ANISH ADHANI
D17B/01

EXPERIMENT 11

Aim: Distributed file system

Theory:

HDFS:

Hadoop is a free, java based programming framework that supports the processing of large data sets in a satisfied distributed computing environment

Hadoop makes all possible to run application on systems with thousands of nodes involving thousands of terabytes

Its distributed file system, facilitates rapid data transfer rate among nodes to the system to continue operating uninterrupted in case of node failure

Hadoop was inspired by google's mapreduce, a software framework in which all application is taken down into numerous small parts.

Any of these parts can run on any node in the cluster:

Hadoop is used by
- Google
- Yahoo
- IBM

- Hadoop file system was developed using distributed file system design. It is run on commodity hardware. unlike other distributed system, HDFS is highly fault tolerant and designed using low-cost hardware.

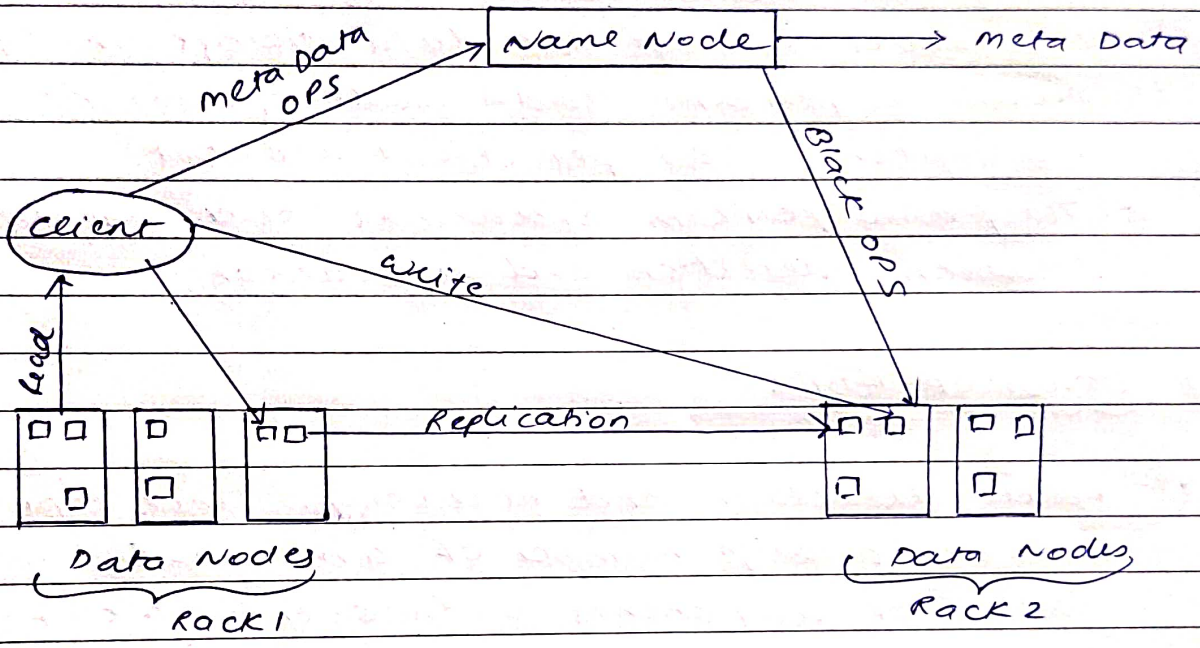
Hadoop holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS also makes application available to parallel processing

Preferred OS
- Windows
- Linux

Features of HDFS:

- It is suitable for the distributed storage & processing
- Hadoop provides a command interface to interact with HDFS
- The built in servers of namenodes and datanode help users to easily check the status of cluster
- Streaming access to file system data.

HDFS Architecture



HDFS follows the master-slave architecture and it has the following elements.

Namenode :

The namenode is the commodity hardware that contains the GNU/Linux operating system and the namenode software. The system having the namenode acts as the master server and does following task

- managing the file system namespace
- Regulate client access to files

Datanode

The datanode is a commodity hardware having the GNU/Linux OS and datanode software These nodes manage the data storage of system
- datanode perform read-write operations on the file systems, as per client request
- They also perform operations such as block creation, detection and replication.

Goals of HDFS

1) Fault detection and Recovery: Since HDFS includes a large number of commodity hardware, failure of components is frequent. Therefore HDFS should have mechanism for quick and automatic fault detection and recovery.

2) Huge datasets: HDFS should have hundreds of nodes per cluster to manage the application having huge datasets

3) Hardware at data: A requested task can be done efficiently, when the computation takes place near the data.

Conclusion:

We understood the concept of Hadoop and implemented all basic hadoop command successfully.

Starting HDFS

Initially, you have to format the configured HDFS file system, open namenode (HDFS server), and execute the following command.

```
$ hadoop namenode -format
```

After formatting the HDFS, start the distributed file system. The following command will start the namenode as well as the data nodes as the cluster.

```
$ start-dfs.sh
```

Listing Files in HDFS

After loading the information in the server, we can find the list of files in a directory, status of a file, using 'ls'. Given below is the syntax of ls that you can pass to a directory or a filename as an argument.

```
$ $HADOOP_HOME/bin/hadoop fs -ls <args>
```

Inserting Data into HDFS

Assume we have data in the file called file.txt in the local system which is ought to be saved in the hdfs file system. Follow the steps given below to insert the required file in the Hadoop file system.

Step 1:

You have to create an input directory.

```
$ $HADOOP_HOME/bin/hadoop fs -mkdir /user/input
```

Step 2:

Transfer and store a data file from local systems to the Hadoop file system using the put command.

```
$ $HADOOP_HOME/bin/hadoop fs -put /home/file.txt /user/input
```

Step 3:

You can verify the file using ls command.

```
$ $HADOOP_HOME/bin/hadoop fs -ls /user/input
```

Retrieving Data from HDFS

Assume we have a file in HDFS called outfile. Given below is a simple demonstration for retrieving the required file from the Hadoop file system.

Step 1:

Initially, view the data from HDFS using cat command.

```
$ $HADOOP_HOME/bin/hadoop fs -cat /user/output/outfile
```

Step 2:

Get the file from HDFS to the local file system using get command.

```
$ $HADOOP_HOME/bin/hadoop fs -get /user/output/ /home/hadoop_tp/
```

Shutting Down the HDFS

You can shut down the HDFS by using the following command.

```
$ stop-dfs.sh
```