AWS S3 101

AWS S3

Amazon S3 (Simple Storage Service) is one of the foundational services in the AWS suite and is widely used by businesses and individuals to store and retrieve any amount of data, at any time, from anywhere.

Overview of AWS S3

Feature	Description
Object Storage	S3 is an object storage service, meaning it is designed to store unstructured data (like photos, videos, backups, etc.) as objects within resources called "buckets".
Durability and Availability	AWS S3 is designed for 99.999999999% (11 9's) durability over a given year. This ensures that your data remains safe and intact.
Scalability	There's no limit to the amount of data you can store in S3, and it's designed to handle high request rates and traffic.
Data Organization	Data in S3 is organized into buckets (similar to directories) and objects (files).
Versioning	S3 supports versioning, allowing you to retain, retrieve, and restore every version of every object in your bucket.
Security	Offers features like bucket policies, ACLs (Access Control Lists), and server-side encryption (SSE) for data. Integrated with AWS Identity and Access Management (IAM) for access control.
Event Configuration	You can set up event notifications to trigger workflows, alerts, or other automated processes based on changes to your data.

AWS S3 Pricing Factors

AWS S3 pricing is based on several factors:

Factor	Description
Storage	You're billed per GB per month based on the amount of data stored.
Requests	Costs associated with the number and type of requests made (GET, PUT, COPY, etc.).
Data Transfer	While transferring data into S3 is typically free, transferring data out of S3 to the internet or other AWS regions incurs charges.
Additional Features	Features like versioning, monitoring with CloudWatch, data transfer acceleration, and others might have associated costs.
Storage Management	Using features like S3 Inventory, S3 Analytics, and S3 Object Tagging will also influence the total cost.

S3 Commands with AWS CLI

The AWS Command Line Interface (CLI) is a powerful tool that allows users to interact with AWS services, including S3, directly from the command line. Here's a list of some commonly used AWS S3 CLI commands:

AWS CLI Commands for S3

Command	Description
`aws configure`	Setup CLI with credentials, region, output format.
`aws s3 ls`	List all buckets.
`aws s3 mb s3://my-bucket-name`	Create new bucket.
`aws s3 rb s3://my-bucket-name`	Delete bucket.
`aws s3 ls s3://my-bucket-name`	List bucket contents.
`aws s3 cp localfile.txt s3://my-bucket-name/`	Copy local file to bucket.
`aws s3 cp s3://my-bucket-name/file.txt localfile.txt`	Copy file from bucket to local.
`aws s3 mv localfile.txt s3://my-bucket-name/`	Move local file to bucket (removes local copy).
`aws s3 rm s3://my-bucket-name/file.txt`	Delete file from bucket.

Amazon S3

Overview

Amazon S3 (Simple Storage Service) is one of the foundational building blocks of AWS, providing scalable object storage for the cloud.

Key Characteristics

- **Infinitely scaling** storage service
- Global service appearance, but buckets are region-specific
- Backbone for many websites and AWS services
- Primary use case: Object storage in the cloud

Primary Use Cases

- **Backup and Storage** - Reliable data backup solution
- **Disaster Recovery** - Geographic redundancy for critical data
- **Archive** - Long-term data retention
- **Hybrid Cloud Storage** - Bridge between on-premises and cloud
- **Application Hosting** - Host application assets and files
- **Media Hosting** - Store and serve images, videos, audio
- **Data Lakes & Big Data Analytics** - Centralized data repository
- **Software Delivery** - Distribute software packages
- **Static Website** - Host static web content

S3 Buckets

Buckets are containers that hold objects (files) in Amazon S3.

Key Properties

- **Global Namespace**: Bucket names must be globally unique across all AWS accounts and regions
- **Regional Resource**: Despite global namespace, buckets are created in specific regions
- **Directory-like Structure**: Conceptually similar to directories, but technically different

Naming Conventions

✅ Valid Naming Rules:

- 3-63 characters long
- Only lowercase letters, numbers, and hyphens
- Must start with lowercase letter or number
- Cannot be formatted as IP address

❌ Invalid Naming Rules:

- No uppercase letters
- No underscores
- Cannot start with `xn--` prefix
- Cannot end with `-s3alias` suffix

Example Valid Bucket Names

```
my-company-data-2024
user-uploads-prod
backup-logs-us-east
```

S3 Objects

Objects are the fundamental entities stored in S3 buckets.

