Paper Critique

Title: The Google File System
Name: Yash Aggarwal | yagga004 | 862333037

The paper describes a filesystem used by google internally to store its files. The idea is to use commodity servers with a layer of software to provide necessary abstractions such that applications can use it as a traditional file system supporting read/ write/ append/ open/ close and other operations. The proposed design uses a master server and multiple (up to 1000) chunk servers to store files ranging from (100MB to a few GBs). The files are broken into chunks (64MB by default, with a 64-bit globally unique ID) and stored in the chunk and replica chunk servers. All the metadata is stored in the master server. As the servers used are commodity servers and can go down or have any issues(disk failure, IO failure, general outage, etc.), multiple replicas (default 3, configured by client application) are stored on different chunk servers to avoid losing the data in case of an outage. Any user application would contact the master server to get the file metadata. The master server would provide metadata along with the IP of the chunk server, and the user application and chunk server can interact with each other (send/ receive data). The replicas are made from this primary chunk server to save bandwidth. As commodity hardware is used, failure detection using a heartbeat mechanism is in place. In case of a fault, recovery, and prevention mechanisms that copy the destroyed replicas to another chunk server to get to the defined number of copies are in place. The paper describes the GFS architecture in detail regarding the role of the master and chunk server, single master principle, shadow master, append-only operation log file, file operations, and append-only system. GFS also provides new operations like snapshot creation and record appending for large files. The file system aims to be scalable using commodity hardware and still be high-performance, reliable, and fault tolerant.

Strengths :

- The paper suggests an excellent scalable design and is the choice for Hadoop and aws' flexible file system.
- The architecture is scalable using cheap commodity hardware.
- The architecture is fault-tolerant, with easy fault detection and techniques for data replication.

Weaknesses:

- The paper did not discuss issues related to data security and privacy in detail.

Comments:

- As the paper is old and did not have access to SSDs at the time, it would be interesting to know how the systems can be improved with the implementation for SSDs