Cloud Storage (Console)

13 June 2021 19:47

Create Bucket

Key Points

- **Availability**
- Location (regional, dual-regional, multi-regional). Location cannot be changed once assigned.
- **Feature**
  - Storage Class
    - Below is availability of each storage class, durability is 99.999999999% for each
      - Standard -> 99.95% multiregional, 99.90% regional
      - Nearline, Coldline -> 99.90%, multi-regional, 99% regional
      - Archival-> None
      -
    - Applied at bucket level, It may take upto 24hr for lifecycle policy change to take effect.
    - Object inherit bucket storage class, but an object can be set a different storage class from bucket. Changing the storage class of bucket does not affect object which are already present.
  - Retention/ Object Hold
    - Retention period can be set
    - Retention Lock can be set
    - Object hold (event based , temporary based)
  - Trigger
- Get or set lifecylce policy
  gsutil lifecycle get gs://<bucket_name>

  gsutil lifecycle set <config-json-file> gs://<bucket_name>...

```
{
 "rule":
 [
  {
   "action": {"type": "Delete"},
   "condition": {"age": 365}
  }
 ]
}
```

- We can configure pub/sub notification on any bucket to any pub/sub topic of any project for which user has access.
- GMEK, CMEK(using encryption_key in boto config) and CSEK supported. We can supply CSEK while restarting vm in console or using gcloud command –csek-file-path option.
- Metadata not encrypted by kms key, (object data, crc3 checksum and md5 hash encrypted)
- Below is how we set metadata

```
bucket.upload(imgPath, {
    destination: join(bucketDir, newImgName),
    metadata: {
        contentType: 'image/jpeg',
    }
});
```

- In transit TLS and HTTPS encrypt data
- Bucket lock similar to retention lock but applied on bucket level

Overview of storage classes

	Standard	Nearline	Coldline	Archive
Use case	"Hot" data and/or stored for only brief periods of time like data-intensive computations	Infrequently accessed data like data backup, long-tail multimedia content, and data archiving	Infrequently accessed data that you read or modify at most once a quarter	Data archiving, online backup, and disaster recovery
Minimum storage duration	None	30 days	90 days	365 days
Retrieval cost	None	$0.01 per GB	$0.02 per GB	$0.05 per GB
Availability SLA	99.95% (multi/dual) 99.90% (region)	99.90% (multi/dual) 99.00% (region)		None
Durability	99.999999999%			

Data storage pricing per storage class:

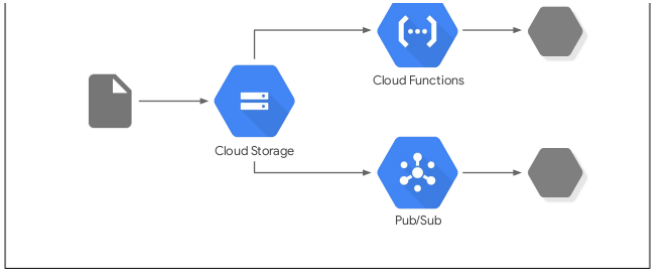
Storage Class	Standard Storage	Nearline Storage	Coldline Storage	Archive Storage
Price / GB / Month	$0.026	$0.010	$0.007	$0.004

Cloud Storage features

- Customer-supplied encryption key (CSEK)
  - Use your own key instead of Google-managed keys
- Object Lifecycle Management
  - Automatically delete or archive objects
- Object Versioning
  - Maintain multiple versions of objects
- Directory synchronization
  - Synchronizes a VM directory with a bucket
- Object change notifications using Pub/Sub

Pub/Sub notifications for Cloud Storage

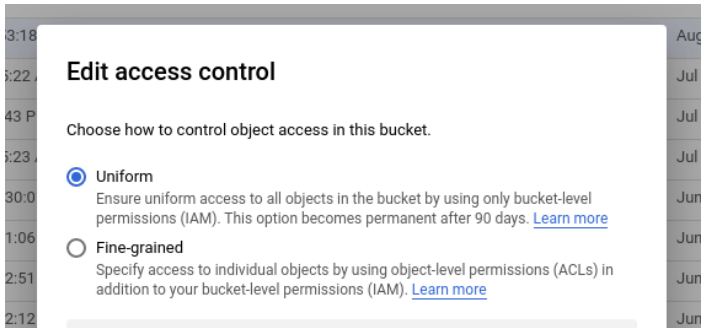
- Versioning supported, it is enabled at bucket level, each version have a generation no, it increase cost, lifecycle policy can help reduce cost. Object version Determined by data and metadata version
- Versioning and Retention policy, only one can be active at a time
- Parallel composite upload(gsutil -m)(divided to 32 chunk before e upload) of composite object (md5 hash not supported, crc32c supported, CMEK not supported, require same storage class)
- Parallel Composit upload command: gsutil -o GSUtil:parallel_composite_upload_threshold=150M cp bigfile gs://your-bucket
- Large no of files can be uploaded using multi-threaded approach with recursion using below command
  gsutil -m cp -R top-level-dir/subdir/image* gs://example-bucket
- Cloud storage support max file size of 5TB
- There is no limit to read and write. Bucket initial support 1000 write/sec and then scale as needed and initially support 5000 read/se and then scale as needed.
- Gsutil can perform incremental sync like rsync. Below is the command
  gsutil -m rsync -r data gs://mybucket/data
- Single File upload : small file, no metadata in request require, multi-part upload : small file, metadata in request required, resumable upload(flow small bandwidth, changing region can result in slow upload, must be completed within a week, Integrity check required by comparing md5 hash is required if streaming happened for many days for resumable upload.
) and streaming upload supported (when final file size is not known, for communication with process). ACL (old method) , IAM supported
- **Costing**
- Standard: 0.02$/GB/Month, nearline: 0.01$/GB/Month, coldline:0.007$/GB/Month, Archival: 0.004$/GB/Month.
- **Access Management**
- Signed url provide limited time access. It is signed with account service account private key. (for both download and upload)
- Signed policy can specify what can be uploaded to bucket (for upload only from HTML f)
- Uniform (Using IAM, Recommended) and Fined grained access using IAM and ACL
- Cors configuration are allowed at bucket level only
- Changes to object can be tracked using pub/sub or audit log (data access log)
- Strong consistency (upload, metadata change, read-after-(write, delete, metadata-update)), bucket and object listing
- Eventual consistency (granting or revoking access)
- GCS autoscaling work well if request increase gradually rather than sudden spike
- Avoid sequential naming
- For static website hosting bucket name should be a verified domain name.
- We can set trigger for pub/sub, cloud function, scan with DLP
- Data migration: less than 10 TB and 200mbps gsutil, cloud to cloud -> Storage Transfer Service, more than 20 TB transfer application.
- Cloud storage fuse allow mounting cloud storage bucket as file system. But it is not POSIX compliant and cannot be used as boot disk. There can be overwriting of data.



