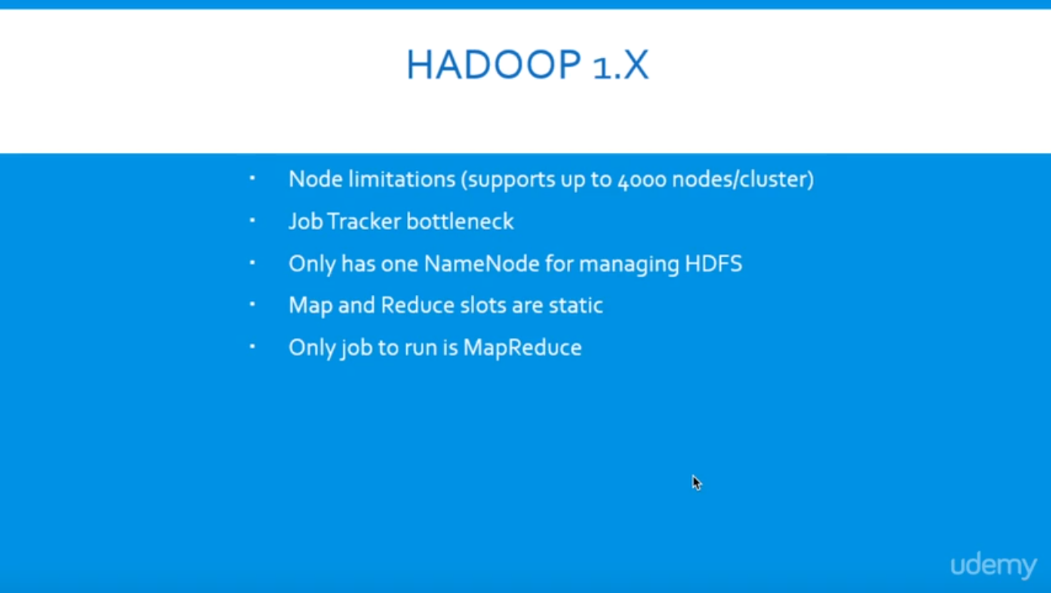
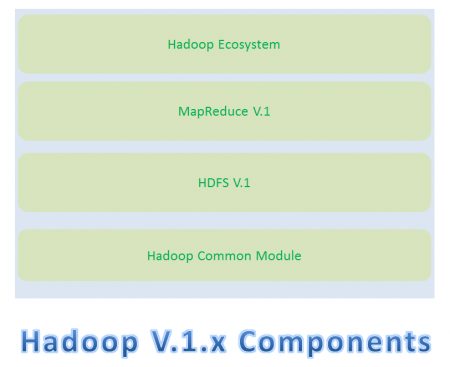
**HADOOP 1.X**

