# HDFS Architecture

## Hadoop

## =======

Hadoop is an open source software framework for developing distributed applications that enables parallel processing of large data sets against cluster of commodity hardware.

## How Hadoop is solving Big data challenges.

## ===================================

* Supports huge volume of data storage
* Cost Effective
* Good data recovery
* Horizontal scaling
* Distributed and parallel processing
* 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.

## Hadoop core components

## =====================

Hadoop 1

========

HDFS -- Storage system

MapReduce -- Processing and resource allocation Framework

Hadoop 2

========

HDFS -- Storage system

MapReduce -- Processing Framework

YARN --- Resource allocation

HDFS

====

1)HDFS is primary storage system and designed for storing very large files with streaming data access patterns, running on cluster of commodity hardware's.

2)HDFS takes the data in form of blocks and distributes them in cluster for parallel processing.

3)HDFS files interact each other and passing info from one file to other.

4)HDFS is not physical file system. It's virtual abstraction on top your file system.

HDFS Architecture

================

It's a master-slave architecture.

Master is NameNode and slave is DataNode. Also, we have secondary NameNode.

Files are divided into blocks and stored across in DataNode.

By default, block size is 128 MB.

By default, replication factor is 3.

## NameNode

## =========

1)It's the single point of contact. Any such kind of activity as adding files, removing files, any action happened in Hadoop cluster, NN is the decider.

2)NN maintains the metadata information blocks (i.e Blockid, Size, location, Permission details) all are stored in NN.

3)NN metadata consist of two things "Edit logs" and "FS image".

4)Entire metadata information stores in memory of NN for faster retrieval of blocks info.

5)It contains Metadata and Block Location.

6) 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.

7)NN also keep track of usage of cluster. DN send its heartbeat to NN and send its info about their live status, block report.

8)“Balancer” will be skewed and balancing the blocks stored across the cluster.

9)As NN is a in memory component, when cluster accidentally shuts down, it loses its metadata.

10)When server or cluster brings up NN losses it contents in memory, NN maintain two file structures “FS image” and “Edit Logs”.

## Metadata

## ========

1)It's includes Filename, File size, No.of Blocks, Block ID's, User, Group, Permission, Replication, Block size etc.

2)Block Id is unique across the cluster.

NN is recovered using Edit logs and FS Image, it recover only the metadata not the block location.

## Block Location

## ============

1)Using the block report received from DN, NN built "block id to location mapping" called Block location.

2)It also important for HDFS writes.

## EDIT logs

## ========

1)changes done in HDFS like placing file or changing attributes of a file called "Transactional logs".

2)Whatever changes in HDFS files happen are edited in edit logs for durability.

3)Each operation in edit logs called "Transaction" and each transaction have "Transaction id".

## FS image

## ========

1)It just like an "iso image file" like our windows, linux iso images.

2)It represents the complete snapshot of entire HDFS system and it occurs at regular intervals based on parameter configuration.

## Secondary NameNode (check point node)

## ==================================

1)It acts as the book keeping of the NN. It takes the latest FS image snapshot and merge the edit logs, it creates the new FS image and send back to NN for recovery process in case NN restarts or failure.

2)It only used backup and recovery.

3)By default it's check point is one hour.

4)We call this mechanism as "check point mechanism".

## Check point

## ==========

1)It periodically merges the FS Image and Edit logs and created the new FS image and send back to NN.

2)It helps for sooner recover of NN during its bootup or restarts.

Parameter:

==========

dfs.namenode.checkpoint.period (1hr)& dfs.namenode.checkpoint.txns (1 million)

3)Either of the check point parameter is reached and it triggers the checkpoint and created the new FS image.

## DataNode

## ========

1)Files are splits into small chunks called blocks and stored in DN.

2)Data in HDFS are stored as "Blocks" where the actual data stored.

3)It responsible for block creation, deletion and replication of files.

4)It gets its instruction from NN and stores the blocks.

5)During start up and periodically, it scans the whole disk and send its report to NN called "Block Report".

6)Along with Block ID, it sends generation stamp and block length (size of block)

7)Generation stamp is used to determine block version and used for append operation.

8)DN is slave node where the actual stored in form of blocks.

## Heartbeats(HB)

## ============

1)DN send it heartbeats to NN for every "3 sec" default, said hey NN am alive and having this much of "free space, used space, Total space, Data Transfer in progress".

2)HB also carries information of storage in use, data transfer happens in DN.

3)NN send its instruction to DN through heartbeats.

4)Using the HB form DN, NN allocates blocks and load balancing.

## HDFS Block

## ==========

By default, its size is 128 MB.

1)So, HDFS file is chopped up into 128 mb chunks.

2)"dfs.blocksize" is parameter to change the block size.

## Replication Factor

## ===============

By default, it is 3.

1)so that your blocks get replicated three times and stored in three different machines(DN's).

2)dfs.replication is parameter to configure if changes needed.

3)It's takes care fault tolerance at node level.

# 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).