AWS S3 Object Versioning

# S3 Object Versioning

* S3 Object Versioning can be used to protect from **unintended overwrites and deletions**
* Versioning helps to keep multiple variants of an object in the same bucket and can be used to preserve, retrieve, and restore every version of every object stored in the S3 bucket.
* **As Versioning maintains multiple copies of the same objects as whole and charges accrue for multiple versions for e.g. for a 1GB file with 5 copies with minor differences would consume 5GB of S3 storage space and you would be charged for the same.**
* Versioning is not enabled by default and has to be explicitly enabled for each bucket
* **Versioning once enabled, cannot be disabled and can only be suspended**
* **Versioning enabled on a bucket applies to all the objects within the bucket**
* **Permissions are set at the version level. Each version has its own object owner; an AWS account that creates the object version is the owner. So, you can set different permissions for different versions of the same object.**
* **Irrespective of the Versioning, each object in the bucket has a version.**
  + **For Non Versioned bucket, the version ID for each object is null**
  + **For Versioned buckets, a unique version ID is assigned to each object**
* **With Versioning, version ID forms a key element to define uniqueness of an object within an bucket along with the bucket name and object key**

## Object Retrieval

* For Non Versioned bucket
  + An Object retrieval always return the only object available
* For Versioned bucket
  + An object retrieval returns the Current object.
  + Non Current object can be retrieved by specifying the version ID.

## Object Addition

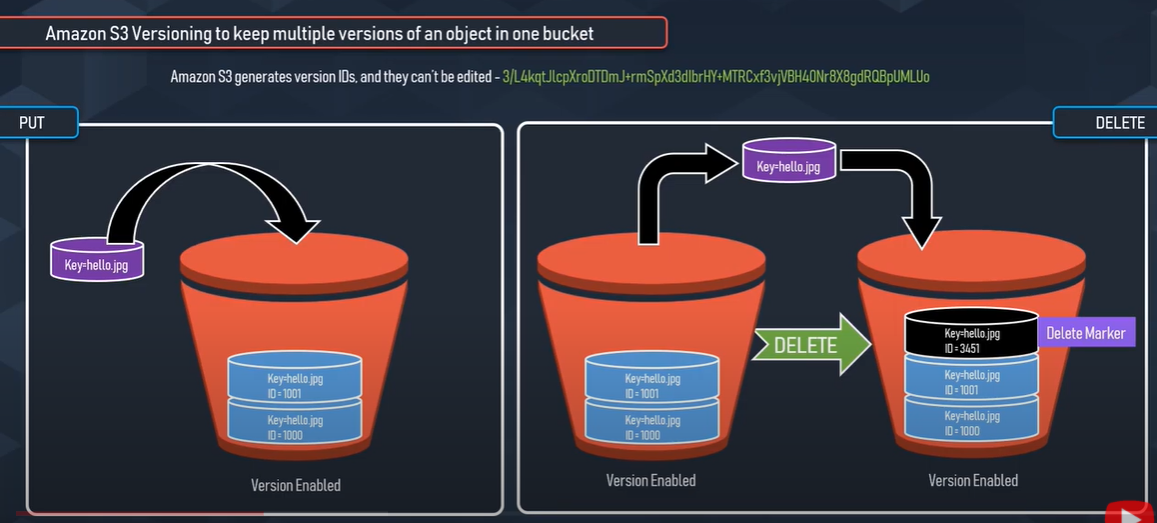
* For Non Versioned bucket
  + If an object with the same key is uploaded again it overwrites the object
* For Versioned bucket
  + If an object with the same key is uploaded the new uploaded object becomes the Current version and the previous object becomes the Non current version.
  + A non current versioned object can be retrieved and restored hence protecting against **accidental overwrites**

## Object Deletion

* For Non Versioned bucket
  + An object is permanently deleted and cannot be recovered
* For Versioned bucket,
  + All versions remain in the bucket and Amazon inserts a **delete marker** which becomes the Current version
  + A non current versioned object can be retrieved and restored hence protecting against **accidental overwrites**
  + If an Object with a specific version ID is deleted, a permanent deletion happens and the object cannot be recovered

## Delete marker

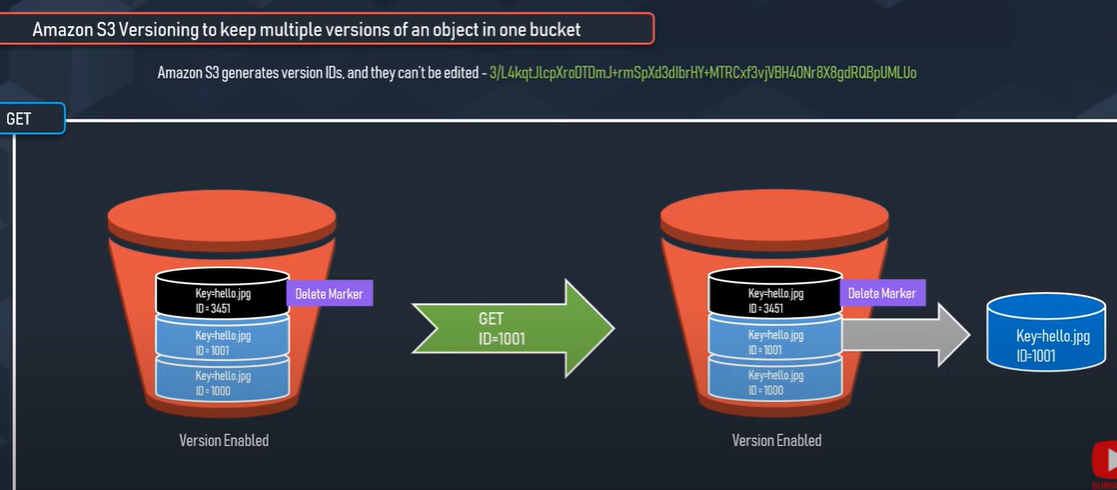
* Delete Marker object does not have any data or ACL associated with it, just the key and the version ID



* An object retrieval on a bucket with delete marker as the Current version would return a 404



* Non Current object can be retrieved by specifying the version ID.



