Contents

[Components of Apache Hadoop 1](#_Toc464946763)

[Understanding the Apache Hadoop daemons 1](#_Toc464946764)

[Namenode 1](#_Toc464946765)

# Components of Apache Hadoop

Apache Hadoop is composed of two core components. They are:

HDFS

MapReduce

# Understanding the Apache Hadoop daemons

Apache Hadoop 1.x (MRv1) consists of the following daemons

1. Namenode
2. Secondary Namenode
3. Jobtracker
4. Tasktracker
5. Datanode

Hadoop stores and process data in a distributed fashion. To achieve this it employs a master-slave architecture.

The namenode and Jobtracker are master daemons whereas Datanode and Tasktracker are slave or worker daemons.

## Namenode

The namenode is a master daemon and is responsible for storing all the location information of the files present in HDFS. It doesnot store the actual data but only the metadata.

This entire metadata information is stored in RAM and it also maintains a persistant checkpoint of the metadata in a file stored on the disk called **fsimage** file

The fsimage file is not updated for every write operation.

Whenever a file is placed/deleted/updated in the cluster an entry of this action is updated in a file called **edits logFile** .

After updating the edits logs the metadata present in the memory is also updated accordingly.

In case the Namenode is restarted the following sequence of events take place

1. Reads the fsimage file from the disk and load it into the memory (RAM)
2. Reads the actions that are present in the edits log and apply each action to the in-memory representation of the fsimage file.
3. Write the modified in-memory fsimage file to disk

The preceding steps ensures that the in memory representation is up to date

Namenode is the single point of failure in Hadoop 1.x. Administrators can configure the namenode to write the fsimage to the local as well as to a remote disk on the network.

Hadoop 2.x support High Availability (HA) which supports two namenodes in an active/passive configuration.

Since the fsimage file is not updated for every operation, it is possible that theeit logFile would grow to a very large size. The restart of the namenode would also be very slow since every action the edit log file has to be applied to the fsimage.

Also since namenode restart are not so frequent hence we need a way to keep the fsimage updated at regular intervals not just at system restart.

This is the job done by the Secondary Namenode

## Secondary Namenode