**Experiment-1**

**Aim:** Case Study on GFS and Hadoop Ecosystem.

**Theory:**

**Google File System (GFS):**

Google File System (GFS) is a proprietary distributed file system developed by Google to provide efficient, reliable access to data using large clusters of commodity hardware.

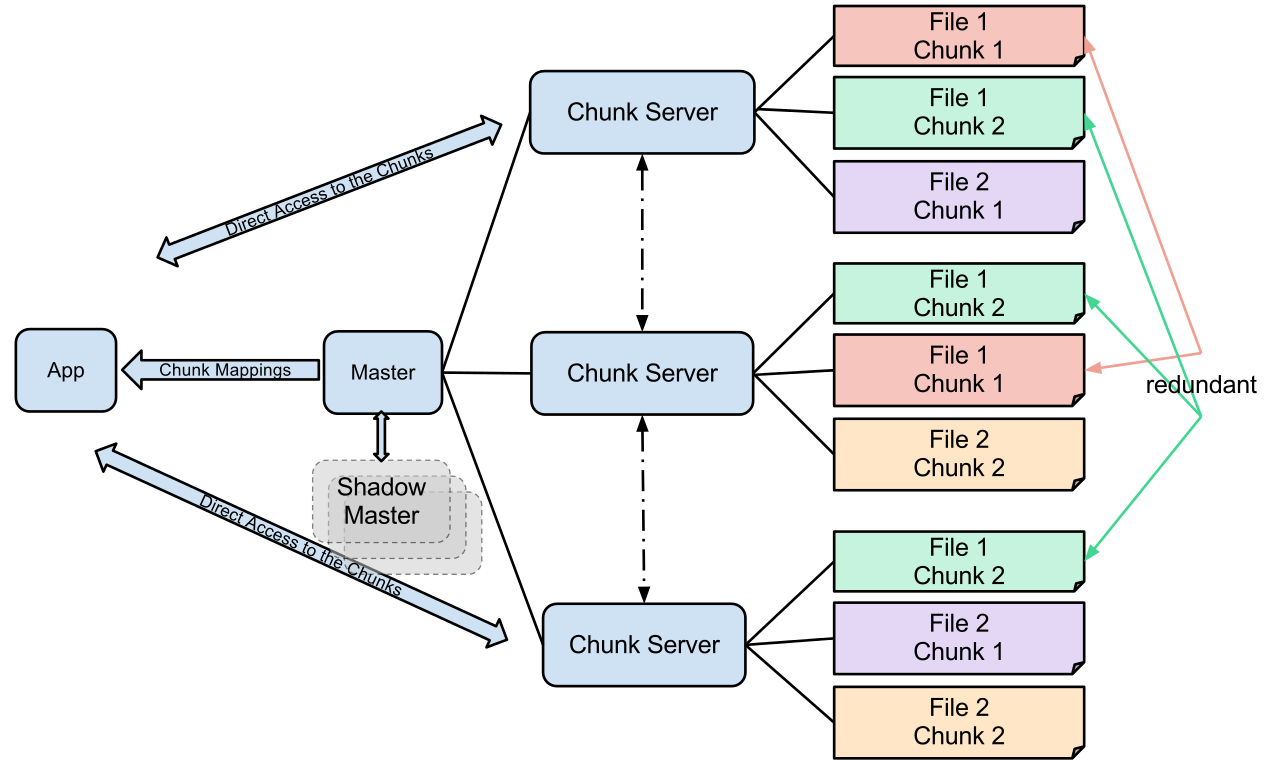
It is a scalable distributed file system (DFS) created by Google Inc. and developed to accommodate Google’s expanding data processing requirements. GFS provides fault tolerance, reliability, scalability, availability and performance to large networks and connected nodes. GFS is made up of several storage systems built from low-cost commodity hardware components. It is optimized to accommodate Google's different data use and storage needs, such as its search engine, which generates huge amounts of data that must be stored.

**Architecture:**

The most influential design choice in GFS is the storage of files in fixed-size chunks, where each chunk is 64 megabytes in size. This is quite large compared to other file system designs. At one level this simply reflects the size of the files stored in GFS. This decision is crucial to providing highly efficient sequential reads and appends of large amounts of data at another level.

Given this design choice, 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. This is achieved with the architecture shown in the figure below, which shows an instance of a GFS file system as it maps onto a given physical cluster. Each GFS cluster has a single master and multiple chunk servers (typically on the order of hundreds), which together provide file service to large numbers of clients concurrently accessing the data.

