Big Data - Concepts

21 December 2023 09:07

3. Hadoop Distributed File System (HDFS)

Lecture

- Distributed File System,
- What is HDFS,
- Where does HDFS fit in,
- Core components of HDFS,
- HDFS Daemons,
- Hadoop Server Roles: Name Node, Secondary Name Node, and Data Node

4. HDFS Architecture

Lecture

- HDFS Architecture,
- Scaling and Rebalancing,
- Replication,
- Rack Awareness,
- Data Pipelining,
- Node Failure Management.
- HDFS High Availability NameNode

Lab-Assignment:

- Run the HDFS commands, and add a one liner understanding for each of the command.
- Execute the provided code using HDFS, step run and understand

5. Hadoop Installation and Cluster Configuration

Getting Started: Hadoop Installation

Lecture

- Hadoop Operation modes
- Setting up a Hadoop Cluster,
- Cluster specification,
- Single and Multi-Node Cluster Setup on Virtual & Physical Machines,
- Remote Login using Putty/Mac Terminal/Ubuntu Terminal.
- Hadoop Configuration, Security in Hadoop, Administering Hadoop,
- HDFS – Monitoring & Maintenance, Hadoop benchmarks,
- Hadoop in the cloud.

6. Hadoop Architecture

Lecture

- Hadoop Architecture,
- Core components of Hadoop,
- Common Hadoop Shell commands.

Hadoop - HDFS Overview

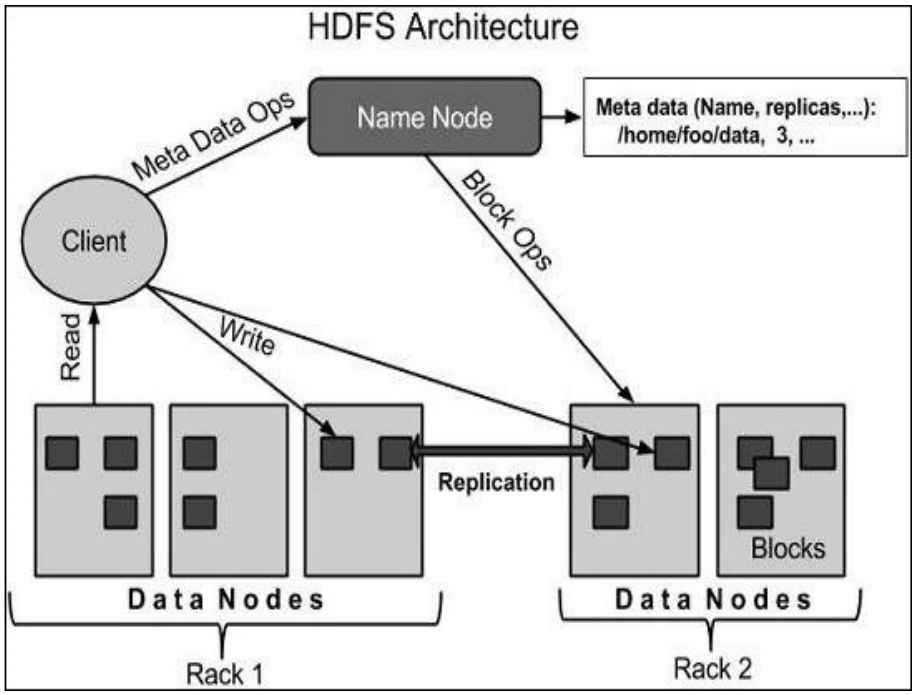
- Hadoop file systems was developed using distributed file system design.
- It is designed to run even on commodity or lower end hardware.
- HDFS is highly FT and designed using lo-cost hardware.
- It can hold very large amount of data and provides easier access.
- To store huge data, the files are spread across multiple machines
- These files are stored in redundant fashion to rescue the system from possible data losses in of failures
- HDFS also makes applications available for parallel processing

Features of HDFS

- IT is suitable for distributed storage and processing
- Hadoop provides a CLI to interact with HDFS
- The built-in servers of namenode and datanode help users to easily check the status of cluster.
- Streaming access to file system data
- HDFS provides file permissions and authentications