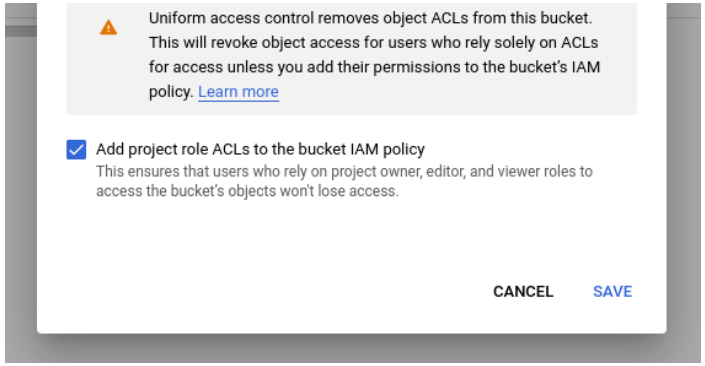
GSUTIL Commands

1. Create Bucket
   - gsutil mb –p project –c storage_class –l bucket_location gs://bucket_name (gsutil mb -p powow-vpp -c nearline -l us gs://testucker)
     Activity audit log generated

Change bucket access control

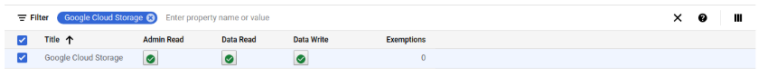


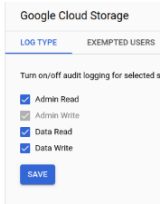
```
request: {1}
requestMetadata: {
  callerIp: "35.247.149.193"
  callerSuppliedUserAgent:
  "apitools Python/3.7.3 gsutil/4.67 (linux) analytics/disabled interactive/True c
  destinationAttributes: {0}
  requestAttributes: {2}
 }
resourceLocation: {1}
resourceName: "projects/_/buckets/testucker"
serviceData: {2}
serviceName: "storage.googleapis.com"
status: {0}
}
receiveTimestamp: "2021-08-30T13:23:19.483960506Z"
resource: {2}
severity: "NOTICE"
timestamp: "2021-08-30T13:23:18.174272800Z"
}
```

2. List Bucket
   a. gsutil ls

Enable Data Access Log in IAM > Audit Log for cloud storage





3. Get bucket size
   a. Gsutil du –s gs://bucket_name
4. Get bucket location/default storage class
   a. Gsutil ls –L –b gs://bucket_name

saraswata_b@cloudshell:~ (powow-vpp)$ gsutil ls -L -b gs://testucker/

gs://testucker/ :

Storage class: NEARLINE
Location type: multi-region
Location constraint: US
Versioning enabled: None
Logging configuration: None
Website configuration: None
CORS configuration: None
Lifecycle configuration: None
Requester Pays enabled: None
Labels: None
Default KMS key: None
Time created: Mon, 30 Aug 2021 13:23:18 GMT
Time updated: Mon, 30 Aug 2021 15:09:32 GMT
Metageneration: 5

Bucket Policy Only enabled: True

Public access prevention: unspecified

ACL: []

Default ACL: []

5. Change bucket default storage class
   a. Gsutil defstorageclass set storage_class gs://bucket_name (gsutil defstorageclass set coldline gs://testucker/)
6. Copy bucket objects to another bucket
   a. Gsutil cp –r gs://source_bucket/* gs://target_bucket/
   b. gsutil cp README-cloudshell.txt gs://testucker/ (copy a local file to bucket)
7. Command to list object from a bucket
   a. gsutil ls -r gs://testucker/**
8. Command to rename an object
   a. gsutil mv gs://testucker/README-cloudshell.txt gs://testucker/Readme.txt
9. When object are created they inherit the bucket default storage class we can change it at object level
   a. gsutil rewrite -O -s nearline gs://testucker/Readme.txt (-O flag is provided if uniform level access is enabled)
10. View Object metadata
    a. gsutil ls -L gs://testucker/Readme.txt

gs://testucker/Readme.txt:

Creation time: Mon, 30 Aug 2021 16:36:51 GMT

Update time: Mon, 30 Aug 2021 16:36:51 GMT

Storage class: NEARLINE

Content-Length: 913

Content-Type: text/plain

Hash (crc32c): lJfvug==

Hash (md5): +bqpwgYMTRm0kWnp5HXRMw==

ETag: CMTdtIWX2fICEAE=

Generation: 1630341411188420

Metageneration: 1

Your company is planning to upload several important files to Cloud Storage. After the upload is completed, they want to verify that the uploaded content is identical to what they have on-premises. You want to minimize the cost and effort of performing this check. What should you do?

D. 1. Use gsutil -m to upload the files to Cloud Storage. 2. Use gsutil hash -c FILE_NAME to generate CRC32C hashes of all on-premises files. 3. Use gsutil ls -L gs://[YOUR_BUCKET_NAME] to collect CRC32C hashes of the uploaded files. 4. Compare the hashes.

Pricing

Data storage pricing per storage class:

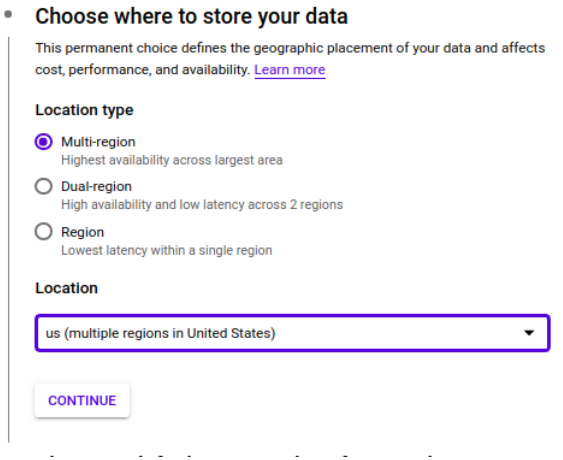
Storage Class	Standard Storage	Nearline Storage	Coldline Storage	Archive Storage
Price / GB / Month	$0.026	$0.010	$0.007	$0.004

Class A operations (**object adds, bucket and object listings**) 10,000 operations. Operations.
Class B operations (object gets, retrieving bucket and object metadata)

Storage Class[1]	Class A operations (per 10,000 operations)	Class B operations (per 10,000 operations)
Standard Storage	$0.05	$0.004
Nearline Storage and **Durable Reduced Availability (DRA) Storage**	$0.10	$0.01
Coldline Storage	$0.10	$0.05
Archive Storage	$0.50	$0.50

Source: Google Cloud

1. Choice of location is permanent



Minimum storage duration: This is the minimum duration for which an object must be stored in a storage class there is a early deletion free.

