**Intro**

HDFS, whose full name is Hadoop Distributed File System, is an open-source, distributed, scalable and portable file system written in Java for the Hadoop framework [1]. HDFS is designed for commodity hardware to store and process big data across multiple machines to achieve parallel computing. It provides an ideal pattern to store large amounts of data overcoming the limit of traditional data storage pattern. It provides a foundational file system structure for MapReduce to run its algorithm. Basic concepts of HDFS including HDFS cluster, NameNode, DataNode, data block, data block replication, secondary NameNode, standby node will be explained in detail in below sections.

**HDFS**

**Limit of traditional database**

Prior to the emergence of HDFS, traditional relational database pattern is broadly used in industry and academics. It nicely solves the problem of storing structured data of small size. However, it is not ideal enough to process large amount of data and it does not provide sufficient support to store structured data. Each block of data for traditional relational database model is small of approximately 51 bytes [2]. Another problem is there is huge amount of I/O operations while large data is being read, which consumes processing power greatly.

The birth of HDFS is to solve the problem. Each block of data for a HDFS file system is large. By default, it is 64MB and could be increased. Also, it reads large data sequentially after single seek operation, which saves processing power greatly.

**Typical architecture of HDFS (Hadoop 1.0)**

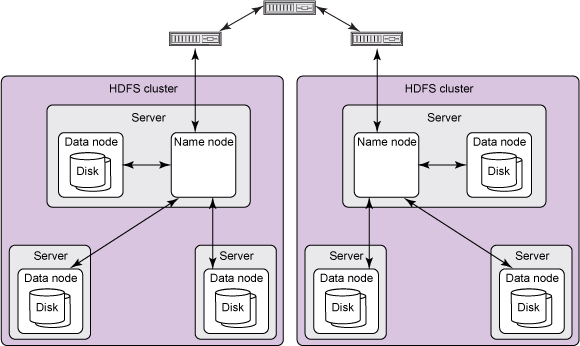


Figure [3]

The diagram shown above represents a typical architecture HDFS. Despite versions update from time to time, the basic architecture is maintained. An HDFS consists of multiple server. There is one server holding the NameNode which is very crucial as there is only one NameNode existing in the whole cluster. Several DataNodes belong to the NameNode. They can reside in the server with the NameNode and they can also be stored in different server. DataNodes hold data blocks which are where the real files are stored. This architecture demonstrates an implementation of parallel computing.

**NameNode**

A NameNode is unique within a cluster in Hadoop 1.0. It is used to store metadata such as directory structure of file system as well as basic information of the file system. A NameNode is aware of what certain file data is stored in which curtained DataNode. In version 1.0 it was a single point of failure, which means, if NameNode becomes unavailable, the whole system will go down.

A NameNode holds two persistent files which are transaction log namely edit logs and namespace image namely fsimage. Edit logs keeps recording changes of metadata (e.g. creating new file) while fsimage maintains the information of the entire file system namespace (e.g. mapping of blocks and file system properties) [4].

**Secondary NameNode**

Secondary NameNode is not a backup daemon for the NameNode [5]. That means, we cannot treat it as a back-up NameNode and we cannot completely depend on secondary NameNode for the recovery process of a NameNode. It is configured more like a check-pointer [6]. Secondary NameNode stores the latest checkpoint in a directory by maintaining two persistent file edit logs and fsimage which is structured the same way as the primary NameNode’s directory. Such check-pointed image is always ready to be read by the primary NameNode [7] and thus reduce the restart time of NameNode [5].

**DataNode**

DataNode stores and maintains data blocks which hold the actual file data. It is responsible to store and retrieve data blocks upon request from NameNode or the client. There can be multiple numbers of DataNodes depending on the system requirement. Operations such as read, write, block creation and replication can be performed on DataNode.

**Data Block**

Data block holds actual file data. Each file is split into one or more blocks stored and replicated in DataNodes. By default, each block is 64MB and the size could be increased to 128MB.