The role of the master is to manage metadata about the file system defining the namespace for files, access control information and the mapping of each particular file to the associated set of chunks. In addition, all chunks are replicated (by default on three independent chunk servers, but the level of replication can be specified by the programmer). The location of the replicas is maintained in the master. Replication is important in GFS to provide the necessary reliability in the event of (expected) hardware and software failures. This is in contrast to NFS and AFS, which do not provide replication with updates



**Performance:**

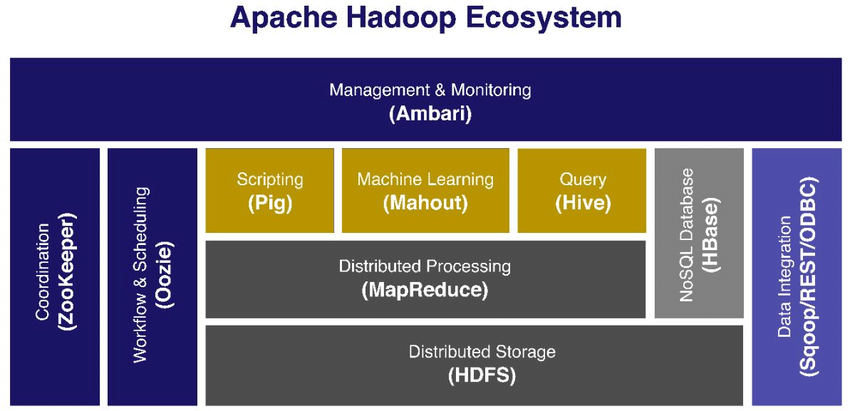
Deciding from benchmarking results, when used with a relatively small number of servers (15), the file system achieves reading performance comparable to that of a single disk (80–100 MB/s), but has a reduced write performance (30 MB/s), and is relatively slow (5 MB/s) in appending data to existing files. The authors present no results on random seek time. As the master node is not directly involved in data reading (the data are passed from the chunk server directly to the reading client), the read rate increases significantly with the number of chunk servers, achieving 583 MB/s for 342 nodes. Aggregating multiple servers also allows big capacity, while it is somewhat reduced by storing data in three independent locations (to provide redundancy).

**Hadoop:**

**What is Hadoop Ecosystem?**

​​Apache Hadoop ecosystem refers to the various components of the Apache Hadoop software library; it includes open source projects as well as a complete range of complementary tools. Some of the most well-known tools of the Hadoop ecosystem include HDFS, Hive, Pig, YARN, MapReduce, Spark, HBase, Oozie, Sqoop, Zookeeper, etc.

Hadoop Ecosystem is neither a programming language nor a service, it is a platform or framework which solves big data problems. You can consider it as a suite which encompasses a number of services (ingesting, storing, analyzing and maintaining) inside it.



**HDFS:**

HDFS is the primary or major component of Hadoop ecosystem and is responsible for storing large data sets of structured or unstructured data across various nodes and thereby maintaining the metadata in the form of log files.

Name Node is the prime node which contains metadata (data about data) requiring comparatively fewer resources than the data nodes that stores the actual data. These data nodes are commodity hardware in the distributed environment. Undoubtedly, making Hadoop cost effective.

**YARN:**

Yet Another Resource Negotiator, as the name implies, YARN is the one who helps to manage the resources across the clusters. In short, it performs scheduling and resource allocation for the Hadoop System.

Resource manager has the privilege of allocating resources for the applications in a system whereas Node managers work on the allocation of resources such as CPU, memory, bandwidth per machine and later on acknowledges the resource manager. Application manager works as an interface between the resource manager and node manager and performs negotiations as per the requirement of the two.

**MapReduce:**

By making the use of distributed and parallel algorithms, MapReduce makes it possible to carry over the processing’s logic and helps to write applications which transform big data sets into a manageable one.

MapReduce makes the use of two functions i.e. Map() and Reduce() whose task is:

1. Map() performs sorting and filtering of data and thereby organizing them in the form of group. Map generates a key-value pair based result which is later on processed by the Reduce() method.
2. Reduce(), as the name suggests does the summarization by aggregating the mapped data. In simple, Reduce() takes the output generated by Map() as input and combines those tuples into smaller set of tuples.

**Hive:**

With the help of SQL methodology and interface, HIVE performs reading and writing of large data sets. However, its query language is called as HQL (Hive Query Language).

**Mahout:**

Mahout, allows Machine Learnability to a system or application. Machine Learning, as the name suggests helps the system to develop itself based on some patterns, user/environmental interaction or om the basis of algorithms.

It provides various libraries or functionalities such as collaborative filtering, clustering, and classification which are nothing but concepts of Machine learning. It allows invoking algorithms as per our need with the help of its own libraries.

**Conclusion:**

Hence, we have successfully studied GFS and Hadoop Ecosystem.