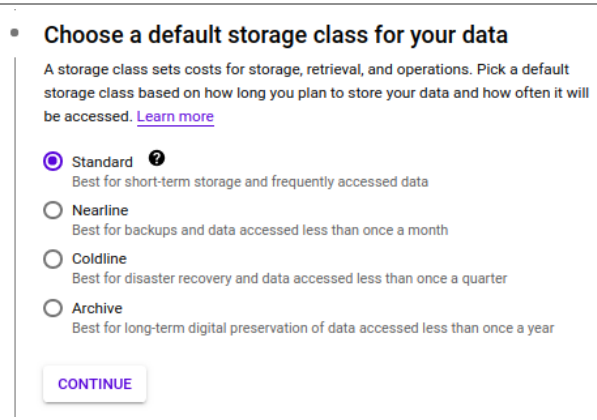
We can move object between storage class using lifecycle policy

Pricing is applied based on storage class

An SLA outlines what the provider and customer are responsible for in regards to using the service

Standard: (Highest SLA- > 99.99 % availability in multi-region and 99.99% for regional)
Minimum storage duration : None
For frequent data access.(This is the new term of regional and multi regional storage class)

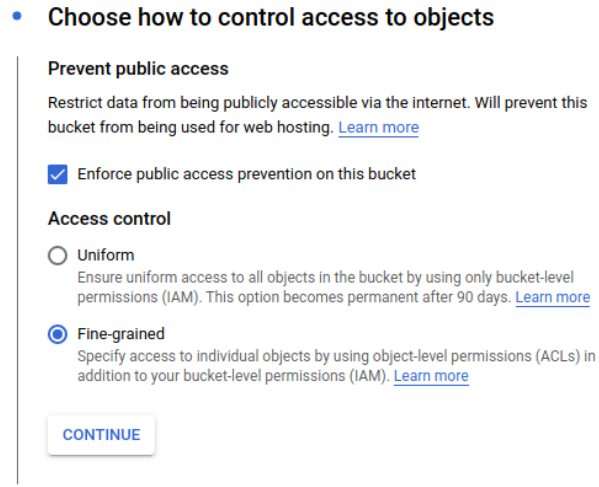
Nearline: (SLA- > 99.95 % availability in multi-region and 99.95% for regional)
Minimum storage duration : 30 days
Accessed once a month



Coldline: (SLA- > 99.95 % availability in multi-region and 99.95% for regional)
Minimum storage duration : 90 days
Accessed once a quarter

Archival: (SLA- > 99.95 % availability in multi-region and 99.95% for regional)
Minimum storage duration : 365 days
Accessed once a year

When an object is transitioned to Nearline Storage, Coldline Storage, or Archive Storage using the SetStorageClass feature, any subsequent early deletion and associated charges are based on the original creation time of the object, regardless of when the storage class changed.



Retention Policy

You can include a retention policy when creating a new bucket, or you can add a retention policy to an existing bucket. Placing a retention policy on a bucket ensures that all current and future objects in the bucket cannot be deleted or replaced until they reach the age you define in the retention policy. Attempts to delete or replace objects whose age is less than the retention period fail with a 403 - retentionPolicyNotMet error.

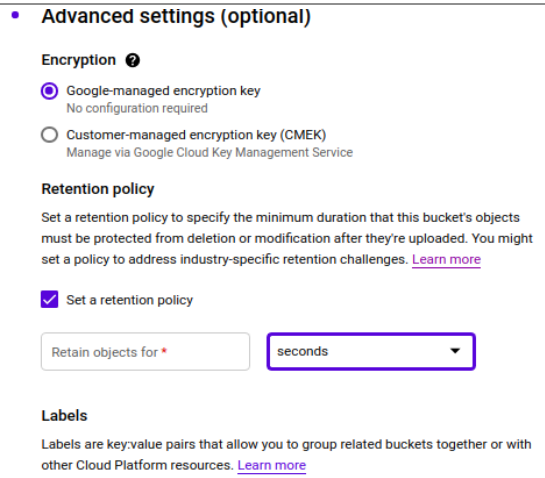
Retention policies and Object Versioning are mutually exclusive features in Cloud Storage: for a given bucket, only one of these can be enabled at a time

A lifecycle rule won't delete an object until after the object fulfills the retention policy.

Customer Managed Encryption Key

When you apply a customer-managed encryption key to an object, Cloud Storage uses the key when encrypting:

- The object's data.
- The object's CRC32C checksum.
- The object's MD5 hash.



Cloud Storage uses standard server-side keys to encrypt the remaining metadata for the object, including the object's name. Thus, if you have sufficient permission, you can perform actions such as reading most metadata, listing objects, and deleting objects even after you've disabled or destroyed the associated customer-managed encryption key.

Limitation:

- You must create the Cloud KMS key in the same location as the data you intend to encrypt. For example, if your bucket is located in US-EAST1, any Cloud KMS key encrypting objects in that bucket must also be created in US-EAST1. For available Cloud KMS locations, see Cloud KMS locations.
- You cannot encrypt an object with a customer-managed encryption key by updating the object's metadata. Include the key as part of a rewrite of the object instead.
- You cannot specify a customer-managed encryption key as part of a Storage Transfer Service transfer, and any such keys on source objects are not applied to the transferred objects. Set a default customer-managed key on your bucket prior to performing the transfer.

Customer supplied encryption key

In addition to customer-managed encryption, Cloud Storage offers Customer-Supplied Encryption Keys as a way of controlling your data encryption. You can encrypt different objects in a single bucket with different encryption methods, but note that:

- A single object can only be encrypted by one of these methods at a time.
- If you have a default customer-managed key set for your bucket and specify a customer-supplied key in a request, Cloud Storage uses the customer-supplied key to encrypt the object.
- You can set a default customer-**managed** key on your bucket, but you cannot set a default customer-**supplied** key on your bucket.

