachable even if one server fails |
| **Scalability** | Can handle **millions of requests per second** | Supports global traffic spikes during a big sale |
| **Domain Registration** | You can **buy and manage domain names** directly | Register mybusiness.com in AWS |
| **Traffic Routing** | Supports **latency-based, geolocation, weighted, and failover routing** | Users in India routed to Mumbai server, users in US routed to Virginia server |
| **Health Checks & Failover** | Automatically **routes traffic away from unhealthy endpoints** | If your EC2 instance fails, traffic moves to a healthy instance |
| **Integration with AWS Services** | Works with **S3, CloudFront, ELB, EC2**, etc. | Route a custom domain to your S3 static website or ELB load balancer |
| **Private Hosted Zones** | Route traffic within your **VPC privately** | Internal corporate apps only accessible within AWS network |

**Simple Analogy**

Think of **Route 53 as a smart traffic controller for the Internet** 🛣️:

* **Public hosted zone** = Streets leading to your public buildings (website)
* **Private hosted zone** = Roads inside your private campus (VPC internal apps)
* **Routing policies** = Decide which road each visitor takes for fastest or safest route
* **Health checks** = Close blocked roads and redirect traffic automatically

### AWS Global Accelerator

**AWS Global Accelerator** is a **network service that improves the availability and performance of your applications for global users** by directing traffic through the **AWS global network**.

**In short:**  
 *Global Accelerator = Smart traffic manager that routes users to the nearest healthy endpoint globally.*

**How It Works:**

1. **Users connect to static Anycast IP addresses** provided by Global Accelerator.
2. Global Accelerator automatically **routes traffic to the nearest healthy AWS endpoint** (like EC2, ELB, or S3).
3. Traffic travels over the **AWS global network**, bypassing congested Internet paths.
4. Automatically **fails over to healthy endpoints** if one fails.

**Key Benefits**

| **Benefit** | **Explanation** | **Example / Analogy** |
| --- | --- | --- |
| **Improved Performance** | Routes traffic over **AWS backbone** instead of public Internet | Faster response for users worldwide |
| **Global Availability** | Directs users to **healthy endpoints** automatically | If one region fails, traffic is routed to another |
| **Static IP Addresses** | Provides **fixed IPs** for your app globally | No need to update DNS every time endpoints change |
| **Seamless Failover** | Detects unhealthy endpoints and **redirects traffic instantly** | Ensures high availability |
| **Easy Integration** | Works with **ALB, NLB, EC2, S3** | Global apps like e-commerce or SaaS benefit |

**Exam-Friendly Use Cases**

| **Use Case** | **Description** |
| --- | --- |
| **Global Applications** | Users worldwide access your application with lower latency |
| **Disaster Recovery** | Automatically failover to healthy regions if one region goes down |
| **Static IP Requirements** | Apps need fixed IPs for whitelisting in firewalls |
| **Multi-Region Web Apps** | Route traffic intelligently to the fastest or nearest endpoint |

**Simple Analogy**

Think of **Global Accelerator as an international airport network**:

* Users = Passengers
* AWS endpoints = Airports around the world
* Global Accelerator = Air traffic control
* Directs passengers to the **nearest, available, and fastest airport**, avoiding delays or closed airports

Faster, smarter, and more reliable than regular Internet routing (like CloudFront focuses on content delivery, Global Accelerator focuses on **applications**).

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Directs users to the nearest healthy endpoint globally” | **Global Accelerator** |
| “Provides static Anycast IPs” | **Global Accelerator** |
| “Improves availability and performance of multi-region applications” | **Global Accelerator** |
| “Works at the application level, not content delivery” | **Global Accelerator** |

### CloudFront vs Route 53 vs Global Accelerator

| **Feature** | **CloudFront** | **Route 53** | **Global Accelerator** |
| --- | --- | --- | --- |
| **Type of Service** | Content Delivery Network (CDN) | DNS service | Global traffic manager / Application Accelerator |
| **Primary Purpose** | Deliver web content, videos, APIs **fast to users globally** | Translate **domain names → IP addresses** and route traffic | Improve **application performance & availability** for global users |
| **Focus** | Content caching & low latency | DNS resolution & routing | Low latency & high availability for apps |
| **Traffic Routing** | Sends content from **nearest edge location** | Routes users based on **DNS queries** (latency, geolocation, weighted) | Sends traffic to **nearest healthy endpoint** over AWS global network |
| **Static IPs** | No | No | Yes, Anycast IPs |
| **Use Case** | Websites, videos, APIs, static & dynamic content | Domain registration, DNS routing, health checks | Multi-region apps, disaster recovery, low-latency global apps |
| **Integration** | S3, EC2, ELB, Lambda, API Gateway | Any AWS resource, external IP | ALB, NLB, EC2, S3 |
| **Failover** | Via cached content & origin failover | Yes, using health checks | Automatic routing to healthy endpoints |
| **Internet Dependency** | Yes, delivers content over Internet | Yes, resolves DNS queries | Uses AWS global backbone to bypass public Internet |

# Storage

## Types of Storage: Block, Object, and File Storage

### 1. Block Storage

Block storage stores data in **fixed-size blocks (chunks)**, similar to a **hard drive**.  
Each block acts like a small piece of a file system and can be managed or formatted by the operating system.

**In short:**  
 *Block storage = Like your computer’s disk drive — data stored in chunks (blocks) with fast access.*

**Key Features:**

* Data stored in **blocks** with unique addresses.
* Very **fast and low-latency**.
* Ideal for **databases, operating systems, and applications** that need frequent read/write.
* Accessed as a **mounted disk** to an EC2 instance.
* **Not shared** between multiple instances directly.

**Best for:** Databases (MySQL, Oracle), boot volumes, transactional workloads.

### 2. Object Storage

Object storage stores data as **objects** (not files or blocks).  
Each object includes the **data**, **metadata**, and a **unique identifier** (key).

**In short:**  
 *Object storage = Store and retrieve files (objects) using a web API — perfect for scalability and durability.*

**Key Features:**

* Data stored as **objects** in **buckets**.
* Highly **scalable** and **durable (11 nines — 99.999999999%)**.
* Ideal for **backups, images, videos, logs, and large unstructured data**.
* Accessed via **HTTP/HTTPS API**, not as a mounted drive.
* Supports **versioning** and **lifecycle policies**.

**Best for:** Backups, static website hosting, data lakes, multimedia storage.

**3. File Storage**

File storage organizes data in a **hierarchical file and folder structure**, like your computer’s file system.

**In short:**  
 *File storage = Shared file system accessed by multiple users or servers over a network.*

**Key Features:**

* Uses **folders, directories, and file paths**.
* Supports **file-level permissions (NFS, SMB protocols)**.
* Can be **shared among multiple EC2 instances**.
* Suitable for **content management, shared work environments, home directories**.

**Best for:** Shared storage, file servers, content management systems, development environments.

### Quick Comparison Table

| **Feature** | **Block Storage (EBS)** | **Object Storage (S3)** | **File Storage (EFS/FSx)** |
| --- | --- | --- | --- |
| **Data Format** | Blocks (fixed-size chunks) | Objects (data + metadata) | Files (hierarchical folders) |
| **Access Method** | Attached to instance (mount) | Access via API (HTTP/HTTPS) | Shared via network (NFS/SMB) |
| **Scalability** | Limited to instance | Highly scalable | Scalable (limited compared to S3) |
| **Performance** | Very high, low latency | Moderate, depends on access | Moderate to high |
| **Use Case** | Databases, OS, transactional apps | Backups, media, data lakes | Shared file systems, app data |
| **AWS Service** | EBS | S3 | EFS / FSx |

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Stores data as objects in buckets” | **S3 (Object Storage)** |
| “Used for EC2 root volume or databases” | **EBS (Block Storage)** |
| “Shared file system across instances” | **EFS / FSx (File Storage)** |
| “Most scalable and durable storage type” | **Object Storage (S3)** |

## Amazon EC2 Storage Options: Instance Store vs EBS

Both are **block storage** types used to store data for **EC2 instances**, but they differ in **persistence, durability, and use cases**.

### 1. Amazon EC2 Instance Store

**Instance Store** provides **temporary block-level storage** that is **physically attached to the EC2 host (the actual hardware server)**.

**In short:**  
 *Instance Store = Temporary storage directly on the EC2 host — fast but data is lost if instance stops or fails.*

**Key Features / Benefits:**

| **Feature** | **Description** |
| --- | --- |
| **High Performance** | Data stored on physical disks attached to the host → very fast read/write speeds. |
| **Low Latency** | Since it’s local to the instance, performance is higher than network storage. |
| **No Extra Cost** | Included in the EC2 price (no separate EBS charge). |
| **Ephemeral Storage** | Data is **deleted when the instance stops, terminates, or fails**. |
| **Best for Temporary Data** | Ideal for caches, buffers, or temporary files that can be recreated easily. |

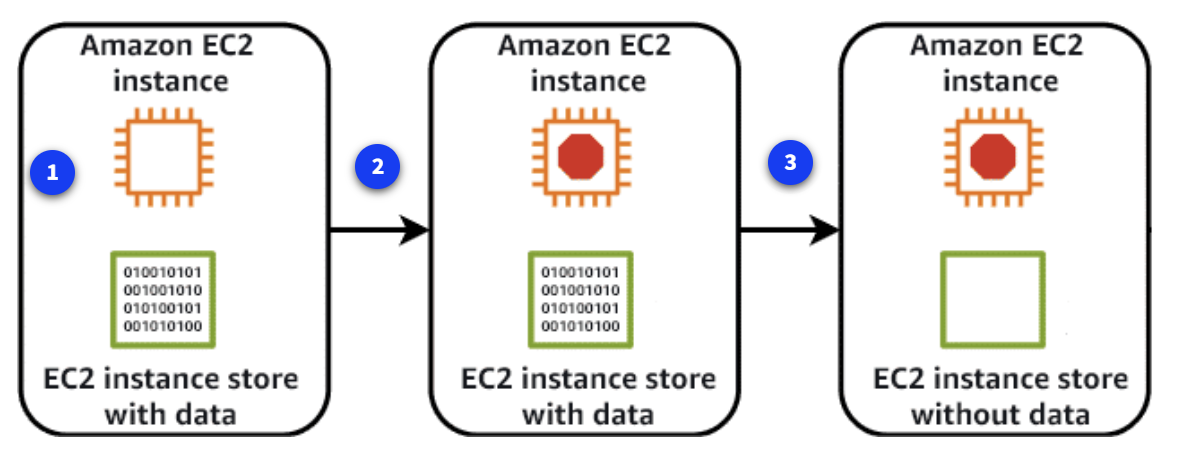
**Use Cases:**

* Temporary data storage (like cache, scratch space, or buffers).
* High-performance workloads that process large data sets (like big data analytics).
* Applications that **replicate data elsewhere** (so loss isn’t critical).

**Important Exam Note:**

Data on instance store **is lost when the EC2 instance stops or terminates.**  
Only **EBS** provides persistent storage.

**Best For:** Temporary, high-speed local storage needs (e.g., cache, batch processing).



### 2. Amazon EBS (Elastic Block Store)

**Amazon EBS** provides **persistent, high-availability block storage volumes** that can be **attached to EC2 instances**.

**In short:**  
 *EBS = Persistent storage for EC2 — data stays even if instance stops.*

**Key Features / Benefits:**

| **Feature** | **Description** |
| --- | --- |
| **Persistent Storage** | Data remains even after instance stop or termination (if volume detached properly). |
| **High Availability** | Data replicated automatically within the Availability Zone (AZ). |
| **Backup & Snapshots** | You can take **snapshots to S3** for backup or creating new volumes. |
| **Flexible & Scalable** | Resize volumes or change performance type anytime. |
| **Attach/Detach Easily** | Volumes can be attached to or detached from EC2 instances. |
| **Multiple Volume Types** | Choose between SSD (fast) or HDD (throughput-optimized). |

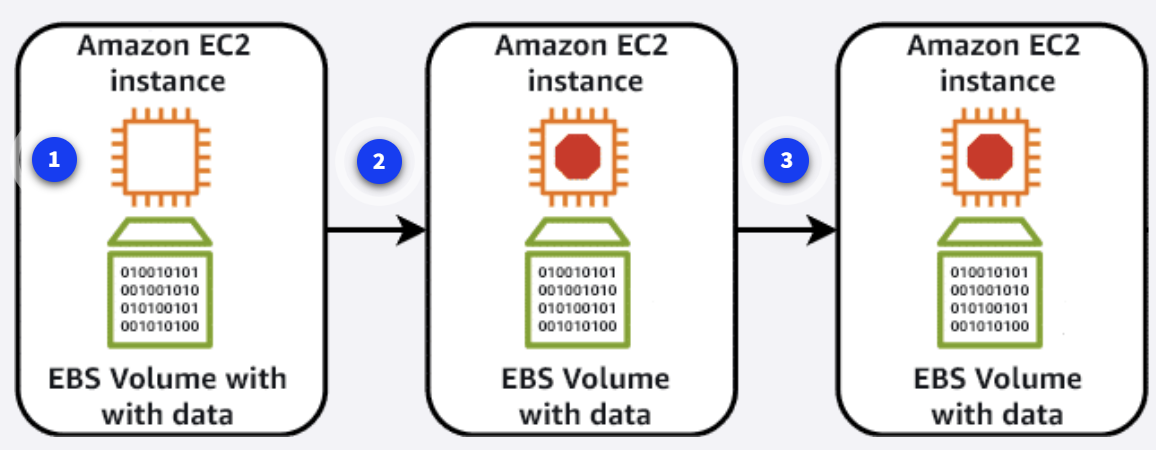
**Use Cases:**

* EC2 root volume (boot disk).
* Databases that need consistent, fast I/O performance.
* Application data that must **persist between instance stops/restarts**.
* Backup and restore using EBS snapshots.

**Important Exam Note:**

**EBS volumes are AZ-specific**, but you can create snapshots and **restore them in another AZ or region.**

**Best For:** Databases, OS storage, long-term data, or applications needing data persistence.



### Quick Comparison: Instance Store vs EBS

| **Feature** | **Instance Store** | **EBS (Elastic Block Store)** |
| --- | --- | --- |
| **Persistence** | Temporary (data lost on stop/terminate) | Persistent (data retained) |
| **Performance** | Very high (local disk) | High (network attached) |
| **Durability** | Low | High (replicated within AZ) |
| **Cost** | Included with instance | Charged separately |
| **Backup** | No backup | Snapshots to S3 supported |
| **Attach/Detach** | Fixed to one instance | Can attach/detach freely |
| **Best For** | Temporary or cache data | Databases, OS, app data |

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Data lost when instance stops or terminates” | **Instance Store** |
| “Persistent block storage for EC2” | **EBS** |
| “Can take snapshots to S3” | **EBS** |
| “Used for temporary, high-speed data” | **Instance Store** |
| “Automatically replicated within an AZ” | **EBS** |

## Amazon EBS Snapshots & Data Lifecycle Management

### 1. What are Amazon EBS Snapshots?

An **EBS Snapshot** is a **backup (point-in-time copy)** of an Amazon EBS volume that is **stored in Amazon S3**.

**In short:**  
 *Snapshots = Backups of your EBS volumes stored in S3.*

**Key Features:**

| **Feature** | **Description** |
| --- | --- |
| **Incremental Backups** | Only changes since the last snapshot are saved → saves time and storage. |
| **Durable Storage** | Snapshots are stored in Amazon S3, which is highly durable (99.999999999%). |
| **Easy Restore** | You can create a new EBS volume from a snapshot at any time. |
| **Cross-Region Copy** | Snapshots can be copied to another AWS Region for disaster recovery. |
| **Automation Support** | Snapshots can be automated using **Amazon Data Lifecycle Manager (DLM)**. |

**Common Use Cases:**

| **Use Case** | **Description** |
| --- | --- |
| **Backup & Recovery** | Protect data from loss or corruption by taking regular backups. |
| **Disaster Recovery (DR)** | Copy snapshots to another Region for DR planning. |
| **Data Migration** | Create volumes in another Region or AZ from a snapshot. |
| **Version Control** | Keep multiple snapshots as different “versions” of your data. |

**Example:**  
You have an EBS volume attached to an EC2 instance running a database.  
You take daily **snapshots** so if data is lost or corrupted, you can **restore from the snapshot** quickly.

### 2. EBS Data Lifecycle & Integration with AWS Services

The **EBS Data Lifecycle** describes how data moves through stages — from **creation → snapshot → backup → archive/deletion** — using automation and integrations.

