# Amazon Cloud Computing

The on demand delivery of IT resources over the internet with pay as you go pricing.

# Amazon EC2 Service

Amazon EC2 (Elastic Compute Cloud) is a web service that provides secure, resizable virtual servers (instances) in the cloud. It allows users to run applications, store data, and scale resources up or down as needed, paying only for the compute power they use.

## Types

1. General Purpose: Balanced CPU, memory, and storage for a variety of workloads, such as web servers and app development.

2. Compute Optimized: High CPU performance for compute intensive tasks like gaming servers and batch processing.

3. Memory Optimized: Extra memory for data intensive applications, such as large databases and in memory caching.

4. Storage Optimized: High disk performance for tasks that require fast local storage, like big data and NoSQL databases.

5. Accelerated Computing: GPU and specialized hardware for machine learning, graphics rendering, and complex scientific computations.

## Pricing

Amazon EC2 offers flexible pricing options, which include:

1. On Demand : Pay only for the time the instance is running, with no upfront cost.

Example : Launch a virtual server for a few hours to test an app, paying hourly or per second pricing.

2. Savings Plans : Commit to a 1 or 3 year term to get discounted rates (up to 72% savings) on On Demand prices.

Example : A business commits to a 1 year term for a 30% discount on its web server costs.

3. Reserved Instances : Reserve instances for a 1 or 3 year term, reducing costs (up to 75% savings).

Example : A company reserves an instance for a 3 year term to save on a consistently needed database server.

4. Spot Instances : Use spare capacity at a reduced price (up to 90% off) but can be interrupted when demand rises.

Example : A data analyst uses spot instances to perform large computations overnight at a lower cost.

5. Dedicated Hosts : Rent physical servers for security or compliance requirements.

Example : A financial company needs dedicated servers for compliance, paying for exclusive hardware.

These pricing models give flexibility based on usage patterns and budget needs.

Here’s a simple overview of each:

1. \*\*Elastic Load Balancer (ELB)\*\*:

- Distributes incoming traffic across multiple EC2 instances to improve application availability and fault tolerance.

- Types include Application Load Balancer (ALB), Network Load Balancer (NLB), and Gateway Load Balancer (GLB), each designed for different use cases.

2. \*\*Simple Queue Service (SQS)\*\*:

- A fully managed message queuing service that lets you decouple and scale microservices or distributed systems.

- It ensures messages are stored reliably until they’re processed, allowing for asynchronous communication between services.

3. \*\*Simple Notification Service (SNS)\*\*:

- A fully managed publish-subscribe messaging service that sends notifications to multiple subscribers at once.

- Useful for sending alerts, pushing updates, or triggering workflows across distributed systems.

Together, these services help manage, scale, and streamline distributed application architecture on AWS.

# Selecting Region

Compliance: Location bounded data 🡪 (data must not get out of UK)🡪 Choose London.

Proximity: The nearest the faster.

### **Available services within a Region**

### **Pricing**

# **Amazon CloudFront**

a content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to users worldwide with low latency and high transfer speeds. It caches content at edge locations globally, reducing the load on origin servers and speeding up content delivery to users closer to these locations.

# Edge Location

An **edge location** is a site that Amazon CloudFront uses to store cached copies of your content closer to your customers for faster delivery.

# **Amazon Outposts**

a fully managed service that brings AWS infrastructure, services, and tools to your on-premises data center or facility. It allows you to run AWS services locally, giving you a consistent hybrid experience with low-latency access to on-premises applications.

# AWS Elastic Beanstalk

a platform-as-a-service (PaaS) that helps you deploy and manage applications without needing to handle the underlying infrastructure. You simply upload your code, and Elastic Beanstalk automatically manages the deployment, load balancing, scaling, and monitoring for you.

# AWS CloudFormation

a service that allows you to model, provision, and manage AWS resources using code. By defining infrastructure as code (IaC) in templates (JSON or YAML), CloudFormation automates the setup of complex AWS environments, making it easy to create, update, and replicate resources in a consistent manner.

VPC: Your private network in AWS to control and isolate resources.

