High-level Architecture

This lesson gives a brief overview of GFS's architecture.

We'll cover the following

- Chunks
- Chunk handle
- Cluster
- ChunkServer
- Master
- Client

A GFS cluster consists of a single master and multiple ChunkServers and is accessed by multiple clients.

Chunks#

As files stored in GFS tend to be very large, GFS breaks files into multiple fixed-size chunks where each chunk is 64 megabytes in size.

Chunk handle#

Each chunk is identified by an immutable and globally unique 64-bit ID number called chunk handle. This allows for 2^{64} unique chunks. If each chunk is 64 MB, total storage space would be more than 10^9 exa-bytes.

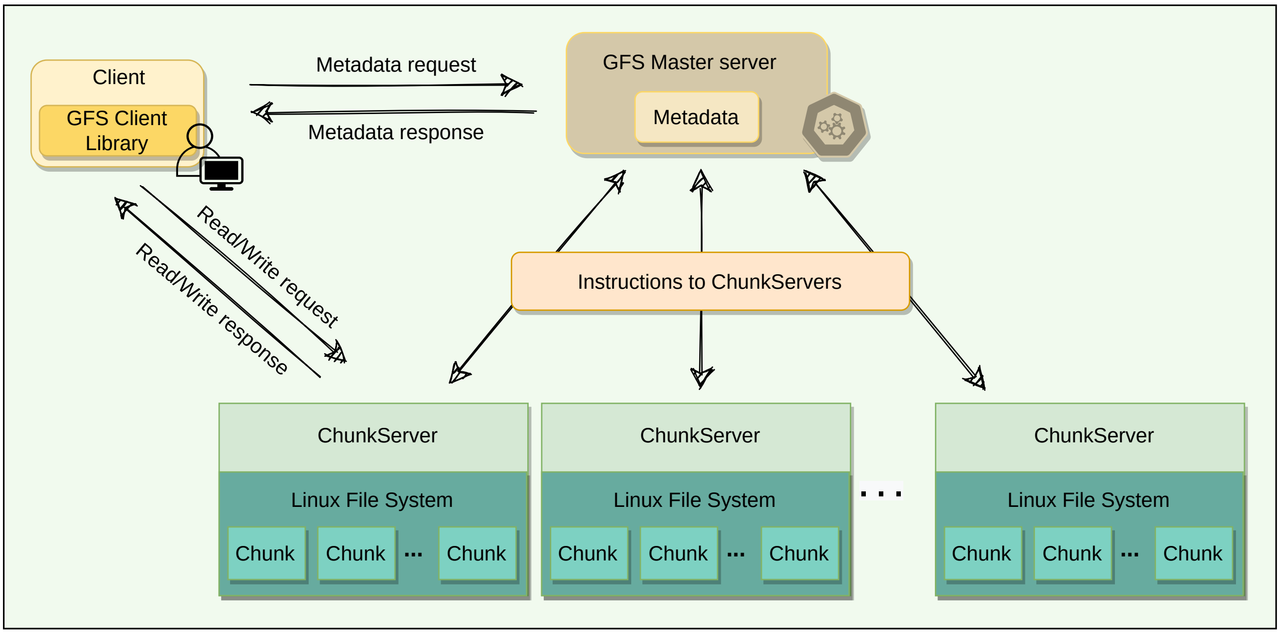
As files are split into chunks, therefore, the job of GFS is to provide a mapping from files to chunks, and then to support standard operations on files, mapping down to operations on individual chunks.

Cluster#

GFS is organized into a simple network of computers called a cluster. All GFS clusters contain three kinds of entities:

1. A single master server
2. Multiple ChunkServers
3. Many clients

The master stores all metadata about the system, while the ChunkServers store the real file data.