**EBS Data Lifecycle Stages:**

| **Stage** | **Description** |
| --- | --- |
| **Create Volume** | You create an EBS volume and attach it to an EC2 instance. |
| **Take Snapshot** | You take a backup (manual or automated via DLM). |
| **Store Snapshot in S3** | Snapshot is stored in durable Amazon S3 automatically (you don’t see it in your S3 bucket, AWS manages it). |
| **Manage Lifecycle** | Use **Amazon Data Lifecycle Manager (DLM)** to create policies for automated snapshot creation and deletion. |
| **Restore Volume** | You can create new volumes from snapshots when needed — even in other AZs or Regions. |
| **Copy or Share Snapshots** | You can share snapshots with other accounts or copy them across Regions. |

**Integration with AWS Services:**

| **AWS Service** | **Integration with Snapshots** |
| --- | --- |
| **Amazon S3** | Snapshots are stored in S3 (AWS-managed). |
| **Amazon EC2** | Snapshots used to create new EBS volumes for EC2 instances. |
| **AWS Backup** | Centralized service for managing EBS snapshot backups. |
| **Amazon DLM (Data Lifecycle Manager)** | Automates creation, retention, and deletion of snapshots. |
| **AWS IAM** | Controls who can create, share, or delete snapshots. |

**Example:**  
Using **DLM**, you can automatically take a snapshot of a production EC2 volume **every 12 hours** and **delete snapshots older than 7 days** — no manual work needed.

### 3. Customer Responsibility (Shared Responsibility Model)

AWS manages **the infrastructure** that keeps snapshots secure and durable.  
**You (the customer)** are responsible for **how and when you use them.**

**Division of Responsibility:**

| **Responsibility** | **AWS** | **Customer** |
| --- | --- | --- |
| **Infrastructure durability** (S3 storage, replication, availability) | Yes | ❌ No |
| **Snapshot creation, scheduling, deletion** | ❌ No | Yes |
| **Setting DLM policies** | ❌ No | Yes |
| **Data encryption management (KMS keys)** | Shared (depends on who manages KMS key) | Shared |
| **Access control (IAM permissions)** | ❌ No | Yes |
| **Cross-region copies and sharing snapshots** | ❌ No | Yes |

**In Short:**

AWS ensures **snapshots are stored safely** in S3,  
but **you must manage when to take, delete, share, or encrypt them.**

### Amazon Data Lifecycle Manager (DLM):

* AWS tool that **automates snapshot lifecycle** (creation → retention → deletion).
* Helps reduce storage costs by removing old snapshots automatically.
* You set **policies** like:
  + “Create a snapshot every 24 hours.”
  + “Retain last 7 snapshots only.”
  + “Delete snapshots older than 30 days.”

**Exam Tips:**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Backup of EBS volume stored in S3” | **EBS Snapshot** |
| “Only changes since last backup are saved” | **Incremental snapshot** |
| “Automates snapshot creation and deletion” | **Data Lifecycle Manager (DLM)** |
| “Who manages snapshot retention?” | **Customer** |
| “Where are snapshots stored?” | **Amazon S3 (AWS-managed)** |
| “Used for disaster recovery or backup” | **EBS Snapshots** |

## Amazon S3 (Simple Storage Service)

**Amazon S3** is an **object storage service** that lets you store and retrieve **any amount of data** from **anywhere on the web**.

Each object typically includes the *data*itself, *metadata*, and a unique identifier, or *key*. Objects can be of any file type, such as images, videos, documents, or application data, and can range in size from a few bytes to several terabytes.

In short:  
 *S3 = Scalable, durable, and secure cloud storage for files (objects).*

**Key Features / Benefits**

| **Feature** | **Description** |
| --- | --- |
| **Object Storage** | Stores data as *objects* (file + metadata) in *buckets* (like folders). |
| **Scalable** | Can store unlimited data — scales automatically. |
| **Highly Durable** | 99.999999999% (11 nines) durability — data is safe. |
| **Highly Available** | Data stored across multiple Availability Zones (AZs). |
| **Cost-Effective** | Pay only for what you use. Multiple storage classes for cost savings. |
| **Secure by Design** | Encryption, access control, and logging built-in. |
| **Easy Integration** | Works with many AWS services — EC2, CloudFront, Lambda, Athena, etc. |
| **Versioning Support** | Keeps multiple versions of an object for recovery or rollback. |

**Common Use Cases**

| **Use Case** | **Description** |
| --- | --- |
| **Backup and Restore** | Store backups, snapshots, and archives safely and durably. |
| **Static Website Hosting** | Host static websites (HTML, CSS, images) directly from S3. |
| **Data Lake / Analytics** | Store large raw datasets for analytics (e.g., AWS Athena, Redshift). |
| **Media Storage & Distribution** | Store videos, photos, and serve them globally via CloudFront. |
| **Disaster Recovery (DR)** | Keep copies of critical data in multiple regions. |
| **Application Data Storage** | Apps use S3 to store user uploads, logs, etc. |

**Integration with Other AWS Services**

| **AWS Service** | **Integration** |
| --- | --- |
| **CloudFront** | Distribute S3 content globally with low latency. |
| **Lambda** | Trigger code execution when new files are uploaded. |
| **Athena** | Query data directly in S3 using SQL. |
| **Backup / DLM** | Manage EBS and EC2 backups that are stored in S3. |
| **Glacier / S3 Glacier Deep Archive** | Store infrequently accessed data at lower cost. |

## Amazon S3 Storage Classes and S3 Lifecycle

### 1. S3 Standard (General Purpose)

Default S3 class for **frequently accessed** data.

**Key Features:**

* **Durability:** 99.999999999% (11 nines)
* **Availability:** 99.99%
* **Stored in:** Multiple Availability Zones (AZs)
* **Retrieval time:** Instant

**Use Cases:**

* Active websites, mobile apps
* Frequently accessed files or media
* Big data analytics

**Tip:** If you don’t choose a class, AWS uses **S3 Standard by default**.

### 2. S3 Intelligent-Tiering

Automatically moves objects between access tiers **based on usage** to optimize cost.

**Key Features:**

* Moves data between tiers automatically:
  + **Frequent Access Tier**
  + **Infrequent Access Tier**
  + **Archive Instant Access Tier**
  + **Archive Access Tier**
  + **Deep Archive Access Tier**
* **No retrieval fees**
* **Monitoring charge per object** (small cost)

**Use Cases:**

* Data with **unpredictable access patterns**
* Data lakes, analytics, long-lived data

**Tip:** Best for when you **don’t know** how often your data will be accessed.

### 3. S3 Standard-IA (Infrequent Access)

For data that is **accessed less often**, but needs to be **available immediately**.

**Key Features:**

* **Retrieval Time:** Instant
* **Lower cost** than Standard
* **Retrieval fee** per GB

**Use Cases:**

* Backups, disaster recovery
* Long-term storage for infrequently used data

**Tip:** Cheap storage for data you rarely read, but can’t afford to lose.

### 4. S3 One Zone-IA

Like Standard-IA, but data stored in **only one AZ** (not multi-AZ).

**Key Features:**

* **Stored in:** One AZ only
* **Lowest cost among instant-access classes**
* **Retrieval fee** per GB

**Use Cases:**

* Non-critical data
* Secondary backups
* Re-creatable data sets

**Tip:** Cheapest option if you can tolerate losing data in case that AZ fails.

### 5. S3 Glacier Instant Retrieval

For **archived data** that’s **rarely accessed**, but when needed, **must be available immediately**.

**Key Features:**

* **Retrieval Time:** Milliseconds
* **Very low cost** storage
* **Retrieval fee applies**

**Use Cases:**

* Medical images, archives that may need fast access
* Compliance or reference data

**Tip:** “Archive storage that’s instantly available.”

### 6. S3 Glacier Flexible Retrieval

(Previously **S3 Glacier**)

For long-term archives that **can wait minutes or hours** for retrieval.

**Key Features:**

* **Retrieval Time:**
  + Expedited: 1–5 minutes
  + Standard: 3–5 hours
  + Bulk: 5–12 hours
* **Very low cost**

**Use Cases:**

* Backups
* Regulatory archives
* Old project files

**Tip:** Use if you can **wait a few hours** to get the data back.

### 7. S3 Glacier Deep Archive

**Lowest-cost** S3 storage — for **very long-term archives** rarely accessed.

**Key Features:**

* **Retrieval Time:**
  + Standard: 12 hours
  + Bulk: 48 hours
* **Cheapest storage in S3**

**Use Cases:**

* Compliance archives (e.g., 7+ year retention)
* Old research data
* Government records

**Tip:** Use if you **rarely or never** need to retrieve data.

### 8. S3 Express One Zone *(NEW — 2023)*

**High-performance storage** for latency-sensitive applications; stored in **one AZ**.

**Key Features:**

* **Stored in:** Single AZ
* **Latency:** Millisecond-level — fastest among all classes
* **Cost:** Higher than Standard

**Use Cases:**

* AI/ML workloads
* Real-time analytics
* Data processing pipelines

**Tip:** “Fast like local storage, but still S3.”

### Quick Comparison Table

| **Storage Class** | **Access Frequency** | **Stored In** | **Retrieval Time** | **Cost** | **Use Case** |
| --- | --- | --- | --- | --- | --- |
| **S3 Standard** | Frequent | Multi-AZ | Instant | 💰💰 | Websites, active data |
| **Intelligent-Tiering** | Variable | Multi-AZ | Instant | 💰💰 | Unpredictable access |
| **Standard-IA** | Infrequent | Multi-AZ | Instant | 💰 | Backups, DR |
| **One Zone-IA** | Infrequent | One AZ | Instant | 💰 | Re-creatable data |
| **Glacier Instant Retrieval** | Rare | Multi-AZ | Instant | 💵 | Archives needing fast access |
| **Glacier Flexible Retrieval** | Rare | Multi-AZ | 1 min–12 hrs | 💵 | Archive data |
| **Glacier Deep Archive** | Very Rare | Multi-AZ | 12–48 hrs | 💵 (cheapest) | Compliance archives |
| **Express One Zone** | Frequent | One AZ | Milliseconds | 💰💰💰 | AI/ML, analytics |
| **RRS (Legacy)** | Rare | Multi-AZ | Instant | 💵 | Non-critical, deprecated |

**Exam Memory Trick:**

“Hot → Cold → Frozen”  
**Standard → IA → Glacier → Deep Archive**

And remember:

**Multi-AZ = More durable, safer.**  
**One Zone = Cheaper, less durable.**

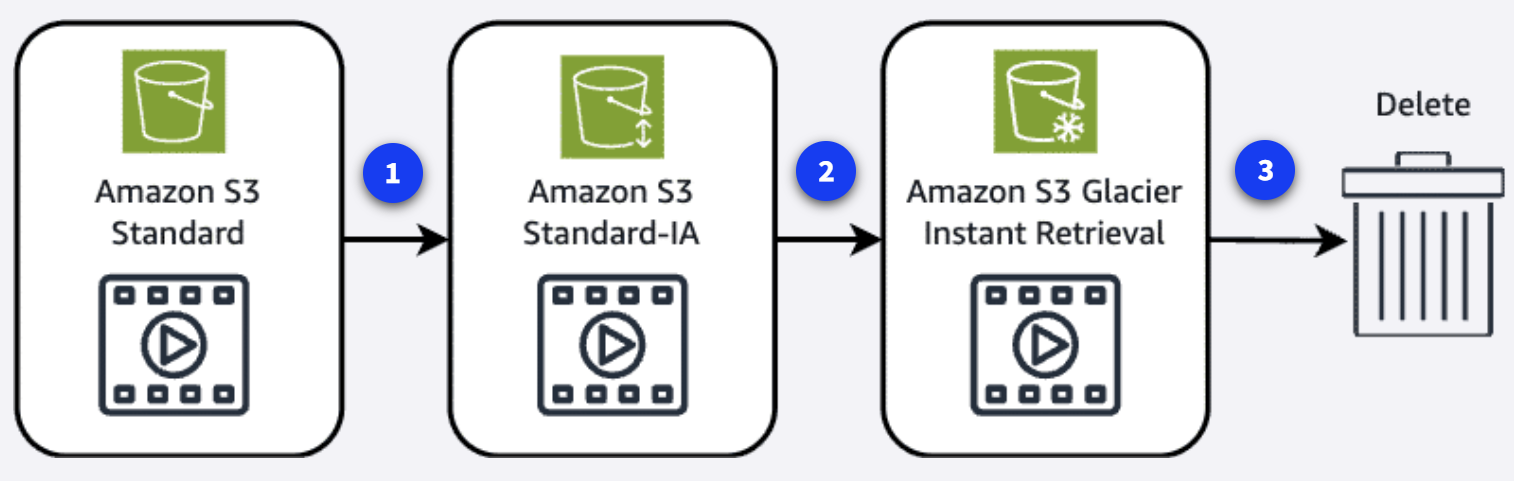
## S3 Lifecycle

To avoid manually managing your object storage tier configurations, you can use S3 Lifecycle configurations to automate the process. When you define a lifecycle configuration for an object or group of objects, you can choose to automate between two types of actions, as follows:

* *Transition actions:* define when objects should transition to another storage class.
* *Expiration actions:* define when objects expire and should be permanently deleted.

For example, you might transition objects to S3 Standard-IA storage class 30 days after you create them. Or you might archive objects to the S3 Glacier Deep Archive storage class 1 year after creating them.

To learn more about an example S3 Lifecycle configuration, choose each of the three numbered markers.



**Use cases**

The following situations are candidates for the use of S3 lifecycle configuration rules:

* **Periodic logs:** If you upload periodic logs to a bucket, your application might need them for a week or a month. After that, you might want to delete them.
* **Data that changes in access frequency:** Some documents are frequently accessed for a limited period of time. After that, they are infrequently accessed. At some point, you might not need real-time access to them. However, your organization or regulations might require you to archive them for a specific period. After that, you can delete them.

## Amazon EFS (Elastic File System)

**Amazon EFS** is a **fully managed, scalable, shared file storage service** that can be **mounted on multiple EC2 instances** at the same time.

**In short:**  
 *EFS = Shared “network drive” for multiple EC2 instances.*

**How It Works:**

* It provides **Network File System (NFS)** storage, accessible across multiple EC2 instances.
* Storage **automatically grows and shrinks** as you add or remove files.
* Can be used by **multiple instances across multiple Availability Zones (AZs)** in a Region.
* You **don’t need to manage servers or file systems** — AWS handles it.

**Key Benefits of Amazon EFS**

| **Benefit** | **Description** |
| --- | --- |
| **Fully Managed** | No need to provision or maintain file servers. AWS handles scaling and maintenance. |
| **Scalable** | Automatically grows and shrinks as you add/remove files (no need to set capacity). |
| **Shared Access** | Multiple EC2 instances can access the same files simultaneously. |
| **High Availability & Durability** | Data is stored across multiple AZs in a Region. |
| **Elastic Performance** | Supports thousands of concurrent connections. |
| **Integrated with AWS Services** | Works with EC2, Lambda, EKS, ECS, and more. |
| **Secure** | Supports encryption (at rest and in transit) and IAM access control. |

## EFS Storage Classes

| **Storage Class** | **Description** | **Best For** |
| --- | --- | --- |
| **EFS Standard** | Multi-AZ, high availability and durability | Frequently accessed data |
| **EFS Infrequent Access (IA)** | Lower-cost tier for rarely accessed files | Backup or archive data |

You can enable **Lifecycle Management** to automatically move files between **Standard** and **IA** to save cost.

**Amazon EFS benefits**

* **Multi-AZ redundancy:** Amazon EFS automatically replicates data across multiple Availability Zones in a region for high availability. This built-in redundancy protects against AZ failures and provides continuous access to your file systems.
* **Shared access:** Amazon EFS supports thousands of concurrent NFS connections, so multiple EC2 instances can access the same file system simultaneously. This shared access model makes EFS ideal for collaborative workloads and distributed applications.
* **Elastic storage:** Amazon EFS automatically grows and shrinks as you add and remove files, with no need to provision or manage storage capacity. And since you only pay for the storage you use, it's cost-effective for varying workload demands.

**EFS vs EBS vs S3 (Quick Comparison)**

| **Feature** | **EFS** | **EBS** | **S3** |
| --- | --- | --- | --- |
| **Storage Type** | File storage | Block storage | Object storage |
| **Access Type** | Multiple EC2 instances | One EC2 instance at a time | Access via APIs |
| **Scalability** | Auto-scales | Fixed size (manually resized) | Auto-scales |
| **Durability** | Multi-AZ | AZ-specific | Multi-AZ |
| **Use Case** | Shared file storage | Database, OS | Backups, static files |

**Tip for Exam:**  
If the question mentions **shared access** or **file system across multiple EC2s**, the correct answer is → **Amazon EFS**.

## Amazon FSx

**Amazon FSx** is a **fully managed file storage service** that provides **specialized file systems** built on **popular file system technologies** (like Windows File Server or Lustre).