Subnet: IP range within a VPC; public subnets connect to the internet, private stay internal.

Direct Connect: Private, fast, and secure link between your data center and AWS.

* **Security Groups**: Virtual firewalls for controlling inbound and outbound traffic to AWS resources. They allow or deny access at the instance level.
* **Network ACLs (Access Control Lists)**: Optional layer of security for controlling traffic at the subnet level. They allow or deny IP-based traffic rules for entire subnets.

Use **Security Groups** for instance-specific access and **ACLs** for broader subnet security.

* **DNS (Domain Name System)**: System for translating domain names (like example.com) into IP addresses so browsers can load resources.
* **Route 53**: AWS’s scalable DNS and domain registration service. It routes traffic to AWS resources and external websites and can also be used for health checks and load balancing.
* **CDN (Content Delivery Network)**: Network of servers that caches and delivers content from locations closer to users, reducing latency.

Together, **Route 53** handles routing, while a **CDN** like CloudFront delivers content quickly to global users.

### **Instance Store**

* **Definition**: Temporary block-level storage attached directly to an Amazon EC2 instance.
* **Data Lifecycle**: Data is deleted when the instance is stopped or terminated.
* **Best Use**: Temporary data storage (e.g., buffers or caches) that doesn’t need to persist beyond the instance's lifespan.

### **Amazon Elastic Block Store (EBS)**

* **Definition**: Persistent block-level storage for Amazon EC2 instances.
* **Data Lifecycle**: Data persists even if the EC2 instance is stopped or terminated.
* **Configuration**: Customize size, type, and performance settings for the volume.
* **Backups**: Use **EBS Snapshots** for incremental backups to save changes without duplicating unchanged data.

### **Amazon EBS Snapshots**

* **Definition**: Incremental backups of EBS volumes to protect data over time.
  + **First Snapshot**: Captures all data in the volume.
  + **Subsequent Snapshots**: Only saves blocks that have changed since the last snapshot.
* **Benefits**: Efficient storage and faster backups, since only modified data blocks are backed up.

Top of Form

Bottom of Form

### **Amazon S3 Overview**

* **Type**: Object storage; stores data as objects with data, metadata, and a unique key.
* **Uses**: Ideal for storing files like images, videos, documents, backups, and media files.
* **Features**: Unlimited storage (up to 5 TB per object), customizable permissions, versioning for tracking changes.
* **Update Behavior**: Entire object is updated when modified (unlike block storage where only changes are saved).

### **Amazon S3 Storage Classes**

1. **Amazon S3 Standard**
   * **For**: Frequently accessed data.
   * **Availability**: Minimum of 3 Availability Zones.
   * **Use Cases**: Websites, content distribution, analytics.
2. **Amazon S3 Standard-Infrequent Access (S3 Standard-IA)**
   * **For**: Infrequently accessed data with high availability.
   * **Availability**: Minimum of 3 Availability Zones.
   * **Cost**: Lower storage cost, higher retrieval cost.
3. **Amazon S3 One Zone-Infrequent Access (S3 One Zone-IA)**
   * **For**: Infrequent access data stored in a single Availability Zone.
   * **Cost**: Lower than Standard-IA.
   * **Use Case**: Data you can easily reproduce in case of an Availability Zone failure.
4. **Amazon S3 Intelligent-Tiering**
   * **For**: Data with changing or unknown access patterns.
   * **Features**: Automatically moves data to frequent/infrequent tiers based on access patterns.
   * **Cost**: Small monthly monitoring and automation fee.
5. **Amazon S3 Glacier Instant Retrieval**
   * **For**: Archived data needing immediate access.
   * **Retrieval Time**: Within milliseconds.
   * **Use Cases**: Archival storage needing fast access, e.g., regulatory documents.
6. **Amazon S3 Glacier Flexible Retrieval**
   * **For**: Data archiving with retrieval in minutes to hours.
   * **Cost**: Low-cost archival storage.
   * **Use Cases**: Long-term archives like older customer records.
7. **Amazon S3 Glacier Deep Archive**
   * **For**: Long-term data preservation accessed rarely (1-2 times per year).
   * **Retrieval Time**: 12–48 hours.
   * **Cost**: Lowest cost in AWS for archival storage.
8. **Amazon S3 Outposts**
   * **For**: Object storage on AWS Outposts for local data residency and performance needs.
   * **Use Cases**: Storing and accessing data close to on-premises applications.