Object Structure

Object Key (Path)

The key is the **full path** to the object within the bucket:

```
s3://my-bucket/my_file.txt
s3://my-bucket/my_folder1/another_folder/my_file.txt
```

Key Components:

- **Prefix**: `my_folder1/another_folder/`
- **Object Name**: `my_file.txt`

> **Important**: S3 has no true directory concept. What appears as folders are just keys with forward slashes.

Object Properties

- **Value**: The actual content/data of the file
- **Maximum Size**: 5TB (5,000 GB)
- **Large File Upload**: Files >5GB require multipart upload
- **Metadata**: Key-value pairs (system or user-defined)
- **Tags**: Up to 10 Unicode key-value pairs (useful for security/lifecycle)
- **Version ID**: Present when versioning is enabled

Security

S3 provides multiple layers of security through various mechanisms.

Security Model Overview

User-Based Security

- **IAM Policies**: Define which API calls specific users can make

Resource-Based Security

- **Bucket Policies**: Bucket-wide rules configured in S3 console
- **Object ACLs**: Fine-grained object-level permissions (can be disabled)
- **Bucket ACLs**: Less common bucket-level permissions (can be disabled)

Access Evaluation Logic

An IAM principal can access an S3 object if:

1. **User IAM permissions ALLOW it** OR **Resource policy ALLOWS it**
2. **AND** there's no explicit DENY

Bucket Policies

Bucket policies are JSON-based policies with the following structure:

Policy Components

- **Resources**: Specify buckets and objects
- **Effect**: Allow or Deny
- **Actions**: Set of API operations to Allow/Deny
- **Principal**: Account or user the policy applies to

Common Use Cases

- Grant public access to bucket
- Force encryption on object uploads
- Grant cross-account access

Example Policy Structure

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::my-bucket/*"
    }
  ]
}
```

Block Public Access Settings

Critical security feature designed to prevent accidental data leaks:

- **Company Data Protection**: Prevents unintended public access
- **Account-Level Setting**: Can be applied across entire AWS account
- **Best Practice**: Leave enabled unless bucket explicitly needs public access

---

Static Website Hosting

S3 can host static websites directly from bucket storage.

Configuration

- Enable static website hosting on bucket
- Specify index document (usually `index.html`)
- Configure error document (optional)

Website URLs

S3 provides region-specific website URLs:

```
http://bucket-name.s3-website-aws-region.amazonaws.com
http://bucket-name.s3-website.aws-region.amazonaws.com
```

Troubleshooting

- **403 Forbidden Error**: Ensure bucket policy allows public reads
- **404 Not Found**: Check object key and index document configuration

---

Why we need explicit bucket policy when public access is unblocked already?

Short Answer:

Simply unblocking public access during bucket creation is not enough to make objects publicly readable. You still need a **bucket policy** (or ACLs) that explicitly grants public read permissions. AWS separates *access control settings* from *actual permissions*, and both must align for public access to work.

1. "Block Public Access" Settings ≠ Permissions

When you create an S3 bucket, AWS enables **"Block Public Access"** by default (for security). If you disable those blocks (e.g., uncheck "Block all public access"), you're only **allowing** public access — **not granting it**.

Think of it like:

- **Block Public Access = A locked gate**
  → Disabling it = **unlocking the gate**
  → But people still need **permission (a ticket)** to enter → that's the **bucket policy**.

2. Bucket Policy is What Actually Grants Access

Even with public access "unblocked," S3 still requires an explicit policy that says:

> "Yes, anonymous users (`"*"` principal) can perform `s3:GetObject` on objects in this bucket."

Without this policy, requests to your files (e.g., `https://your-bucket.s3-website-region.amazonaws.com/index.html`) will return **403 Forbidden** — because S3 denies access by default.

3. Static Website Hosting Adds a Twist

When you enable **S3 static website hosting**, S3 serves content via a special **website endpoint** (e.g., `http://your-bucket.s3-website-us-east-1.amazonaws.com`).
→ This endpoint **only works if objects are publicly readable**.
→ And objects are only publicly readable if you have a **bucket policy (or ACLs)** allowing it.

> ⚠ Note: The website endpoint **does not respect IAM policies** — it relies solely on **resource-based policies (bucket policy)** or **ACLs**.

What Happens If You Skip the Bucket Policy?

- You'll see **"Access Denied"** errors when visiting your site.
- The S3 console might even show your files as accessible (because you're logged in as the owner), but anonymous users (i.e., visitors) cannot load them.

Best Practice

Use a minimal bucket policy like this (replace `your-bucket-name`):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::your-bucket-name/*"
    }
  ]
}
```

This grants public read access to all objects — exactly what a static website needs.