Option A is the correct choice because the client doesn't want to store the encryption keys on Google Cloud. Supplying your own encryption keysis best suited in this scenario, Google uses your key to protect the Google-generated keys used to encrypt and decrypt your data.

Signed Policy Document

Use signed policy documents to specify what can be uploaded to a bucket. Policy documents allow greater control over size, content type, and other upload characteristics than signed URLs, and can be used by website owners to allow visitors to upload files to Cloud Storage.

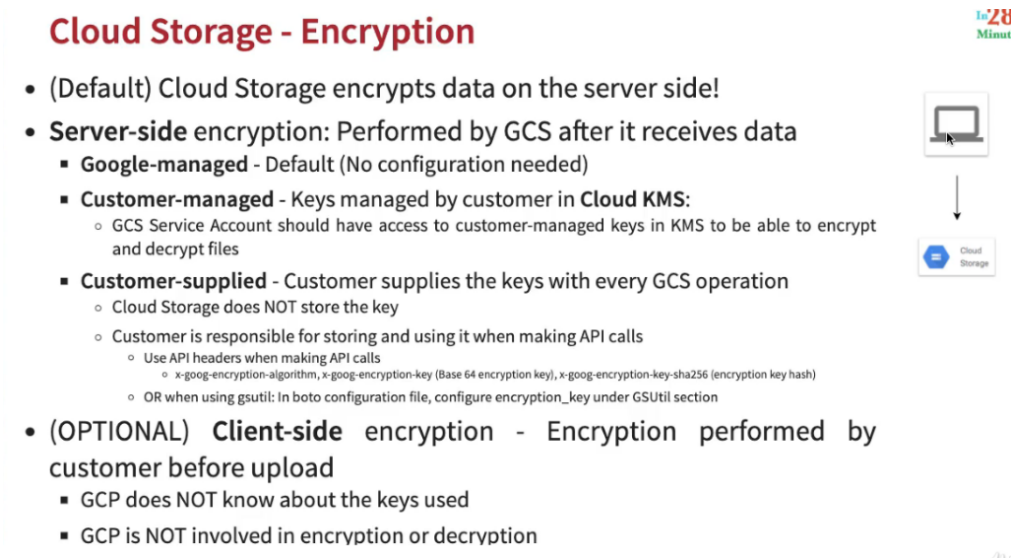
You can use signed policy documents in addition to IAM and ACLs. For example, you can use IAM to allow people in your organization to upload any object, then create a signed policy document that allows website visitors to upload only objects that meet specific criteria.

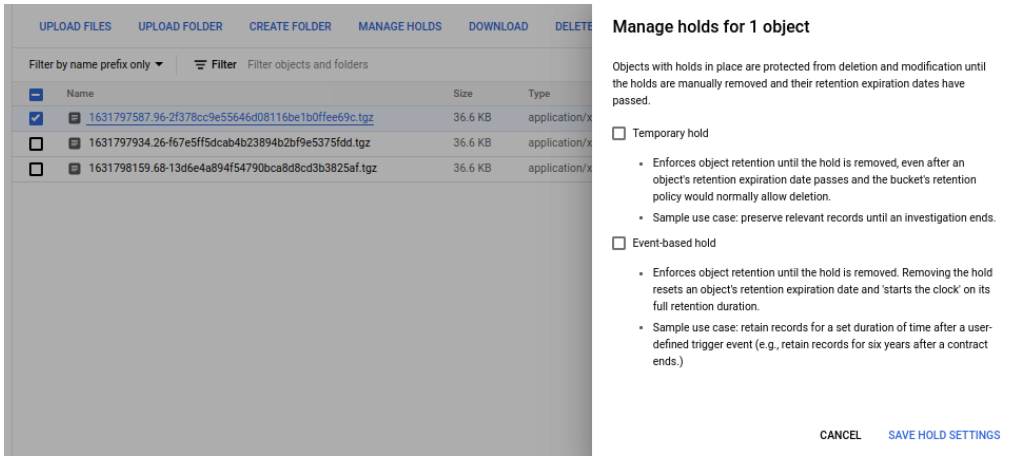
Bucket Lock

This page discusses the Bucket Lock feature, which allows you to configure a data retention policy for a Cloud Storage bucket that governs how long objects in the bucket must be retained. The feature also allows you to lock the data retention policy, permanently preventing the policy from being reduced or removed.

This feature can provide immutable storage on Cloud Storage. In conjunction with Detailed audit logging mode, which logs Cloud Storage request and response details, Bucket Lock can help with regulatory and compliance requirements, such as those associated with FINRA, SEC, and CFTC. Bucket Lock may also help you address certain health care industry retention regulations.

Data Encryption and Managed Encryption Keys





Object hold

Say you have two objects - Object A and Object B - in a bucket with a 1-year retention period. When you added the objects to the bucket, you placed an event-based hold on Object A and a temporary hold on Object B. A year passes, and while you'd normally be able to delete them at this point, because both objects still have a hold on them, you can't delete either of them.

Let's say at this point you release the hold from both objects. For Object A, its time in the bucket starts from scratch for the purposes of the retention period. This means it must stay in the bucket for another year before it can be deleted or replaced. Object B, on the other hand, can immediately be deleted or replaced, because the temporary hold has no effect on when the object fulfilled its retention time.

This behavior allows you to use event-based holds in conjunction with retention policies to control retention based on the occurrence of some event, such as holding loan documents for a certain period after loan was paid. Temporary holds can be used for regulatory or legal purposes, such as holding trading documents for legal investigation.