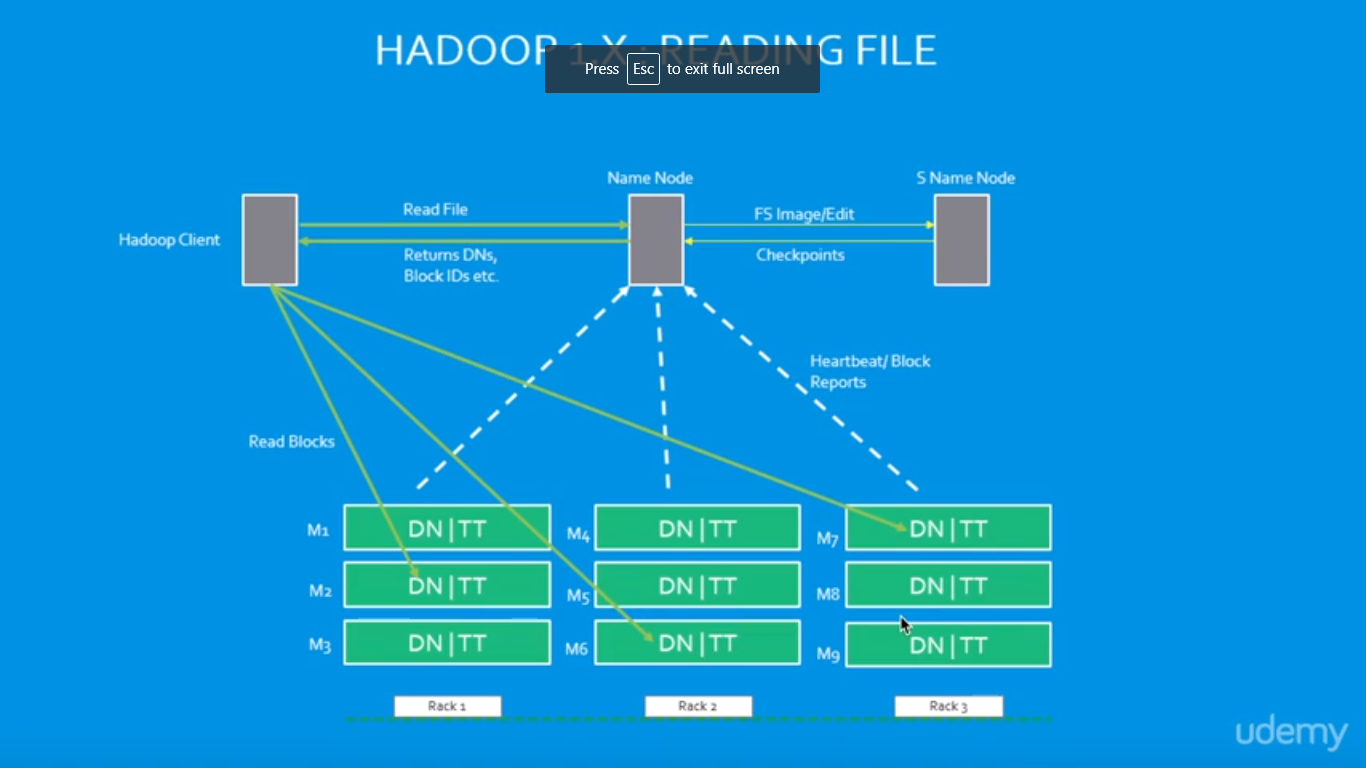


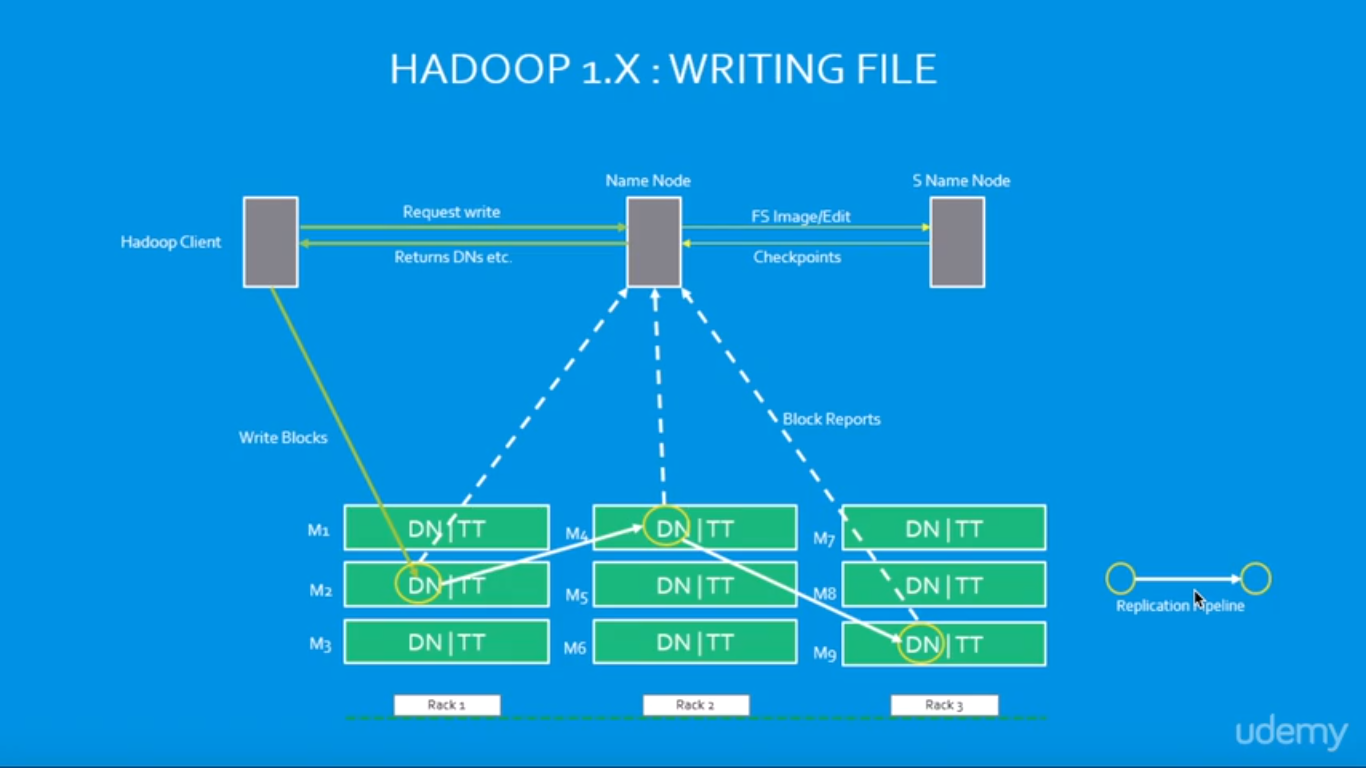
### **Hadoop 1.x Architecture**

Apache Hadoop 1.x or earlier versions are using the following Hadoop Architecture. It is a Hadoop 1.x High-level Architecture. We will discuss in-detailed Low-level Architecture in coming sections.

[](https://cdn.journaldev.com/wp-content/uploads/2015/08/hadoop1.x-components.png)If you don’t understand this Architecture at this stage, no need to worry. Read next sections in this post and also coming posts to understand it very well.

* Hadoop Common Module is a Hadoop Base API (A Jar file) for all Hadoop Components. All other components works on top of this module.
* HDFS stands for Hadoop Distributed File System. It is also know as HDFS V1 as it is part of Hadoop 1.x. It is used as a Distributed Storage System in Hadoop Architecture.
* MapReduce is a Batch Processing or Distributed Data Processing Module. It is built by following Google’s MapReduce Algorithm. It is also know as “MR V1” or “Classic MapReduce” as it is part of Hadoop 1.x.
* Remaining all Hadoop Ecosystem components work on top of these two major components: HDFS and MapReduce. We will discuss all Hadoop Ecosystem components in-detail in my coming posts.





### Hadoop 1.x Major Components

* Hadoop 1.x Major Components components are: HDFS and MapReduce. They are also know as “Two Pillars” of Hadoop 1.x.

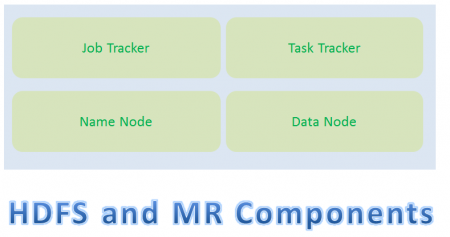
**HDFS:**  
HDFS is a Hadoop Distributed FileSystem, where our BigData is stored using Commodity Hardware. It is designed to work with Large DataSets with default block size is 64MB (We can change it as per our Project requirements).

