Google File System: Introduction

Let’s explore Google File System and its use cases.

We'll cover the following

Goal#

Design a distributed file system to store huge files (terabyte and larger). The system should be scalable, reliable, and highly available.

What is Google File System (GFS)?#

GFS is a scalable distributed file system developed by Google for its large data-intensive applications.

Background#

GFS was built for handling batch processing on large data sets and is designed for system-to-system interaction, not user-to-system interaction.

Google built GFS keeping the following goals in mind:

- **Scalable**: GFS should run reliably on a very large system built from commodity hardware.
- **Fault-tolerant**: The design must be sufficiently tolerant of hardware and software failures to enable application-level services to continue their operation in the face of any likely combination of failure conditions.
- **Large files**: Files stored in GFS will be huge. Multi-GB files are common.
- **Large sequential and small random reads**: The workloads primarily consist of two kinds of reads: large, streaming reads and small, random reads.
- **Sequential writes**: The workloads also have many large, sequential writes that append data to files. Typical operation sizes are similar to those for reads. Once written, files are seldom modified again.
- **Not optimized for small data**: Small, random reads and writes do occur and are supported, but the system is not optimized for such cases.
- **Concurrent access**: The level of concurrent access will also be high, with large numbers of concurrent appends being particularly prevalent, often accompanied by concurrent reads.
- **High throughput**: GFS should be optimized for high and sustained throughput in reading the data, and this is prioritized over latency. This is not to say that latency is unimportant; rather, GFS needs to be optimized for high-performance reading and appending large volumes of data for the correct operation of the system.

GFS use cases#

- GFS is a distributed file system built for large, distributed data-intensive applications like **Gmail** or **YouTube**.

intensive applications like **Gmail** or **YouTube**.

- Originally, it was built to store data generated by Google's large **crawling and indexing system**.
- Google's **BigTable** uses the distributed Google File System to store log and data files.

APIs#

GFS does not provide standard POSIX-like APIs; instead, user-level APIs are provided. In GFS, files are organized hierarchically in directories and identified by their pathnames. GFS supports the usual file system operations:

`create` – To create a new instance of a file.
`delete` – To delete an instance of a file.
`open` – To open a named file and return a handle.
`close` – To close a given file specified by a handle.
`read` – To read data from a specified file and offset.
`write` – To write data to a specified file and offset.

In addition, GFS supports two special operations:

- **Snapshot**: A snapshot is an efficient way of creating a copy of the current instance of a file or directory tree.
- **Append**: An append operation allows multiple clients to append data to the same file concurrently while guaranteeing atomicity. It is useful for implementing multi-way merge results and producer-consumer queues that many clients can simultaneously append to without additional locking.

← Back
Mock Interview: Chubby

Next →
High-level Architecture

Mark as Completed

Report an Issue