Object metadata

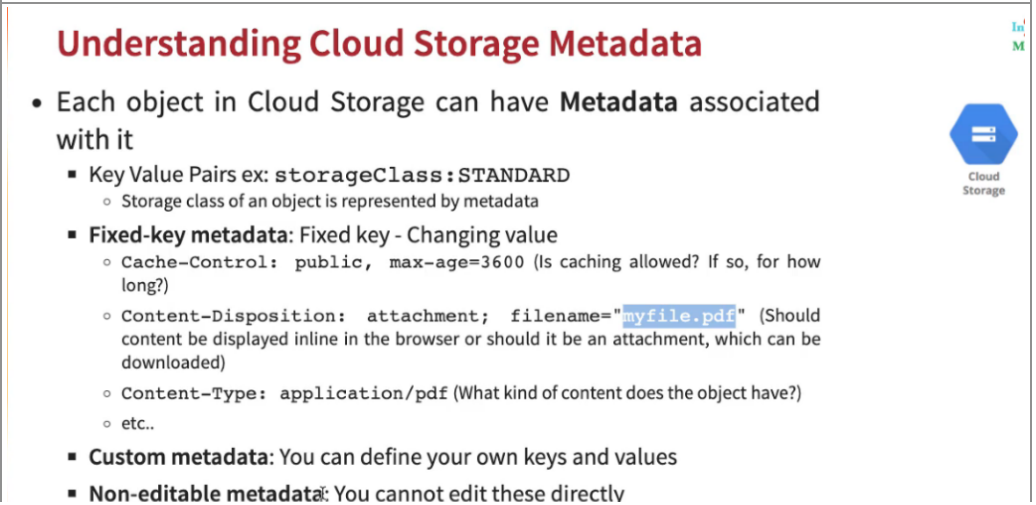
The mutability of metadata varies: some metadata you can edit at any time, some metadata you can only set at the time the object is created, and some metadata you can only view.

MD5 Hash of object

If you know the MD5 of a file before uploading, you can specify it in the Content-MD5 header, which enables the cloud storage service to reject the upload if the MD5 doesn't match the value computed by the service. For example:

1. It prevents the corrupted object from becoming visible. If you don't specify the header, the object is visible for 1-3 seconds before gsutil deletes it.
2. If you don't specify the Content-MD5 header, it's possible for the gsutil process to complete the upload but then be interrupted or fail before it can delete the corrupted object, leaving the corrupted object in the cloud.

Cloud Storage Metadata



- Storage class of the object, customer-managed encryption keys etc

Cloud Storage Bucket Lock

Cloud Storage Bucket Lock - Meet Compliance Needs

In 24 Minutes

- How do you ensure that you **comply with regulatory and compliance requirements** around immutable storage in a Cloud Storage bucket?
- Configure **data retention policy** with retention period:
  - How long should objects in the bucket be retained for?
    - "Objects in the bucket can only be deleted or replaced once their age is greater than the retention period"
  - You can set it **while creating a bucket or at a later point in time**
    - Applies automatically to existing objects in the bucket (as well as new objects added in)
  - Once a retention policy is locked:
    - You **CANNOT remove retention policy** or **reduce** retention period (You can increase retention period)
    - You **CANNOT delete the bucket** unless all objects in bucket have age greater than retention period

Transfer data from on-promesis to Cloud

Data rehydration

Data rehydration is the process by which you fully reconstitute the files so you can access and use the transferred data. To rehydrate data, the data is first copied from the Transfer Appliance to your Cloud Storage staging bucket. The data uploaded to your staging bucket is still compressed, deduplicated, and encrypted. Data rehydration reverses this process and restores your data to a usable state.

Transferring data from on premises to cloud

In 28 Minutes

- **Most popular** data destination is **Google Cloud Storage**
- **Options:**
  - **Online Transfer**: Use gsutil or API to transfer data to Google Cloud Storage
    - Good for one time transfers
  - **Storage Transfer Service**: Recommended for large-scale (petabytes) online data transfers from your private data centers, AWS, Azure, and Google Cloud
    - You can set up a repeating schedule
    - Supports incremental transfer (only transfer changed objects)
    - Reliable and fault tolerant - continues from where it left off in case of errors
  - **Storage Transfer Service vs gsutil:**
    - gsutil is recommended only when you are transferring less than 1 TB from on-premises or another GCS bucket
    - Storage Transfer Service is recommended if either of the conditions is met:
      - Transferring more than 1 TB from anywhere
      - Transferring from another cloud

Migrating Data with Transfer Appliance

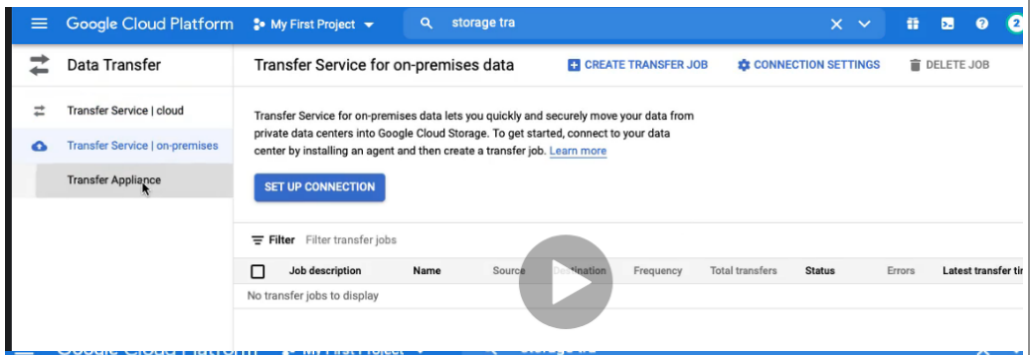
In 28 Minutes

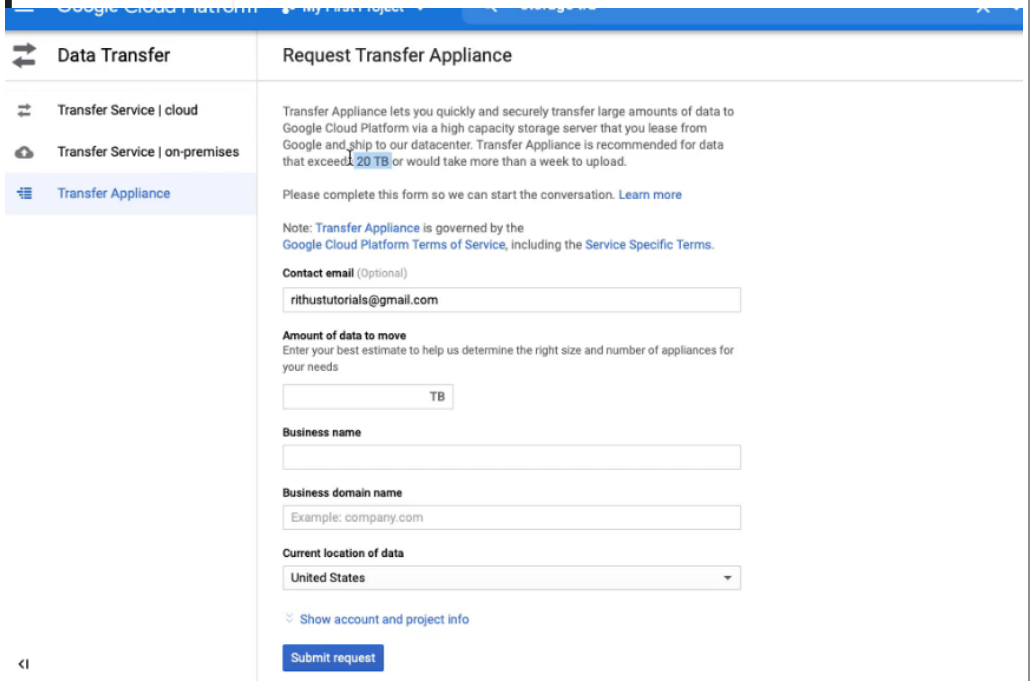
- **Transfer Appliance**: Copy, ship and upload data to GCS
  - **Recommended** if your data size is **greater than 20TB**
    - OR **online transfer takes > 1 week**
  - **Process:**