* HDFS component is again divided into two sub-components:
* Name Node
* Name Node is placed in Master Node. It used to store Meta Data about Data Nodes like “How many blocks are stored in Data Nodes, Which Data Nodes have data, Slave Node Details, Data Nodes locations, timestamps etc” .
* Data Node
* Data Nodes are places in Slave Nodes. It is used to store our Application Actual Data. It stores data in Data Slots of size 64MB by default.

**MapReduce:**  
MapReduce is a Distributed Data Processing or Batch Processing Programming Model. Like HDFS, MapReduce component also uses

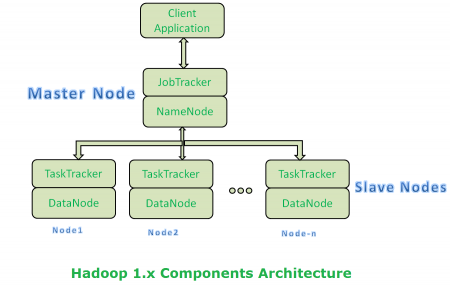
Commodity Hardware to process “High Volume of Variety of Data at High Velocity Rate” in a reliable and fault-tolerant manner.

* MapReduce component is again divided into two sub-components:
* Job Tracker
* Job Tracker is used to assign MapReduce Tasks to Task Trackers in the Cluster of Nodes. Sometimes, it reassigns same tasks to other Task Trackers as previous Task Trackers are failed or shutdown scenarios.
* Job Tracker maintains all the Task Trackers status like Up/running, Failed, Recovered etc.
* Task Tracker
* Task Tracker executes the Tasks which are assigned by Job Tracker and sends the status of those tasks to Job Tracker.

[](https://cdn.journaldev.com/wp-content/uploads/2015/08/hadoop1.x-hdfs-mr-components.png)

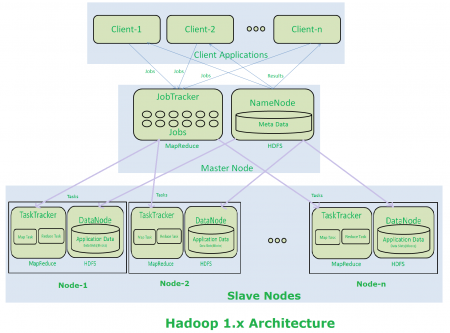
* We will discuss these four sub-component’s responsibilities and how they interact each other to perform a “Client Application Tasks” in detail in next section.
* How Hadoop 1.x Major Components Works
* Hadoop 1.x components follow this architecture to interact each other and to work parallel in a reliable and fault-tolerant manner.

**Hadoop 1.x Components High-Level Architecture**

[](https://cdn.journaldev.com/wp-content/uploads/2015/08/hadoop1.x-components-architecture.png)

* Both Master Node and Slave Nodes contain two Hadoop Components:
* HDFS Component
* MapReduce Component
* Master Node’s HDFS component is also known as “Name Node”.
* Slave Node’s HDFS component is also known as “Data Node”.
* Master Node’s “Name Node” component is used to store Meta Data.
* Slave Node’s “Data Node” component is used to store actual our application Big Data.
* HDFS stores data by using 64MB size of “Data Slots” or “Data Blocks”.
* Master Node’s MapReduce component is also known as “Job Tracker”.
* Slave Node’s MapReduce component is also known as “Task Tracker”.
* Master Node’s “Job Tracker” will take care assigning tasks to “Task Tracker” and receiving results from them.
* Slave Node’s MapReduce component “Task Tracker” contains two MapReduce Tasks:
* Map Task
* Reduce Task
* We will discuss in-detail about MapReduce tasks (Mapper and Reducer) in my coming post with some simple End-to-End Examples.
* Slave Node’s “Task Tracker” actually performs Client’s tasks by using MapReduce Batch Processing model.
* Master Node is a Primary Node to take care of all remaining Slave Nodes (Secondary Nodes)