Bonus Tip

If you use **CloudFront** (recommended), you can actually keep the bucket **private** and grant access only to CloudFront using an **Origin Access Identity (OAI)** — which is more secure. But for pure S3 static hosting, the public bucket policy is necessary.

Versioning

Versioning protects against accidental file deletion and enables easy rollback.

Key Features

- **Bucket-Level Setting**: Enabled/disabled at bucket level
- **Version Tracking**: Same key creates new versions (1, 2, 3...)
- **Rollback Capability**: Easy restoration to previous versions
- **Delete Protection**: Accidental deletions can be reversed

Important Notes

- **Pre-versioning Files**: Files uploaded before versioning have version "null"

- **Suspending Versioning**: Doesn't delete existing versions
- **Best Practice**: Enable versioning for important data

Version Management

- Multiple versions of same object consume storage
- Each version is billed separately
- Delete markers are created when objects are "deleted"

Replication

S3 replication automatically copies objects between buckets.

Prerequisites

- **Versioning Required**: Must be enabled on both source and destination buckets
- **IAM Permissions**: Proper permissions for S3 replication service
- **Cross-Account Support**: Buckets can be in different AWS accounts

Replication Types

Cross-Region Replication (CRR)

- **Purpose**: Replicate across different AWS regions
- **Use Cases**:
  - Compliance requirements
  - Lower latency access
  - Cross-account replication

Same-Region Replication (SRR)

- **Purpose**: Replicate within same AWS region
- **Use Cases**:
  - Log aggregation
  - Live replication between production and test accounts

Replication Behavior

- **Asynchronous**: Copying happens in background
- **New Objects Only**: Only replicates objects created after enabling
- **Existing Objects**: Use S3 Batch Replication for existing objects
- **No Chaining**: Replication doesn't chain across multiple buckets

Delete Operations

- **Delete Markers**: Can optionally replicate delete markers
- **Version Deletions**: Deletions with version ID are NOT replicated (prevents malicious deletes)

Storage Classes

S3 offers multiple storage classes optimized for different use cases and cost requirements.

Overview of Classes

1. **S3 Standard** - General Purpose
2. **S3 Standard-IA** - Infrequent Access
3. **S3 One Zone-IA** - One Zone Infrequent Access
4. **S3 Glacier Instant Retrieval**
5. **S3 Glacier Flexible Retrieval**
6. **S3 Glacier Deep Archive**
7. **S3 Intelligent-Tiering**

Durability and Availability Concepts

Durability

- **Definition**: Probability that object won't be lost
- **S3 Durability**: 99.999999999% (11 9's) across multiple AZs
- **Practical Meaning**: If you store 10 million objects, expect to lose 1 object every 10,000 years
- **Consistency**: Same durability across all storage classes

Availability

- **Definition**: How readily available the service is
- **Measurement**: Percentage of time service is accessible
- **Variation**: Differs by storage class

Storage Class Details

S3 Standard - General Purpose

- **Availability**: 99.99% (53 minutes downtime/year)
- **Use Case**: Frequently accessed data
- **Performance**: Low latency, high throughput
- **Resilience**: Sustains 2 concurrent facility failures
- **Best For**: Big data analytics, mobile & gaming applications, content distribution

S3 Standard-IA (Infrequent Access)

- **Availability**: 99.9%
- **Purpose**: Less frequently accessed data requiring rapid access when needed
- **Cost**: Lower storage cost than Standard, but retrieval fees apply
- **Use Cases**: Disaster recovery, backups

S3 One Zone-IA

- **Availability**: 99.5%
- **Durability**: 99.999999999% in single AZ (data lost if AZ destroyed)
- **Cost**: Lower cost than Standard-IA

- **Use Cases**: Secondary backup copies, recreatable data

S3 Glacier Instant Retrieval

- **Retrieval**: Millisecond retrieval
- **Minimum Storage**: 90 days
- **Use Case**: Data accessed once per quarter
- **Cost**: Lower storage cost, retrieval fees apply