Here’s a cheatsheet for **Amazon Elastic File System (Amazon EFS)** and a brief comparison with **Amazon EBS**:

### **Amazon Elastic File System (Amazon EFS)**

* **Type**: Scalable, fully managed file storage.
* **Access**: Supports multiple clients (applications, servers, users) accessing shared file folders.
* **Scaling**: Automatically grows and shrinks as files are added/removed, up to petabyte scale.
* **Availability**: Stores data across multiple Availability Zones, allowing concurrent access from all zones within a region.
* **On-Premises Access**: Accessible from on-premises resources using **AWS Direct Connect**.

### **Comparison: Amazon EBS vs. Amazon EFS**

| **Feature** | **Amazon EBS** | **Amazon EFS** |
| --- | --- | --- |
| **Storage Type** | Block storage, single Availability Zone | File storage, multiple Availability Zones |
| **Usage** | Persistent storage for a single EC2 instance | Shared file system across multiple EC2 instances |
| **Availability** | Single AZ; requires both EBS and EC2 in same AZ | Multi-AZ; accessible from all zones in a region |
| **Scaling** | Manual resizing | Automatic scaling |
| **On-Premises** | Not directly accessible | Accessible using AWS Direct Connect |

### **Amazon Relational Database Service (Amazon RDS) Cheat Sheet**

### **Overview**

* **Type**: Managed relational database service.
* **Purpose**: Allows running and managing relational databases in AWS without complex setup.
* **Key Features**:
  + Automated hardware provisioning, database setup, patching, and backups.
  + SQL-based storage and querying, making data easily accessible and scalable.

### **Benefits of Amazon RDS**

* **Time-Saving**: Reduces administrative tasks like provisioning and backup.
* **Scalability**: Optimized for performance, memory, and I/O needs.
* **Security**: Encryption at rest and in transit for data protection.
* **Integration**: Works well with other AWS services (e.g., AWS Lambda for serverless apps).

### **Supported Database Engines**

1. **Amazon Aurora**: High-performance, compatible with MySQL and PostgreSQL; replicates across 3 AZs.
2. **PostgreSQL**: Open-source, supports complex queries and data types.
3. **MySQL**: Widely used, ideal for web applications.
4. **MariaDB**: MySQL-compatible, optimized for performance.
5. **Oracle Database**: Enterprise-grade, suitable for large workloads.
6. **Microsoft SQL Server**: Ideal for .NET applications and Windows-based setups.

### **Amazon Aurora Specifics**

* **Compatibility**: MySQL and PostgreSQL.
* **Performance**: Up to 5x faster than MySQL and 3x faster than PostgreSQL.
* **Reliability**: Replicates 6 copies of data across 3 Availability Zones (AZs), continuous backup to Amazon S3.

### **Amazon DynamoDB Cheat Sheet**

### **Overview**

* **Type**: Nonrelational (NoSQL) key-value database.
* **Purpose**: Stores and queries large-scale, flexible data structures with rapid response times.

### **Key Features**

* **Data Structure**:
  + Uses tables for data organization.
  + Stores data as **key-value pairs** with attributes (e.g., Name, Address).
  + Flexible: Each item doesn’t need to have the same attributes.
* **Serverless**:
  + Fully managed with no need to handle server maintenance or software updates.
* **Automatic Scaling**:
  + Dynamically adjusts capacity to match data needs.
  + Maintains single-digit millisecond performance at any scale.

### **Best For**

* Applications needing **high performance** and **scalability**.
* Use cases like gaming, IoT, mobile apps, and real-time data analytics.

# Amazon Redshift

 is a data warehousing service that you can use for big data analytics. It offers the ability to collect data from many sources and helps you to understand relationships and trends across your data.

# Additional Database Services

### **Amazon DocumentDB**

[**Amazon DocumentDB**](https://aws.amazon.com/documentdb) is a document database service that supports MongoDB workloads. (MongoDB is a document database program.)

