Storage

S3 (Simple Storage Service)

- S3 Bucket is an **object-based** storage, used to manage data as objects
- **S3 Object** is having:-
  - Value - data bytes of object (photos, videos, documents, etc.)
  - Key - full path of the object in bucket e.g. `/movies/comedy/abc.avi`
  - Version ID - version object, if versioning is enabled
  - Metadata - additional information
- S3 Bucket holds objects. S3 console shows virtual folders based on key.
- S3 is a universal namespace so bucket names must be globally unique (think like having a domain name)

```
https://<bucket-name>.s3.<aws-region>.amazonaws.com
or
https://s3.<aws-region>.amazonaws.com/<bucket-name>
```

- Unlimited Storage, Unlimited Objects from **0 Bytes** to **5 Terabytes** in size. You should use **multi-part upload** for Object size **> 100MB**
- All new buckets are **private** when created by default. You should enable **public access** explicitly.
- Access control can be configured using **Access Control List (ACL)** (deprecated) and **S3 Bucket Policies** (recommended)
- **S3 Bucket Policies** are **JSON** based policy for complex access rules at user, account, folder, and object level
- Enable **S3 Versioning** and **MFA delete** features to protect against accidental delete of S3 Object.
- Use **Object Lock** to store object using write-once-read-many (WORM) model to prevent objects from being deleted or overwritten for a fixed amount of time (**Retention period**) or indefinitely (**Legal hold**). Each version of object can have different retention-period.
- You can **host static websites** on S3 bucket consisting of HTML, CSS, **client-side JavaScript**, and images. You need to enable Static website hosting and Public access for S3 to avoid 403 forbidden error. Also you need to add **CORS Policy** to allow cross-origin request.

```
https://<bucket-name>.s3-website[.-]<aws-region>.amazonaws.com
```

- Generate a **pre-signed URL** from CLI or SDK (can't from the web) to provide temporary access to an S3 object to either upload or download object data. You specify expiry (say 5 sec) while generating url:-

```
aws s3 presign s3://mybucket/myobject --expires-in 300
```

S3 Select or **Glacier Select** can be used to query subset of data from S3 Objects using SQL query. S3 Objects can be CSV, JSON, or Apache Parquet. GZIP & BZIP2 compression is supported with CSV or JSON format with server-side encryption.

using `Range` HTTP Header in a GET Request to download the specific range of bytes of S3 object, known as **Byte Range Fetch**

You can create **S3 event notification** to push events e.g. `s3:ObjectCreated:*` to SNS topic, SQS queue or execute a Lambda function. It is possible that you receive single notification for two writes to a non-versioned object at the same time. Enable versioning to ensure you get all notifications.

Enable **S3 Cross-Region Replication** for asynchronous replication of object across buckets in another region. You must have **versioning** enabled on both **source** and **destination** side. Only **new S3 Objects** are replicated after you enable them.

Enable **Server access logging** for logging object-level fields object-size, total time, turn around time, and HTTP referrer. Not available with CloudTrail.

Use **VPC S3 gateway endpoint** to access S3 bucket within AWS VPC to reduce the overall data transfer cost.

Enable **S3 Transfer Acceleration** for faster transfer and high throughput to S3 bucket (mainly uploads), Create **CloudFront** distribution with OAI pointing to S3 for faster-cached content delivery (mainly reads)

Restrict the access of S3 bucket through CloudFront only using **Origin Access Identity (OAI)**. Make sure user can’t use a direct URL to the S3 bucket to access the file.

S3 Storage Class Types

Standard: Costly choice for very high availability, high durability and fast retrieval

Intelligent Tiering: Uses ML to analyze your Object’s usage and move to the appropriate cost-effective storage class automatically

Standard-IA: Cost-effective for infrequent access files which cannot be recreated

One-Zone IA: Cost-effective for infrequent access files which can be recreated

Glacier: Cheaper choice to Archive Data. You must purchase **Provisioned capacity**, when you require guaranteed **Expedite retrievals**.

