Handwritten Notes: Lecture 03 - Amazon S3

1. What is Amazon S3?

Amazon Simple Storage Service (S3) is an object storage service offering industry-leading scalability, data availability, security, and performance. It's designed to store and retrieve any amount of data from anywhere on the web.

S3 was one of the first services launched by AWS in 2006 and has become a fundamental building block for cloud applications. It provides a simple web interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.

Key characteristics:

- **Object-based storage**: Stores files as objects (not a file system or database)
  - Each object consists of data, metadata, and a key
  - Objects can range from 0 bytes to 5 terabytes in size
  - Metadata includes system metadata (e.g., date last modified) and user metadata (custom key-value pairs)
- **Virtually unlimited storage capacity**:
  - No limit to the total amount of data you can store
  - Individual buckets can contain an unlimited number of objects
- **Highly durable**: 99.999999999% (11 nines) durability
  - Data is automatically distributed across a minimum of three physical facilities
  - Designed to sustain the concurrent loss of data in two facilities
- **Highly available**: 99.99% availability SLA (Standard storage class)
  - Built for mission-critical applications
  - Redundant infrastructure ensures continuous access
- **Regional service**: Data stored within a specific AWS region
  - Data never leaves the region unless explicitly transferred
  - Helps meet data residency and compliance requirements
- **Pay-as-you-go pricing model**:
  - Only pay for what you use with no minimum fee

- Pricing factors include: storage used, requests made, data transferred, and storage management features

- **Strong consistency**: Read-after-write consistency for all operations
    - Any data written to S3 is immediately readable
    - No need to wait for data propagation

2. Different Components of S3

Core Components:

- **Buckets**: Container for objects stored in S3
    - Primary storage unit in S3
    - Must have a globally unique name across all of AWS
    - Owned by the AWS account that created it
    - Can contain an unlimited number of objects
    - Cannot be nested (buckets within buckets are not allowed)
    - Limited to 100 buckets per AWS account by default (can be increased)
- **Objects**: Files and metadata stored in S3 (up to 5TB in size)
    - Basic entity stored in S3
    - Consists of object data (the file) and metadata
    - Identified by a key within a bucket
    - Can be versioned to maintain multiple states
    - Can be encrypted at rest and in transit
    - Can have user-defined metadata attached (up to 2KB)
- **Keys**: Unique identifier for an object within a bucket
    - Similar to a file path in traditional file systems
    - Used to retrieve objects from S3
    - Can include prefixes (like folders) for organization
    - Maximum key length is 1,024 bytes
    - UTF-8 encoding supported
- **Regions**: Geographic location where S3 buckets are physically stored
    - Determines physical location of your data
    - Affects latency, costs, and regulatory compliance
    - Data never leaves a region unless explicitly transferred
    - Each region is completely independent
- **Endpoints**: URL entry point for the S3 web service
    - Region-specific access points for S3

    - Format: s3..amazonaws.com
    - Virtual-hosted style: .s3..amazonaws.com
    - Path-style: s3..amazonaws.com/
- **Access Points**: Named network endpoints with dedicated access policies
    - Simplify managing data access at scale
    - Each access point has its own DNS name
    - Can restrict access to specific VPCs
    - Allows for customized access policies per application

Storage Classes:

- **S3 Standard**: General-purpose storage for frequently accessed data
    - 99.99% availability and 11 nines durability
    - Low latency and high throughput
    - Designed for frequent access
    - No minimum storage duration or retrieval fees
    - Use cases: websites, content distribution, mobile applications
- **S3 Intelligent-Tiering**: Automatic cost optimization for data with unknown/changing access patterns
    - Automatically moves objects between access tiers based on usage patterns
    - Small monthly monitoring and automation fee per object
    - No retrieval charges or operational overhead
    - Same performance as S3 Standard
    - Use cases: long-lived data with unpredictable access patterns
- **S3 Standard-IA** (Infrequent Access): Infrequently accessed data with rapid access when needed
    - Lower storage cost than Standard, but retrieval fee applies
    - 99.9% availability (slightly lower than Standard)
    - Minimum storage duration of 30 days
    - Use cases: backups, disaster recovery files, older data still needed quickly
- **S3 One Zone-IA**: Lower-cost option for infrequently accessed data (single AZ)
    - Stores data in only one Availability Zone (less redundant)
    - 20% cheaper than Standard-IA
    - 99.5% availability
    - Same durability as other classes but vulnerable to AZ destruction
    - Use cases: secondary backups, easily reproducible data
- **S3 Glacier Instant Retrieval**: Archive storage with millisecond retrieval
    - Lowest cost storage for long-lived data accessed once per quarter
    - Millisecond retrieval times

