**What is Amazon S3 (Simple Storage Service)?**

**Amazon S3** is a **scalable, durable, object storage** service used to store and retrieve any amount of data from anywhere on the web.

It’s designed for **99.999999999% (11 9s)** durability and is widely used for **backups, file storage, hosting static websites, logs, big data, and more.**

**🧱 Core Concepts of S3**

| **Concept** | **Description** |
| --- | --- |
| **Bucket** | A container for storing objects (like folders) |
| **Object** | Any file (e.g., image, document, video) stored in a bucket |
| **Key** | The unique name (path) of an object in a bucket |
| **Region** | Location where the bucket is hosted (e.g., us-east-1) |

**🧠 Key Features**

| **Feature** | **Description** |
| --- | --- |
| **Object Storage** | Stores files as **objects**, not blocks or filesystems |
| **Highly Durable** | 11 9s durability — stored across multiple AZs |
| **Versioning** | Keep multiple versions of the same file |
| **Lifecycle Policies** | Automatically move or delete objects (e.g., archive after 30 days) |
| **Encryption** | Supports server-side and client-side encryption |
| **Access Control** | Bucket policies, IAM, and ACLs for fine-grained access |
| **Event Notifications** | Trigger AWS Lambda, SNS, SQS when an object is uploaded or deleted |
| **Static Website Hosting** | Host HTML/CSS/JS websites directly from S3 |

**📁 Storage Classes in S3**

S3 offers **different storage tiers** for cost and access frequency:

| **Storage Class** | **Use Case** | **Durability** | **Availability** | **Cost** |
| --- | --- | --- | --- | --- |
| **S3 Standard** | Frequent access (e.g., websites, logs) | 11 9s | 99.99% | 💲💲 |
| **S3 Intelligent-Tiering** | Auto-moves data between tiers | 11 9s | 99.9–99.99% | 💲→💲💲 |
| **S3 Standard-IA** | Infrequent access (monthly access) | 11 9s | 99.9% | 💲 |
| **S3 One Zone-IA** | Low-cost, single AZ infrequent data | 11 9s | 99.5% | 💲 |
| **S3 Glacier** | Archive (retrieval in minutes/hours) | 11 9s | Variable | 💲 |
| **S3 Glacier Deep Archive** | Long-term archival (retrieval in hours) | 11 9s | Variable | 💲 |

**🚀 Common Use Cases**

| **Use Case** | **How S3 Helps** |
| --- | --- |
| 🗃️ Backup and Restore | Central storage for system or DB backups |
| 📦 Data Lake | Store structured & unstructured data for analytics |
| 🌐 Static Website Hosting | Host frontend websites (HTML, CSS, JS) |
| 📈 Big Data & AI | Store large datasets for ML training and analytics |
| 📁 Application Storage | Store documents, logs, videos, etc. for apps |
| 🔁 Disaster Recovery | Geo-redundant backup of critical data |

**🔒 Security in S3**

| **Feature** | **Description** |
| --- | --- |
| **IAM Policies** | Control who can access which buckets/objects |
| **Bucket Policies** | JSON rules directly applied to buckets |
| **ACLs (Legacy)** | Object- and bucket-level access control |
| **S3 Block Public Access** | Block all unintended public access to buckets |
| **Encryption** | SSE-S3, SSE-KMS, SSE-C (for client-side) |

**What is Versioning in S3?**

**Versioning** allows you to preserve, retrieve, and restore **every version** of every object stored in a bucket.

**🔹 Key Benefits:**

* Recover from accidental deletions or overwrites.
* Retain multiple versions of a file for auditing or rollback.
* Works perfectly with **CRR**.

**🛠️ Steps to Enable Versioning**

**🔧 Using AWS Console:**

1. Go to **S3 console** → Select your **bucket**
2. Go to **Properties** tab.
3. Scroll to **Bucket Versioning**
4. Click **Edit** → **Enable** versioning → Save changes.

**What Are Lifecycle Policies in S3?**

**Lifecycle policies** automate the **transition** of data between storage classes or the **deletion** of objects after a certain period.

**🎯 Use Cases:**

* Move infrequently accessed data to cheaper storage (e.g., S3 Glacier)
* Automatically delete old logs or backups
* Reduce costs for long-term stored data

**🛠️ Example: Lifecycle Rule**

| **Object Prefix** | **Action** | **Time Condition** |
| --- | --- | --- |
| /logs/ | Transition to **S3 Standard-IA** | After 30 days |
| /logs/ | Transition to **S3 Glacier** | After 90 days |
| /logs/ | **Delete** permanently | After 365 days |

**🚀 How to Configure Lifecycle Rule (AWS Console)**

1. Go to **S3 Console** → Select your bucket
2. Go to the **Management** tab
3. Click **Create Lifecycle Rule**
4. Name your rule (e.g., archive-logs)
5. Scope:
   * Apply to all objects or use a **prefix** like logs/
6. Choose **Transitions**:
   * Move to **S3 Standard-IA** after 30 days
   * Move to **S3 Glacier Flexible Retrieval** after 90 days
7. Set **Expiration**:
   * Delete objects after 365 days
8. Review and Save

✅ Done! Now your bucket will manage data automatically based on your rules.

**✅ What is S3 Intelligent-Tiering?**

**S3 Intelligent-Tiering** is an **automatic, smart storage class** that monitors object access patterns and **automatically moves** objects between access tiers **to optimize cost**.

**🔹 Tiers in Intelligent-Tiering:**

| **Tier** | **Use Case** | **Cost** |
| --- | --- | --- |
| **Frequent Access** | Actively used data | High |
| **Infrequent Access** | Rarely accessed data | Lower |
| **Archive Instant Access** | Archived but instantly needed | Cheaper |
| **Deep Archive** (opt-in) | Long-term cold storage | Cheapest |

**💡 Why Use Intelligent-Tiering?**

* **No need to set rules manually** (unlike Lifecycle)
* Best for unpredictable access patterns
* Automatically reduces cost over time

**⚙️ How to Use S3 Intelligent-Tiering**

**During Upload:**

* Choose storage class: S3 Intelligent-Tiering

**Or Use Lifecycle Rule to Transition:**

* Create a rule to **move objects** to Intelligent-Tiering after X days.

**Requirements:**

* Object must be **over 128 KB**
* Must stay for **minimum 30 days** (to avoid early delete charges)

**🆚 Lifecycle Policies vs Intelligent-Tiering**

| **Feature** | **Lifecycle Policy** | **Intelligent-Tiering** |
| --- | --- | --- |
| Setup Effort | Manual rule configuration | Auto-managed after selection |
| Flexibility | Custom logic (prefix, tags, age) | Fully automated by AWS |
| Cost Optimization | Manual transitions based on expected use | Auto-optimizes based on real access patterns |
| Best For | Predictable data lifecycle | Unpredictable or varied access patterns |