### **Amazon Neptune**

[**Amazon Neptune**](https://aws.amazon.com/neptune) is a graph database service.

You can use Amazon Neptune to build and run applications that work with highly connected datasets, such as recommendation engines, fraud detection, and knowledge graphs.

### **Amazon Quantum Ledger Database (Amazon QLDB)**

[**Amazon Quantum Ledger Database (Amazon QLDB)**](https://aws.amazon.com/qldb) is a ledger database service.

You can use Amazon QLDB to review a complete history of all the changes that have been made to your application data.

### **Amazon Managed Blockchain**

[**Amazon Managed Blockchain**](https://aws.amazon.com/managed-blockchain) is a service that you can use to create and manage blockchain networks with open-source frameworks.

Blockchain is a distributed ledger system that lets multiple parties run transactions and share data without a central authority.

### **Amazon ElastiCache**

[**Amazon ElastiCache**](https://aws.amazon.com/elasticache) is a service that adds caching layers on top of your databases to help improve the read times of common requests.

It supports two types of data stores: Redis and Memcached.

### **Amazon DynamoDB Accelerator**

[**Amazon DynamoDB Accelerator (DAX)**](https://aws.amazon.com/dynamodb/dax/) is an in-memory cache for DynamoDB.

It helps improve response times from single-digit milliseconds to microseconds.

### Shared Responsibility Model

* **AWS** and **Customer** share responsibility for cloud security.
* Responsibilities are divided into:
  + **Security of the Cloud** (AWS)
  + **Security in the Cloud** (Customer)

### AWS: **Security of the Cloud**

AWS secures the infrastructure hosting cloud resources:

* **Physical security:** Data centers and hardware
* **Network security:** Protects global infrastructure, including AWS Regions, Availability Zones, and edge locations
* **Virtualization security:** Host operating systems, virtualization layer

AWS provides **compliance reports** from third-party auditors for customer verification.

### Customer: **Security in the Cloud**

Customers secure everything they create and manage within AWS:

* **Content control:** Managing stored data and permissions
* **Service configuration:** Choosing and configuring AWS services
* **Access management:** Defining and managing user access rights
* **Resource security:** Configuring security groups, patching, and operating system updates on Amazon EC2

**AWS Identity and Access Management (IAM)** concepts and best practices:

### Key Components of IAM:

1. **IAM Users**:
   * Individual identities representing a person or application that interacts with AWS resources.
   * **Best Practice**: Create individual IAM users for each person needing access, giving each a unique set of credentials.
2. **IAM Policies**:
   * Documents that define permissions for AWS services and resources.
   * **Best Practice**: Follow the **least privilege** principle by granting only the permissions necessary for users to perform their tasks.
3. **IAM Groups**:
   * Collections of IAM users with assigned policies.
   * **Example**: A "Cashiers" group for users needing access to certain resources, simplifying permissions management.
   * **Best Practice**: Use groups to manage permissions based on job roles. For role changes, simply adjust group membership.
4. **IAM Roles**:
   * Temporary identities used to gain specific permissions.
   * Useful for situations where users need to switch tasks or access resources temporarily.
   * **Example**: An employee at a coffee shop switching from a "Cashier" to an "Inventory" role based on tasks.
   * **Best Practice**: Assign roles for short-term access rather than long-term permissions.
5. **Multi-Factor Authentication (MFA)**:
   * Adds an extra layer of security by requiring a second form of verification, like a one-time code.
   * **Best Practice**: Enable MFA for the root user and all IAM users to protect against unauthorized access.

### AWS Account Root User:

* The root user has complete access to all AWS resources and should not be used for daily tasks.
* **Best Practice**: Use the root user only for initial setup or account-specific tasks, such as changing account settings.

### AWS Organizations

* **Purpose**: Enables companies with multiple AWS accounts to manage them centrally, controlling permissions and applying policies across accounts.
* **Root**: The parent container for all accounts within the organization. When you create an organization, AWS sets up this root by default.

