ASSIGNMENT

BIG DATA TECHNOLOGIES

Submitted by,

Jesliya p v

S4,MCA

Roll.no:21

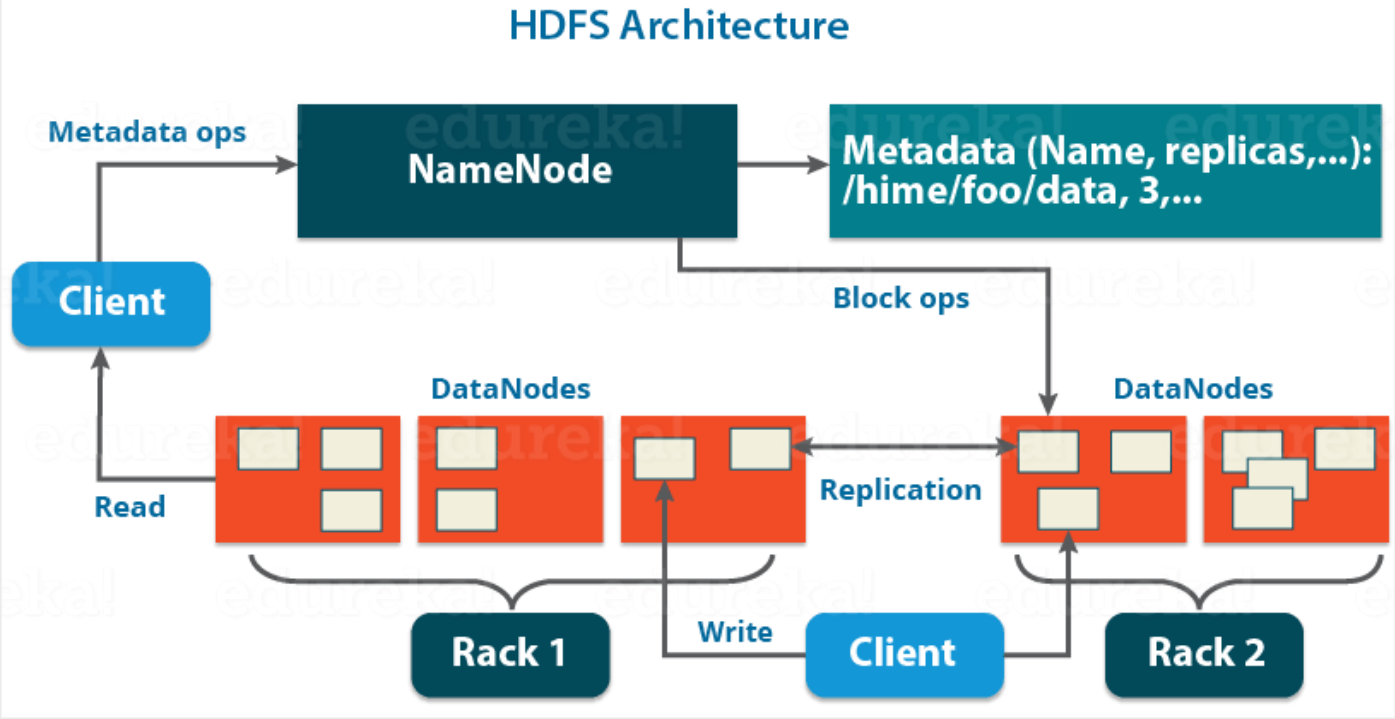
HDFS ARCHITECTURE

HDFS – Hadoop Distributed File System, It is a cluster of storage solutions that is highly reliable, more efficient and economical and provides facilities to manage files containing related data across machines.

HDFS stores very large files running on a cluster of commodity hardware. It works on the principle of storage of less number of large files rather than the huge number of small files. HDFS stores data reliably even in the case of hardware failure. It provides high throughput by providing the data access in parallel.

HDFS Architecture:

Hadoop Distributed File System follows the **master-slave architecture**. Each cluster comprises a **single master node** and **multiple slave nodes**. Internally the files get divided into one or more **blocks**, and each block is stored on different slave machines depending on the **replication factor** .



The master node stores and manages the file system namespace, that is information about blocks of files like block locations, permissions, etc. The slave nodes store data blocks of files.

The Master node is the NameNode and DataNodes are the slave nodes.

