**ACD\_BDDOF\_Session\_1\_Assignment\_5\_Main**

**Problem Statement:**

List the Components of Hadoop 2.x and explain each component in detail.

**Components of Hadoop 2.x:**

Hadoop 2.x has the following three Major Components:

* HDFS – data storage
* YARN – Yet Another Resource Navigator(Managing Resources)
* MapReduce – processing framework

When compared with Hadoop 1.x, major key component change is YARN. And High availability concept is also introduced in 2.x to avoid SPOF.

**1)HDFS:**

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

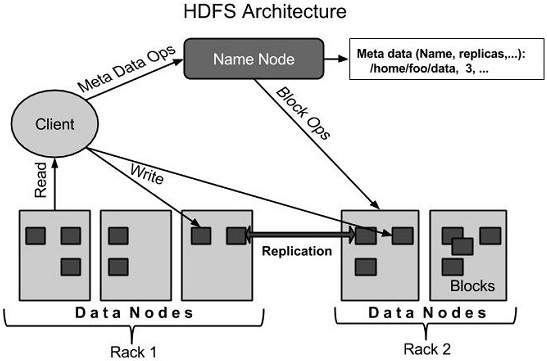
To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure.

**Features of HDFS:**

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

**HDFS Architecture**

Given below is the architecture of a Hadoop File System.



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

**Namenode:**

It is a software that can be run on commodity hardware. The system having the namenode acts as the master server and it does the following tasks:

* Manages 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.
* Contains Hadoop FileSystem Tree and other metadata information about files and directories.
* Contains in-memory mapping of which blocks are stored in which datanode.

**Secondary Namenode:**

* Performs house-keeping activities for namenode, like periodic merging of namespace and edits.
* This is not a back up for namenode.

**Datanode:**

Datanodes manage the data storage of their system.For every node (Commodity hardware/System) in a cluster, there will be a datanode. Data nodes perform read-write operations on the file systems, as per client request.

* Stores actual data blocks of file in HDFS on its own local disk.
* Sends signals to NameNode periodically (called as Heartbeat) to verify it is active.
* Sends block reporting to the nameode on cluster startup as well as periodically at every 10th Heartbeat.
* The data node are the workhorse of the system.
* They perform all the block operation including periodic checksum. They receive instructions from the
* name node of where to put the blocks and how to put the blocks.
* They also perform operations such as block creation, deletion, and replication according to the instructions of the namenode.

**Block:**

Generally the user data is stored in the files of HDFS. It is the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

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.

HDFS blocks are a collection of contiguous blocks on the hard disk

* To reduce seek times
* To read data in less rotations of the hard disk

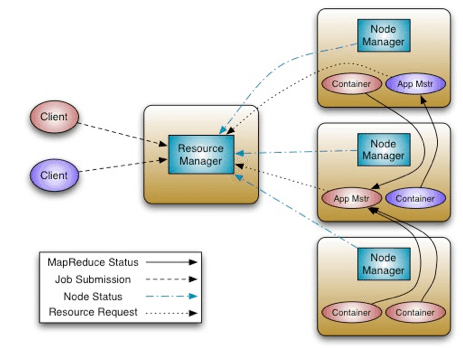
**2)YARN**

In Hadoop 1.x architecture Job tracker and task tracker are overloaded. Job Tracker was submitting an monitoring the job.

YARN is implemented to split up the two major responsibilities of the JobTracker.

1. Resource management and
2. Job scheduling/monitoring - with separate daemons:
   1. ResourceManager (global) and
   2. Application Master(per-application)

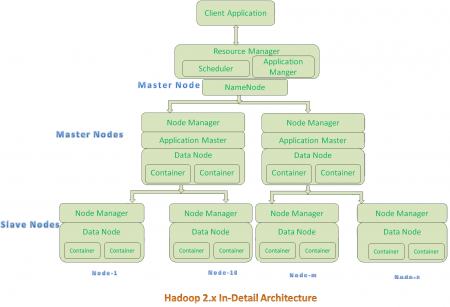
Resource manager is like job tracker and Node manager is like Task tracket of 1.x



* All Master Nodes and Slave Nodes contains both MapReduce and HDFS Components.
* One Master Node has two components:
  1. Resource Manager
  2. HDFS
* NameNode is used to store Meta Data.
* In Hadoop 2.x, some more Nodes acts as Master Nodes as shown in the above diagram. Each this 2nd level Master Node has 3 components:
  1. Node Manager
  2. Application Master
  3. Data Node
* Each this 2nd level Master Node again contains one or more Slave Nodes as shown in the above diagram.
* These Slave Nodes have two components:
  1. Node Manager
  2. HDFS

Data Node component is used to store actual our application Big Data. These nodes does not contain Application Master component.

**Hadoop 2.x Components In-detail Architecture**

[](https://cdn.journaldev.com/wp-content/uploads/2015/08/hadoop2.x-indetail-architecture.png)

**Hadoop 2.x Architecture Description**

**Resource Manager:**

* Resource Manager is a Per-Cluster Level Component.
* This will submits the job to all the data node’s
* Creates a job id for tracking
* Resource Manager is again divided into two components:
  1. Scheduler
  2. Application Manager

**Scheduler:**

* Resource Manager’s Scheduler is :
  1. Responsible to schedule required resources to Applications (that is Per-Application Master).
  2. It does only scheduling.
  3. It does care about monitoring or tracking of those Applications.

**Application Master:**

* Application Master is a per-application level component. It is responsible for:
  1. Managing assigned Application Life cycle.
  2. It interacts with both Resource Manager’s Scheduler and Node Manager
  3. It interacts with Scheduler to acquire required resources.
  4. It interacts with Node Manager to execute assigned tasks and monitor those task’s status. It pings Resource manager and asks for resources needed.
  5. Once it receives information, creates a container with all the acquired resources and submits & monitors the job.

**Node Manager:**

* Node Manager is a Per-Node Level component.
* This gets the job from Resource manager and creates application master with id.
* It is responsible for:
  1. Managing the life-cycle of the Container.
  2. Monitoring each Container’s Resources utilization.

**Container:**

* Each Master Node or Slave Node contains set of Containers
* Real job execution happens here
* Container is a portion of Memory in HDFS (Either Name Node or Data Node).
* As soon as job is completed container and Application master will be freed.
* In Hadoop 2.x, Container is similar to Data Slots in Hadoop 1.x.

**High Availability:**

In Hadoop 1.x Name node is a Single Point of Failure and there is no way to recover that without manual intervention. In 2.x it is been avoided by introducing High availability – Journal node.

**Journal node(JN):**

* This node will receive heart beat from name node at frequent intervals.
* If JN doesn’t receive heart beat from Name Node, JN will become name node using latest FSImage from Secondary name node.

**3)MapReduce:**

MapReduce is a processing technique and a program model for distributed computing based on java.

The MapReduce algorithm contains two important tasks, namely **Map and Reduce.**

* Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).
* Reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples.
* As the name implies, Reduce task is always performed after the map job.

The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes.

* During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.
* The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.
* Most of the computing takes place on nodes with data on local disks that reduces the network traffic.
* After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