### Key Features of AWS Organizations:

1. **Service Control Policies (SCPs)**:
   * SCPs are central policies that define and restrict what services, resources, and API actions are accessible across accounts within an organization.
   * **Example**: Use an SCP to restrict certain accounts from accessing services that don’t meet specific security or regulatory standards.
2. **Consolidated Billing**:
   * Allows all accounts in an organization to be billed together under one payment method, simplifying billing management.
3. **Organizational Units (OUs)**:
   * **Purpose**: OUs allow you to group accounts with similar needs (business, security, or regulatory) to streamline management.
   * **Inheritance**: Policies applied to an OU are inherited by all accounts within that OU.
   * **Example**: If HR and Legal departments have similar AWS resource needs, grouping their accounts in an OU allows you to apply the same policy to both, while preventing them from accessing unrelated services.

### Example of AWS Organizations Use:

Imagine a company with AWS accounts for Finance, IT, HR, and Legal:

* **Finance & IT**: Managed under the organization for billing purposes but without specific OUs.
* **HR & Legal**: Placed in a shared OU, allowing a single SCP to apply permissions that meet both departments' requirements.

AWS Organizations supports centralized permissions and policy management, enhancing security and efficiency across multiple accounts.

### AWS Artifact

AWS Artifact provides on-demand access to AWS compliance reports and agreements, ensuring companies can meet industry-specific compliance standards.

#### 1. **AWS Artifact Agreements**

* **Purpose**: Allows companies to review, accept, and manage legal agreements with AWS.
* **Use Cases**: This section is particularly relevant for companies needing to comply with specific regulations, such as **HIPAA** for healthcare data.
* **Scope**: Agreements can be managed individually or across multiple accounts within AWS Organizations.

#### 2. **AWS Artifact Reports**

* **Purpose**: Provides compliance reports from third-party auditors verifying AWS’s adherence to global and industry-specific standards.
* **Use Cases**: Developers and compliance officers can access these reports to understand compliance requirements.
* **Common Reports Available**:
  + Compliance standards like **SOC**, **ISO**, **PCI DSS**, and industry-specific certifications.
* **Audit Evidence**: AWS Artifact Reports can be shared with auditors and regulators as proof of AWS’s security practices.

AWS Artifact is useful for maintaining compliance visibility, especially in regulated industries, by ensuring access to the latest compliance reports and relevant agreements.

### Denial-of-Service (DoS) Attack

* **Description**: An attempt to overload a website or application, making it unavailable for legitimate users by flooding it with excessive network traffic.
* **Example**: A single prank caller continuously calls a coffee shop, blocking other legitimate customers.

### Distributed Denial-of-Service (DDoS) Attack

* **Description**: A DoS attack using multiple sources (often a network of compromised devices, or “bots”) to generate traffic. This makes it harder to block the attack since it comes from many points.
* **Example**: Multiple prank callers flooding the coffee shop lines from different numbers, making it challenging to block each individually.

### AWS Shield

AWS Shield is a managed service designed to protect AWS applications from DDoS attacks.

#### AWS Shield Levels of Protection

1. **AWS Shield Standard**:
   * **Cost**: Free for all AWS customers.
   * **Protection**: Automatically safeguards against common, frequently occurring DDoS attacks. Uses analysis techniques to detect and mitigate malicious traffic in real-time.
2. **AWS Shield Advanced**:
   * **Cost**: Paid service for enhanced protection.
   * **Features**: Provides in-depth attack diagnostics and sophisticated attack mitigation, including detailed analytics.
   * **Integration**: Works with services like Amazon CloudFront, Amazon Route 53, and Elastic Load Balancing, allowing for custom rules in AWS WAF to counter complex DDoS attacks.

AWS Shield enables companies to minimize the risk of downtime due to DoS or DDoS attacks by offering robust, layered protection for different needs.

### AWS Key Management Service (AWS KMS)

* **Purpose**: AWS KMS enables encryption of data both **at rest** (in storage) and **in transit** (during transfer), using cryptographic keys.
* **Key Features**:
  + **Key Management**: Create, manage, and use cryptographic keys securely within AWS.
  + **Access Control**: Assign specific IAM users or roles to manage keys, or temporarily disable keys.
  + **Security**: Keys never leave AWS KMS, giving you full control.