NameNode

NameNode is the centerpiece of the Hadoop Distributed File System. It maintains and manages the **file system namespace** and provides the right access permission to the clients.

The NameNode stores information about blocks locations, permissions, etc. on the local disk in the form of two files:

- **Fsimage:** Fsimage stands for File System image. It contains the complete namespace of the Hadoop file system since the NameNode creation.
- **Edit log:** It contains all the recent changes performed to the file system namespace to the most recent Fsimage.

Functions of HDFS NameNode:

1. It executes the file system namespace operations like opening, renaming, and closing files and directories.
2. NameNode manages and maintains the DataNodes.
3. It determines the mapping of blocks of a file to DataNodes.
4. NameNode records each change made to the file system namespace.
5. It keeps the locations of each block of a file.
6. NameNode takes care of the replication factor of all the blocks.
7. NameNode receives heartbeat and block reports from all DataNodes that ensure DataNode is alive.
8. If the DataNode fails, the NameNode chooses new DataNodes for new replicas.

Before Hadoop2, NameNode was the single point of failure. The High Availability Hadoop cluster architecture introduced in Hadoop 2, allows for two or more NameNodes running in the cluster in a hot standby configuration.

DataNode

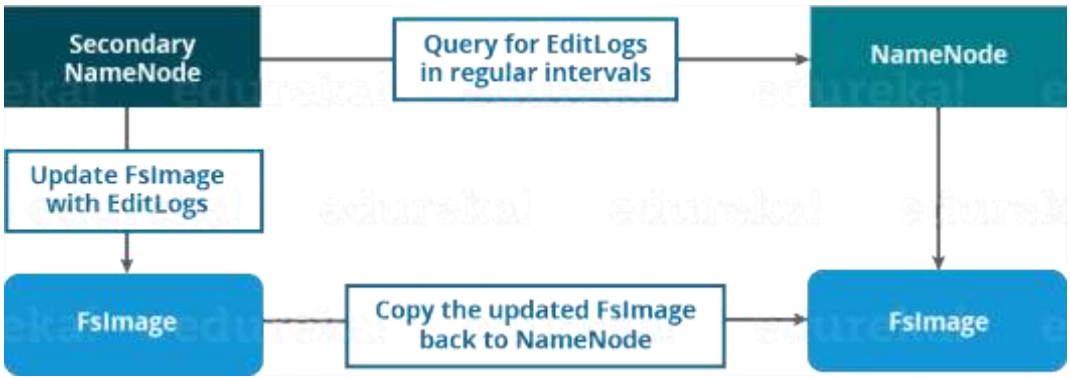
DataNodes are the slave nodes in Hadoop HDFS. DataNodes are **inexpensive commodity hardware**. They store blocks of a file.

Functions of HDFS DataNode:

1. DataNode is responsible for serving the client read/write requests.
2. Based on the instruction from the NameNode, DataNodes performs block creation, replication, and deletion.
3. DataNodes send a heartbeat to NameNode to report the health of HDFS.
4. DataNodes also sends block reports to NameNode to report the list of blocks it contains.

Secondary NameNode:

Apart from these two daemons, there is a third daemon or a process called Secondary NameNode. The Secondary NameNode works concurrently with the primary NameNode as a **helper daemon.** And don't be confused about the Secondary NameNode being a backup NameNode because it is not.



Functions of Secondary NameNode:

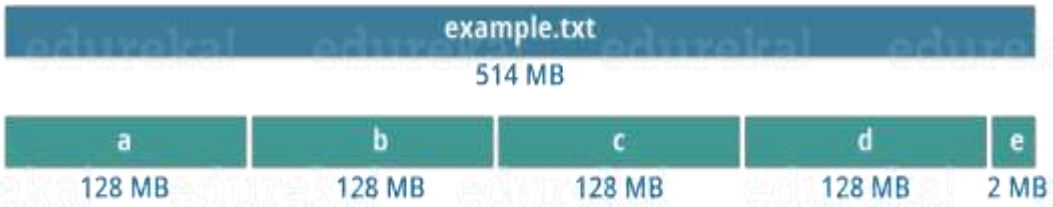
1. The Secondary NameNode is one which constantly reads all the file systems and metadata from the RAM of the NameNode and writes it into the hard disk or the file system.
2. It is responsible for combining the EditLogs with FsImage from the NameNode.
3. It downloads the EditLogs from the NameNode at regular intervals and applies to FsImage. The new FsImage is copied back to the NameNode, which is used whenever the NameNode is started the next time.