S3 Glacier Flexible Retrieval

- **Retrieval Options**:
  - Expedited: 1-5 minutes
  - Standard: 3-5 hours
  - Bulk: 5-12 hours (free)
- **Minimum Storage**: 90 days
- **Use Case**: Archive data with flexible retrieval needs

S3 Glacier Deep Archive

- **Retrieval Options**:
  - Standard: 12 hours
  - Bulk: 48 hours
- **Minimum Storage**: 180 days
- **Use Case**: Long-term archival, compliance

S3 Intelligent-Tiering

- **Automatic Optimization**: Moves objects between tiers based on access patterns
- **No Retrieval Charges**: No fees for accessing data
- **Monitoring Fee**: Small monthly monitoring cost
- **Tiers**:
  - Frequent Access (automatic): Default tier
  - Infrequent Access (automatic): Objects not accessed for 30 days
  - Archive Instant Access (automatic): Objects not accessed for 90 days
  - Archive Access (optional): Configurable 90-700+ days
  - Deep Archive Access (optional): Configurable 180-700+ days

S3 Storage Classes Comparison

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durability	99.999999999% == (11 9's)						
Availability	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99%	99.9%	99.9%
Availability Zones	>= 3	>= 3	>= 3	1	>= 3	>= 3	>= 3
Min. Storage Duration Charge	None	None	30 Days	30 Days	90 Days	90 Days	180 Days
Min. Billable Object Size	None	None	128 KB	128 KB	128 KB	40 KB	40 KB
Retrieval Fee	None	None	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Storage Cost (per GB per month)	$0.023	$0.0025 - $0.023	$0.0125	$0.01	$0.004	$0.0036	$0.00099
Retrieval Cost (per 1000 request)	**GET:** $0.0004 **POST:** $0.005	**GET:** $0.0004 **POST:** $0.005	**GET:** $0.001 **POST:** $0.01	**GET:** $0.001 **POST:** $0.01	**GET:** $0.01 **POST:** $0.02	**GET:** $0.0004 **POST:** $0.03 **Expedited:** $10 **Standard:** $0.05 **Bulk:** free	**GET:** $0.0004 **POST:** $0.05 **Standard:** $0.10 **Bulk:** $0.025
Retrieval Time	Instantaneous					**Expedited** (1 – 5 mins) **Standard** (3 – 5 hours) Bulk (5 – 12 hours)	**Standard** (12 hours) **Bulk** (48 hours)
Monitoring Cost (pet 1000 objects)		$0.0025					

Best Practices

Security Best Practices

1. **Enable MFA Delete** for versioned buckets
2. **Use Bucket Policies** for fine-grained access control
3. **Enable Block Public Access** unless public access is required
4. **Implement least privilege** access principles
5. **Use IAM roles** instead of user credentials when possible

Performance Best Practices

1. **Use appropriate storage class** for access patterns
2. **Enable Transfer Acceleration** for global users
3. **Use multipart upload** for large files (>100MB)
4. **Implement request rate optimization** for high-request workloads

Cost Optimization

1. **Lifecycle Policies**: Automatically transition objects to cheaper storage classes

2. **Delete Incomplete Multipart Uploads**: Clean up failed uploads
3. **Monitor Storage Usage**: Use S3 Analytics for insights
4. **Use S3 Intelligent-Tiering** for unpredictable access patterns

Operational Best Practices

1. **Enable versioning** for important data
2. **Set up replication** for critical data
3. **Use meaningful naming conventions** for buckets and objects
4. **Tag resources** for better organization and cost tracking
5. **Monitor access logs** for security and compliance

Key Takeaways

1. **S3 is foundational** - Core AWS service used by many other services
2. **Global namespace, regional buckets** - Understand the scope of resources
3. **Security is multi-layered** - Combine IAM, bucket policies, and encryption
4. **Storage classes matter** - Choose based on access patterns and cost requirements
5. **Versioning and replication** - Essential for data protection and compliance
6. **Static website hosting** - Simple way to serve web content
7. **Lifecycle management** - Automate cost optimization and compliance

---

Additional Resources:

- AWS S3 Storage Classes Comparison
- AWS S3 Pricing
- S3 Best Practices Guide