HDFS Architecture

Architecture of Hadoop File System



- **HDFS follows Master-Slave architecture**

Components / Elements of HDFS

- **Namenode**
  - ◇ This is the commodity hardware that contains the GNU/Linux OS and namenode software
  - ◇ Namename acts as a master server and performs the following tasts
    - ○ Manage the file system namespace
    - ○ Regulates client;s access to files
    - ○ It also executes file system operations such as renaming, closing and opening files and directories

- **Datanode**
  - ◇ This is the commodity hardware that contains the GNU/Linux OS and datanode software
  - ◇ For every node in a cluster, there will be a datanode.
  - ◇ These nodes manages the data storage of their systems.
  - ◇ Perform read-write operations on the file systems as per the client request
  - ◇ Perfrom operations such as block creation, deletion and replications according to the instructions of the namenode

- **Blocks**
  - ◇ User data is stored in files of HDFS and the file in a file system will be divided into one or more segments and/or stored in individual data nodes.
  - ◇ These file segments are called as blocks.
  - ◇ The minimum amount of data that HDFS can read or write is called a block
  - ◇ Default block size is 64MB , but can be increased as per the need , by chaining the HDFS configuration

(preferred block size is 128MB)

- **Goals of HDFS**

Fault detection and recovery - HDFS includes a very large number of commodity hardware because of this failure of components is frequent hence there should be mechanisms for quick and automatic fault detection and recovery.
Huge datasets - HDFS should have hundreds of nodes per cluster to manage the applications
A requested task can be done efficiently, when the computation takes place near the data. Where huge datasets are involved, it reduces the network traffic and increases the throughput. (perform are calculated based on the IOPS - input output operations per second)

HDFS operations

1. Starting HDFS -
   a. Before you start you need to intially format the confgured HDFS file systems, to be on namenode (HDFS server) by executing the following command

      $ hadoop namenode -format

   b. Now after formatting the HDFS start the distributed file systems by sing the following command.

      $ start-dfs.sh (this command will start the namenode as well as the datanodes as cluster)
2. List files in HDFS
   a. To view the Hadoop home directory ----> **$ echo $HADOOP_HOME**
   b. To list the contents present the hadoop root directory ---> **$ $HADOOP_HOME/bin/hadoop fs - ls /**
3. Insert data into HDFS
   a. To create a directory ---> **$HADOOP_HOME/bin/hadoop fs -mkdir /mydata**
   b. To create a sub-directory under /mydata ---> **$HADOOP_HOME/bin/hadoop fs -mkdir /mydata/input**
   c. To transfer or store data form the local system to the hadoop file systm --> **$HADOOP_HOME/bin/hadoop fs -put /etc/*.conf /mydata/input**
   d. To transfer a file created in the present location to the Hadoop file system --> $ cat > <filename> after inserting the content save by suing ctrl+d options only once eg: **cat > test.txt** { add your relevent text and save it} now use the following command **$HADOOP_HOME/bin/hadoop fs -put ./test.txt /mydata/input**
4. Verify the file in HDFS
   a. To verify the file present in Hadoop ---> **$HADOOP_HOME/bin/hadoop fs -ls /mydata/input/**
5. Retrieve data form HDFS
   a. To view the data from the HDFS ---> **$HADOOP_HOME/bin/hadoop fs -cat /mydata/input/test.txt**
   b. To download or get the file from HDFS to local file system ---> **$HADOOP_HOME/bin/hadoop fs -get /mydata/input/test.txt**
6. Shutdown the HDFS
   a. We can shutdown the HDFS by using the following command ---> **$ stop-dfs.sh**