- Request an appliance
- Upload your data
- Ship the appliance back
- Google uploads the data

- Fast copy (upto 40Gbps)
- AES 256 encryption - Customer-managed encryption keys
- Order **multiple devices** (TA40, TA300) if need

https://cloud.google.com





Cloud Storage Best Practices

Understanding Cloud Storage Best Practices

- Avoid use of sensitive info in bucket or object names
- Store data in the **closest region** (to your users)
- Ramp up **request rate gradually**
  - No problems upto 1000 write requests per second or 5000 read requests per second
  - BUT beyond that, take at least 20 minutes to double request rates
- Use **Exponential backoff** if you receive 5xx (server error) or 429 (too many requests) errors
  - Retry after 1, 2, 4, 8, 16, .. seconds
- **Do NOT use sequential** numbers or timestamp as object keys
  - Recommended to use completely random object names
  - Recommended to add a hash value before the sequence number or timestamp
- **Use Cloud Storage FUSE** to enable file system access to Cloud Storage
  - Mount Cloud Storage buckets as file systems on Linux or macOS systems

Gsutil Command Line

Objects can have associated metadata, which control aspects of how GET requests are handled, including Content-Type, Cache-Control, Content-Disposition, and Content-Encoding. In addition, you can set custom key:value metadata for use by your applications.

```
gsutil -h "Content-Type:text/html" \
    -h "Cache-Control:public, max-age=3600" cp -r images \
    gs://bucket/images
```

Cloud Storage - Command Line - gsutil - 2

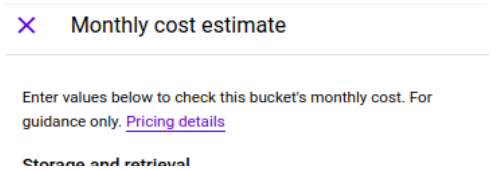
- Cloud Storage (gsutil)
  - *gsutil **versioning set on/off** gs://BKT_NAME* (Enable/Disable Versioning)
  - *gsutil **uniformbucketlevelaccess set on/off** gs://BKT_NAME*
  - *gsutil **acl ch*** (Set Access Permissions for Specific Objects)
    - *gsutil acl ch -u AllUsers:R gs://BKT_NAME/OBJ_PATH* (Make specific object public)
    - *gsutil acl ch -u john.doe@example.com:WRITE gs://BKT_NAME/OBJ_PATH*
      - Permissions - READ (R), WRITE (W), OWNER (O)
      - Scope - User, allAuthenticatedUsers, allUsers(-u), Group (-g), Project (-p) etc
    - gsutil acl set JSON_FILE gs://BKT_NAME
  - *gsutil **iam ch** MBR_TYPE:MBR_NAME:IAM_ROLE gs://BKT_NAME* (Setup IAM role)
    - *gsutil iam ch user:me@myemail.com:objectCreator gs://BKT_NAME*
    - *gsutil iam ch allUsers:objectViewer gs://BKT_NAME* (make the entire bucket readable)
  - *gsutil **signurl -d 10m** YOUR_KEY gs://BUCKET_NAME/OBJECT_PATH* (Signed URL for temporary access)

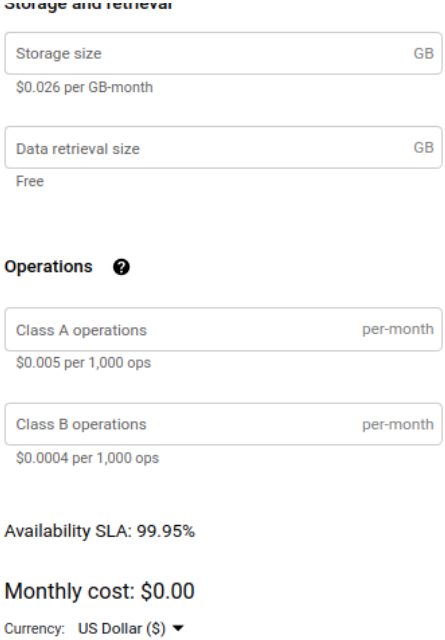
```
gcloud --version
gsutil mb gs://my_bucket_in28minutes_shell
gcloud config set project glowing-furnace-304608
gsutil mb gs://my_bucket_in28minutes_shell
gsutil ls gs://my_bucket_in28minutes_shell
```

Sample class A operations

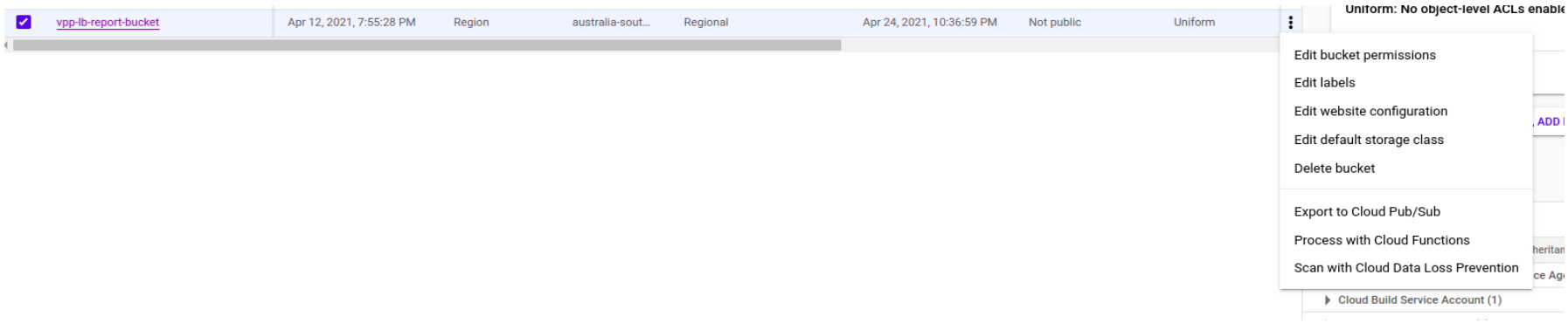
Create buckets; upload objects; set bucket permissions; delete object permissions