**Data Block Replication**

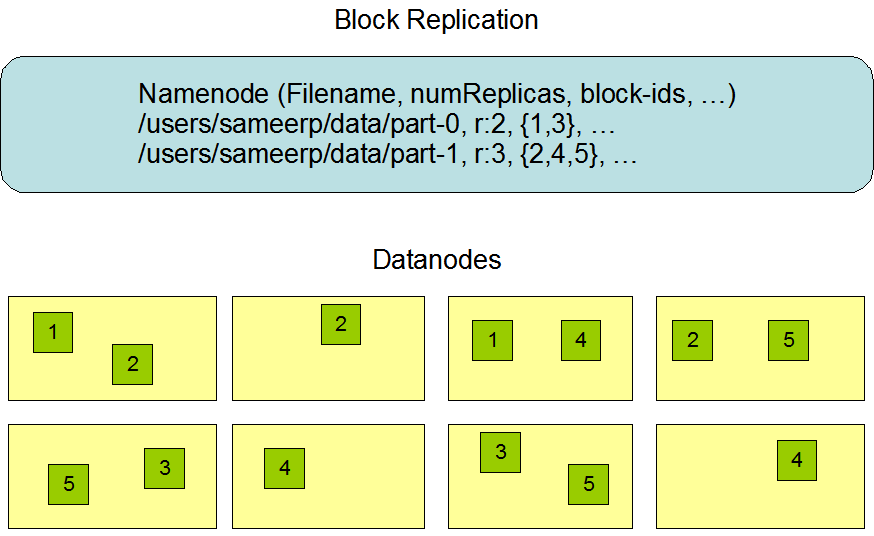


Figure [4]

The data blocks are replicated for fault tolerance. The data blocks are distributed in DataNode system within the cluster and thus ensures the replica of data is maintained. The administrator of HDFS can designate which rack a node belongs. This is called rack awareness.

**Interactions of Each Components**

All HDFS communication protocols build on TCP/IP protocol. Client communicate with the name node using a Remote Procedure Call (RPC)-based protocol. Each data node serves up blocks of data over the networking using block protocol specific to HDFS [3].

**Weakness of HDFS in Hadoop 1.0**

There was a single point of failure problem existing in version 1.0. Once NameNode server goes down, the whole cluster becomes unavailable. Measures need to be taken to ensure High Availability.

**Support of Standby NameNode in later version**

In later version, standby NameNode is supported to ensure the High Availability feature of Hadoop [8]. The standby NameNode is in the same cluster. Two or more nodes in the cluster are in Active/Passive state and only one NameNode is allowed to be in Active state at a time.

**Reference**

[1] Apache Hadoop. (n.d.). Retrieved October 18, 2016, from <https://en.wikipedia.org/wiki/Apache_Hadoop\>

[2] Hadoop Architecture Tutorial | Hadoop Tutorial For Beginners. (2016). Retrieved October 18, 2016, from <https://www.youtube.com/watch?v=iIPItenMIOw>

[3] An introduction to the Hadoop Distributed File System. (n.d.). Retrieved October 18, 2016, from <https://www.ibm.com/developerworks/library/wa-introhdfs/#communications_protocols_sidebar>

[4] HDFS Architecture Guide. (n.d.). Retrieved October 18, 2016, from <http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html>

[5] Multi-host SecondaryNameNode Configuration - Cloudera Engineering Blog. (2009). Retrieved October 18, 2016, from <http://blog.cloudera.com/blog/2009/02/multi-host-secondarynamenode-configuration/>

[6] [HADOOP-4539] Streaming Edits to a Backup Node. - ASF JIRA. (n.d.). Retrieved October 18, 2016, from <https://issues.apache.org/jira/browse/HADOOP-4539>

[7] HDFS Users Guide. (n.d.). Retrieved October 18, 2016, from <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html>

[8] B. (n.d.). Apache Hadoop HDFS - Hortonworks. Retrieved October 18, 2016, from <http://hortonworks.com/apache/hdfs/>

[] The Hadoop Distributed File System. (n.d.). Retrieved October 18, 2016, from <http://www.aosabook.org/en/hdfs.html>