In simple words:

**EFS** = for Linux (general shared storage)  
**FSx** = for specific file system needs like **Windows**, **Lustre (high performance)**, **NetApp ONTAP**, and **OpenZFS**

**What Amazon FSx Does**

* Provides **fully managed file systems** optimized for **different workloads**.
* You don’t need to install or manage file servers.
* Offers **Windows-compatible**, **high-performance**, or **enterprise** file systems.

### Types and When to Use Each

| **Scenario** | **Use This** |
| --- | --- |
| Your app runs on **Windows** and needs SMB file shares | **FSx for Windows File Server** |
| You need **high-speed file access** for compute workloads (AI/ML, HPC) | **FSx for Lustre** |
| You use **NetApp on-premises** and want hybrid or cloud storage | **FSx for NetApp ONTAP** |
| You use **Linux** and need advanced file features (snapshots, cloning) | **FSx for OpenZFS** |

**Key Benefits of Amazon FSx**

| **Benefit** | **Description** |
| --- | --- |
| **Fully Managed** | No need to set up or maintain file servers |
| **High Performance** | Designed for demanding workloads like HPC or ML |
| **Integration with AWS** | Works with EC2, S3, Lambda, and others |
| **Security** | Data encrypted at rest (KMS) and in transit |
| **Scalable & Reliable** | Automatically replicates data within AZs |
| **Hybrid Ready** | Integrates with on-prem systems (especially NetApp & Windows) |

### Comparison: EFS vs FSx

| **Feature** | **Amazon EFS** | **Amazon FSx** |
| --- | --- | --- |
| **OS Type** | Linux | Windows, Linux, or specialized |
| **Protocols** | NFS | SMB (Windows), Lustre, NFS |
| **Use Case** | General-purpose file storage | Specialized or enterprise file systems |
| **Performance** | Standard file performance | High or specialized performance |
| **Integration** | EC2 (Linux), Lambda | Windows, HPC, or enterprise workloads |
| **Managed By AWS** | Yes | Yes (fully managed) |

## AWS Storage Gateway

**AWS Storage Gateway** is a **hybrid cloud storage service** that connects **on-premises environments** (like your local data center or office) with **AWS Cloud storage**.

**In simple words:**

It acts as a **bridge** between your **on-prem storage** and **AWS cloud storage**.

**Purpose**

Many businesses still have on-prem servers but want to:

* Backup data to the cloud
* Use AWS storage services (like S3 or EBS)
* Gradually migrate to AWS

**AWS Storage Gateway** makes this possible — **smoothly and securely**.

**How It Works (Simple View)**

1. You **install a gateway** (as software or hardware) in your data center or office.
2. It connects securely to **AWS storage services** (like S3, EBS, Glacier).
3. It **syncs and caches** data between your on-prem system and AWS.

**Benefits of AWS Storage Gateway**

| **Benefit** | **Description** |
| --- | --- |
| **Hybrid Integration** | Seamlessly connects on-premises apps to AWS storage. |
| **Cost-Effective** | Uses cloud storage instead of expensive local hardware. |
| **Durable & Scalable** | Data stored in AWS (S3/Glacier) is highly durable and scalable. |
| **Secure** | Data is encrypted in transit and at rest. |
| **Low-Latency Access** | Local caching ensures quick access to frequently used data. |
| **Simple Migration** | Helps migrate data gradually to the cloud. |

### 1. File Gateway

* Acts like a **file server** (using NFS or SMB protocols).
* Files are stored in **Amazon S3** as objects.
* You can access data both **locally** and in the **AWS console**.
* Supports **file caching** for faster access.

**Use Case:**

* Cloud backups, file shares, data archiving.

**Think of it as:**

“Your local file system that secretly stores everything in S3.”

### 2. Volume Gateway

* Provides **block storage** volumes to on-prem apps (like iSCSI drives).
* Two operating modes:

| **Mode** | **Description** |
| --- | --- |
| **Cached Volume Mode** | Keeps most data in AWS and caches frequently used data locally. *(Cheaper and scalable)* |
| **Stored Volume Mode** | Keeps all data locally and asynchronously backs it up to AWS. *(Faster local performance)* |

**Use Case:**

* Database or application data that needs cloud backup or disaster recovery.

**Think of it as:**

“Your local hard drive backed up automatically to AWS.”

### 3. Tape Gateway

* Emulates a **physical tape library** (Virtual Tape Library — VTL).
* Works with existing **backup software** (like NetBackup, Veeam, etc.).
* Stores “virtual tapes” in **S3** or **Glacier Deep Archive** for long-term retention.

**Use Case:**

* Backup and archival replacement for traditional tape systems.

**Think of it as:**

“Your old tape backup, but now in the cloud.”

**Types of AWS Storage Gateways**

| **Type** | **What It Does** | **Best For / Use Case** | **Connects To** |
| --- | --- | --- | --- |
| **File Gateway** | Stores files in AWS S3 using SMB or NFS protocols | File shares, backups, archiving | Amazon S3 |
| **Volume Gateway** | Presents block storage volumes to on-prem apps | Database backups, disaster recovery | Amazon EBS snapshots (stored in S3) |
| **Tape Gateway** | Replaces physical backup tapes with virtual tapes in AWS | Backup and archive to cloud | Amazon S3 and Glacier |

**Exam Tips**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Hybrid storage between on-prem and AWS” | **AWS Storage Gateway** |
| “Access S3 files using NFS or SMB” | **File Gateway** |
| “Local disk with backup in AWS” | **Volume Gateway** |
| “Virtual tapes stored in S3 or Glacier” | **Tape Gateway** |
| “Caches frequently used data locally for fast access” | **File or Volume Gateway** |

**Integration with AWS Services**

| **AWS Service** | **How It’s Used** |
| --- | --- |
| **Amazon S3** | File and Tape Gateways store data here |
| **Amazon EBS Snapshots** | Volume Gateway uses these for backups |
| **Amazon Glacier / Deep Archive** | Tape Gateway stores long-term backups |
| **AWS Backup** | Centralized backup management for all Gateways |

**Real-World Analogy**

Imagine your company’s office has:

* A **file server** → You use **File Gateway** to store those files in S3.
* A **database server** → You use **Volume Gateway** to back it up to AWS.
* An **old tape backup system** → You use **Tape Gateway** to move it to Glacier.

## AWS Elastic Disaster Recovery (AWS DRS)

**AWS Elastic Disaster Recovery (DRS)** is a **fully managed service** that helps you **quickly recover your applications and servers** (from on-premises or other clouds) to **AWS** after an outage or disaster.

**In simple words:**

It continuously replicates your on-premises or cloud servers to AWS, so if something fails, you can **launch them instantly** in the AWS cloud.

**Purpose**

* To minimize **downtime** and **data loss** during disasters.
* To provide a **cost-effective, fast, and reliable** recovery solution.

**How It Works (Simple 4-Step Flow)**

1. **Install Agent**
   * You install a **lightweight agent** on your on-premises or cloud servers (physical or virtual).
2. **Continuous Replication**
   * The agent continuously replicates your data, applications, and configurations to a **low-cost staging area** in AWS (using EBS and S3).
3. **Failover (Disaster Happens)**
   * When a disaster occurs, you can **launch your replicated servers instantly** as **fully functional EC2 instances** in minutes.
4. **Failback**
   * Once your original environment is fixed, you can **replicate data back** from AWS to your local servers.

**Exam Tips**

| **Question Hint** | **Correct Answer** |
| --- | --- |
| “Service for recovering servers quickly to AWS after failure” | **AWS Elastic Disaster Recovery (DRS)** |
| “Continuously replicates servers from on-premises to AWS” | **AWS Elastic Disaster Recovery** |
| “Launch recovery instances in minutes during an outage” | **Elastic Disaster Recovery** |
| “Minimize downtime and data loss” | **Elastic Disaster Recovery** |
| “Replacement for AWS CloudEndure Disaster Recovery” | **Elastic Disaster Recovery** |

**Elastic Disaster Recovery vs Backup vs High Availability**

| **Feature** | **Elastic DR (DRS)** | **AWS Backup** | **High Availability (HA)** |
| --- | --- | --- | --- |
| **Purpose** | Recover entire systems after disaster | Back up data for long-term storage | Keep systems running continuously |
| **Downtime** | Minutes | Hours | Near-zero |
| **Replication** | Continuous | Periodic | Real-time |
| **Use Case** | Site failure or outage | Data protection | Live failover systems |

# Databases

## Amazon RDS (Relational Database Service)

**Amazon RDS** is a **fully managed relational database service** that makes it easy to **set up, operate, and scale** databases in AWS.  
AWS handles **backups, patching, and maintenance**, so you can just focus on using your database.

**In simple words:**

RDS = A cloud database that AWS manages for you.

**What It Does**

* Automatically **creates, runs, and maintains** relational databases.
* Supports **automatic backups** and **software patching**.
* Offers **scaling** (compute & storage).
* Provides **high availability** with **Multi-AZ** and **read replicas**.

**Key Points to Remember (for Exam)**

* RDS is for **relational databases (SQL-based)**.
* It’s a **managed service** — AWS handles **backups, patching, and failover**.
* Supports **6 database engines** (MySQL, PostgreSQL, MariaDB, Oracle, SQL Server, Aurora).
* **Multi-AZ** = High availability (automatic failover).
* **Read Replica** = Performance boost for read-heavy workloads.
* **Backups are automatic**, with **point-in-time recovery**.
* **Encryption** is optional but recommended (KMS).
* RDS runs **inside a VPC** for network security.
* You pay for **instance type, storage, and I/O**.
* For **high performance and cost efficiency**, AWS recommends **Aurora (RDS family)**.

## Amazon Aurora

Amazon Aurora is a **relational database** service that is **fast, reliable, and fully managed**. It works with **MySQL** and **PostgreSQL**, but performs much better than regular databases.

**Real-Life Analogy:**

Imagine a high-speed train (Aurora) that runs on tracks designed for regular trains (MySQL/PostgreSQL). It’s faster, safer, and maintained by AWS — you just buy a ticket and enjoy the ride.

**Use Case:**

* E-commerce websites needing fast and reliable databases.
* Applications with high traffic and need for quick data access.
* Replacing traditional MySQL/PostgreSQL with a more scalable solution.

**Key Points to Remember:**

* Aurora is **compatible with MySQL and PostgreSQL**.
* It is **5x faster than MySQL** and **3x faster than PostgreSQL**.
* **Fully managed** by AWS (no need to worry about backups, patching, etc.).
* **Automatically scales** storage up to 128 TB.
* **High availability** with replication across multiple Availability Zones.
* **Pay only for what you use** (storage and I/O).

## Aurora vs RDS – Comparison Table

| **Feature** | **Amazon Aurora** | **Amazon RDS (MySQL/PostgreSQL)** |
| --- | --- | --- |
| Performance | High (up to 5x faster than MySQL) | Standard |
| Compatibility | MySQL & PostgreSQL | Multiple engines (MySQL, PostgreSQL, etc.) |
| Scalability | Auto-scales storage up to 128 TB | Manual scaling |
| Availability | Multi-AZ by default | Optional Multi-AZ |
| Cost | Slightly higher than RDS | Lower |
| Management | Fully managed | Fully managed |

## Amazon DynamoDB

Amazon DynamoDB is a **NoSQL database** service that is **fast, flexible, and fully managed**. It stores data in **key-value** or **document** format and is designed for **high-performance applications**.

**Real-Life Analogy:**

Think of DynamoDB like a **super-fast filing cabinet** where each file has a unique label (key). You can quickly find or update any file without searching through everything.

**Use Case:**

* Mobile apps storing user profiles.
* Gaming apps tracking scores and sessions.
* IoT devices storing sensor data.
* Real-time bidding systems.

**Key Points to Remember:**

* **NoSQL database** (not relational like MySQL).
* **Serverless** – no need to manage servers.
* **Automatic scaling** based on traffic.
* **Single-digit millisecond response time**.
* Supports **key-value** and **document** data models.
* **Highly available and durable** across multiple regions.
* **Pay-per-request** or **provisioned capacity** pricing.

**DynamoDB vs RDS – Comparison Table**

| **Feature** | **Amazon DynamoDB** | **Amazon RDS** |
| --- | --- | --- |
| Database Type | NoSQL (key-value/document) | Relational (SQL-based) |
| Schema | Flexible (no fixed schema) | Structured (tables, rows, columns) |
| Scaling | Automatic | Manual or auto with Aurora |
| Performance | Millisecond latency | Varies by engine |
| Management | Fully managed, serverless | Fully managed |
| Use Case | Real-time apps, IoT, mobile | Traditional apps, ERP, CRM |
| Query Language | API-based (no SQL) | SQL |

## In-Memory Caches

An **in-memory cache** stores frequently accessed data in the **RAM (memory)** instead of slower storage like disks. This makes data retrieval **very fast**.

When applications need specific information, they first check the cache before requesting it from the original data source. This reduces the load on primary databases and speeds up response times for end users.

**Key Points to Remember:**

* **Very fast** because it uses memory (RAM).
* Used to **reduce latency** and **improve performance**.
* Common AWS service: **Amazon ElastiCache**.
* Supports **Redis** and **Memcached** engines.
* Ideal for **temporary data** that changes often.

## Amazon ElastiCache

Amazon ElastiCache is a **fully managed in-memory caching service** that helps speed up applications by storing frequently accessed data in **RAM** instead of slower disk storage.

It automatically detects and replaces failed nodes, which makes it ideal for applications that need consistent high performance.

**Use Case:**

* Caching user session data for websites.
* Reducing database load by storing frequent queries.
* Real-time gaming leaderboards.
* Fast retrieval of product catalog data in e-commerce.

**Key Points to Remember:**

* **In-memory cache** = super fast data access.
* Supports **Redis** and **Memcached** engines.
* **Fully managed** by AWS (no need to install or maintain).
* Improves **application performance** and **scalability**.
* Can be used with **RDS**, **DynamoDB**, or other databases.
* Ideal for **temporary, frequently accessed data**.

**ElastiCache vs RDS vs DynamoDB – Quick Comparison**

| **Feature** | **ElastiCache (In-memory)** | **RDS (Relational)** | **DynamoDB (NoSQL)** |
| --- | --- | --- | --- |
| Speed | Very fast (RAM-based) | Moderate (disk-based) | Fast (SSD-based) |
| Data Type | Temporary, frequently used | Structured, relational | Key-value/document |
| Use Case | Caching, sessions, gaming | Traditional apps, ERP | Real-time apps, IoT |
| Persistence | Optional | Persistent | Persistent |
| Management | Fully managed | Fully managed | Fully managed |

## Additional Database Services

### 1. Amazon DocumentDB

Amazon DocumentDB is a **fully managed NoSQL document database** service that is **compatible with MongoDB**. It stores data in **JSON-like documents**, making it flexible and easy to work with.

**Real-Life Analogy:**

Think of DocumentDB like a **digital filing cabinet** where each file (document) can have different formats and fields. You don’t need to follow a strict structure like in traditional databases.

**Use Case:**

* Applications that use **MongoDB** and want to migrate to AWS.
* Content management systems.
* Catalogs for e-commerce platforms.
* User profiles with flexible data formats.

**Key Points to Remember:**

* **NoSQL document database**.
* **MongoDB-compatible** – you can use MongoDB tools and drivers.
* **Fully managed** by AWS (no server maintenance).
* Stores data in **JSON-like documents**.
* Scales storage and compute independently.
* Built for **high availability and durability**.
* Integrated with **AWS security and monitoring tools**.

**DocumentDB vs DynamoDB vs RDS – Comparison Table**

| **Feature** | **Amazon DocumentDB** | **Amazon DynamoDB** | **Amazon RDS** |
| --- | --- | --- | --- |
| Type | NoSQL (Document-based) | NoSQL (Key-value/document) | Relational (SQL-based) |
| Compatibility | MongoDB | Proprietary API | MySQL, PostgreSQL, Oracle, etc. |
| Schema | Flexible | Flexible | Structured |
| Use Case | Content, user profiles | Real-time apps, IoT | Traditional apps, ERP |
| Query Language | MongoDB query language | API-based | SQL |
| Management | Fully managed | Fully managed | Fully managed |

### 2. AWS Backup

AWS Backup is a **fully managed service** that helps you **automate and centralize backups** across AWS services. It ensures your data is protected and recoverable.

**Use Case:**

* Automatically backing up data from services like **Amazon EFS**, **RDS**, **DynamoDB**, **EC2**, and **S3**.
* Meeting compliance and business continuity requirements.
* Restoring data after accidental deletion or corruption.

**Real-Life Analogy:**

* **AWS Backup** = Saving your files to a USB drive every evening.
* **AWS DRS** = Having a second laptop with everything pre-installed, ready to use if your main one crashes.

