Garbage Collection

Let's learn how GFS implements garbage collection.

We'll cover the following

- Garbage collection through lazy deletion
- Advantages of lazy deletion
- Disadvantages of lazy deletion

Garbage collection through lazy deletion#

When a file is deleted, GFS does not immediately reclaim the physical space used by that file. Instead, it **follows a lazy garbage collection strategy**. When the client issues a delete file operation, GFS does two things:

1. The master logs the deletion operation just like other changes.
2. The deleted file is renamed to a hidden name that also includes a deletion timestamp.

The file can still be read under the new, special name and can also be undeleted by renaming it back to normal. To reclaim the physical storage, the master, while performing regular scans of the file system, removes any such hidden files if they have existed for more than three days (this interval is configurable) and also deletes its in-memory metadata. This lazy deletion scheme provides a window of opportunity to a user who

deleted a file by mistake to recover the file.

The master, while performing regular scans of chunk namespace, deletes the metadata of all chunks that are not part of any file. Also, during the exchange of regular HeartBeat messages with the master, each ChunkServer reports a subset of the chunks it has, and the master replies with a list of chunks from that subset that are no longer present in the master's database; such chunks are then deleted from the ChunkServer.

Advantages of lazy deletion#

Here are the advantages of lazy deletion.

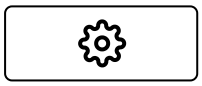
- Simple and reliable. If the chunk deletion message is lost, the master does not have to retry. The ChunkServer can perform the garbage collection with the subsequent heartbeat messages.
- GFS merges storage reclamation into regular background activities of the master, such as the regular scans of the filesystem or the exchange of HeartBeat messages. Thus, it is done in batches, and the cost is amortized.
- Garbage collection takes place when the master is relatively free.
- Lazy deletion provides safety against accidental, irreversible deletions.

Disadvantages of lazy deletion#

As we know, after deletion, storage space does not become available immediately. Applications that frequently create and delete files may not be able to reuse the storage right away. To overcome this, GFS provides following options:

- If a client deletes a deleted file again, GFS expedites the storage

- If a client deletes a deleted file again, GFS expedites the storage reclamation.



- Users can specify directories that are to be stored without replication.
- Users can also specify directories where deletion takes place immediately.

← Back

Fault Tolerance, High Availability, and ...

Next →

Criticism on GFS

Mark as Completed

Report an Issue