### AWS WAF (Web Application Firewall)

* **Purpose**: AWS WAF protects your web applications from common attacks, such as SQL injection and cross-site scripting.
* **How it Works**:
  + Integrates with Amazon CloudFront and Application Load Balancer.
  + Uses a **web ACL (Access Control List)** to filter requests, blocking or allowing specific traffic.
  + Example: You can set up a web ACL to block requests from certain IP addresses known to be malicious while allowing legitimate traffic.

### Amazon Inspector

* **Purpose**: Amazon Inspector automates security assessments to identify vulnerabilities and security best practices within your applications.
* **Features**:
  + **Automated Assessments**: Checks for issues like open access to EC2 instances or outdated software.
  + **Findings Report**: Generates prioritized security findings with recommendations, aiding in remediation.
* **Shared Responsibility Model**: AWS manages infrastructure security, while you’re responsible for securing applications running on AWS.

### Amazon GuardDuty

* **Purpose**: Amazon GuardDuty provides threat detection by continuously monitoring AWS infrastructure and account activity.
* **How it Works**:
  + Monitors sources like VPC Flow Logs and DNS logs without the need for additional security software.
  + **Findings**: Reports threats and provides detailed recommendations for remediation.
  + **Automation**: Can trigger AWS Lambda functions to take automatic action on findings.

AWS’s security services help ensure data protection, robust application security, and proactive threat detection to keep your resources safe.

**Amazon CloudWatch** is a versatile web service for monitoring, managing, and responding to data insights across AWS resources. It offers powerful tools like metrics, alarms, and dashboards to ensure optimal performance and cost-efficiency.

### Key Features of Amazon CloudWatch

1. **Metrics**:
   * **Purpose**: Metrics capture and display key performance data of AWS resources.
   * **Data Representation**: AWS services send performance data points to CloudWatch, which can then be used to create graphs showing performance trends over time.
2. **CloudWatch Alarms**:
   * **Purpose**: Automate actions or notifications based on metric thresholds.
   * **Example Use Case**: If developers forget to stop EC2 instances after testing, CloudWatch alarms can automatically stop an instance when CPU utilization falls below a specified threshold, reducing unnecessary charges.
   * **Notifications**: Set up notifications to alert you when an alarm is triggered.
3. **CloudWatch Dashboards**:
   * **Purpose**: Provide a unified view of all metrics for your resources in a single, customizable interface.
   * **Examples**:
     + Monitor CPU utilization for EC2 instances.
     + Track request counts for S3 buckets.
   * **Customization**: Design separate dashboards for different applications, resources, or business needs.

Amazon CloudWatch ensures that you can monitor, automate, and take action on AWS performance data, making it a critical tool for effective AWS resource management.

4o

**AWS CloudTrail** is an essential auditing service that records API calls for your AWS account, providing a transparent and detailed activity history to enhance security and compliance. CloudTrail logs capture various details of each API call, including:

1. **Identity of the API Caller**: Identifies who made the API request.
2. **Time of the API Call**: Logs the exact time, helping you correlate actions to incidents.
3. **Source IP Address**: Shows the IP address used to make the call, allowing you to spot suspicious activity.
4. **Detailed Event Information**: Contains the type of resource involved, specific API actions, and any changes made.

### Practical Example: AWS CloudTrail Event

Imagine a scenario where a coffee shop owner discovers a new IAM user named Mary in their AWS account. Using **CloudTrail Event History**, they filter for the "CreateUser" action in IAM and find that on January 1, 2020, at 9:00 AM, IAM user John created this account through the AWS Management Console. This comprehensive detail helps them understand when and how the new user was added.

### CloudTrail Insights

For added security, **CloudTrail Insights** can automatically detect and alert on unusual API activity. For instance, if there is a sudden spike in the number of EC2 instances launched, Insights will notify you, allowing you to quickly assess and act on potential security incidents.

CloudTrail’s event history and Insights help ensure accountability, making it a fundamental part of any security-conscious AWS environment.