  - Minimum storage duration of 90 days
  - Use cases: medical images, news media assets

- **S3 Glacier Flexible Retrieval** (formerly S3 Glacier): Low-cost archival storage

  - Retrieval options:
    - Expedited (1-5 minutes)
    - Standard (3-5 hours)
    - Bulk (5-12 hours)
  - Minimum storage duration of 90 days
  - Use cases: archival data, long-term backups

- **S3 Glacier Deep Archive**: Lowest-cost storage for long-term retention

  - Retrieval time of 12+ hours (standard)
  - Designed for data that might be accessed once or twice a year
  - Minimum storage duration of 180 days
  - Use cases: regulatory archives, long-term data preservation

- **S3 Outposts**: Object storage on AWS Outposts on-premises environments

  - Provides S3 APIs and features for on-premises data
  - Designed for workloads with local data processing needs
  - Use cases: applications requiring local data access, processing, and data residency

Management Features:

- **Lifecycle policies**: Automate transitions between storage classes

  - Define rules to automatically transition objects to different storage classes
  - Set expiration dates for objects to be automatically deleted
  - Apply rules based on object prefixes or tags

- **Versioning**: Keep multiple versions of objects

  - Preserves, retrieves, and restores every version of every object
  - Provides protection against accidental deletions and modifications
  - Once enabled, cannot be disabled (only suspended)
  - Maintains a complete version history even for deleted objects
  - Each version is charged at standard S3 rates
  - MFA Delete feature provides additional protection against accidental deletions

- **Replication**: Cross-region and same-region replication

  - Cross-Region Replication (CRR): Replicate objects across different AWS regions
  - Same-Region Replication (SRR): Replicate objects within the same region
  - Requires versioning to be enabled on both source and destination buckets
  - Replication can be configured for entire buckets or specific object prefixes

  - Can be used for compliance, disaster recovery, or data locality

- **Event notifications**: Trigger workflows based on bucket events

  - Notifications can be sent to SNS, SQS, or Lambda functions
  - Events include object creation, deletion, restoration, and more
  - Enable automated workflows and real-time processing

- **Analytics**: Data analysis capabilities for storage patterns

  - S3 Storage Class Analysis: Helps identify optimal lifecycle policies
  - S3 Storage Lens: Organization-wide visibility into object storage
  - S3 Inventory: Audit and report on objects and their metadata

3. Creating an S3 Bucket and Considerations

Creation Process:

1. Sign in to AWS Management Console
2. Navigate to S3 service
3. Click "Create bucket"
4. Choose a globally unique bucket name
5. Select the AWS Region
6. Configure bucket settings:
   - Object ownership
   - Public access settings
   - Bucket versioning
   - Default encryption
   - Object lock
7. Create the bucket

Key Considerations:

- **Bucket naming rules**:

  - Globally unique across all AWS accounts
  - 3-63 characters long
  - Only lowercase letters, numbers, dots, and hyphens
  - Must start with letter/number
  - Cannot be formatted as IP address
  - Should avoid using dots for SSL certificate compatibility with virtual-hosted style URLs

- **Region selection**:

    - Choose based on latency, compliance, and cost requirements
    - Data stored in a region never leaves that region unless explicitly transferred

- **Performance considerations**:

    - S3 automatically scales to high request rates
    - For high-throughput applications, use multipart uploads
    - Consider request rate design for prefix organization

4. Access Control on S3 Bucket

Multiple Layers of Security:

1. **IAM Policies**: Define user/role permissions to S3 resources
2. **Bucket Policies**: JSON-based access policies attached to buckets
3. **Access Control Lists (ACLs)**: Legacy method to grant permissions to buckets and objects
4. **S3 Block Public Access**: Override settings to prevent public access
5. **Presigned URLs**: Time-limited URLs for temporary access

Common Access Control Scenarios:

- **Private buckets**: Default setting, accessible only with proper AWS credentials
- **Public read access**: Allow anyone to read objects (e.g., for website hosting)
- **Public write access**: Allow anyone to upload objects (rarely used due to security concerns)
- **Authenticated read/write**: Allow specific AWS accounts or IAM users/roles to access

Example Bucket Policy (Public Read):

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

5. Hosting a Static Website Using S3

Setup Process:

1. Create an S3 bucket with a name matching your domain (e.g., example.com)
2. Enable "Static website hosting" in bucket properties
3. Specify index and error documents
4. Upload website files to the bucket
5. Set appropriate permissions:
    - Either make objects public
    - Or use a bucket policy to grant public read access
6. Access website via S3 website endpoint URL

For Custom Domain:

1. Create a bucket matching your domain name
2. Set up Amazon CloudFront distribution (optional but recommended)
3. Configure Route 53 to point to S3 website endpoint or CloudFront distribution
4. Upload SSL certificate via AWS Certificate Manager (if using CloudFront)

Benefits:

- Highly available and scalable
- Cost-effective (pay only for storage and data transfer)
- No server maintenance required
- Integrates with CloudFront for global content delivery

6. Best Practices While Creating S3 Bucket

Security Best Practices:

- **Block all public access** unless explicitly required
- Use **bucket policies** and IAM policies for access control
- Enable **default encryption** for all objects
- Use **VPC endpoints** for private access from VPC
- Enable **access logging** to track requests
- Implement **least privilege principle** for all users/roles

Performance Best Practices:

- Use **appropriate storage class** based on access patterns
- Implement **prefix naming strategy** for high-throughput use cases
- Use **multipart uploads** for large objects

- Consider **S3 Transfer Acceleration** for fast, global uploads
- Use **CloudFront** for frequently accessed content

Cost Optimization:

- Set up **lifecycle policies** to transition objects to cheaper storage classes
- Enable **S3 Intelligent-Tiering** for unpredictable access patterns
- Use **S3 Analytics** to identify cost-saving opportunities
- Configure **object expiration** for temporary data
- Monitor usage with **AWS Cost Explorer** and **S3 Storage Lens**

Data Management:

- Enable **versioning** for critical data
- Configure **cross-region replication** for disaster recovery
- Set up **event notifications** for workflow automation
- Use **object tagging** for fine-grained management
- Implement **inventory reports** for large buckets

7. S3 Bucket Broad Usage

Data Storage:

- **Backup and restore**: Durable storage for backups
- **Data archiving**: Long-term retention with Glacier
- **Data lakes**: Foundation for big data analytics
- **Content storage**: Media files, documents, and application assets

Application Integration:

- **Static asset hosting**: Images, videos, JavaScript, CSS files
- **Application data storage**: User-generated content
- **Software delivery**: Distribution of software packages
- **Mobile app backend**: Storage for mobile applications

Data Processing:

- **ETL workflows**: Source or destination for data pipelines
- **Big data processing**: Input/output for analytics jobs
- **Machine learning**: Training data and model storage
- **Log processing**: Centralized logging repository

Content Distribution:

- **Website hosting**: Static websites
- **Media distribution**: Video, audio, and image content
- **Software updates**: Patches and updates distribution
- **Global content delivery**: When combined with CloudFront

8. Production Use Cases for S3 Bucket

Case 1: Media Streaming Platform

- Store high-resolution video files in S3 Standard
- Use S3 event notifications to trigger transcoding workflows
- Implement CloudFront for global content delivery
- Transition older content to S3 Standard-IA after 30 days
- Generate presigned URLs for secure, time-limited access

Case 2: Enterprise Data Lake

- Store raw data in S3 buckets organized by data source
- Use S3 Select for efficient querying of specific data
- Implement S3 Access Points for team-specific access patterns
- Process data with AWS analytics services (Athena, EMR, Redshift Spectrum)
- Apply lifecycle policies to transition historical data to Glacier

Case 3: Mobile Application Backend

- Store user-generated content (photos, videos, documents)
- Use S3 Transfer Acceleration for fast global uploads
- Implement server-side encryption for sensitive data
- Generate thumbnails automatically using Lambda functions triggered by S3 events
- Scale seamlessly to millions of users without infrastructure management

Case 4: Compliance and Archiving Solution

- Store regulated data with S3 Object Lock for WORM (Write Once Read Many) compliance
- Implement S3 Inventory for regular auditing
- Use S3 Glacier Deep Archive for long-term retention (7+ years)
- Apply bucket policies enforcing encryption and access controls
- Enable S3 Access Logs for comprehensive audit trails

These notes were created based on Lecture 03 on Amazon S3. They cover the fundamental concepts, implementation details, and best practices for using Amazon S3 in various scenarios.

9. S3 Versioning In-Depth

Understanding S3 Versioning:

Versioning is a means of keeping multiple variants of an object in the same bucket. It enables you to preserve, retrieve, and restore every version of every object stored in your S3 bucket, providing an additional layer of protection against accidental deletions and modifications.