**Key Points for Exam:**

* AWS Backup is ideal for **data protection and compliance**.
* AWS DRS is ideal for **business continuity and minimal downtime**.
* AWS DRS offers **faster recovery** but is **more complex and costly**.
* Both are **fully managed** and support **automation**.

| **Feature** | **AWS Backup** | **AWS Elastic Disaster Recovery (AWS DRS)** |
| --- | --- | --- |
| **Purpose** | Centralized backup and restore of AWS resources | Rapid recovery of entire applications and systems |
| **Data Type** | Files, volumes, databases | Full system state (OS, apps, data) |
| **Recovery Scope** | Point-in-time recovery of data | Full failover to AWS with minimal downtime |
| **Recovery Time Objective (RTO)** | Hours to days | Minutes |
| **Recovery Point Objective (RPO)** | Hours | Seconds |
| **Use Case** | Regular backups, compliance, long-term retention | Disaster recovery for critical workloads |
| **Supported Services** | EFS, RDS, EC2, DynamoDB, S3, etc. | On-premises servers, EC2, VMware, Hyper-V |
| **Pricing Model** | Pay-as-you-go (based on storage used) | More complex (licenses + storage + compute) |
| **Management** | Fully managed, easy setup | Requires setup of replication agents and staging area |
| **Automation** | Backup plans and policies | Continuous replication and automated failover |
| **Integration** | Deep integration with AWS services | Works across hybrid and multi-cloud environments |

### 3. Amazon Neptune

Amazon Neptune is a **fully managed graph database** service. It helps you store and query data that is connected — like relationships between people, products, or systems.

**Use Case:**

* Social networking apps (finding mutual friends).
* Fraud detection (tracking suspicious connections).
* Recommendation engines (suggesting related products).
* Knowledge graphs (linking concepts and entities).

**Key Points to Remember:**

* **Graph database** – stores data as nodes and relationships.
* Supports **open graph models**: **Property Graph** (via Gremlin) and **RDF** (via SPARQL).
* **Fully managed** by AWS.
* Optimized for **connected data** and complex queries.
* High performance and **low latency** for graph queries.
* Integrated with **IAM**, **CloudWatch**, and **VPC** for security and monitoring.

# AI/ML and Data Analysis

***AI***

Artificial Intelligence is a broad field focused on the development of intelligent computer systems capable of performing humanlike tasks.

***ML***

Machine learning is a type of AI for training machines to perform complex tasks without explicit instructions. Machine learning training finds the patterns hidden in vast amounts of historical data to produce an ML model. This ML model can then be applied to new data to make predictions or decisions based on the patterns it's learned.

## AWS AI/ML Solutions

### Language services

#### 1. Amazon Comprehend

Amazon Comprehend uses natural language processing to extract key insights from documents.  It develops these insights by recognizing key phrases, language, sentiment, and other common elements in documents.

* **Use Case**: Analyzing customer reviews to find sentiment (positive/negative).
* **Key Features**:
  + Sentiment analysis
  + Entity recognition (names, places, etc.)
  + Language detection
  + Topic modeling

#### 2. Amazon Transcribe

Amazon Transcribe converts speech into text. It supports multiple languages and offers features such as speaker identification, custom vocabulary, and real-time transcription.

* **What it does**: Converts **speech to text**.
* **Use Case**: Transcribing customer service calls or voice notes.
* **Key Features**:
  + Real-time and batch transcription
  + Speaker identification
  + Custom vocabulary support

**Analogy**: Like a voice assistant that writes down everything you say.

#### 3. Amazon Translate

Amazon Translate is a text translation service. This service is ideal for global communication because it supports real-time and batch text translation across multiple languages.

* **Use Case**: Translating product descriptions for global customers.
* **Key Features**:
  + Real-time and batch translation
  + Supports many languages
  + Custom terminology

**Analogy**: Like Google Translate, but built into your AWS apps.

#### 4. Amazon Polly

Amazon Polly converts text into lifelike speech. It supports multiple languages, different genders, and a variety of accents.

* **What it does**: Converts **text to speech**.
* **Use Case**: Reading out articles or instructions aloud.
* **Key Features**:
  + Realistic voice generation
  + Supports multiple languages and voices
  + Can create MP3 files

**Analogy**: Like an audiobook narrator that reads anything you type.

#### Quick Comparison Table

| **Service** | **Purpose** | **Input Type** | **Output Type** | **Example Use Case** |
| --- | --- | --- | --- | --- |
| Comprehend | Text understanding | Text | Insights | Sentiment analysis |
| Transcribe | Speech to text | Audio | Text | Call transcription |
| Translate | Language translation | Text | Translated text | Multilingual websites |
| Polly | Text to speech | Text | Audio |  |

### Computer Vision & Search AI Services in AWS

These services help you work with **images, documents, and intelligent search** using machine learning.

#### 1. Amazon Rekognition

Amazon Rekognition is a video analysis service. It can identify objects, people, text, scenes, and activities within images and videos stored in Amazon Simple Storage Service (Amazon S3).

* **What it does**: Analyzes images and videos to detect objects, faces, text, and more.
* **Use Case**: Face detection for security, image tagging, content moderation.
* **Key Features**:
  + Face and object detection
  + Text in images
  + Celebrity recognition
  + Unsafe content detection

**Analogy**: Like a smart camera that can tell who’s in a photo and what’s happening.

#### 2. Amazon Textract

Amazon Textract detects and extracts typed and handwritten text found in documents, forms, and even tables within documents.

* **What it does**: Extracts text, tables, and forms from scanned documents and PDFs.
* **Use Case**: Automating data entry from invoices, forms, and receipts.
* **Key Features**:
  + Reads printed and handwritten text
  + Detects form fields and tables
  + Structured output for easy processing

**Analogy**: Like a scanner that not only reads text but understands the layout and meaning.

#### 3. Amazon Kendra

Amazon Kendra uses natural language processing to search for answers within large amounts of enterprise content. Because it understands the context of a query, it can return more precise and relevant answers than just a list of documents with matching keywords.

* **What it does**: Intelligent enterprise search service that uses ML to deliver accurate answers.
* **Use Case**: Internal company search portals, knowledge base search.
* **Key Features**:
  + Natural language search
  + Connects to multiple data sources (SharePoint, S3, etc.)
  + Relevance tuning and FAQs support

**Analogy**: Like a super-smart search engine that understands your question and gives the best answer from your documents.

#### Quick Comparison Table

| **Service** | **Purpose** | **Input Type** | **Output Type** | **Example Use Case** |
| --- | --- | --- | --- | --- |
| Rekognition | Image/video analysis | Images, videos | Labels, faces, text | Face detection, content filtering |
| Textract | Document text extraction | PDFs, scans | Structured text | Invoice processing, form reading |
| Kendra | Intelligent search | Text documents | Ranked answers |  |

### Conversational AI & Personalization Services

These services help you build **smart chatbots** and **personalized user experiences** using machine learning.

#### 1. Amazon Lex – Conversational AI

With Amazon Lex, you can add voice and text conversational interfaces to your applications. This service uses both natural language understanding (NLU) and automatic speech recognition (ASR) to create lifelike conversations.

**What it does:**

Builds **chatbots and voice assistants** that understand natural language.

**Use Case:**

* Customer support bots
* Appointment booking assistants
* Voice-enabled apps

**Key Features:**

* Natural language understanding (NLU)
* Speech-to-text and text-to-speech
* Integrates with **Amazon Connect**, **Lambda**, and **Polly**

**Analogy**: Like building your own Alexa for your app or website.

#### 2. Amazon Personalize – *Personalization AI*

With Amazon Personalize, you can use historical data to build intelligent applications with personalized recommendations for your customers.

**What it does:**

Creates **real-time personalized recommendations** for users, similar to what Netflix or Amazon.com does.

**Use Case:**

* Product recommendations in e-commerce
* Personalized content feeds
* Tailored marketing messages

**Key Features:**

* Uses user behavior and item data
* Real-time recommendations
* No ML expertise needed

**Analogy**: Like a smart shop assistant who remembers what you like and suggests things just for you.

#### Quick Comparison - Lex vs Personalize

| **Feature** | **Amazon Lex** | **Amazon Personalize** |
| --- | --- | --- |
| Purpose | Build chatbots and voice assistants | Create personalized recommendations |
| Input Type | Text or speech | User behavior and item data |
| Output Type | Text responses or actions | Ranked recommendations |
| Use Case | Customer support, voice apps | E-commerce, media, marketing |
| ML Expertise Needed | No | No |
| Integration | Connect, Lambda, Polly | S3, Lambda, real-time apps |

### ML Services

**Amazon SageMaker – *Build, Train, and Deploy ML Models***

With this fully managed service, you can build, train, and deploy your own ML models without worrying about infrastructure. The SageMaker AI integrated development environment (IDE) provides simplified access control and transparency over your ML projects. You can track model training experiments, visualize data, and debug and monitor your workflows all within one environment. SageMaker AI even offers access to hundreds of pre-trained models that you can deploy in a few quick steps.

**Real-Life Analogy:**

Think of SageMaker like a **machine learning factory** — it provides all the tools, machines, and automation to build smart products (ML models) without needing to set up everything yourself.

**Use Case:**

* Predicting customer churn
* Fraud detection
* Image classification
* Forecasting sales

**Key Points to Remember:**

* Supports **end-to-end ML workflow**: data prep → training → tuning → deployment.
* Offers **pre-built algorithms** and **Jupyter notebooks**.
* Can use **AutoML** (SageMaker Autopilot) for beginners.
* Integrates with **S3**, **Lambda**, **CloudWatch**, and more.
* Pay-as-you-go pricing for compute and storage.

## Generative AI on AWS

### Amazon SageMaker JumpStart – *Quick Start for ML Projects*

A **low-code/no-code ML hub** that helps you quickly start machine learning projects using **pre-trained models**, **built-in algorithms**, and **solution templates**.

**Analogy:**

Like a **starter kit** for machine learning — everything is pre-packed so you can begin without building from scratch.

**Use Case:**

* Deploying Hugging Face models
* Sentiment analysis
* Fraud detection
* Image classification

**Key Features:**

* One-click deployment of models
* Prebuilt solutions for common ML tasks
* Foundation models from Hugging Face, Meta, Stability AI, etc.
* Works with SageMaker Studio and Python SDK [[aws.amazon.com]](https://aws.amazon.com/sagemaker/ai/jumpstart/)

### Amazon Bedrock – Foundation Models for Generative AI

A **fully managed service** to access and customize **foundation models (FMs)** from top AI providers like Anthropic, Meta, Stability AI, and Amazon Titan — without managing infrastructure.

**Analogy:**

Like a **model marketplace** where you pick a powerful AI model, customize it with your data, and plug it into your app.

**Use Case:**

* Building chatbots and AI assistants
* Text and image generation
* Retrieval-Augmented Generation (RAG)
* Customizing models for enterprise tasks

**Key Features:**

* Unified API for multiple models
* Serverless architecture
* Fine-tuning and RAG support
* Secure integration with AWS services [[docs.aws.amazon.com]](https://docs.aws.amazon.com/bedrock/latest/userguide/what-is-bedrock.html)

### Amazon Q – Generative AI Assistant for Developers & Businesses

Amazon Q is a **generative AI assistant** that helps **developers write code** and **business users get insights** from internal data.

**Analogy:**

Like having a **smart teammate** who can code, answer questions, generate reports, and automate tasks.

**Use Case:**

* Developers: Code generation, debugging, AWS CLI help
* Business users: Search across company data, generate dashboards
* Build custom AI apps with natural language

**Key Features:**

* Two products: **Q Developer** and **Q Business**
* Q Developer: IDE integration, security scans, code transformation
* Q Business: Connects to 40+ enterprise systems
* Q Apps: Build AI apps using natural language (preview) [[aws.amazon.com]](https://aws.amazon.com/q/)

**Amazon Q Business**

Amazon Q Business can answer pressing questions, help solve problems, and take actions using the data and expertise found in your company's information repositories. Amazon Q Business provides this tailored assistance with a secure connection to commonly used systems.

**Use cases:**Information requests, automated workflows, and insight extraction

**Amazon Q Developer**

Amazon Q Developer provides code recommendations to accelerate development for coding languages including C#, Java, JavaScript, Python, and TypeScript applications. It integrates with multiple IDEs and helps developers write code faster by generating entire functions and logical blocks of code.

**Use cases:**Faster code generation, improved reliability and security, and automated code reviews

### Summary:

**SageMaker JumpStart**: Great for **quickly starting ML projects** using prebuilt models and templates — ideal for beginners or prototyping.

**Amazon Bedrock**: Use this when you want to build **chatbots or GenAI apps** using powerful prebuilt models (like Anthropic, Meta, etc.) without managing servers.

**Amazon Q Developer**: Your **AI coding assistant** — helps write, debug, and understand code, and provides AWS service guidance.

**Amazon Q Business**: A **smart assistant for business users** — helps search across company data, generate insights, and build AI-powered apps using natural language.

## AWS data pipeline services

Let's dive a bit deeper into the AWS services used in a typical AWS data pipeline.

### Data Ingestion Services

A couple of rings with blue background

AI-generated content may be incorrect.

Data ingestion involves moving data from source systems into your chosen storage solution. Use real-time ingestion when the data is needed immediately. Use batch ingestion when some latency is tolerable.

#### Amazon Kinesis Data Streams

You can use Kinesis Data Streams for real-time ingestion of terabytes of data from applications, streams, and sensors. This serverless service even provides automatic provisioning and scaling in on-demand mode.

#### Amazon Data Firehose

Firehose is an option for data ingestion in near real-time. This fully managed service provides automatic provisioning and scaling. It also delivers data within seconds to data lakes, warehouses, and analytics services.

### Data Storage Services

A couple of rings on a blue background

AI-generated content may be incorrect.

Data can come from many different sources. To gain insights, data is commonly consolidated into a single location. There are two storage options for this. *Flexible data lakes* store vast amounts of raw data. Alternatively, the more structured *data warehouses* are optimized for business intelligence.

#### Amazon S3

Amazon S3 is a popular choice for data lakes. This object storage service can securely house virtually any amount of structured or unstructured data. Amazon S3 is also fully elastic, automatically scaling as you add and remove data.

#### Amazon Redshift

Amazon Redshift is a fully managed data warehouse service that can store petabytes of structured or semistructured data. With the scalability and pay-as-you-go pricing model, organizations can cost-effectively analyze large datasets.

### Data cataloging services

A blue square with white logo

AI-generated content may be incorrect.

Cataloging your data with metadata provides an inventory of your organization's data.

#### AWS Glue Data Catalog

AWS Glue Data Catalog provides a centralized, scalable, and managed metadata repository that enhances data discovery. It improves the overall efficiency of data pipelines by delivering metadata to various data stores and analytics services.

### Data Processing Services

A couple of rings on a blue background

AI-generated content may be incorrect.

Data processing services clean and transform your data so it's ready to be analyzed.

#### AWS Glue

AWS Glue is a fully managed ETL (Extract, Transform and Load) service that makes data preparation simpler, faster, and cost effective. AWS Glue ETL jobs can use the AWS Glue Data Catalog to access metadata about data sources, which can help inform transformations defined in the ETL script.

#### Amazon EMR

Amazon EMR is ideal for large-scale data processing and organizations with existing big data expertise. It automatically handles infrastructure provisioning, cluster management, and scaling. Amazon EMR supports popular big data frameworks like Apache Spark, Apache Hadoop, and Apache Hive.

### Data Analysis and Visualization Services

A couple of rings on a blue background

AI-generated content may be incorrect.

Queries and visualization tools help you to develop important insights about your data.

#### Amazon Athena

With Athena, you can run SQL queries to analyze data in relational, nonrelational, object, and custom data sources. This fully managed serverless service can access data hosted on Amazon S3, on premises, or even in multi-cloud environments. It offers a cost-effective solution for data analysis because you only pay for the queries you run.

#### Amazon Redshift

Amazon Redshift is a fully managed data warehouse solution. Its columnar storage and massively parallel processing architecture make it ideal for analyzing large datasets. You can use it to perform complex SQL queries on large datasets for frequent, high-performance analytical workloads.

#### Amazon QuickSight

With QuickSight, both technical and non-technical users can quickly create modern interactive dashboards and reports from various data sources without managing infrastructure. Amazon Q in QuickSight provides natural language queries so business analysts and users can build, discover, and share meaningful insights in seconds.

#### Amazon OpenSearch Service

With OpenSearch Service, you can search for relevant content through precise keyword matching or natural language queries. Unified dashboards provide real-time data visualization as you analyze and monitor logs, traces, and metrics for various applications.

A diagram of a process

AI-generated content may be incorrect.