Other useful Command list

```
$HADOOP_HOME/bin/hadoop fs -help
 82 $HADOOP_HOME/bin/hadoop fs -ls /
 83 $HADOOP_HOME/bin/hadoop fs -ls /mydata/
 84 $HADOOP_HOME/bin/hadoop fs -ls /mydata/input
 85 $HADOOP_HOME/bin/hadoop fs -ls /mydata/input/test.txt
 86 $HADOOP_HOME/bin/hadoop fs -mv /mydata/input/test.txt /mydata/input/oldfile.txt
 87 $HADOOP_HOME/bin/hadoop fs -ls /mydata/input/test.txt
 88 $HADOOP_HOME/bin/hadoop fs -ls /mydata/input/oldfile.txt
 89 $HADOOP_HOME/bin/hadoop fs -du /mydata/input/oldfile.txt
 90 $HADOOP_HOME/bin/hadoop fs -du /mydata/input
 91 $HADOOP_HOME/bin/hadoop fs -dus /mydata/input
 92 du -hs /home/hadoop
 93 $HADOOP_HOME/bin/hadoop fs -duh /mydata/input
 94 $HADOOP_HOME/bin/hadoop fs -dus -m /mydata/input
 95 $HADOOP_HOME/bin/hadoop fs -dus -help
 96 $HADOOP_HOME/bin/hadoop fs -du -s /
 97 $HADOOP_HOME/bin/hadoop fs -du -sh /
 98 $HADOOP_HOME/bin/hadoop fs -du -s -h /
```

```
 99  $HADOOP_HOME/bin/hadoop fs -mkdir /mydata/newdir1
100  $HADOOP_HOME/bin/hadoop fs -cp /mydata/input/oldfile.txt  /mydata/newdir1/newfile.txt
101  $HADOOP_HOME/bin/hadoop fs -cat  /mydata/newdir1/newfile.txt
102  $HADOOP_HOME/bin/hadoop fs -cat >> /mydata/newdir1/newfile.txt
103  $HADOOP_HOME/bin/hadoop fs -rm  /mydata/input/oldfile.txt
104  $HADOOP_HOME/bin/hadoop fs -stat /mydata/newdir1/newfile.txt
105  $HADOOP_HOME/bin/hadoop fs -stat %b /mydata/newdir1/newfile.txt
106  $HADOOP_HOME/bin/hadoop fs -stat %o /mydata/newdir1/newfile.txt
107  $HADOOP_HOME/bin/hadoop fs -stat %n /mydata/newdir1/newfile.txt
108  $HADOOP_HOME/bin/hadoop fs -stat %r /mydata/newdir1/newfile.txt
109  ls
110  nano hadoopcmd.txt
111  $HADOOP_HOME/bin/hadoop fs -put ./hadoopcmd.txt /mydata/newdir1
112  $HADOOP_HOME/bin/hadoop fs -tail -f /mydata/newdir1/hadoopcmd.txt
113  cp hadoopcmd.txt listofcmd.txt
114  $HADOOP_HOME/bin/hadoop fs -copyFromLocal ./listofcmd.txt/mydata/newdir1
115  $HADOOP_HOME/bin/hadoop fs -copyFromLocal ./listofcmd.txt /mydata/newdir1
116  $HADOOP_HOME/bin/hadoop fs -lsr ./listofcmd.txt /mydata/newdir1
```

Hadoop - Multi-Node Cluster

Hadoop Master --> 192.168.56.100 (mainserver1)
Hadoop Slave1 ---> 192.168.56.101 (slave1)
Hadoop Slave2 ---> 192.168.56.102 (slave2)

Perform the following activity on all the nodes

1. create and account by name Hadoop and provide the password
   $ sudo passwd root
   $ sudo useradd -d /home/hadoop -m hadoop
   $ sudo passwd hadoop
   $ su - root
2. Setup the Network and verify the connectivity

```
cd /etc/netplan/
ls
cat 00-installer-config.yaml
nano 00-installer-config.yaml
```

```
root@slave1:/etc/netplan# cat 00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: no
      addresses:
        - 192.168.56.101/24
  version: 2
root@slave1:/etc/netplan# _
```

$ netpaln apply
$ ip a

```
slaveNode2 [Running] - Oracle VM VirtualBox
root@slave2:/etc/netplan# cat 00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: no
      addresses:
        - 192.168.56.102/24
  version: 2
root@slave2:/etc/netplan# netplan apply
root@slave2:/etc/netplan#
```

$ ping 192.168.56.101 -c 4
$ ping 192.168.56.100 -c 4
Note: you should be able to communicate with all the nodes

3. perform the name resolution using the local hosts file ie., /etc/hosts

   Add the following entry in all the nodes (do not remove the existing entries present in this file)

   $ nano /etc/hosts

   192.168.56.100 mainserver1
   192.168.56.101 slave1
   192.168.56.102 slave2
   <save and exit>

4. Establish key based login
   Login as Hadoop user and perform the following activity

   $ ssh-keygen -t rsa
   $ ssh-copy-id hadoop@<specify the node names> { do not include the current node }

5. We need to configure java
6. Install Hadoop

Hadoop Master-Slave Configuration Setup

1. Configure Hadoop in an master-slave setup