**Hadoop 1.x Components In-detail Architecture**

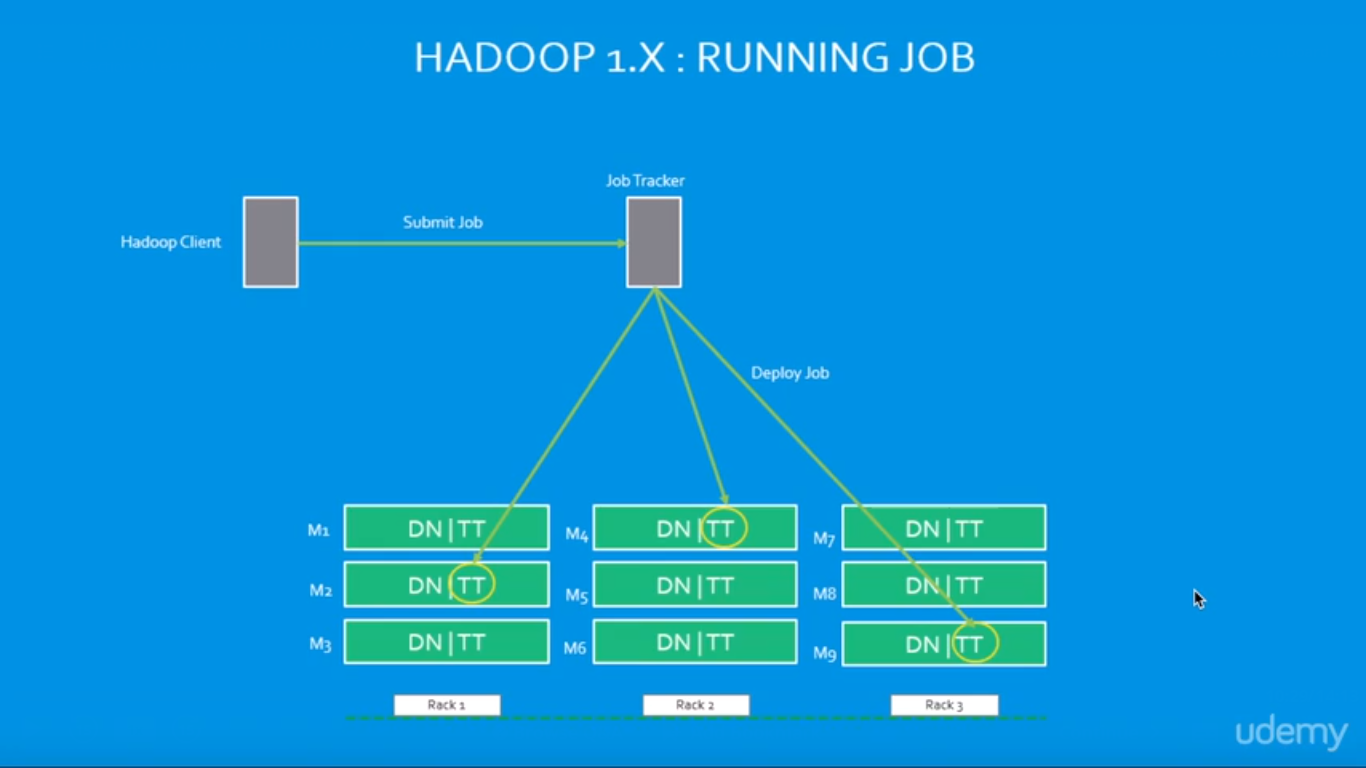
[](https://cdn.journaldev.com/wp-content/uploads/2015/08/hadoop2.x-components-architecture.png)

**Hadoop 1.x Architecture Description**

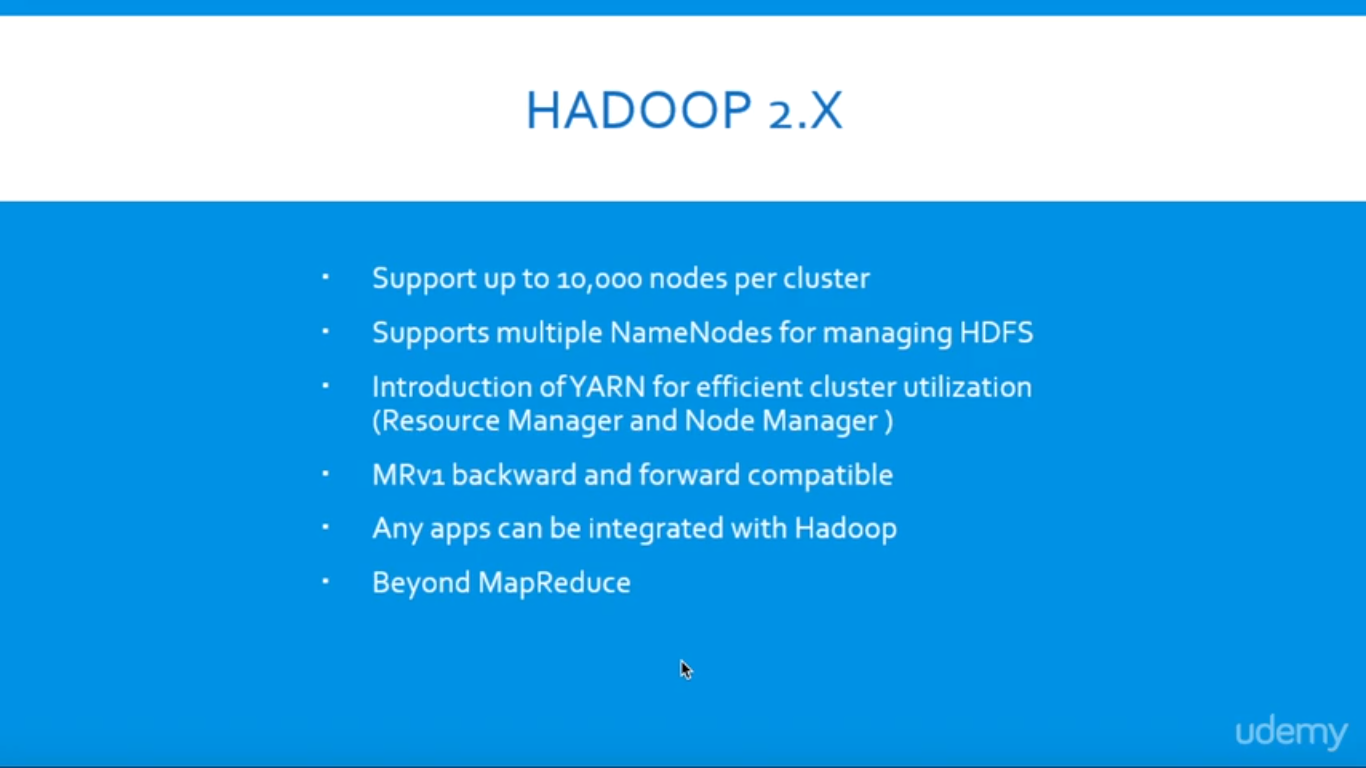
* Clients (one or more) submit their work to Hadoop System.
* When Hadoop System receives a Client Request, first it is received by a Master Node.
* Master Node’s MapReduce component “Job Tracker” is responsible for receiving Client Work and divides into manageable independent Tasks and assign them to Task Trackers.
* Slave Node’s MapReduce component “Task Tracker” receives those Tasks from “Job Tracker” and perform those tasks by using MapReduce components.
* Once all Task Trackers finished their job, Job Tracker takes those results and combines them into final result.
* Finally Hadoop System will send that final result to the Client.
* How Store and Compute Operations Work in Hadoop
* All these Master Node and Slave Nodes are organized into a Network of clusters. Each Cluster is again divided into Racks. Each rack contains a set of Nodes (Commodity Computer).
* When Hadoop system receives “Store” operation like storing Large DataSets into HDFS, it stores that data into 3 different Nodes (As we configure Replication Factor = 3 by default). This complete data is not stored in one single node. Large Data File is divided into manageable and meaningful Blocks and distributed into different nodes with 3 copies.
* If Hadoop system receives any “Compute” operation, it will talk to near-by nodes to retrieve those blocks of Data. While Reading Data or Computing if one or more nodes get failed, then it will automatically pick-up performing those tasks by approaching any near-by and available node.
* That’s why Hadoop system provides highly available and fault tolerant BigData Solutions.

