**COMPONENTS OF HADOOP 2.x:-**

There are 3 components in Hadoop 2.x. They are

1. HDFS
2. YARN
3. MapReduce

**HDFS:-**

Hadoop Distributed File System provides Storage and access to data across Hadoop Clusters. It works **based on master slave architecture.**

There are master daemon and slave daemons present in HDFS.

1. **Name Node:-**

* Its generally **one in number.**
* Master Daemon is known as Name Node.
* It will work on **high end machine.** i.e., the machine that has strong processor.
* Records Metadata such as location of blocks stored, the size of the files, permission etc.,
* Maintains and manages Data Node.

1. **Data Node:-**

* Its **many in number.**
* Its runs on **commodity machine.** Commodity machine is a machine that is used by normai users and it can crash at any time.
* Data Node **stores** actual data.
* Data Node serves read and write requests from clients, because in Data Node, data is stored.

1. **Secondary Name Node:-**

* It’s a **backup Master Daemon.**
* If Name node fails, it is used as backup and **it will be replaced by Hadoop Administrator.**
* It runs on **High End Machine**.
* **Connects** to Name Node **every hour.**
* Saved Meta Data can rebuild a failed node.

**BLOCKS IN HDFS:-**

* It’s the physical division of datafile done by HDFS while storing the data.
* In Hadoop 2.x, **default block size is 128 MB.**

**BLOCK REPLICATION IN HDFS:-**

* Block Replication provides **redundancy and fault tolerance** to the data saved.
* The default value is **3** but can be changed to any number that we want.
* Out of 3 replica’s, 2 replica’s gets stored on one rack and another one gets stored on another rack.

**MapReduce:-**

MapReduce is a java programming model. Map Reduce runs based on Mappers and Reducers.

1. **Mappers:-**

* Mapper segregates the data.
* It’s a java programming unit which analyses one block of data of the Big Data File.
* For example, Consider 1 GB (2^30 B) File.

**No of mappers for 1GB file = Size of the file**

**-------------------------------**

**Size of the Block**

**=**2^30 B

---------------------------- 2^27 B

=8 Mappers.

* + So, no of Mappers required for the file is equal to No of Blocks required for storing the file.

1. **Reducers:-**
   * It’s a **java programming** unit which **aggregates** the output coming out of **Mappers** , combines it and presents the final output.
   * No of Reducers required can be set by **setNumReduceTasks().**
2. **Shuffle and Sort:**

* Shuffle and Sort is the **automatic functionality in MapReduce** provided by framework that takes the output of Mappers and will **assimilate** it and pass it to reducer to aggregate the data.

In Hadoop 1.x. in MapReduce, the master daemon is **JOB TRACKER** and slave daemon is **TASK TRACKER.** While processing the data, it was realized that processing daemon **job tracker is overloaded.** It was doing too many things such as Job Allocation, Job Scheduling, Monitoring, Resource Allocation etc.,

Also the processing part of Hadoop 1.x, supported only one processing framework called as MR1.

So, Hadoop 2.x came up with something called YARN (or MR2).

**YARN:-**

YARN is a latest processing management framework of Hadoop. It was created by dividing the processing engine of Hadoop into smaller, more manageable parts. It also monitors and manages jobs submitted by user in highly performance efficient manner.

There are 6 components in YARN. They are

* Resource Manager
* Node Manager
* Container
* Application Master
* Scheduler
* Application Manager

1. **Resource Manager:-**

* Resource Manager is the **master of processing part.**
* It runs on **high end machine.**
* Maintains **metadata about Node Manager.**
* Its **one in number.**

1. **Node Manager:-**

* Node Manager is the **slave daemon in processing part.**
* When it starts, it **announces itself to Resource Manager by means of HeartBeat Mechanism.**
* It contains **container and Application Master.**
* It runs on **commodity machine.**
* Its **many in number.**

1. **Container:-**

* Container is an abstraction and runtime on commodity machines that help run a programming unit.
* **Slave machine provides container** Abstraction.
* No of Container available on each slave machine is based on core of the machine. For example, If one slave machine in cluster is **octacore** , then **eight container will be available in that slave machine.** In these eight containers, we can run 8 java programs.

1. **Application Master:-**

* Application Master is **one per job.**
* It is a **java program** which is **provided by Framework.**
* It does all the responsibilities of **JOB TRACKER used to perform in Hadoop 1.x.**
* Since it is Java Program, it needs **Container to run**.
* It has java class name of **MrAppMaster.**
* It has temporary life time ((i.e)., It starts when job starts and ends when job ends.)

1. **Scheduler:-**

* Its a daemon and is **part of Master.**
* Scheduler keeps the **reference of all the containers of commodity machines in that particular cluster.**
* If the jobs needs container, then it will request scheduler to give the container reference to run java programs.

1. **Application Manager:-**

* Application Manager is the **master of all application Masters.**
* If any job is submitted, then that job will initialized one Application Master. The **moment that, Application Master is initialized, it will register itself with Application Manager.**