Glacier Deep Archive: Cheapest choice for Long-term storage of large amount of data for compliance

S3 Storage Class	Durability	Availability	AZ	Min. Storage	Retrieval Time	Retrieval fee
S3 Standard (General Purpose)	11 9’s	99.99%	≥3	N/A	milliseconds	N/A
S3 Intelligent Tiering	11 9’s	99.9%	≥3	30 days	milliseconds	N/A
S3 Standard-IA (Infrequent Access)	11 9’s	99.9%	≥3	30 days	milliseconds	per GB
S3 One Zone-IA (Infrequent Access)	11 9’s	**99.5%**	**1**	30 days	milliseconds	per GB

S3 Storage Class	Durability	Availability	AZ	Min. Storage	Retrieval Time	Retrieval fee
S3 Glacier	11 9's	99.99%	≥3	90 days	**Expedite (1-5 mins)** Standard (3-5 hrs) Bulk (5-12 hrs)	per GB
S3 Glacier Deep Archive	11 9's	99.99%	≥3	180 days	**Standard (12 hrs)** Bulk (48 hrs)	per GB

- You can upload files in the same bucket with different **Storage Classes** like *S3 standard, Standard-IA, One Zone-IA, Glacier* etc.
- You can setup **S3 Lifecycle Rules** to transition current (or previous version) objects to cheaper storage classes or delete (expire if versioned) objects after certain days e.g.
    - transition from S3 Standard to S3 Standard-IA or One Zone-IA can only be done after 30 days.
    - transition from S3 Standard to S3 Intelligent Tiering, Glacier, or Glacier Deep Archive can be done immediately.
- You can also setup lifecycle rule to **abort multipart upload**, if it doesn't complete within certain days, which auto delete the parts from S3 buckets associated with multipart upload.

Encryption

- **Encryption is transit** between client and S3 is achieved via SSL/TLS
- You can add default encryption at bucket level and also override encryption at file level.
- **Encryption at rest - Server Side Encryption (SSE)**
    - **SSE-S3** AWS S3 managed keys, use AES-256 algorithm. Must set header: `"x-amz-server-side-encryption":"AES-256"`
    - **SSE-KMS** Envelope Encryption using AWS KMS managed keys. Must set header: `"x-amz-server-side-encryption":"aws:kms"`
    - **SSE-C** Customer provides and manage keys. HTTPS is mandatory.
- **Encryption at rest - Client Side Encryption** client encrypts and decrypts the data before sending and after receiving data from S3.
- To meet **PCI-DSS or HIPAA** compliance, encrypt S3 using SSE-C and Client Side Encryption

Data Consistency

- S3 provides **strong read-after-write consistency for PUTs and DELETEs** of objects. PUTs applies to both writes to new objects as well as overwrite existing objects.
- **Updates to a single key are atomic.** For example, if you PUT to an existing key from one thread and perform a GET on the same key from a second thread concurrently, you will get either the old data or the new data, but never partial or corrupt data.

AWS Athena

- You can use **AWS Athena** (Serverless Query Engine) to perform analytics directly against S3 objects using SQL query and save the analysis report in another S3 bucket.
- **Use Case:** one-time SQL query on S3 objects, S3 access log analysis, serverless queries on S3, IoT data analytics in S3, etc.

Instance Store

- Instance Store is temporary **block-based** storage physically attached to an EC2 instance
- Can be attached to an EC2 instance only when the instance is launched and cannot be dynamically resized
- Also known as **Ephemeral Storage**
- Deliver very low-latency and high random I/O performance
- Data persists on instance reboot, data **doesn't persist on stop or termination**

EBS (Elastic Block Store)