**NOTE:-**

* Hadoop 1.x Architecture has lot of limitations and drawbacks. So that Hadoop Community has evaluated and redesigned this Architecture into Hadoop 2.x Architecture.
* Hadoop 2.x Architecture is completely different and resolved all Hadoop 1.x Architecture’s limitations and drawbacks.

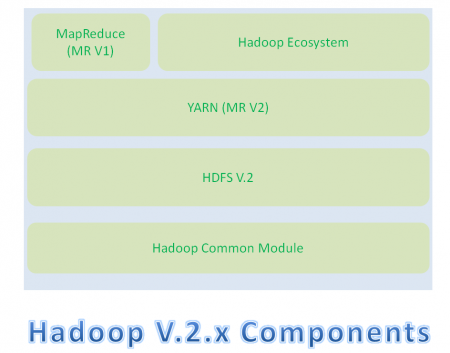


**HADOOP 2.X**



**Hadoop 2.x Architecture**

* Apache Hadoop 2.x or later versions are using the following Hadoop Architecture. It is a Hadoop 2.x High-level Architecture. We will discuss in-detailed Low-level Architecture in coming sections.

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

* Hadoop Common Module is a Hadoop Base API (A Jar file) for all Hadoop Components. All other components works on top of this module.
* HDFS stands for Hadoop Distributed File System. It is also know as HDFS V2 as it is part of Hadoop 2.x with some enhanced features. It is used as a Distributed Storage System in Hadoop Architecture.
* YARN stands for Yet Another Resource Negotiator. It is new Component in Hadoop 2.x Architecture. It is also know as “MR V2”.
* MapReduce is a Batch Processing or Distributed Data Processing Module. It is also know as “MR V1” as it is part of Hadoop 1.x with some updated features.
* Remaining all Hadoop Ecosystem components work on top of these three major components: HDFS, YARN and MapReduce. We will discuss all Hadoop Ecosystem components in-detail in my coming posts.

When compared to Hadoop 1.x, Hadoop 2.x Architecture is designed completely different. It has added one new component : YARN and also updated HDFS and MapReduce component’s Responsibilities.

### **Hadoop 2.x Major Components**

Hadoop 2.x has the following three Major Components:

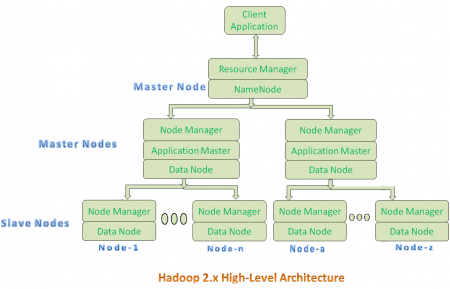
* HDFS
* YARN
* MapReduce

These three are also known as Three Pillars of Hadoop 2. Here major key component change is YARN. It is really game changing component in BigData Hadoop System.

### How Hadoop 2.x Major Components Works

Hadoop 2.x components follow this architecture to interact each other and to work parallel in a reliable, highly available and fault-tolerant manner.

**Hadoop 2.x Components High-Level Architecture**

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

* All Master Nodes and Slave Nodes contains both MapReduce and HDFS Components.
* One Master Node has two components:
  1. Resource Manager(YARN or MapReduce v2)
  2. HDFS

It’s HDFS component is also knows as NameNode. It’s 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:

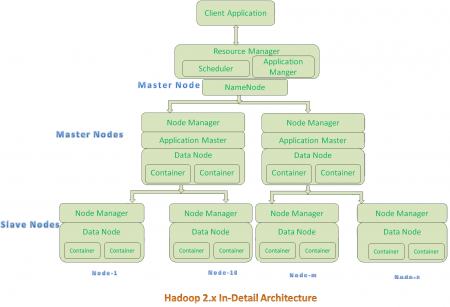
* 1. Node Manager
  2. Application Master
  3. Data Node

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

It’s HDFS component is also knows as Data Node. It’s 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.
* Resource Manager is again divided into two components:
  1. Scheduler
  2. Application Manager
* 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.