**8. Train model**

SageMaker AI reads the same dataset directly from Amazon S3. ML engineers can then train new versions of the recommendation model using the latest information.

**7. Perform data analytics**

Data scientists use Athena to gather insights through queries.

**6. Catalog data**

AWS Glue Data Catalog serves as a metadata repository with tables that describe the schema and location of the Amazon S3 data.

**5. Deliver data**

Firehose then delivers the data to the company's Amazon S3 data lake, where it is available for multiple consumers.

**4. Process data**

The data is in JSON format, so Firehose invokes an AWS Lambda function that transforms the data into .csv format.

**2. Store app data**

An Amazon DynamoDB database stores the historical customer data gathered through the app. This makes sense for low latency reads and writes but isn't ideal for ML model training.

**3. Ingest data**

Kinesis Data Streams ingests the data from DynamoDB. Amazon Data Firehose then aggregates the data.

**1. Make recommendations**

An e-commerce company uses an ML model to make product recommendations.

# Security

### Authentication and Authorization

*Authentication* is the process of verifying the identity of a user or entity through credentials like a username and password combination.

* **Use case:** An employee logs in to an employee portal.

*Authorization* grants users certain access rights and permissions that determine which actions they can perform in a system or application.

* **Use case:** An employee can only access their own employee records inside the employee portal.

A diagram of a diagram

AI-generated content may be incorrect.

## AWS shared responsibility model

Cloud security is a shared responsibility between customers and AWS. Let's examine this relationship using the AWS shared responsibility model.

### Customers: Security *in* the cloud

When using AWS services, customers maintain complete control over their content. As a result, customers are responsible for securing everything they create and manage in the AWS Cloud. This includes the following:

* Managing the security of data, systems, and applications
* Deciding what data and workloads to store or run in AWS
* Determining which AWS services to use
* Controlling who has access to environments and resources

### AWS: Security *of* the cloud

AWS is responsible for security *of*the cloud. AWS operates, manages, and controls the components at all layers of the infrastructure. This includes securing the following:

* The foundational software that powers AWS services
* The virtualization layer
* The hardware and global infrastructure that supports the data centers from which services operate. This includes protection for AWS Regions, Availability Zones, and edge locations.

A diagram of a software company

AI-generated content may be incorrect.

### AWS security controls

AWS offers multiple security mechanisms to protect your cloud resources. These controls can help you do the following:

* *Prevent* security incidents through proper permission and access management.
* *Protect*networks, applications, and data.
* *Detect and respond* to security incidents as they occur.

## AWS Identity and Access Management (IAM)

**IAM** is a service that helps you **securely control access** to AWS services and resources. It lets you decide **who can do what** in your AWS account.

**Real-Life Analogy:**

Think of IAM like a **security gate at an office**. It decides **who can enter**, **what rooms they can access**, and **what actions they can perform** inside.

**Use Case:**

* Allowing developers to access only specific AWS services (e.g., EC2, S3).
* Giving read-only access to auditors.
* Creating temporary access for contractors.

#### Comparison Table - IAM Components

| **Feature** | **IAM User** | **IAM Group** | **IAM Role** | **IAM Policy** |
| --- | --- | --- | --- | --- |
| **What is it?** | A person or app with long-term credentials | A collection of users | A set of permissions assumed temporarily | A document that defines permissions |
| **Used By** | Individuals or applications | Users | AWS services, users, or external identities | Attached to users, groups, or roles |
| **Credentials** | Username + password or access keys | No credentials (inherits from users) | No credentials (assumed when needed) | Not a user or identity — just permission rules |

#### IAM vs Resource-Based Policies

| **Feature** | **IAM Policies** | **Resource-Based Policies** |
| --- | --- | --- |
| Attached To | Users, Groups, Roles | AWS resources (e.g., S3 bucket) |
| Use Case | Control what a user can do | Control who can access a resource |
| Example | Allow user to access S3 | Allow another account to access S3 |

**Key Points for Exam:**

* IAM is **global** (not region-specific).
* **Free to use** (no extra cost).
* Supports **MFA (Multi-Factor Authentication)** for extra security.
* Use **IAM roles** for services like EC2 to access other AWS services securely.
* Follow **least privilege** principle.

## Additional Access Management Services

Let's examine some additional AWS services that can help you enforce the principle of least privilege across AWS environments. Collectively, these services help streamline the administration of AWS environments while bolstering your security practices.

### AWS IAM Identity Center

IAM Identity Center centralizes identity and access management across AWS accounts and applications. IAM Identity Center can also connect to an existing identity source and provide your workforce with single sign-on access to all your connected AWS services and accounts. This is called federated identity management.

***Federated identity management****is a system that allows users to access multiple applications, services, or domains using a single set of credentials.*

### AWS Secrets Manager

Secrets Manager provides a secure way to manage, rotate, and retrieve database credentials, API keys, and other secrets throughout their lifecycle. This helps keep your applications, services, and IT resources safe.

***Secrets****are confidential or private information intended to be known only to specific individuals or groups. Examples include passwords, database credentials, and API keys.*

### AWS Systems Manager

Systems Manager provides a centralized view of nodes across your organization’s accounts and Regions and multi-cloud and hybrid environments. With this service, you can quickly access node information, such as ID and operating system details, and automate registry edits, user management, and security patching.

***Nodes****are connection points in a network, system, or structure****.***

### Comparison Table - IAM Identity Center vs Secrets Manager vs Systems Manager

| **Feature** | **IAM Identity Center** | **AWS Secrets Manager** | **AWS Systems Manager** |
| --- | --- | --- | --- |
| **Purpose** | Manage **user access** to AWS accounts & apps | Securely store and manage **secrets** (e.g., passwords, API keys) | Manage and automate **infrastructure operations** |
| **Use Case** | Centralized login for multiple AWS accounts or apps | Store DB credentials, API keys, tokens securely | Patch management, automation, inventory, session mgmt |
| **Replaces** | AWS SSO (Single Sign-On) | Manual secret storage in code or config files | Manual server management via SSH or RDP |
| **Key Features** | - Centralized user directory - SSO to AWS & SaaS apps - Integration with AD & external IdPs | - Automatic secret rotation - Fine-grained access control - Audit logging | - Run commands remotely - Patch & compliance - Parameter Store |
| **Security Focus** | Identity and access management | Credential and secret protection | Operational security and automation |
| **Integration** | AWS Organizations, AD, IAM | IAM, Lambda, RDS, EC2 | EC2, Lambda, CloudWatch, IAM |
| **Pricing** | Free for AWS account access; charges for app assignments | Pay per secret stored and API calls | Mostly free; some features may incur charges |

## Protecting Networks and Applications

### Network And Application Attacks

Network and application protection is another vital component of a secure environment on AWS.

#### Dos Attacks

In a denial of service attack, an attacker floods a web application with excessive network traffic. Legitimate customer requests are denied if the web application becomes overloaded and can no longer respond.

A red arrow pointing to the right

AI-generated content may be incorrect.

#### Ddos Attacks

In a distributed denial of service (DDoS) attack, an attacker can use multiple infected computers (called *zombie bots*) to unknowingly send excessive traffic to a web application.

A diagram of a zombie bot

AI-generated content may be incorrect.

### AWS Protection through Infrastructure

#### 1. Security Groups

Security groups only allows in proper request traffic. They operate at the AWS network level so they can shrug off massive attack using the entire AWS Region’s capacity.

#### 2. Elastic Load Balancing (ELB)

ELB handles traffic first before handing it off, so your frontend server is not overwhelmed. Like security groups, it runs at the Region level.

#### 3. AWS Regions

The enormous capacity of Regions makes them extremely difficult to overwhelm. It would be massively expensive to achieve.

### AWS Protection through Services

#### 1. AWS Shield

*AWS Shield Standard* is designed to automatically protect AWS customers from the most common, frequently occurring types of DDoS attacks at no cost. It uses a variety of analysis techniques to detect and mitigate incoming malicious network traffic in real time.

*AWS Shield Advanced* is a paid service that provides detailed attack diagnostics and the ability to detect and mitigate sophisticated DDoS attacks. It also integrates with other services, such as Amazon CloudFront, Amazon Route 53, and ELB.

Additionally, you can integrate AWS Shield with AWS WAF by writing custom rules to mitigate complex DDoS attacks.

#### 2. AWS WAF

AWS WAF is a web application firewall that monitors network requests that come into your web applications. When a request comes into AWS WAF, it checks the IP address against a web access control list (web ACL). If the request comes from a blocked IP address on the web ACL, AWS WAF denies access.Legitimate requests are allowed access.

## Protecting Data

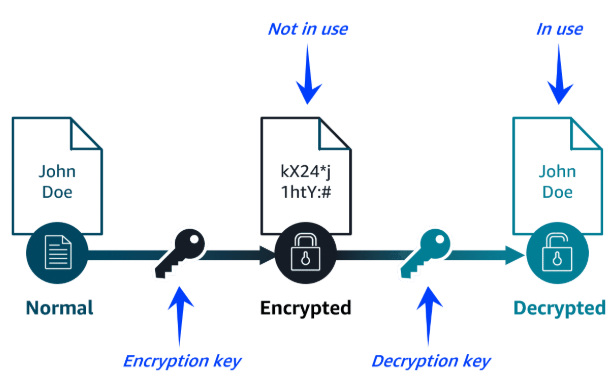
**Data encryption**

So much of what you do on AWS is driven by data. Keeping your data safe is important to make sure your applications run smoothly and to maintain customer trust.

Encryption is a key component of data protection. Let's review how data encryption works.

### Encryption Basics

Data encryption works like a lock and key mechanism. If you have the right key, you can access the encrypted data. Otherwise, you cannot access the data. For example, let's say you are protecting a customer's profile. An encryption key is used to turn the profile information into a randomized set of characters. A decryption key is used to access the customer's information, such as their name, only when it's needed by your application.



### Types of data encryption

Data encryption comes in the following two forms:

* **Data encryption at rest**: The data is idle and not moving, like when it's stored in a database.
* **Data encryption in transit:** The data is moving between locations, like when it's being sent from a database to an application. SSL/TLS certificates are used to establish encrypted network connections from one system to another.

A screen shot of a computer

AI-generated content may be incorrect.

## AWS Data Protection

### 1. AWS Key Management Service (KMS)

AWS KMS helps you **create, manage, and control encryption keys** used to protect your data.

**Real-Life Analogy:**

Think of KMS like a **digital safe** that stores the keys used to lock (encrypt) and unlock (decrypt) your data.

**Use Case:**

* Encrypting data in **S3**, **RDS**, **EBS**, or **Lambda**.
* Managing keys for secure communication between services.

**Key Features:**

* Create and rotate encryption keys.
* Integrated with most AWS services.
* Supports **customer-managed** and **AWS-managed** keys.
* Logs key usage with **CloudTrail**.

### 2. Amazon Macie

Amazon Macie is a **data security and privacy service** that uses machine learning to **discover and protect sensitive data** like names, emails, and credit card numbers in **Amazon S3**.

**Real-Life Analogy:**

Macie is like a **data detective** that scans your storage and flags anything sensitive or private.

**Use Case:**

* Identifying **Personally Identifiable Information (PII)** in S3.
* Monitoring for accidental data exposure.

**Key Features:**

* Uses ML to detect sensitive data.
* Provides dashboards and alerts.
* Helps with **compliance** (e.g., GDPR, HIPAA).

### 3. AWS Certificate Manager (ACM)

ACM helps you **provision, manage, and renew SSL/TLS certificates** to secure your websites and applications.

**Real-Life Analogy:**

ACM is like a **digital ID card** for your website that proves it’s secure and trustworthy.

**Use Case:**

* Securing websites with **HTTPS**.
* Managing certificates for **Elastic Load Balancers**, **CloudFront**, and **API Gateway**.

**Key Features:**

* Free public certificates from AWS.
* Automatic renewal.
* Easy integration with AWS services.

### Quick Comparison Table

| **Feature** | **AWS KMS** | **Amazon Macie** | **AWS Certificate Manager (ACM)** |
| --- | --- | --- | --- |
| Purpose | Manage encryption keys | Discover and protect sensitive data | Manage SSL/TLS certificates |
| Main Use Case | Encrypt data in AWS services | Scan S3 for PII and sensitive content | Secure websites and APIs with HTTPS |
| Integration | S3, RDS, EBS, Lambda, etc. | S3 (primary) | ELB, CloudFront, API Gateway |
| Security Focus | Data encryption | Data classification and privacy | Secure communication |
| Pricing | Pay per key and usage | Pay per GB scanned | Free for public certs issued by ACM |

## Detection and response services

Preventing and protecting against security threats are two methods for securing your AWS resources. You should also be prepared to detect and respond to security incidents that might occur. AWS offers a variety of services you can use to detect and respond to security incidents.

### 1. Amazon Inspector – Vulnerability Scanner

Amazon Inspector helps improve the security and compliance of applications by running automated security assessments for Amazon EC2 instances, containers, and Lambda functions.It checks applications for security vulnerabilities and deviations from security best practices, such as open access to EC2 instances and installations of vulnerable software versions.

**Analogy:**

Like a **security guard checking doors and windows** to make sure nothing is broken or unlocked.

**Use Case:**

* Scanning EC2 instances and container images for known vulnerabilities.
* Ensuring compliance with security standards.

**Key Features:**

* Automated, continuous scanning.
* Integrates with EC2, Lambda, ECR.
* CVE (Common Vulnerabilities and Exposures) detection.

### 2. Amazon GuardDuty – *Threat Detection*

Amazon GuardDuty provides intelligent threat detection across your infrastructure and resources. GuardDuty identifies threats by continuously monitoring streams of your account metadata and network activity in your environment. It uses known malicious IP addresses, anomaly detection, and machine learning to identify threats more accurately.

**Analogy:**

Like a **security camera with AI** that alerts you when someone behaves strangely.

**Use Case:**

* Detecting unusual API calls, port scanning, or compromised credentials.
* Monitoring logs from CloudTrail, VPC Flow Logs, DNS.

**Key Features:**

* No agents required.
* Continuous monitoring.
* Sends alerts to CloudWatch or Security Hub.

### 3. Amazon Detective – Security Investigation Tool

After a threat has been detected, you can use Amazon Detective to further investigate the root cause. Detective helps you analyze threats with interactive visualizations contained in a unified AWS Management Console view. These visualizations include resource and user interactions over a configurable timeline with recommended steps for remediation.

**Analogy:**

Like a **detective analyzing clues** to understand what happened during a security incident.

**Use Case:**

* Investigating GuardDuty findings.
* Tracing suspicious user or network activity.

**Key Features:**

* Visual graphs of relationships and events.
* Helps identify root cause of threats.
* Integrates with GuardDuty and Security Hub.

### 4. AWS Security Hub – Central Security Dashboard

Security Hub gives you a **centralized view of your security posture** across AWS services.

Security Hub brings multiple security services together into a single place and format. With this service, you can quickly see your security and compliance state in one comprehensive view.

**Analogy:**

Like a **control room dashboard** showing all security alerts and compliance checks in one place.

**Use Case:**

* Aggregating findings from GuardDuty, Inspector, Macie, etc.
* Checking compliance with standards like CIS, PCI-DSS.

**Key Features:**

* Unified dashboard for security findings.
* Automated compliance checks.
* Integrates with third-party tools.

### Quick Comparison Table

| **Service** | **Purpose** | **Main Function** | **Analogy** |
| --- | --- | --- | --- |
| **Inspector** | Vulnerability scanning | Finds weaknesses in EC2, containers | Security guard checking locks |
| **GuardDuty** | Threat detection | Detects suspicious behavior | AI-powered security camera |
| **Detective** | Security investigation | Analyzes and traces incidents | Detective solving a case |
| **Security Hub** | Centralized security view | Aggregates findings and checks compliance | Control room dashboard |

# Monitoring, Compliance, and Governance in the AWS Cloud

## Amazon CloudWatch

CloudWatch monitors your AWS resources and the applications that you run on AWS in real time. With CloudWatch, you gain system-wide visibility into resource utilization, application performance, and operational health.

**Real-Life Analogy:**

Think of CloudWatch like a **fitness tracker** for your AWS environment — it monitors everything (CPU, memory, errors, etc.) and alerts you when something goes wrong.

**Use Case:**

* Monitor EC2 instance CPU usage.
* Set alarms for high error rates in Lambda functions.
* View logs from applications and services.
* Automatically respond to issues (e.g., restart an instance).