Hence, Secondary NameNode performs regular checkpoints in HDFS. Therefore, it is also called CheckpointNode.

Blocks:

Blocks are the nothing but the smallest continuous location on your hard drive where data is stored.

In general, in any of the File System, you store the data as a collection of blocks. Similarly, HDFS stores each file as blocks which are scattered throughout the Apache Hadoop cluster. The default size of each block is 128 MB in Apache Hadoop 2.x (64 MB in Apache Hadoop 1.x) which you can configure as per your requirement.



It is not necessary that in HDFS, each file is stored in exact multiple of the configured block size (128 MB, 256 MB etc.).

Let's take an example where I have a file "example.txt" of size 514 MB as shown in above figure. Suppose that we are using the default configuration of block size, which is 128 MB. Then, how many blocks will be created? 5, Right. The first four blocks will be of 128 MB. But, the last block will be of 2 MB size only.The file of a smaller size does not occupy the full block size space in the disk.

HDFS is highly fault-tolerant.

Replication Management

For a distributed system, the data must be redundant to multiple places so that if one machine fails, the data is accessible from other machines.

In Hadoop, HDFS stores replicas of a block on multiple DataNodes based on the replication factor.The replication factor is the number of copies to be created for blocks of a file in HDFS architecture.

If the replication factor is 3, then three copies of a block get stored on different DataNodes. So if one DataNode containing the data block fails, then the block is accessible from the other DataNode containing a replica of the block.If we are storing a file of 128 Mb and the replication factor is 3, then (3*128=384) 384 Mb of disk space is occupied for a file as three copies of a block get stored.

This replication mechanism makes HDFS fault-tolerant.

Rack Awareness in HDFS Architecture

Rack is the collection of around 40-50 machines (DataNodes) connected using the same network switch. If the network goes down, the whole rack will be unavailable.

Rack Awareness is the concept of choosing the closest node based on the rack information.

To ensure that all the replicas of a block are not stored on the same rack or a single rack, NameNode follows a rack awareness algorithm to store replicas and provide latency and fault tolerance.

Suppose if the replication factor is 3, then according to the rack awareness algorithm:

1. The first replica will get stored on the local rack.
2. The second replica will get stored on the other DataNode in the same rack.
3. The third replica will get stored on a different rack.

HDFS Read and Write Operation

1. Write Operation

When a client wants to write a file to HDFS, it communicates to the NameNode for metadata. The Namenode responds with a number of blocks, their location, replicas, and other details.

Based on information from NameNode, the client directly interacts with the DataNode.The client first sends block A to DataNode 1 along with the IP of the other two DataNodes where replicas will be stored.

When Datanode 1 receives block A from the client, DataNode 1 copies the same block to DataNode 2 of the same rack. As both the DataNodes are in the same rack, so block transfer via rack switch.

Now DataNode 2 copies the same block to DataNode 4 on a different rack. As both the DataNoNes are in different racks, so block transfer via an out-of-rack switch.When DataNode receives the blocks from the client, it sends write confirmation to Namenode.The same process is repeated for each block of the file.

2. Read Operation

To read from HDFS, the client first communicates with the NameNode for metadata. The Namenode responds with the locations of DataNodes containing blocks. After receiving the DataNodes locations, the client then directly interacts with the DataNodes.

The client starts reading data parallelly from the DataNodes based on the information received from the NameNode. The data will flow directly from the DataNode to the client.

When a client or application receives all the blocks of the file, it combines these blocks into the form of an original file.

Summary

- Hadoop Distributed File System follows the *master-slave architecture*. Each cluster comprises a *single master node and multiple slave nodes.*
- The HDFS divides the files into blocks.
- The size of the block is 128 Mb by default, which we can configure as per the requirements.
- The master node (NameNode) stores and manages the metadata about block locations, blocks of a file, etc.
- The DataNode stores the actual data blocks. The Master Node manages the DataNodes.
- HDFS creates replicas of blocks and stores them on different DataNodes in order to provide fault tolerance.
- Also, NameNode uses the Rack Awareness algorithm to improve cluster performance.