**Node Manager:**

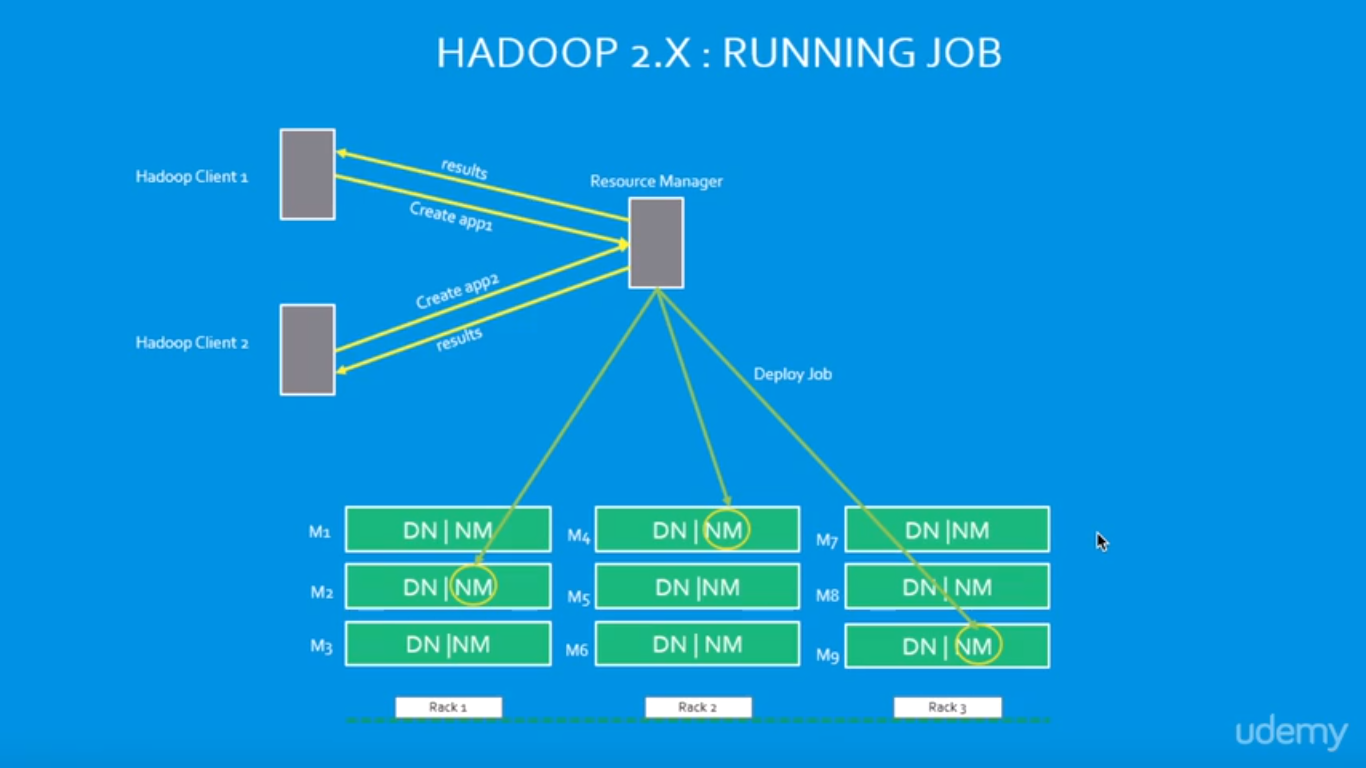
* Node Manager is a Per-Node Level component.
* It is responsible for:
  1. Managing the life-cycle of the Container.
  2. Monitoring each Container’s Resources utilization.