**Key Features:**

| **Feature** | **Description** |
| --- | --- |
| **Metrics** | Tracks performance data (e.g., CPU, memory, latency). |
| **Logs** | Collects and stores logs from AWS services and apps. |
| **Alarms** | Sends alerts when metrics cross thresholds. |
| **Dashboards** | Visualize metrics and logs in real-time. |
| **Events** | Respond to changes (e.g., auto-restart EC2 on failure). |
| **Insights** | Analyze logs using queries (CloudWatch Logs Insights). |

## What is Auditing in AWS?

**Auditing** means **tracking and reviewing actions** taken in your AWS environment — like who accessed what, when, and how. It helps ensure **security, compliance, and accountability**.

**Real-Life Analogy:**

Think of auditing like a **CCTV system** in an office. It records who entered, what they did, and when — so you can review it if something goes wrong.

**Why Auditing is Important:**

1. **Security** – Detect unauthorized access or suspicious activity.
2. **Compliance** – Meet legal and industry standards (e.g., GDPR, HIPAA).
3. **Troubleshooting** – Understand what caused an issue.
4. **Accountability** – Know who made changes and when.
5. **Monitoring** – Keep track of usage and access patterns.

## AWS CloudTrail

AWS CloudTrail is a **logging service** that records **all API calls and actions** taken in your AWS account. It helps you **track who did what, when, and from where**.

**Real-Life Analogy:**

Think of CloudTrail like a **security camera** in your AWS environment — it records every action so you can review it later if something goes wrong.

**Use Case:**

* Auditing user activity and API usage.
* Investigating security incidents.
* Meeting compliance requirements.
* Tracking changes to resources.

**Key Points for Exam:**

* CloudTrail is **enabled by default** for management events.
* Stores logs in **S3 buckets**.
* Can be used with **Amazon Athena** for querying logs.
* Helps with **auditing, compliance, and security investigations**.

## What is Compliance in AWS?

**Compliance** means following **rules, laws, and standards** that protect data, privacy, and security. In AWS, it ensures that your cloud usage meets **industry and government regulations**.

**Real-Life Analogy:**

Think of compliance like **traffic rules** — they keep everyone safe and organized. If you follow them, you avoid penalties and accidents.

**Why Compliance Is Important:**

1. **Legal Protection** – Avoid fines and legal issues.
2. **Customer Trust** – Shows users their data is safe.
3. **Security** – Helps enforce best practices.
4. **Market Access** – Required to operate in regulated industries (e.g., healthcare, finance).
5. **Audit Readiness** – Makes it easier to pass security audits.

**Benefits of Compliance with AWS**

| **Benefit** | **Description** |
| --- | --- |
| **Global Standards Support** | AWS complies with standards like **ISO**, **SOC**, **HIPAA**, **GDPR**, etc. |
| **Shared Responsibility Model** | AWS secures the infrastructure; you manage your data and access. |
| **Automated Tools** | Services like **AWS Config**, **Security Hub**, and **CloudTrail** help track compliance. |
| **Documentation & Reporting** | AWS provides audit reports and compliance documentation. |
| **Scalability & Flexibility** | Easily apply compliance controls across multiple accounts and regions. |

## What Are Artifacts in AWS?

In AWS, **artifacts** refer to **documents and reports** that prove AWS meets **security and compliance standards**. These are used by customers to support their own **audit and compliance needs**.

### AWS Artifact

**AWS Artifact** is a **self-service portal** where you can **access AWS compliance reports**, such as **ISO certifications**, **SOC reports**, and **agreements** like **HIPAA** or **GDPR**.

**Real-Life Analogy:**

Think of AWS Artifact like a **library of official certificates and reports** that show AWS is following the rules — and you can use them to prove your own compliance.

**Use Case:**

* Downloading **SOC 2** or **ISO 27001** reports for audits.
* Reviewing **GDPR** or **HIPAA** agreements.
* Sharing AWS compliance documents with regulators or auditors.

**Key Features:**

| **Feature** | **Description** |
| --- | --- |
| **Compliance Reports** | Access third-party audit reports (SOC, ISO, PCI, etc.) |
| **Agreements** | Accept and manage legal agreements (e.g., BAA for HIPAA) |
| **Self-Service Access** | No need to contact AWS support — download instantly |
| **Audit Support** | Helps customers meet their own compliance requirements |

## Auditing AWS Resources for Compliance

### 1. AWS Config

AWS Config is a **configuration tracking and compliance service**. It helps you **monitor, record, and evaluate changes** to your AWS resources over time.

**Real-Life Analogy:**

Think of AWS Config like a **security camera for your AWS setup** — it records every change made to your resources and checks if those changes follow your rules.

**Use Case:**

* Track changes to EC2 instances, security groups, IAM roles, etc.
* Check if resources comply with company or regulatory policies.
* Troubleshoot issues by reviewing historical configurations.

**Key Features:**

| **Feature** | **Description** |
| --- | --- |
| **Resource Inventory** | Lists all AWS resources and their configurations. |
| **Change Tracking** | Records every change made to supported resources. |
| **Compliance Rules** | Automatically checks if resources follow defined rules. |
| **Snapshots** | Keeps historical versions of configurations. |
| **Integration** | Works with AWS CloudTrail, Security Hub, and Organizations. |

**Key Points for Exam:**

* AWS Config helps with **compliance auditing** and **resource monitoring**.
* You can create **custom rules** or use **managed rules**.
* It supports **multi-account** and **multi-region** tracking via AWS Organizations.
* Useful for **security**, **governance**, and **troubleshooting**.

### 2. AWS Audit Manager

AWS Audit Manager helps you **automate the collection of evidence** needed for **audits and compliance checks**. It continuously monitors your AWS usage and maps it to **industry standards** like **ISO**, **PCI-DSS**, **HIPAA**, and **GDPR**.

**Real-Life Analogy:**

Think of Audit Manager like a **smart assistant for auditors** — it automatically gathers and organizes all the documents and logs needed to prove you're following the rules.

**Use Case:**

* Preparing for **security audits**.
* Demonstrating compliance with **regulatory frameworks**.
* Reducing manual effort in collecting audit evidence.

**Key Features:**

| **Feature** | **Description** |
| --- | --- |
| **Automated Evidence Collection** | Gathers data from AWS services like CloudTrail, Config, IAM, etc. |
| **Prebuilt Frameworks** | Includes templates for common standards (e.g., ISO 27001, SOC 2). |
| **Custom Frameworks** | Create your own audit frameworks based on internal policies. |
| **Continuous Monitoring** | Tracks changes and updates evidence automatically. |
| **Integration** | Works with AWS Organizations, CloudTrail, Config, Security Hub. |

#### Audit Manager vs AWS Artifact vs AWS Config

| **Feature** | **AWS Audit Manager** | **AWS Artifact** | **AWS Config** |
| --- | --- | --- | --- |
| Purpose | Automate audit evidence collection | Access AWS compliance reports | Track resource configurations and compliance |
| Use Case | Internal/external audits | Share AWS certifications with auditors | Monitor changes and enforce rules |
| Data Type | Evidence from AWS services | Reports and legal agreements | Configuration snapshots |
| Audience | Compliance teams, auditors | Auditors, legal teams | DevOps, security teams |

**Key Points for Exam:**

* Helps reduce **manual effort** in audits.
* Supports **prebuilt and custom frameworks**.
* Integrates with other AWS security and logging services.
* Useful for **continuous compliance** and **audit readiness**.

## AWS Organizations

AWS Organizations lets you **manage multiple AWS accounts** from a **central location**. It helps with **billing**, **access control**, and **policy enforcement** across all accounts in your organization.

**Real-Life Analogy:**

Think of AWS Organizations like a **corporate head office** managing multiple branch offices (AWS accounts). It sets rules, controls budgets, and manages access for all branches.

**Use Case:**

* Centralized billing for multiple AWS accounts.
* Applying security policies across accounts.
* Isolating workloads by environment (e.g., dev, test, prod).
* Delegating access and control to different teams.

**Key Features:**

| **Feature** | **Description** |
| --- | --- |
| **Organizational Units (OUs)** | Group accounts by function (e.g., finance, dev). |
| **Service Control Policies (SCPs)** | Set permission boundaries for accounts. |
| **Consolidated Billing** | Combine bills from all accounts into one. |
| **Account Management** | Create and manage AWS accounts centrally. |
| **Integration** | Works with IAM Identity Center, CloudTrail, Config, etc. |

### AWS Organizations vs IAM – Quick Comparison

| **Feature** | **AWS Organizations** | **AWS IAM** |
| --- | --- | --- |
| Scope | Manages multiple AWS accounts | Manages users and permissions within one account |
| Use Case | Centralized control and billing | Access control for services and resources |
| Policies | SCPs (organization-wide) | IAM policies (account-level) |
| Structure | Hierarchical (root → OUs → accounts) | Flat (users, groups, roles) |

**Key Points for Exam:**

* Helps with **multi-account management**.
* SCPs **do not grant permissions**, they **limit** them.
* **Consolidated billing** can reduce costs.
* Supports **delegated administration**.

## Governance in the AWS Cloud

As an organization scales up, they could have difficulty governing the services and accounts, including the new AWS accounts, the AWS services they choose, and even the software licenses.

### AWS Control Tower

AWS Control Tower can help you save time while providing governance. It uses preconfigured controls, which can help you to quickly set up multi-account environments, automation with built-in governance, and integration of third-party software at scale.

**Purpose:**  
Helps set up and govern a secure, multi-account AWS environment based on best practices.

**Key Features:**

* Automates landing zone setup (multi-account architecture)
* Enforces guardrails (rules for security & compliance)
* Centralized logging and monitoring
* Account vending (automated account creation)

**Use Case:**  
Ideal for large organizations needing consistent governance across multiple AWS accounts.

**Analogy:**  
Think of it as a “campus planner” for AWS — it lays out the buildings (accounts), roads (networking), and rules (guardrails) for a well-organized cloud environment.

#### AWS Control Tower landing zone

It is the enterprise-wide container that holds all your organizational units (OUs), accounts, users, and resources that you want to regulate for compliance.

### AWS Service Catalog

With Service Catalog, you can create, share, and organize from a curated catalog of AWS resources. You can deploy baseline networking resources and security tools for new AWS accounts so you can govern consistently.

**Benefits:**Service Catalog saves time by making it quick to find and deploy approved, self-service cloud resources. It also helps you stay agile while improving governance over resources across multiple accounts.

**Purpose:**  
Allows organizations to create and manage catalogs of approved IT services (like EC2 instances, RDS databases, etc.).

**Key Features:**

* Centralized management of approved resources
* Role-based access control
* Version control and updates
* Integration with AWS Control Tower

**Use Case:**  
Useful for giving teams access to pre-approved resources while maintaining control over what they can deploy.

**Analogy:**  
Like an “app store” for internal cloud resources — users can pick from pre-approved items without needing admin access.

### AWS License Manager

License Manager is a service that helps you manage your software licenses and fine-tune your licensing costs.

**Benefits:** License Manager helps with visibility and control, tracking and managing licenses, and reducing the risk of noncompliance with licenses.

**Purpose:**  
Helps manage software licenses (like Windows, SQL Server) across AWS and on-premises environments.

**Key Features:**

* Track license usage
* Enforce licensing rules
* Integrate with AWS services (EC2, Systems Manager)
* Prevent license overuse

**Use Case:**  
Great for enterprises using licensed software and needing to stay compliant with vendor agreements.

**Analogy:**  
Think of it as a “license librarian” — it keeps track of who’s using what, and ensures no one breaks the rules.

### Comparison Table

| **Feature** | **AWS Control Tower** | **AWS Service Catalog** | **AWS License Manager** |
| --- | --- | --- | --- |
| Primary Function | Multi-account governance | Resource provisioning | License tracking & control |
| Target Users | Cloud admins | End users & admins | IT & compliance teams |
| Automation | Landing zone setup | Product deployment | License enforcement |
| Integration | With Service Catalog | With Control Tower | With EC2, Systems Manager |
| Governance | Guardrails | Role-based access | Licensing rules |

## Health of your AWS Cloud resources

**Notifications on service events**

AWS Health is the go-to data source for events and changes affecting the health of your AWS Cloud resources. It notifies you about service events, planned changes, and account notifications to help you manage and take actions.

### AWS Health Dashboard

With AWS Health Dashboard, you can view account-specific health information and get AWS Health event updates. You can also use AWS Health programmatically using the AWS Health API, which is available with AWS Premium Support.

**Benefits:**AWS Health Dashboard provides valuable information as a data source for events and changes. It gives you timely and actionable guidance to remedy issues. It also helps manage service health and is integrated and automated to use at scale.

**Use cases:**Use AWS Health Dashboard to view account-specific health information. You can also use it to plan for lifecycle events or troubleshoot an incident.

## Continuously evaluating your AWS environment

**To improve security, cost optimization, performance, and resilience**

### Trusted Advisor

Optimizing large scale cloud deployments is extremely important to do, and it's not a one-time thing. You must look for ways to optimize for costs, performance, security, and resilience. With AWS Trusted Advisor, you can continuously evaluate your AWS environment by using best practice checks across several categories. All AWS Support plans include access to dozens of Trusted Advisor checks. With Business Support and other advanced plans, you can benefit from hundreds of checks.

**Benefits:** Trusted Advisor helps you align with AWS best practices, prioritize recommendations, and optimize your AWS resources at scale.

**Use cases:**It can be used to optimize cost, efficiency, security, improve performance, and track service limits.

Although Trusted Advisor does check to optimize security, you might need help to check the fine-grained permissions of your AWS Identity and Access Management (IAM). IAM Access Analyzer can help meet your goals for least privilege access within your AWS environment.

### IAM Access Analyzer

IAM Access Analyzer provides capabilities to set, verify, and refine permissions by analyzing external access and validating that your policies match your corporate security standards.

**Benefits:** IAM Access Analyzer provides benefits like refining permissions, validating IAM policies, helping you meet your least privilege goals, and automating IAM policy reviews.

**Use cases:**It can be used to set fine-grained permissions, verify who can access what, remediate unused access, and refine and remove broad access.

# Pricing and Support

AWS offers customers a pay-as-you-go approach for pricing for the vast majority of our cloud services. With AWS, customers pay only for the individual services they need, for as long as they use them, and without requiring long-term contracts or complex licensing.

### Key concepts of AWS pricing

#### 1. Pay as you go

With pay as you go, you can adapt to changing business needs and reduce the risk of overprovisioning or missing capacity.

#### 2. Save when you commit

For certain services, such as Compute services on AWS, Savings Plans offer savings over On-Demand prices when you commit to a 1-year or 3-year plan.

#### 3. Pay less by using more

With AWS, you can realize important savings as your usage increases. For some services, pricing is tiered, meaning the more you use, the less you pay.

### Driving factors of cost

The pricing of AWS services varies based on several factors, such as service category or type, configuration, AWS Region, and which pricing model you choose. Refer to the pricing tab on a service's webpage for details on its specific pricing factors.

There are three fundamental drivers of cost with AWS: compute, storage, and outbound data transfer. These driving factors impact AWS service categories in different ways.

#### Compute

For compute resources, you pay by a certain span of time, like by the hour or by the second. Unless you've made a reservation for which the cost is agreed upon beforehand, you pay from the time you launch a resource until the time you stop the instance.

#### Storage

You can choose from a broad portfolio of storage solutions with deep functionality for storing, accessing, protecting, and analyzing data. Pricing for storage largely depends on how much storage you have provisioned or how much you are using.

For some storage options, such as Amazon Simple Storage Service (Amazon S3), storage cost is tiered. This means you can optimize storage costs based on how frequently and quickly you need to access data. With Amazon S3, consider the following six cost components when storing and managing customer data:

* Storage pricing
* Request and data retrieval pricing
* Data transfer and transfer acceleration pricing
* Data management and analytics pricing
* Replication pricing
* The price to process your data with Amazon S3 Object Lambda

#### Data transfer

In most cases, there is no charge for inbound data transfer or for data transfer between AWS services within the same Region. There are some exceptions, so be sure to verify data transfer rates before beginning.

**Outbound data transfer** is aggregated across services and then charged at the outbound data transfer rate. The more data you transfer, the less you pay per gigabyte. For data storage and transfer, you typically pay per gigabyte.

## AWS pricing and billing services

You learned about various AWS pricing and billing services and tools. These services are purpose-built to help you forecast, track, manage, and view your AWS costs. When you are first starting out with the AWS Cloud, they can be hard to tell apart. To help you differentiate the services, let's review these services and their key uses cases.

### 1. AWS Organizations

AWS Organizations provides centralized management and governance of your AWS environment. Using AWS Organizations, you can create, group, and manage accounts. You can also apply security policies at the account level and consolidate billing with multiple accounts using a single payment method.

Use cases:

* Consolidate multiple AWS accounts into one central organization.
* Implement organization-wide policies.

### 2. AWS Billing and Cost Management dashboard

The AWS Billing and Cost Management dashboard centralizes cost management, showing current charges, usage, forecasts, and detailed breakdowns. It also provides tools to manage payments, view invoices, set budgets, and consolidate billing.