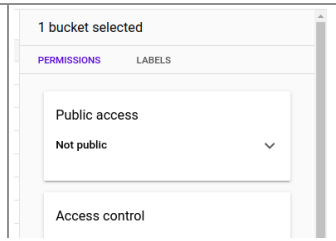
Sample class B operations

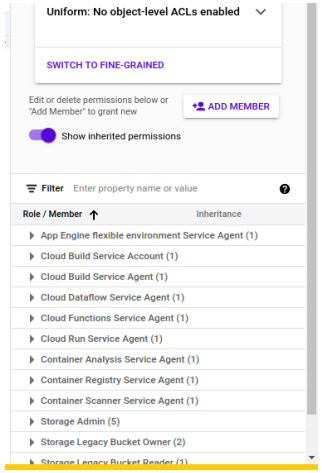


Download objects; view metadata; retrieve bucket and object permissions	

GCP Bucket Operations

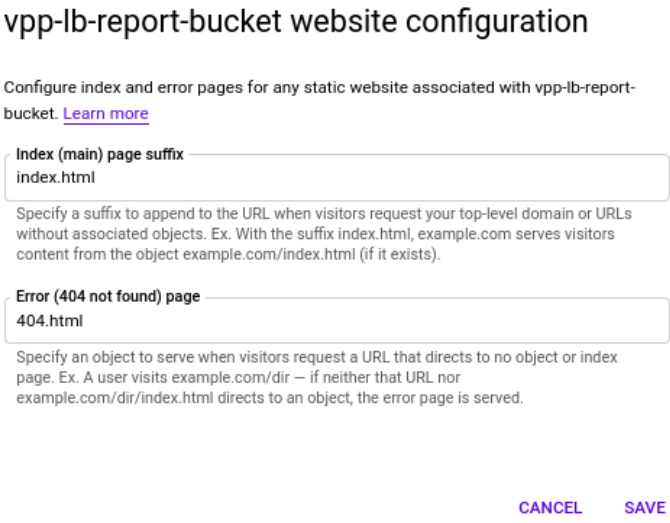


Edit Bucket Permission: Bucket level permission can be configured at resource scope	

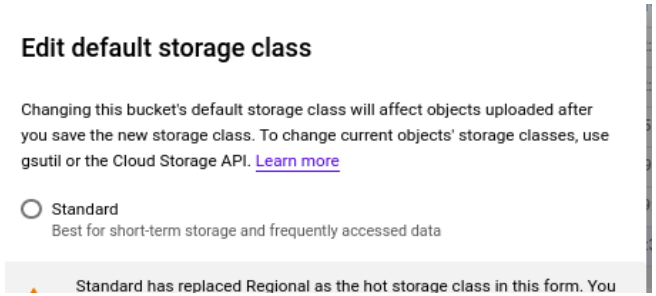


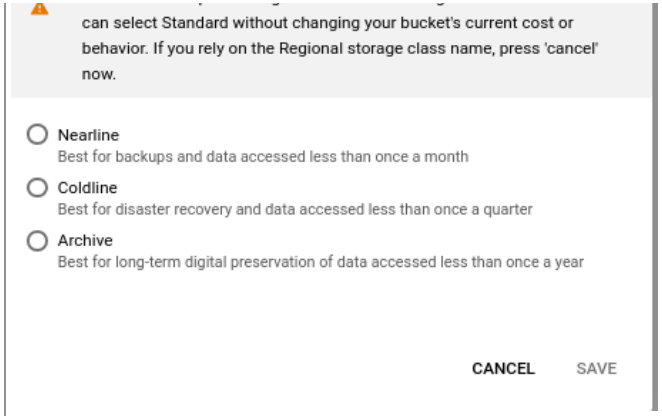
Edit bucket label: Labels help organize your resources (e.g., cost_center:sales or env:prod).