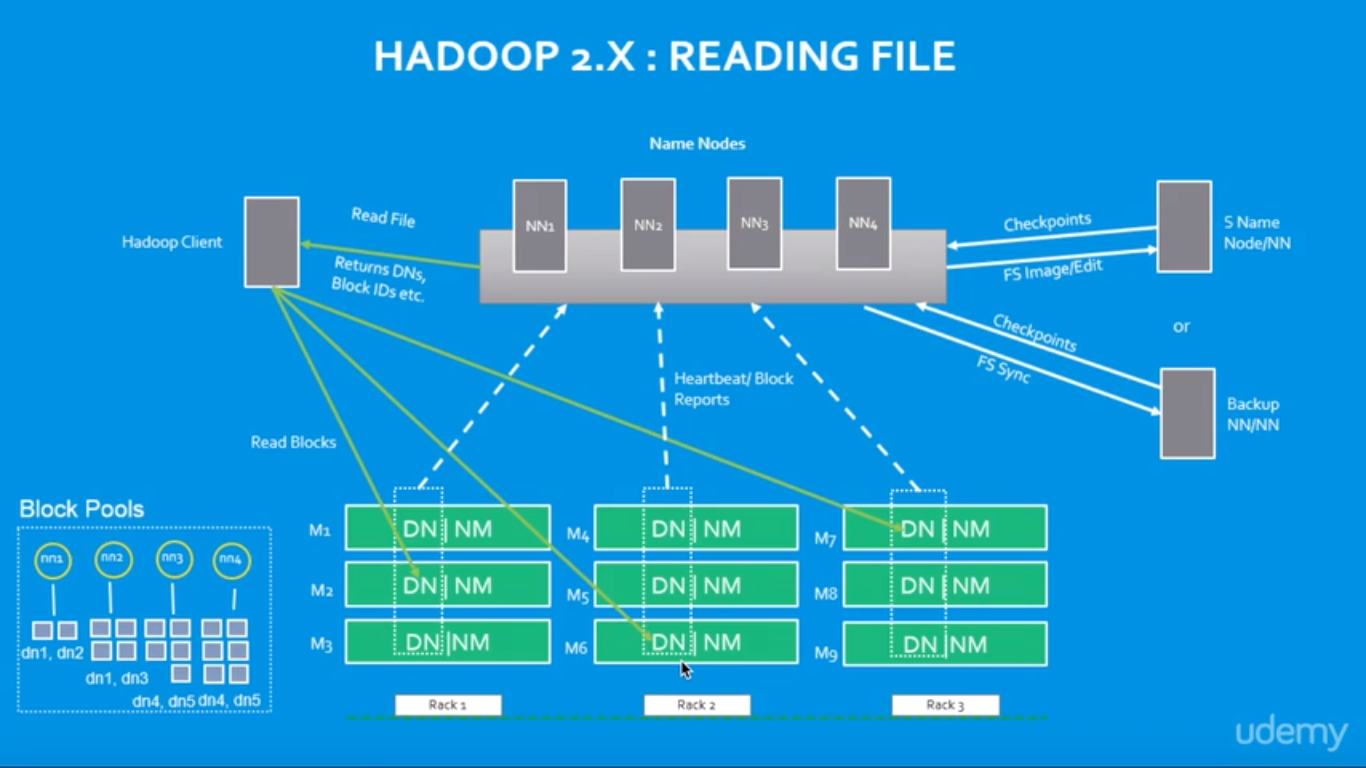
**Container:**

* Each Master Node or Slave Node contains set of Containers. In this diagram, Main Node’s Name Node is not showing the Containers. However, it also contains a set of Containers.
* Container is a portion of Memory in HDFS (Either Name Node or Data Node).
* In Hadoop 2.x, Container is similar to Data Slots in Hadoop 1.x. We will see the major differences between these two Components: Slots Vs Containers in my coming posts.

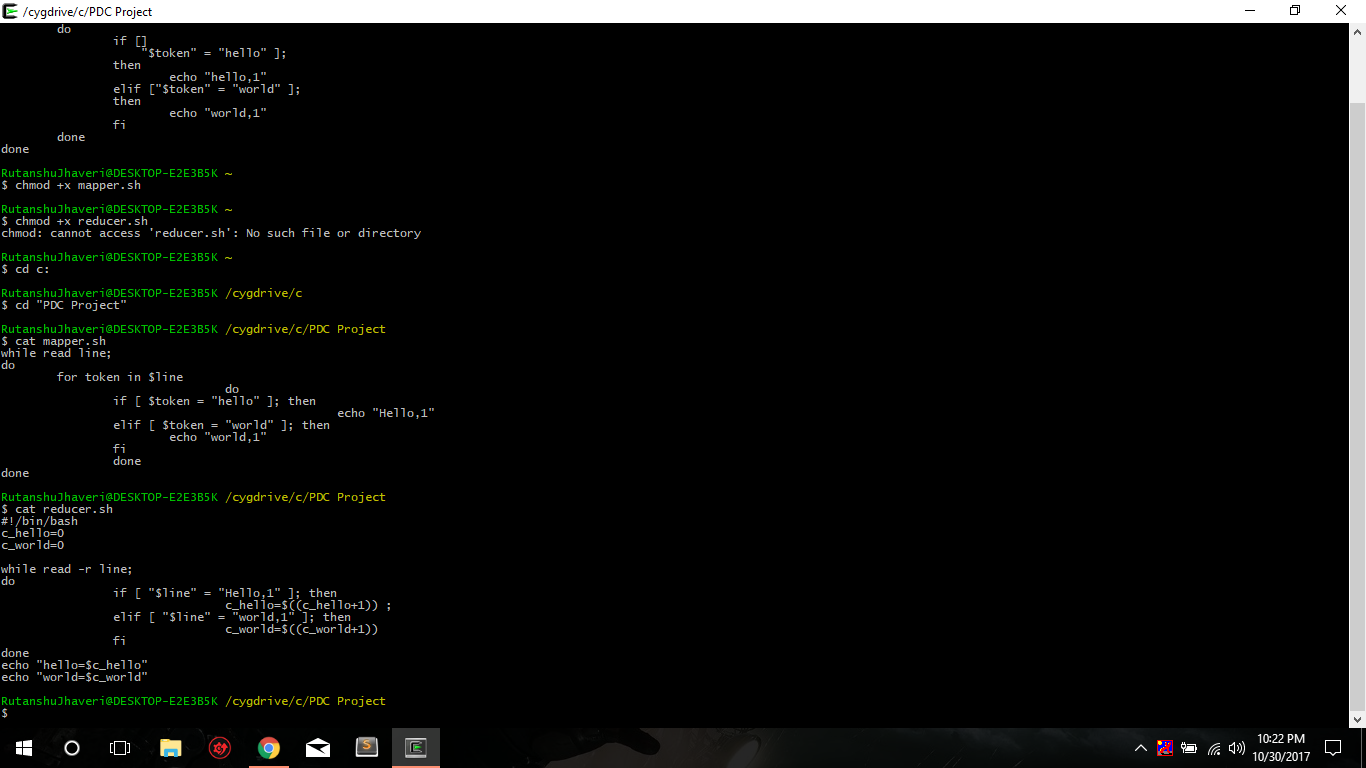
**NOTE:-**

* Resource Manager is Per-Cluster component where as Application Master is per-application component.
* Both Hadoop 1.x and Hadoop 2.x Architectures follow Master-Slave Architecture Model.