Use cases:

* Use helpful visualizations and billing reports of monthly AWS spend.
* Set up and manage payment methods.

### 3. AWS Budgets

AWS Budgets helps set custom budgets and sends alerts when costs, usage, or Savings Plans and Reserved Instances (RIs) utilization or coverage exceed defined thresholds.

Use cases:

* Set up alerts for when projected costs exceed predefined thresholds.
* Forecast future expenses based on current usage trends.

### 4. AWS Cost Explorer

AWS Cost Explorer helps visualize, analyze, and manage AWS costs and usage with interactive graphs, reports, and forecasts. It provides insights into spending patterns, trends, and Reserved Instance recommendations.

Use cases:

* Analyze historical spending trends to identify cost-saving opportunities.
* Forecast future AWS costs based on current usage patterns to budget effectively.

### 5. AWS Pricing Calculator

Another helpful tool is the AWS Pricing Calculator. The AWS Pricing Calculatoris a web-based planning tool that you can use to create estimates. You input specific configurations such as instance types, storage options, and data transfer volumes. Then, based on your configurations, you receive a detailed cost breakdown to help you budget for your AWS resource allocation.

Use cases:

* Estimate potential costs before deployment.
* Compare costs of different AWS services and configurations.

### Comparison Table

| **Feature / Tool** | **AWS Organizations** | **Billing & Cost Management Dashboard** | **AWS Budgets** | **AWS Cost Explorer** | **AWS Pricing Calculator** |
| --- | --- | --- | --- | --- | --- |
| **Purpose** | Manage multiple AWS accounts centrally | View and manage billing, payments, and cost data | Set custom budgets and alerts | Analyze historical and forecasted costs | Estimate costs before using AWS services |
| **Key Use Case** | Centralized governance and billing | Track charges, pay bills, manage payment methods | Monitor spending and prevent overages | Visualize usage trends and cost drivers | Plan and compare pricing for services |
| **Scope** | Multi-account structure | Account-level billing and cost data | Budgeting across services/accounts | Detailed cost breakdowns | Pre-deployment cost estimation |
| **Timeframe Focus** | Ongoing account structure | Current and past billing periods | Future cost control | Past and forecasted usage | Future cost planning |
| **Integration** | IAM, SCPs, consolidated billing | AWS Budgets, Cost Explorer | CloudWatch, SNS | Budgets, Organizations | None (standalone tool) |
| **Pricing** | Free | Free | Free (up to 62 budgets/month) | Free | Free |

**Simple Analogies**

* **AWS Organizations**: Like a company’s HQ managing multiple branch offices.
* **Billing Dashboard**: Like your monthly bank statement showing all transactions.
* **AWS Budgets**: Like setting a monthly spending limit on your credit card.
* **Cost Explorer**: Like a detailed expense tracker app showing where your money goes.
* **Pricing Calculator**: Like checking the price of groceries before going shopping.

## AWS Support Plans

A diagram of a business support

AI-generated content may be incorrect.

AWS offers a range of support plans tailored to meet the needs of different customers, from those just getting started to large enterprises with complex requirements. Each plan builds onto the previous one, adding more advanced tools, personalized support, and faster response times to help you get the most out of your AWS experience.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Basic Support** | **Developer Support** | **Business Support** | **Enterprise On-Ramp Support** | **Enterprise Support** |
| Included for all AWS customers | Recommended for experimenting or testing in AWS | Recommended minimum tier for production workloads in AWS | Recommended for production and business critical workloads in AWS | Recommended for business critical and mission critical workloads in AWS |
| Includes access to docs, whitepapers, and AWS re:Post | Response times: • < 24 hours for general guidance • < 12 hours when systems impaired | Response times: • *Includes previous plan response times* • < 4 hours when production system impaired | Response times: •*Includes previous plan response times* • < 30 minutes when business-critical system is down | Response times: •*Includes previous plan response times* • < 15 minutes when business- or mission-critical system is down |
| Core AWS Trusted Advisor checks | Core AWS Trusted Advisor checks | Full set of AWS Trusted Advisor checks | Full set of AWS Trusted Advisor checks | Full set of AWS Trusted Advisor checks and prioritized recommendations by AWS account team |
| Technical Account Management not included | Technical Account Management not included | Technical Account Management not included | A pool of technical account managers (TAMs) provide proactive guidance | A designated TAM provides consultative architectural and operational guidance |

## AWS Marketplace and AWS Partner Network

### AWS Marketplace

The AWS Marketplace is a digital catalog that includes thousands of software listings from independent software vendors. You can use AWS Marketplace to find, test, and buy software that runs on AWS. For each listing in AWS Marketplace, you can access detailed information on pricing options and reviews from other AWS customers. Solutions and services offered in the AWS Marketplace include the following:

* **Software as a service (SaaS):**
* **Machine learning (ML) and AI**:
* **Data and analytics**:

### AWS Partner Network

The AWS Partner Network (APN) is a global community that uses AWS technologies, programs, expertise, and tools to build solutions and services for customers. Together, partners and AWS provide innovative solutions, solve technical challenges, and deliver customer value.

You can work with AWS Partners to create or use specialized solutions that are tailored to your unique business needs. For example, a retail company might use AWS to host their website. They could then work with an AWS Partner who specializes in advanced analytics and machine learning to improve customer personalization on that website.

# Migrating to Cloud

## Introduction to Migration

Cloud migration refers to the process of moving your organization's digital assets, IT resources, applications, and databases from on-premises infrastructure to the AWS Cloud. This involves strategic planning, implementation, and ongoing management to achieve a smooth and successful transition. It is not typically a *one and done* type of move

### Three Phases Of The Migration Process

AWS helps guide companies through the migration process, which can be divided into three sequential phases. From the first phase to the last, there are AWS services and tools to support your migration.

#### 1. Assess

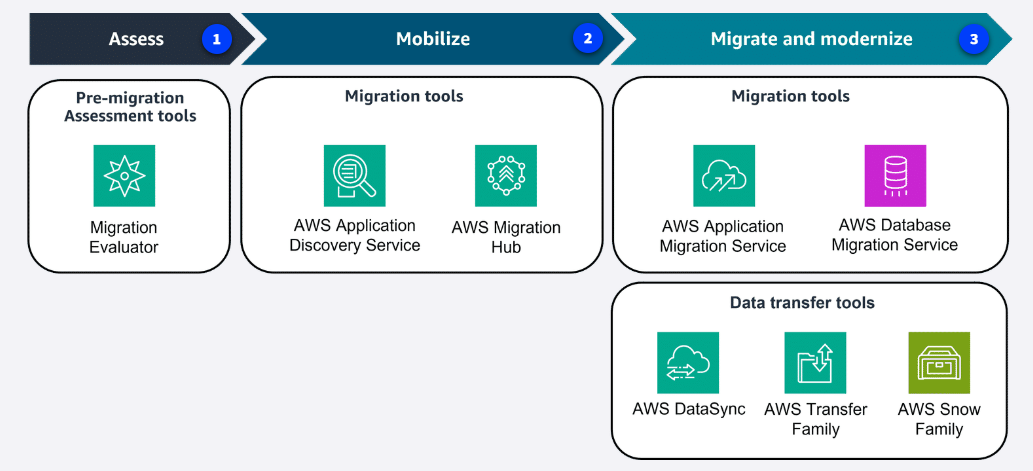
In this phase, you build the business case for the migration and assess your readiness. One of the services used in this phase is the Migration Evaluator.

#### 2. Mobilize

In this phase, you prepare the organization and mobilize the resources needed for the migration. Two services you might use in this phase are AWS Application Discovery Service and the AWS Migration Hub.

#### 3. Migrate and modernize

In this phase, you use your strategy, plan, and the best practices to migrate and modernize. Tools to support you include AWS Application Migration Service and AWS Database Migration Service (AWS DMS). If you are transferring data, you might use AWS DataSync, AWS Transfer Family, and the AWS Snow Family.



## Cloud Adoption Framework

**Improve your cloud readiness**

When you migrate to the AWS Cloud, there are critical planning tasks, many groups of stakeholders, and best practices to consider. With so much to take into account, it can seem overwhelming to even know where to start. That's where the AWS CAF can help.

### AWS CAF

The AWS CAF is a framework that brings AWS experience and best practices to companies preparing to migrate to the AWS Cloud. The framework provides tools to help accelerate the migration journey, organize resources, and align management during the transition.

**Benefits:**AWS CAF provides benefits for migrations to reduce business risk and improve sustainability and corporate transparency. Companies can grow revenue by creating new products and services in their cloud transformation. They can also reduce operational costs, increase productivity, and improve customer experience in their new cloud environment.

**Use cases:** You can use AWS CAF to migrate technology like legacy infrastructure and applications. You can also use it to migrate and optimize business processes, operations, and even create new business models with the move to the cloud.

### Functional And Business-Related Stakeholders

There are several groups of stakeholders and various parts of the business to consider in your migration planning and readiness.

#### 1. Business

The Business perspective makes sure that IT aligns with business needs and that IT investments link to key business results.

Use the Business perspective to create a strong business case for cloud adoption and prioritize cloud adoption initiatives. Make sure that your business strategies and goals align with your IT strategies and goals.

The following are common Business perspective roles:

* Business managers
* Finance managers
* Budget owners
* Strategy stakeholders

#### 2. People

The People perspective supports development of an organization-wide change management strategy for successful cloud adoption.

Use the People perspective to evaluate organizational structures and roles, assess new skill and process requirements, and identify gaps. This helps prioritize training, staffing, and organizational changes.

The following are common People perspective roles:

* Human resources
* Staffing
* People managers

#### 3. Governance

The Governance perspective focuses on skills and processes to align IT strategy with business strategy. This perspective helps you maximize business value and minimize risks.

Use it to understand how to update the staff skills and processes necessary to maintain business governance in the cloud. Manage and measure cloud investments to evaluate business outcomes.

The following are common Governance perspective roles:

* Chief information officer (CIO)
* Program managers
* Enterprise architects
* Business analysts
* Portfolio managers

#### 4. Platform

The Platform perspective includes principles and patterns for implementing new solutions in the cloud and migrating on-premises workloads to the cloud.

Use a variety of architectural models to understand and communicate the structure of IT systems and their relationships. Describe the architecture of the target state environment in detail.

The following are common Platform perspective roles:

* Chief technology officer (CTO)
* IT managers
* Solutions architects

#### 5. Security

The Security perspective makes sure that the organization meets security objectives for visibility, auditability, control, and agility.

Use AWS CAF to structure the selection and implementation of security controls that meet the organization’s needs.

The following are common Security perspective roles:

* Chief information security officer (CISO)
* IT security managers
* IT security analysts

#### 6. Operations

The Operations perspective helps you to enable, run, use, operate, and recover IT workloads to the level agreed upon with your business stakeholders.

Define how day-to-day, quarter-to-quarter, and year-to-year business is conducted. Align with and support the operations of the business. AWS CAF helps these stakeholders define current operating procedures and identify the process changes and training needed to implement successful cloud adoption.

The following are common Operations perspective roles:

* IT operations managers
* IT support managers

## Seven Migration Strategies

**The most common paths to the cloud**

When customers are migrating applications to the cloud, they can use seven common migration strategies. The decision on which strategy depends on factors such as the complexity of existing applications, business goals, time constraints, and available resources. Often, organizations will use a combination of these strategies across their application portfolio. By carefully considering each option and aligning it with specific applications and objectives, organizations can create a tailored migration plan that maximizes the benefits of cloud adoption while minimizing risks and disruptions.

#### 1. Relocate

Relocating is changing the hosting location to the cloud. This could be if applications are already virtual machines (VMs) or containers running on premises and then moving to the cloud.

#### 2. Rehost

Rehosting, also known as lift-and-shift, involves moving applications without changes.

In a scenario of a large legacy migration, in which the company is looking to implement its migration and scale quickly to meet a business case, the majority of applications are rehosted.

#### 3. Replatform

Replatforming, also known as lift, tinker, and shift, involves making a few cloud optimizations to realize a tangible benefit. Optimization is achieved without changing the core architecture of the application.

#### 4. Refactor

Refactoring, also known as re-architecting, involves reimagining how an application is architected and developed by using features built for the cloud. Refactoring is driven by a strong business need to add features, scale, or improve performance that would otherwise be difficult to achieve in the application’s existing environment.

#### 5. Repurchase

Repurchasing involves moving from a traditional license to a software-as-a-service (SaaS) model.

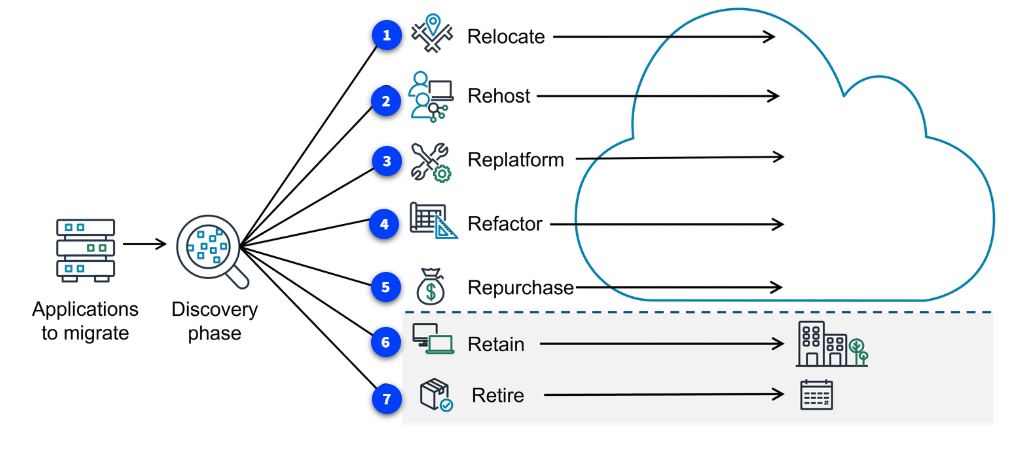
For example, a business might choose to implement the repurchasing strategy by migrating from a customer relationship management (CRM) system to a new sales force software.

#### 6. Retain

Retaining consists of keeping applications that are critical for the business in the source environment. This might include applications that require major refactoring before they can be migrated or work that can be postponed until a later time.

#### 7. Retire

Retiring is the process of removing applications that are no longer needed.



## Migration Services and Tools

### Assess Phase



In the Assess phase, it is important to have the right tools to build the business case for your migration. That's where Migration Evaluator can help.

#### Migration Evaluator

The Migration Evaluator is a migration assessment service that helps you create a business case for AWS Cloud planning and migration. It does this with a data driven approach, analyzing your current state, target state, and developing a migration readiness plan with projected cloud costs.

**Benefits:**The benefits include removing the guesswork when migrating. It provides visibility into multiple cost-effective cloud migration scenarios. It also gives insights on reusing existing software licensing, which can further reduce costs.

**Use cases:** You can use Migration Evaluator to conduct broad-based discovery, take a snapshot of your current on-premises footprint to fine-tune licensing, view server dependencies, and gain visibility into multiple migration scenarios. You can also use it to estimate and reduce your cloud costs.

### Mobilize Phase

A blue rectangle with black text

AI-generated content may be incorrect.

Let's look at two services that you might use in the Mobilize phase for discovery and planning.

#### Application Discovery Service

The Application Discovery Service discovers on-premises server inventory and connections. It gathers configuration, performance, and connection details for both servers and databases to create a detailed migration plan.

**Benefits:**With Application Discovery Service, you get a comprehensive snapshot of your on-premises inventory. You also can integrate discovery data with other services like Migration Hub and protect the data Application Discovery Service collects.

**Use cases:** You can use the AWS Application Discovery Service to conduct discovery and inventory, map the connections and dependencies, and generate a migration plan.

A computer network with a computer and a speaker

AI-generated content may be incorrect.

#### Migration Hub

The Migration Hub is a centralized hub to take you from discovery, assessment, planning, and execution of your migration. It provides tools, guidance, and automated recommendations to collaborate with your team and track your migration.

**Benefits:**With the Migration Hub, you have one location to go for your migration and, expert guidance in the form of prescriptive journey templates. Another benefit is cost savings because there is no charge to use Migration Hub.

**Use cases:** You can use Migration Hub for migration assessment and planning and migration completion and collaboration with teams. You can also use it for modernization efforts like fast-tracking application refactoring.



### Migrate And Modernize Phase

A blue rectangle with white background

AI-generated content may be incorrect.

As you start moving resources to the cloud, you might continue to use Migration Hub to track your progress. You can also use Migration Hub during migration to modernize your applications. In following lessons, you will explore database and data migrations. But first, let's look at another service that focuses on your application migration, Application Migration Service.