* Only a DELETE operation is allowed on the Delete Marker object.
* **If the Delete marker object is deleted by specifying its version ID, the previous non current version object becomes the current version object**
* If a DELETE request is fired on the Bucket with Delete Marker as the current version, the Delete marker object is not deleted but an Delete Marker is added again

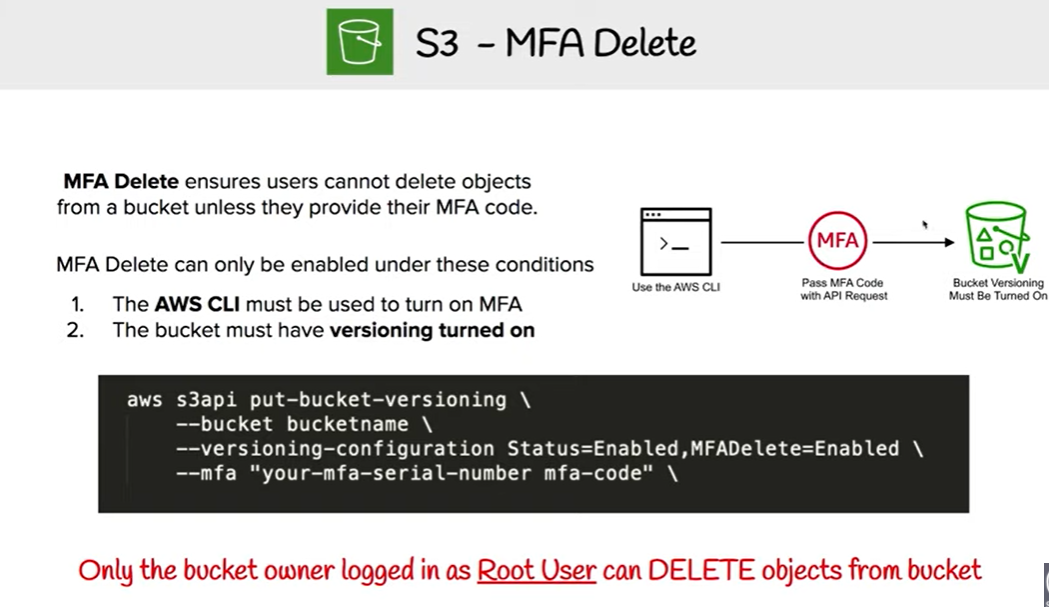
## Restoring Previous Versions

* Copy a previous version of the object into the same bucket. Copied object becomes the current version of that object and all object versions are preserved – **Recommended** as it keeps all the versions
* Permanently delete the current version of the object. When you delete the current object version, you, in effect, turn the previous version into the current version of that object.

## Versioning Suspended Bucket

* Versioning can be suspended to stop accruing new versions of the same object in a bucket
* Existing objects in the bucket do not change and only future requests behavior changes
* For each new object addition, a object with version ID null is added.
* For each object addition with the same key name, the object with the version ID null is overwritten
* An object retrieval request will always return the current version of the object
* A DELETE request on the bucket, would permanently delete the version ID null object and inserts a Delete Marker
* A DELETE request does not delete anything if the bucket does not have an object with version ID null
* A DELETE request can still be fired with a specific version ID for any previous object with version IDs stored

## MFA Delete



* Additional security can be enabled by configuring a bucket to enable MFA (Multi-Factor Authentication) delete
* MFA Delete enabled, requires additional authentication for operations
  + Changing the versioning state of the bucket
  + Permanently deleting an object version
* MFA Delete can be enabled on a bucket to ensure that data in the bucket cannot be **accidentally deleted**
* While the bucket owner, the AWS account that created the bucket (root account), and all authorized IAM users can enable versioning, but only the **bucket owner (root account) can enable MFA delete**.
* MFA Delete however does not prevent deletion or allows restoration.

## AWS Certification Exam Practice Questions

1. Which set of Amazon S3 features helps to prevent and recover from accidental data loss?
   1. Object lifecycle and service access logging
   2. **Object versioning and Multi-factor authentication**
   3. Access controls and server-side encryption
   4. Website hosting and Amazon S3 policies
2. You use S3 to store critical data for your company Several users within your group currently have full permissions to your S3 buckets. You need to come up with a solution that does not impact your users and also protect against the accidental deletion of objects. Which two options will address this issue? Choose 2 answers
   1. **Enable versioning on your S3 Buckets**
   2. **Configure your S3 Buckets with MFA delete**
   3. Create a Bucket policy and only allow read only permissions to all users at the bucket level
   4. Enable object life cycle policies and configure the data older than 3 months to be archived in Glacier
3. To protect S3 data from both accidental deletion and accidental overwriting, you should
   1. **enable S3 versioning on the bucket**
   2. access S3 data using only signed URLs
   3. disable S3 delete using an IAM bucket policy
   4. enable S3 Reduced Redundancy Storage
   5. enable Multi-Factor Authentication (MFA) protected access
4. A user has not enabled versioning on an S3 bucket. What will be the version ID of the object inside that bucket?
   1. 0
   2. There will be no version attached
   3. **Null**
   4. Blank
5. A user is trying to find the state of an S3 bucket with respect to versioning. Which of the below mentioned states AWS will not return when queried?
   1. versioning-enabled
   2. versioning-suspended
   3. unversioned
   4. **versioned**