# HDFS Architecture

* HDFS mean Hadoop Distributed File system.
* It also a logical file system.
* Three main daemons in HDFS are NameNode(NN), DataNode(DN), SecondaryNameNode(SNN).
* Daemons means the background process which runs all the times.
* All of them are java based software’s running in cluster(hosts) nothing is physical one.
* We call NN & SNN called as master node and DN called as worker node.
* It called as cluster topology (Bird side view of cluster).
* You must connect to cluster through NN or DN or Gateway node.
* When a client put files in HDFS, they split into many blocks and these blocks stored physically in HDFS as distributed manner. Blocks are called as physical division of data while splits called as logical division of data.
* NN is a in memory component, metadata information and is best practise to store large files by increasing the block size to avoid NN saturation quickly. So, that NN handle less metadata.
* NN contains the metadata of files(blocks) stored in DN.
* When a job submitted by client through gateway node, it talks to NN first and get the metadata of the files, free memory faces in DN, then client takes care of all things, it splits the files into blocks and stored it in DN.
* Metadata mean the mapping between file to blocks, block id, block location and file permissions of a block.
* NN also keep track of usage of cluster. DN send its heartbeat to NN and send its info about their live status, block report.
* “Balancer” will be skewed and balancing the blocks stored across the cluster.
* DN is slave node where the actual stored in form of blocks.
* As NN is a in memory component, when cluster accidentally shuts down, it loses its metadata.
* When server or cluster brings up NN losses it contents in memory, NN maintain two file structures “FS image” and “Edit Logs”.
* Edit Logs is the one which keep tracks of every entry in memory component of NN(metadata).
* FS image is nothing but filesystem image, it but snapshot of NN in memory contents.
* For the metadata, we have periodic backup of FS image will created once every hour or ten thousand transaction occurs in memory content.
* SNN takes care of creating latest snapshot of FS image, which takes old FS image snapshot and merge the latest edit logs and it happens at regular interval called “check point”.

# Parameter Files

* These are one which controls the behaviour of cluster and jobs running.
* Generally, these files are in “etc” location and are of two types “.sh “and “xml files”.
* For HDFS, “core-site.xml” and “hdfs-site.xml”.

## CORE-Site.xml

* Core-site xml having common properties to both hdfs and yarn.
* In core-site xml, one most important property “fs.defaultFS” tell us where the NN resides in cluster. It basically tell us the NN url(ip address) and port.

<property>

<name>fs.defaultFS</name>

<value>hdfs://sandbox.hortonworks.com:8020</value>

<final>true</final>

</property>

* Compression algorithm, thrash, security parameters under core-site xml common for all.

## HDFS-Site.XML

* In HDFS-Site xml contains property only related to HDFS such as replication factor, block size, http-address (gives NN web interface).

<property>

<name>dfs.blocksize</name>

<value>134217728</value>

</property>

<property>

<name>dfs.heartbeat.interval</name>

<value>3</value>

</property>

<property>

<name>dfs.namenode.http-address</name>

<value>sandbox.hortonworks.com:50070</value>

<final>true</final>

</property>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

* Also, contains two important directory files dfs.namenode.dir, dfs.datanode.dir, where our actual files reside in form of blocks. These are nothing but mount point of NN.

<property>

<name>dfs.namenode.data.dir</name>

<value>/hadoop/hdfs/namenode</value>

<final>true</final>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>/hadoop/hdfs/data</value>

<final>true</final>

</property>

* Edit logs stored in this location of NN (dfs.namenode.data.dir).

[root@sandbox current]# pwd

/hadoop/hdfs/namenode/current

Outputs

======

edits\_0000000000000016519-0000000000000017000

edits\_0000000000000017001-0000000000000017347

edits\_inprogress\_0000000000000017348

fsimage\_0000000000000005674

fsimage\_0000000000000005674.md5

fsimage\_0000000000000010444

* Blocks are stored in this location of DN(dfs.datanode.data.dir).