Edit Website Configuration

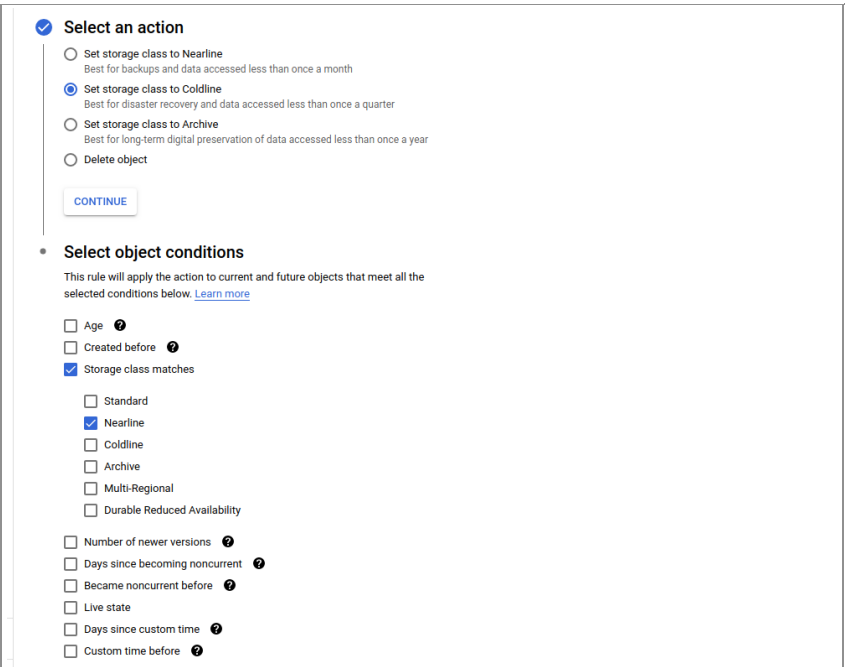


Edit bucket storage class

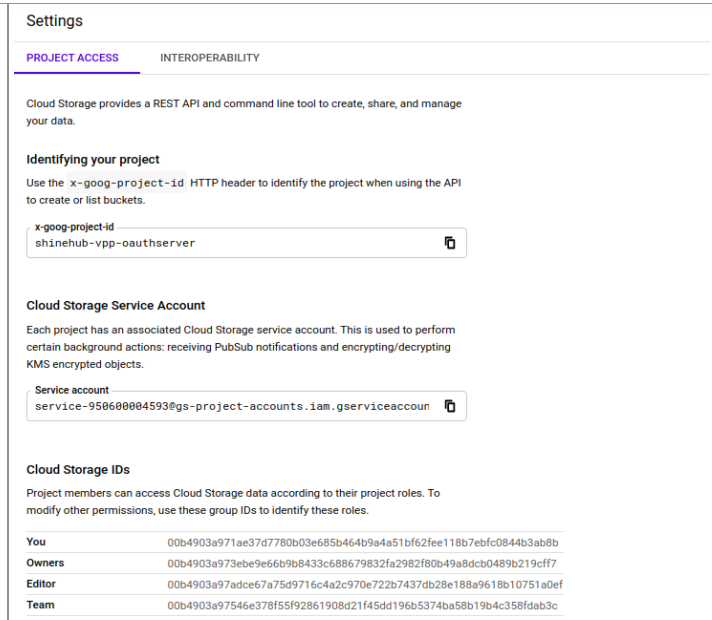


	
Export to pub/sub: Redirected to dataflow where we can use pre-defined template.	
Process with cloud function: Redirected to cloud function with trigger type cloud storage Here when a file is uploaded to bucket, cloud function is triggered which get the file object and get its name or create thumbnail etc.	
Scan with cloud dataloss prevention: Provides methods for detection, risk analysis, and de-identification of privacy-sensitive fragments in text, images, and Google Cloud Platform storage repositories.	

Bucket Lifecycle Policy (Add Rule)

Action: delete, setstorageclass **Age**: Counted from the object's creation date (when the object was added to the current bucket) **Crated Before**: Based on the object's creation date (when the object was added to the current bucket) **Number of version stored**: Limits the number of versions stored, if object versioning has been enabled. **Days since becoming noncurrent:** Counted from when a live object was modified or deleted, if object versioning was enabled. **Become noncurrent before:** Based on when a live object was modified or deleted, if object versioning was enabled. **Days since custom time**: Counted from the object's custom time, for objects that have this custom metadata. The Custom-Time metadata is a user-specified date and time represented in the RFC 3339 format YYYY-MM-DD'T'HH:MM:SS.SS'Z' or YYYY-MM-DD'T'HH:MM:SS'Z' when milliseconds are zero. This metadata is typically set in order to use the DaysSinceCustomTime condition in Object Lifecycle Management. **Custom time before**: Based on the object's custom time, for objects that have this custom metadata.	

Settings

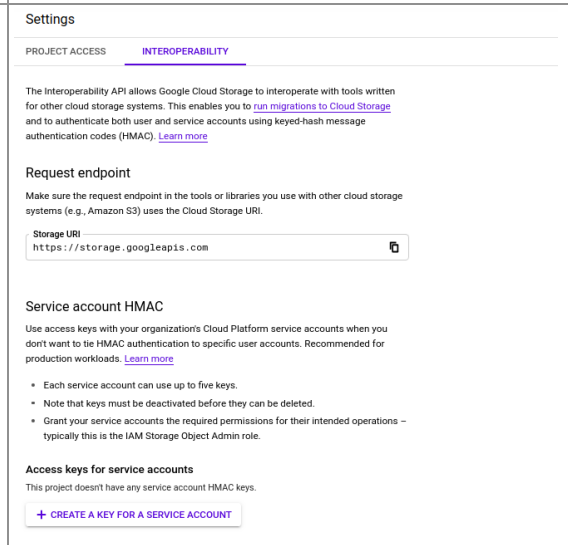


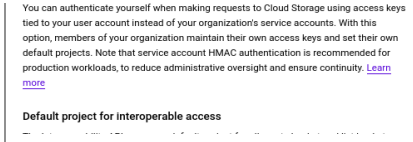
This page describes various Cloud Storage tools that you can use to work with data stored in other cloud providers.

The gsutil tool lets you access Cloud Storage from the command line. It can also be used to access and work with other cloud storage services that use HMAC authentication, like Amazon S3. For example, after you add your Amazon S3 credentials to the .boto configuration file for gsutil, you can start using gsutil to manage objects in your Amazon S3 buckets. The following command lists the objects in the Amazon S3 bucket example-bucket:

gsutil ls s3://example-bucket

gsutil rsync -d -r s3://my-aws-bucket gs://example-bucket





Object Versioning

You can enable Object Versioning to protect your Cloud Storage data from being overwritten or accidentally deleted. Enabling Object Versioning increases storage costs, which can be partially mitigated by configuring Object Lifecycle Management to delete older object versions.

When you enable object versioning, Cloud Storage creates a noncurrent version of an object each time you perform an overwrite or delete of the live version, as long as you do not specify the generation number of the live version. Noncurrent versions have below properties:

Gsutil command to enable object versioning

Enable object versioning for your bucket

gsutil versioning set on gs://cloudaffaire_bucket

gsutil versioning set <on|off> url...

gsutil versioning get url...

View generation and metageneration for your object post versioning

gsutil stat gs://cloudaffaire_bucket/cloudaffaire_object.txt

Note the generation and metageneration for the object does not changed

Generation: 1577366178757673

Metageneration: 1

Composite Object

The compose operation creates a new *composite object* whose contents are the concatenation of a given sequence of source objects. The source objects all must:

- Have the same storage class.
- Be stored in the same Cloud Storage bucket.

When you perform a composition:

- The source objects are unaffected.
- You can use between 1 and 32 source objects.
- Source objects can themselves be composite objects.
- The resulting composite object has the same storage class as the source objects.
- The resulting composite object does not change if the source objects are subsequently replaced or deleted.
- When using gsutil to perform object composition, the Content-Type of the resulting composite object is set to match the Content-Type of the first source object.

With cloud storage, Object composition can be used for uploading an object in parallel: you can divide your data into multiple chunks, upload each chunk to a distinct object in parallel, compose your final object, and delete any temporary source objects. This option helps maximize your bandwidth usage and ensures the file is uploaded as fast as possible.