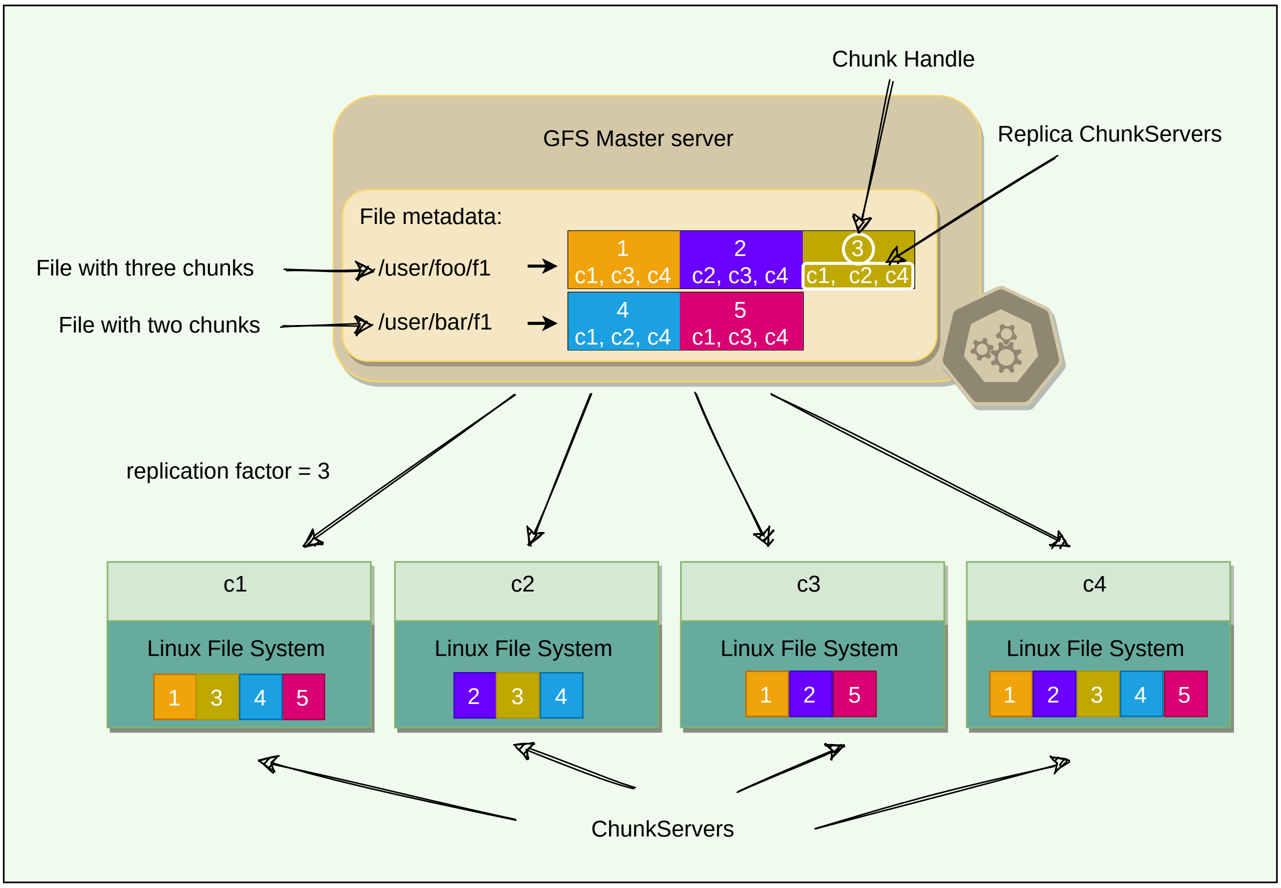


GFS high-level architecture

ChunkServer#

ChunkServers store chunks on local disks as regular Linux files and read or write chunk data specified by a chunk handle and byte-range.

For reliability, each chunk is replicated to multiple ChunkServers. By default, GFS stores three replicas, though different replication factors can be specified on a per-file basis.



Chunk replication

Master#

Master server is the coordinator of a GFS cluster and is responsible for keeping track of filesystem metadata:

1. The metadata stored at the master includes:
    - Name and directory of each file
    - Mapping of each file to its chunks
    - Current locations of chunks

   - Access control information

2. The master also controls system-wide activities such as chunk lease management (locks on chunks with expiration), garbage collection of orphaned chunks, and chunk migration between ChunkServers. Master assigns chunk handle to chunks at time of chunk creation.
3. The master periodically communicates with each ChunkServer in HeartBeat messages to give it instructions and collect its state.
4. For performance and fast random access, all metadata is stored in the master’s main memory. This includes the entire filesystem namespace as well as all the name-to-chunk mappings.
5. For fault tolerance and to handle a master crash, all metadata changes are written to the disk onto an operation log. This operation log is also replicated onto remote machines. The operation log is similar to a journal. Every operation to the file system is logged into this file.
6. The master is a single point of failure, hence, it replicates its data onto several remote machines so that the master can be readily restored on failure.
7. The benefit of having a single, centralized master is that it has a global view of the file system, and hence, it can make optimum management decisions, for example, related to chunk placement.

Client#

Client is an entity that makes a read or write request to GSF. GFS client library is linked into each application that uses GFS. This library communicates with the master for all metadata-related operations like creating or deleting files, looking up files, etc. To read or write data, the client interacts directly with the ChunkServers that hold the data.

Neither the client nor the ChunkServer caches file data. Client caches offer little benefit because most applications stream through huge files or have working sets too large to be cached. ChunkServers rely on the buffer cache in Linux to maintain frequently accessed data in memory.

← Back

Google File System: Introduction

Next →

Single Master and Large Chunk Size

☑ Mark as Completed

Report an Issue