Key Versioning Concepts:

- **Version ID**: Each object version gets a unique identifier
- **Latest Version**: The most recently stored version of an object
- **Delete Markers**: Special placeholders that mark an object as deleted
- **Suspended State**: When versioning is paused but not disabled

Enabling and Managing Versioning:

1. **Enabling Versioning**:

    - Can be enabled at bucket creation or later
    - Once enabled, cannot be disabled (only suspended)
    - Applied at the bucket level (affects all objects)

2. **Versioning States**:

    - Unversioned (default for new buckets)
    - Versioning-enabled
    - Versioning-suspended

3. **Working with Versioned Objects**:

    - GET requests retrieve the latest version by default
    - Specific versions can be retrieved using version IDs
    - Each version is stored as a complete object (not just differences)
    - Each version incurs standard S3 storage charges

Deleting Versioned Objects:

- **Simple DELETE**: Creates a delete marker (latest version)
- **DELETE with version ID**: Permanently removes that specific version
- **Restoring deleted objects**: Remove the delete marker or restore a previous version

MFA Delete:

- Additional protection for versioned objects

- Requires two-factor authentication for:
    - Permanently deleting an object version
    - Suspending or disabling bucket versioning
- Must be enabled by the bucket owner using the AWS CLI

Versioning and Lifecycle Policies:

- Can be used together to manage storage costs
- Transition older versions to cheaper storage classes
- Expire (permanently delete) older versions after a specified time
- Example policy: Keep current version in Standard, move previous versions to Glacier after 30 days, delete versions older than 1 year

Versioning Best Practices:

- Enable versioning for critical data buckets
- Use lifecycle policies to manage costs of storing multiple versions
- Consider using replication with versioning for disaster recovery
- Monitor versioning usage with S3 Storage Lens
- Use MFA Delete for sensitive or regulated data

Versioning Limitations:

- Cannot be disabled once enabled (only suspended)
- Applies to all objects in the bucket
- Increases storage costs as each version is stored as a complete object
- Maximum 100 million versions per object
- No automatic cleanup of old versions (must use lifecycle policies)

10. Advanced S3 Features

S3 Select and Glacier Select:

- Query and retrieve specific data from objects without retrieving the entire object
- Uses simple SQL expressions to filter content
- Supports CSV, JSON, and Parquet formats
- Reduces data transfer and improves query performance
- Example use case: Analyzing log files or large datasets

S3 Object Lock:

- Write-once-read-many (WORM) model
- Prevents objects from being deleted or overwritten

- Two retention modes:
  - Governance mode: Users with special permissions can override
  - Compliance mode: No one can override protection, even the account root user
- Legal holds can be placed independent of retention periods
- Critical for regulatory compliance (SEC, FINRA, HIPAA)

S3 Batch Operations:

- Perform operations on billions of objects in a single request
- Operations include:
  - Copy objects between buckets
  - Set object tags or ACLs
  - Restore objects from Glacier
  - Invoke Lambda functions on objects
- Tracks progress, sends notifications, and provides reports

S3 Access Points:

- Named network endpoints attached to buckets
- Each access point has its own access policy
- Simplify managing access for shared datasets
- Can restrict access to specific VPCs
- Example: Different access points for different departments accessing the same data lake

S3 Multi-Region Access Points:

- Global endpoint that spans multiple regions
- Automatically routes requests to the lowest latency region
- Provides active-active configuration for data access
- Built-in failover if a region becomes unavailable
- Ideal for globally distributed applications

S3 Object Lambda:

- Add custom code to S3 GET requests
- Modify and process data as it is returned to an application
- Use cases:
  - Redacting personally identifiable information
  - Converting data formats
  - Enriching data with information from other sources
  - Compressing or decompressing files

S3 Storage Lens:

- Organization-wide visibility into object storage usage and activity
- Interactive dashboard with customizable metrics
- Provides recommendations for cost optimization
- Tracks metrics across accounts and regions
- Helps identify anomalies in storage patterns

S3 Inventory:

- Scheduled flat-file output of objects and metadata
- Helps audit and report on replication and encryption status
- Can be configured to run daily or weekly
- Output can be queried using Athena, Redshift, or other analytics tools
- Useful for large-scale compliance monitoring

These notes were created based on Lecture 03 on Amazon S3. They cover the fundamental concepts, implementation details, best practices, and advanced features for using Amazon S3 in various scenarios.