S3-Simple Storage Service

* S3 is a highly scalable, object based storage service. It is a REST service. You cannot store DBs and OS on S3 since it would require Block based storage.S3 is for applications where you need to store and retrieve data.
* Each File can be from 0 bytes to 5 TB.
* Files are stored in buckets. Buckets are like region based folders (and that they can stand in a network….)
* S3 is a universal namespace, so identifiers (like bucket name) in S3 must be unique.

NOTE: S3 is a universal namespace but the bucket is created region specific which means that the S3 bucket will be created in the region but will be visible everywhere.

* Bucket links look like:

<https://s3-eu-west-1.amazonaws.com/abcdxyz>

format: s3-region.amazonaws.com/bucket name

* When a file upload is successful it return HTTP 200 OK response

Data Consistency Model for S3

* S3 achieves high availability by replicating data across multiple servers/AZs.
* Read after write consistency for PUTS of new objects. This means:
* For a PUT request ,S3 synchronously stores data across multiple facilities before returning 200 OK response
* A process writes a new object to S3 and will be immediately able to read the object.
* A process writes a new object to S3 and immediately lists keys within its bucket. Until the change is fully propagated, the object might not appear in the list.
* Eventual Consistency for overwrite Puts and Deletes (can take some time to propagate)
* For Updates and Deletes to objects, the changes are eventually reflected and not available immediately.
* So it may happen that after updating, some read requests may show previous version of data or after deletion some requests may still show data for some time.
* Updates are ATOMIC i.e. any read request may return old data or new data but will never return partial or corrupted data.
* S3 does not have an object locking mechanism. If required we would need to build object-locking mechanism in our app ourselves.
* Updates are key based ; there is no way to make atomic updates across keys i.e. you cannot make the update of one key dependent on update of another key unless this is built in your application.

S3 Storage Model

* S3 is object based. It is simply a Key Value Store.
* Amazon guarantees 99.999999999% (11 nines) durability for S3 information.
* S3 stores objects lexicographically. So it is a very important design concept.eg if you want logs to be stored at the same location together, you cannot name them by dates. You should add a salt-like a common letter as a prefix.
* Objects consist of the following:
* Key – name of the object
* Value – the data (sequence of bytes)
* Version ID – (Important for versioning)
* Metadata
* Subresources :
* Access Control Lists – can be applied at object level or/and bucket level
* Torrent – supports Bit Torrent protocol
* Tiered Storage available – different storage classes are available
* Lifecycle Management
* Versioning
* Encryption
* Data can be secured by ACLs or Bucket Policies

S3 Storage Tiers / Classes

* S3 standard
* Offers high durability availability and performance for frequently accessed data.
* Applications - cloud apps, dynamic websites, content distribution, mobile and gaming apps, big data analytics etc
* Data is resilient even if one complete AZ gets destroyed
* Supports SSL encryption
* Lifecycle management for automatic migration of objects
* S3-IA (Infrequent Access)
* For data that is not frequently accessed but would require rapid access if needed.
* S3-RRS (Reduced Redundancy storage)
* Glacier