- EBS is **block-based** storage, referred as EBS Volume
- **EBS Volume** think like a USB stick
    - Can be attached to only one EC2 instance at a time. Can be detached & attached to another EC2 instance in that same AZ only
    - Can attach multiple EBS volumes to single EC2 instance. Data persist after detaching from EC2
- **EBS Snapshot** is a backup of EBS Volume at a point in time. You can not copy EBS volume across AZ but you can create EBS Volume from Snapshot across AZ. EBS Snapshot can copy across AWS Regions.
- Facts about EBS Volume **encryption**:-
    - All data at rest inside the volume is encrypted
    - All data in flight between the volume and EC2 instance is encrypted
    - All snapshots of encrypted volumes are automatically encrypted
    - All volumes created from encrypted snapshots are automatically encrypted
    - Volumes created from unencrypted snapshots can be encrypted at the time of creation
- EBS supports **dynamic changes in live production** volume e.g. volume type, volume size, and IOPS capacity without service interruption
- There are two types of EBS volumes:-
    - **SSD** for small/random IO operations, High IOPS means number of read and write operations per second, Only SSD EBS Volumes can be used as **boot volumes** for EC2
    - **HDD** for large/sequential IO operations, High Throughput means number of bytes read and write per second
- EBS Volumes with two types of RAID configuration:-
    - **RAID 0** (increase performance) two 500GB EBS Volumes with 4000 IOPS - creates 1000GB RAID0 Array with 8000 IOPS and 1000Mbps throughput
    - **RAID 1** (increase fault tolerance) two 500GB EBS Volumes with 4000 IOPS - creates 500GB RAID1 Array with 4000 IOPS and 500Mbps throughput

EBS Volume Types	Description	Usage
General Purpose SSD (gp2/gp3)	Max 16000 IOPS	boot volumes, dev environment, virtual desktop
Provisioned IOPS SSD (io1/io2)	16000 - 64000 IOPS, EBS Multi-Attach	critical business application, large SQL and NoSQL database workloads
Throughput Optimized HDD (st1)	Low-cost, frequently accessed, throughput intensive	Big Data, Data warehouses, log processing

EBS Volume Types	Description	Usage
Cold HDD (sc1)	Lowest-cost, infrequently accessed	Large data with lowest cost

EFS (Elastic File System)

- EFS is a **POSIX-compliant file-based** storage
- EFS supports **file systems semantics** - strong read-after-write consistency and file locking
- **highly scalable** - can automatically scale from gigabytes to petabytes of data without needing to provision storage. With **burst mode**, the throughput increase, as file system grows in size.
- **highly available** - stores data redundantly across multiple Availability Zones
- Network File System (NFS) that can be mounted on and **accessed concurrently by thousands of EC2** in multiple AZs without sacrificing performance.
- EFS file systems can be accessed by Amazon EC2 Linux instances, Amazon ECS, Amazon EKS, AWS Fargate, and AWS Lambda functions via a file system interface such as NFS protocol.
- **Performance Mode:**
  - **General Purpose** for most file system for low-latency file operations, good for content-management, web-serving etc.
  - **Max I/O** is optimized to use with 10s, 100s, and 1000s of EC2 instances with high aggregated throughput and IOPS, **slightly higher latency** for file operations, good for big data analytics, media processing workflow
- **Use case:** Share files, images, software updates, or computing across all EC2 instances in ECS, EKS cluster

FSx for Windows

- Windows-based file system supports **SMB** protocol & Windows **NTFS**
- supports **Microsoft Active Directory (AD) integration**, ACLs, user quotas

FSx for Lustre

- Lustre = Linux + Cluster is a **POSIX-compliant parallel linux file system**, which stores data across multiple network file servers
- High-performance file system for **fast processing of workload** with consistent **sub-millisecond latencies**, up to hundreds of gigabytes per second of throughput, and up to **millions of IOPS**.
- Use it for Machine learning, High-performance computing (HPC), video processing, financial modeling, genome sequencing, and electronic design automation (EDA).
- You can use **FSx for Lustre as hot storage** for your highly accessed files, and **Amazon S3 as cold storage** for rarely accessed files.
- **Seamless integration with Amazon S3** - connect your S3 data sets to your FSx for Lustre file system, run your analyses, write results back to S3, and delete your file system
- FSx for Lustre provides two deployment options:-
  - **Scratch file systems** - for temporary storage and short-term processing
  - **Persistent file systems** - for high available & persist storage and long-term processing