#### Application Migration Service

Application Migration Service is a tool to move and improve your on-premises and cloud-based applications. It helps customers streamline, expedite, and reduce the cost of migrating and modernizing applications.

**Benefits:**The benefits include support to migrate from any source infrastructure that runs a supported operating system (OS). It makes it possible to modernize your applications during migration. You can maintain normal business operations during the application replication process and also reduce costs by using one tool for a wide range of applications.

**Use cases:** You can use the Application Migration Service for on-premises applications running on physical servers or infrastructure, cloud-based applications, or moving between AWS Regions. You can also use it to modernize applications.

## Database Migration

### Migrating Databases

**Services to plan, migrate, and, if needed, convert**

Migrating your database to the cloud can provide an opportunity to redesign and improve your database architecture. Because legacy systems and requirements change over time, you might also consider whether to migrate to an AWS managed database service or an open-source database to reduce licensing costs. It is fairly simple to migrate data from one server to another when both are using the same database engine, known as homogeneous migration. Going to a different engine, or heterogeneous migration, is more complex and might require changes in your application. The good news is, whether your migration is homogenous (from same type of database to same type) or heterogenous (from one type of database to a different type), there are AWS services to support you.

### AWS DMS

The AWS Database Migration Service (AWS DMS) makes it possible to quickly and securely migrate databases and perform ongoing data replication tasks for live databases and data warehouses. It provides a way to plan, assess, convert, and migrate databases even with data warehouses in one central tool.

**Benefits:**AWS DMS provides benefits for migrating databases including maintaining high availability and low downtime during the migration process. It supports homogenous and heterogenous migrations. It also makes it possible to migrate terabyte sized databases at a low cost.

**Use cases:** You can use AWS DMS to move to managed databases, remove licensing costs, replicate ongoing changes in your database, and improve integration with data lakes.

If you want to change from a commercial database to an open-source database during your migration, there are a few important logistics. When the databases you are migrating have different source and target engines, you need to consider how to recreate existing resources, like the database schema, in the target. A schema defines the structure and organization of data inside the database and acts like a blueprint for things like table structures, field types, and relationships between items. It can be time consuming to recreate everything manually. That's where AWS Schema Conversion Tool (AWS SCT) can help.

A cartoon of a duck and a bird

AI-generated content may be incorrect.

### AWS SCT

AWS SCT makes it possible to convert database schemas and code objects (like stored procedures, views, and functions) from one database engine to another. AWS SCT can even give you estimates of how big of an effort a conversion is, which helps with planning.

**Benefits:** AWS SCT provides benefits to simplify database migrations by automating schema analysis, recommendations, and conversion at scale. It is compatible with popular databases and analytics services as source and target engines. It can save weeks or months of manual time and resources, which are typically required in conversions.

**Use cases:** You can use AWS SCT to move from commercial databases to open-source databases. You can also use it for migrating large data warehouse workloads and modernizing or updating database schemas in place.

## AWS Migration Hub vs AWS Database Migration Service

| **Feature** | **AWS Migration Hub** | **AWS Database Migration Service (DMS)** |
| --- | --- | --- |
| **Purpose** | Centralized tracking of migration progress across AWS tools | Migrate databases from on-premises or cloud to AWS |
| **Functionality** | Visibility into application migration status | Actual migration of database data |
| **Scope** | Broad – supports server, application, and database migrations | Narrow – focused only on databases |
| **Migration Execution** | Does **not** perform migration itself | **Performs** the migration of data |
| **Integration** | Works with DMS, Server Migration Service, Application Migration Service | Can be tracked via Migration Hub |
| **Monitoring** | Tracks progress, status, and metrics | Monitors migration tasks and replication |

**Simple Analogy**

* **Migration Hub**: Like a **dashboard** showing the progress of all your moving trucks.
* **DMS**: Like the **truck** that actually moves your database from one place to another.

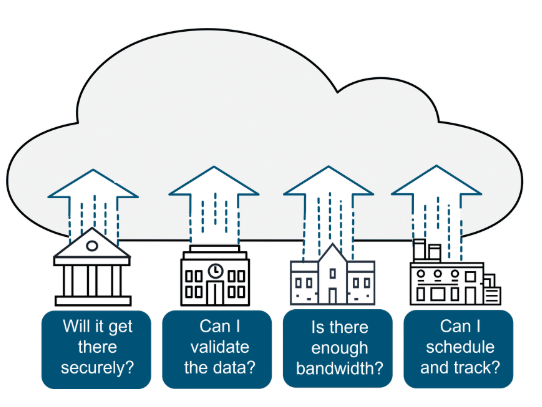
## Transferring Data to And from The AWS Cloud

### Online Data Transfer

Several AWS services facilitate online data transfer to the AWS Cloud. In the last lesson, you learned that AWS Database Migration Service (AWS DMS) transfers the database and its data to the AWS Cloud. In this lesson, you will identify services that can help with the considerations of online transfer for other types of data and files.

In this lesson, you will review three services:

* AWS DataSync
* AWS Transfer Family
* AWS Direct Connect



When you migrate data to the AWS Cloud, there are a few considerations to keep in mind. You need to ensure security (will it get there safely), data validation (will it get there in one piece), scheduling (when is the best time). You would also confirm bandwidth requirements. For the majority of data migration workloads, AWS DataSync will do the job.

#### 1. AWS DataSync

AWS DataSync is specifically designed for automating and accelerating data transfer. DataSync simplifies and accelerates moving large amounts of data between on-premises storage and AWS storage services like Amazon Simple Storage Service (Amazon S3). It automates many aspects of the transfer process, including running instances, encryption, and network optimization.

**Benefits:**The benefits include streamlining and accelerating secure data migrations. DataSync manages data movement workloads with bandwidth throttling, migration scheduling, task filtering, and task reporting. It also provides rapid data replication.

**Use cases:** You can use DataSync to migrate your data, archive your cold data, and manage hybrid data workflows.

The next service provides fully managed support for file transfers into and out of Amazon S3. It supports many different transfer protocols. A transfer protocol is a set of rules or standards that govern the way data is moved from one location to another.

#### 2. AWS Transfer Family

The AWS Transfer Family makes it possible to seamlessly manage and share data with simple, secure, and scalable file transfers. This service provides fully managed support for secure file transfers over FTP, Secure File Transfer Protocol (SFTP), File Transfer Protocol Secure (FTPS), and other protocols. It helps you transfer files directly into and out of AWS storage services like Amazon S3 and Amazon EFS.

**Benefits:**The benefits include simplifying the process of setting up and managing file transfers and reducing the need for complex infrastructure management. The Transfer Family provides secure data transfer with encryption and authentication, to ensure data integrity and confidentiality. It is built to scale and streamline workflows.

**Use cases:** You can use the Transfer Family to modernize and manage your file transfers, simplify data sharing with your workforce and partners, and integrate transactional business data into a unified data lake.

You might remember Direct Connect from the networking module. With the bandwidth of a dedicated connection, it is also a great solution for moving your data online to the AWS Cloud when migrating.

#### 3. Direct Connect

AWS Direct Connect is a service that makes it possible for you to establish a dedicated private connection between your network and virtual private cloud (VPC) in the AWS Cloud. Because it is your dedicated connection, it is a fast, reliable, and secure way to transfer your data or files.

**Benefits:** Direct Connect helps reduce network costs and increase amount of bandwidth.

#### Comparison Table

| **Feature** | **AWS DataSync** | **AWS Transfer Family** | **AWS Direct Connect** |
| --- | --- | --- | --- |
| **Purpose** | Automate data transfer between on-premises and AWS | Enable managed file transfers using SFTP, FTPS, FTP | Establish dedicated network connection to AWS |
| **Use Case** | Bulk data migration, sync jobs | Secure file exchange with partners or legacy systems | Low-latency, high-bandwidth connection for hybrid cloud |
| **Data Type** | Files (NFS, SMB, HDFS) | Files (via FTP/SFTP/FTPS) | All types (network-level access) |
| **Pricing Model** | Pay per GB transferred | Pay per endpoint/hour + data | Port hours + data transfer fees |
| **Ideal For** | Scheduled or automated data sync | Legacy systems, B2B file exchange | Enterprises needing consistent performance |

**Simple Analogies**

* **DataSync**: Like a **moving service** that regularly syncs your files between two homes.
* **Transfer Family**: Like a **secure courier** that picks up and drops off files using standard protocols.
* **Direct Connect**: Like installing a **private road** between your office and AWS for faster, safer travel.

**When to Use What**

| **Scenario** | **Recommended Service** |
| --- | --- |
| Migrating large datasets from on-prem to AWS | **DataSync** |
| Exchanging files with external partners securely | **Transfer Family** |
| Need consistent, low-latency connection to AWS | **Direct Connect** |

### Offline Data Transfer

Many customers prefer online migrations, but there are some customers who need to transfer data offline. An example would be if bandwidth is limited, or in remote locations with no internet and Direct Connect is not an option. Or, in cases with large data volumes, sending petabytes of data over the internet would take longer than simply sending a physical device. In the following section, you will review AWS Snowball Edge Storage Optimized devices.

#### 1. Snowball Edge Storage Optimized devices

AWS Snowball Edge Storage Optimized devices are a great solution for offline data migration where connecting to the internet might not be an option. These devices deliver high performance NVMe storage, making it possible to simplify multi-petabyte data migrations from on-premises locations to AWS.

**Benefits:** The benefits include delivering better compute performance and larger storage capacity with gigabytes of data per second for data migration workloads with offline requirements.

**Use cases:**You can use Snowball Edge devices for data migration when offline migration is required. They can also be used for edge computing when a secure, rugged device is needed.

# AWS Specialized Services

**Types of services**

AWS services are purpose-built for specific use cases. In the following section, you will learn more about the following four types of specialized AWS services:

* Development services
* Business application services
* End-user computing services
* IoT services

## Development Services

AWS offers several services to help developers automate CI/CD pipelines, monitor and debug applications, build GraphQL APIs, and deploy web and mobile applications on AWS. Let's examine these services in a bit more detail.

### What is CI/CD?

* **CI (Continuous Integration)**: Automatically build and test code whenever developers make changes.
* **CD (Continuous Delivery/Deployment)**: Automatically release code to production or staging environments.

**Real-Life Analogy:**

Imagine building and delivering pizzas:

* **CI** is like preparing the dough, adding toppings, and baking automatically every time a new order comes in.
* **CD** is like packaging and delivering the pizza to the customer without manual steps.

### 1. AWS CodeBuild

* **Role in CI/CD**: Builds and tests the code automatically.
* **Use Case**: After code is pushed, CodeBuild compiles and runs tests.
* **Key Points**:
  + Fully managed build service.
  + Scales automatically.
* **Analogy**: Like a **pizza oven** that bakes pizzas (code) automatically when ingredients (code changes) are added.

### 2. AWS CodeDeploy

* **Role in CI/CD**: Deploys the built code to servers or Lambda.
* **Use Case**: Push updates to EC2, Lambda, or on-prem servers.
* **Key Points**:
  + Supports blue/green and rolling deployments.
  + Reduces downtime and errors.
* **Analogy**: Like a **delivery van** that takes the pizza (app) to customers (servers).

### 3. AWS CodeCommit

* **Role in CI/CD**: Stores your source code securely.
* **Use Case**: Developers push code changes to CodeCommit.
* **Key Points**:
  + Git-based repository.
  + Integrated with other AWS DevOps tools.
* **Analogy**: Like a **recipe book** where chefs (developers) write and update pizza recipes (code).

### 4. AWS CodePipeline

* **Role in CI/CD**: Orchestrates the entire CI/CD process.
* **Use Case**: Automates the flow from CodeCommit → CodeBuild → CodeDeploy.
* **Key Points**:
  + Visual workflow.
  + Integrates with AWS and third-party tools.
* **Analogy**: Like a **conveyor belt** in a pizza factory that moves the pizza from prep to oven to delivery.

### 5. AWS X-Ray

* **Role in CI/CD**: Monitors and traces app performance after deployment.
* **Use Case**: Debug and analyze issues in production.
* **Key Points**:
  + Shows latency, errors, and service maps.
  + Works with microservices and Lambda.
* **Analogy**: Like a **quality inspector** who checks if the pizza tastes good and finds what went wrong if it doesn’t.

### 6. AWS Amplify

* **Role in CI/CD**: Simplifies CI/CD for frontend and mobile apps.
* **Use Case**: Automatically build and deploy web/mobile apps from Git.
* **Key Points**:
  + Git-based CI/CD.
  + Hosting + backend integration.
* **Analogy**: Like a **pizza shop builder** that sets up everything—kitchen, delivery, and menu—for web/mobile apps.

### CI/CD Toolchain Comparison Table

| **Service** | **CI/CD Role** | **Use Case** | **Analogy** |
| --- | --- | --- | --- |
| **CodeCommit** | Source control | Store code | Recipe book |
| **CodeBuild** | Build & test | Compile code | Pizza oven |
| **CodeDeploy** | Deploy | Push updates | Delivery van |
| **CodePipeline** | Orchestration | Automate CI/CD | Conveyor belt |
| **X-Ray** | Monitoring | Debug performance | Quality inspector |
| **Amplify** | Full-stack CI/CD | Web/mobile apps | Pizza shop builder |

## Business Application Services

These services are ideal for managing business application needs such as customer service operations and email promotions. Let's review a couple of examples.

### Amazon Connect

Businesses can use this AI-powered contact center service to efficiently set up and operate a scalable customer service call center. Amazon Connect provides capabilities for call routing, recording, and analytics while integrating seamlessly with other AWS services.

**Use Case:**

A company wants to set up a customer support center quickly and cost-effectively, without buying expensive hardware or software.

**Key Points to Remember:**

* Fully managed **contact center** service.
* Supports **voice and chat** interactions.
* Pay-as-you-go pricing (based on usage).
* Integrates with other AWS services like **Lambda**, **Lex**, **S3**, and **DynamoDB**.
* Scalable—can handle 10 or 10,000 agents.
* Easy to set up and customize using a **visual flow designer**.

### Amazon Simple Email Service (Amazon SES)

Amazon SES is a scalable and cost-effective email service provider that can be integrated into any application for reliable, high-volume email automation. It helps businesses optimize the delivery of transactional and marketing emails, resulting in enhanced customer engagement.

## End-user computing services

In modern businesses, IT departments often need to provide remote access to resources like virtual desktops and applications. Let's explore some AWS services that can be used to set up these environments for employees.

### Amazon AppStream 2.0

AppStream 2.0 is a fully managed service that streams applications from the cloud directly to any compatible device. This includes software-as-a-service (SaaS) applications and applications converted from desktop to SaaS without code revisions. This provides instant access to powerful software without the need for high-end local hardware.

*In****SaaS****,**applications are hosted on the cloud and accessed through the internet, without the need for local installation or maintenance.*

### Amazon WorkSpaces

WorkSpaces is a fully managed cloud-based desktop computing service. With WorkSpaces, employees can securely access their work environment from any device with an internet connection. Employees can perform the same tasks as if they were on a physical office computer, while companies can benefit from cost-efficiency and easy administration.

### Amazon WorkSpaces Secure Browser (formerly Amazon WorkSpaces Web)

WorkSpaces Secure Browser is a fully managed remote enterprise browser. It provides a protected environment for employees to access private websites, SaaS applications, and the public internet. With WorkSpaces Secure Browser, IT departments don't need to manage specialized client software, infrastructure, or VPN connections.

### Comparison with Similar Services

| **Feature** | **AppStream 2.0** | **WorkSpaces** | **EC2 with Remote Desktop** |
| --- | --- | --- | --- |
| **Purpose** | Stream individual apps | Full virtual desktop | Custom remote desktop setup |
| **User Experience** | App-only in browser | Full desktop | Full desktop |
| **Setup Complexity** | Medium | Easy | High |
| **Use Case** | Access specific apps remotely | Virtual desktops for employees | Custom remote access |

## IoT services

Internet of Things (IoT) is a network of connected physical devices embedded with sensors and software that collect and exchange data over the internet. These devices can be monitored and controlled remotely to improve efficiency, provide new services, and enhance quality of life.

Let's explore an AWS service specifically designed for IoT management.

### AWS IoT Core

AWS IoT Core is a managed cloud service used to securely connect physical devices with cloud applications. It helps you create efficient IoT solutions by streamlining the complex process of ingesting, processing, and acting on device data. Device connections and data are secured with mutual authentication and end-to-end encryption, and you can choose from several communication protocols.

Some IoT solutions include the following:

* **Smart security cameras** – Home monitoring that sends alerts to your phone
* **Smart pet feeders** – A pet feeder that you can control remotely.
* **Smart irrigation systems** – A rain machine that adjusts watering based on weather and soil conditions.