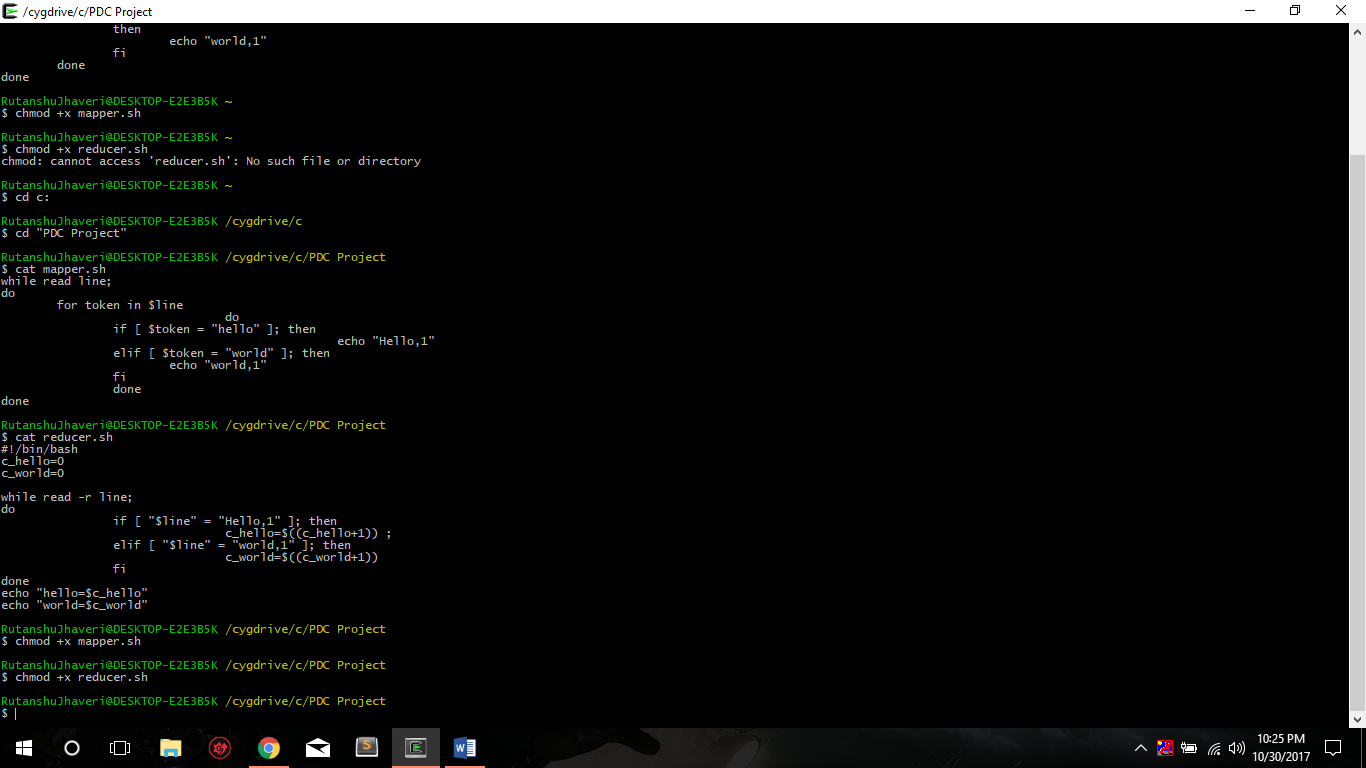




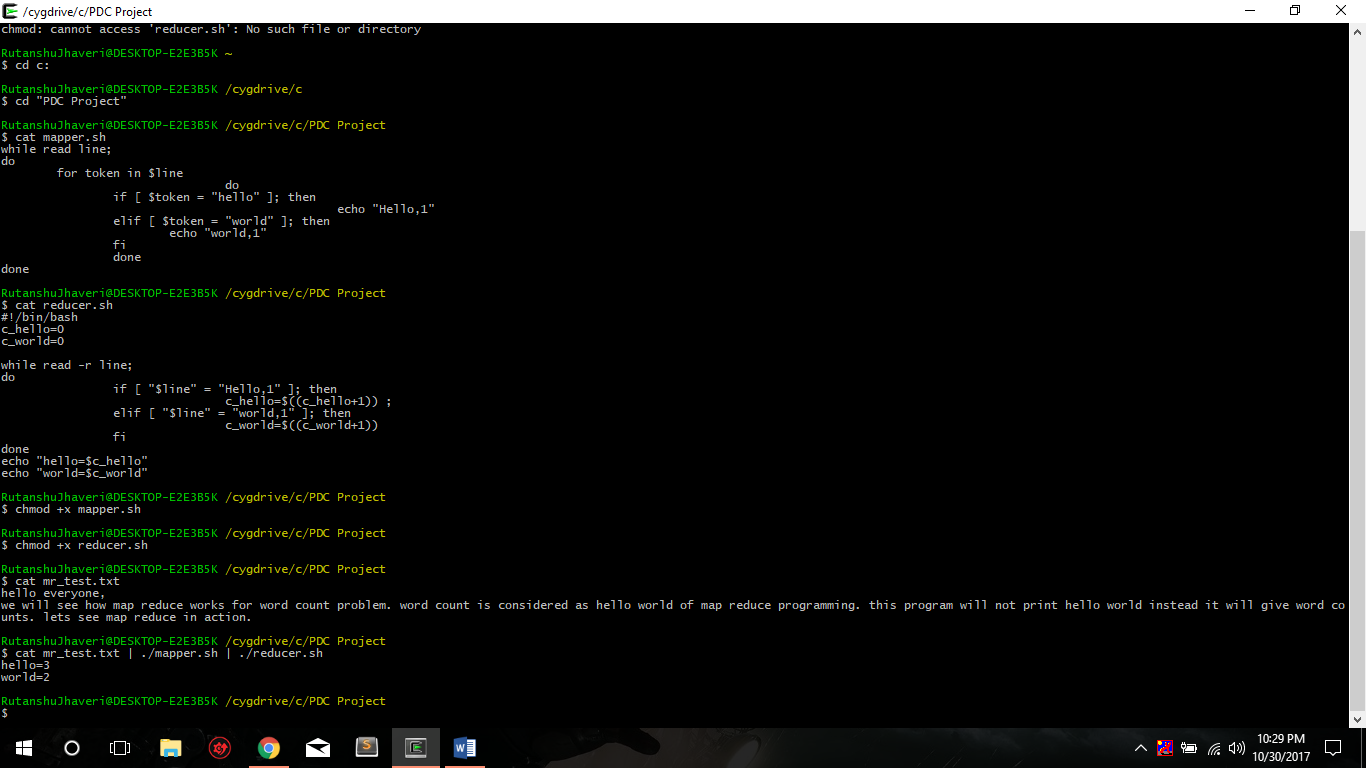
**Implementation Example:-**

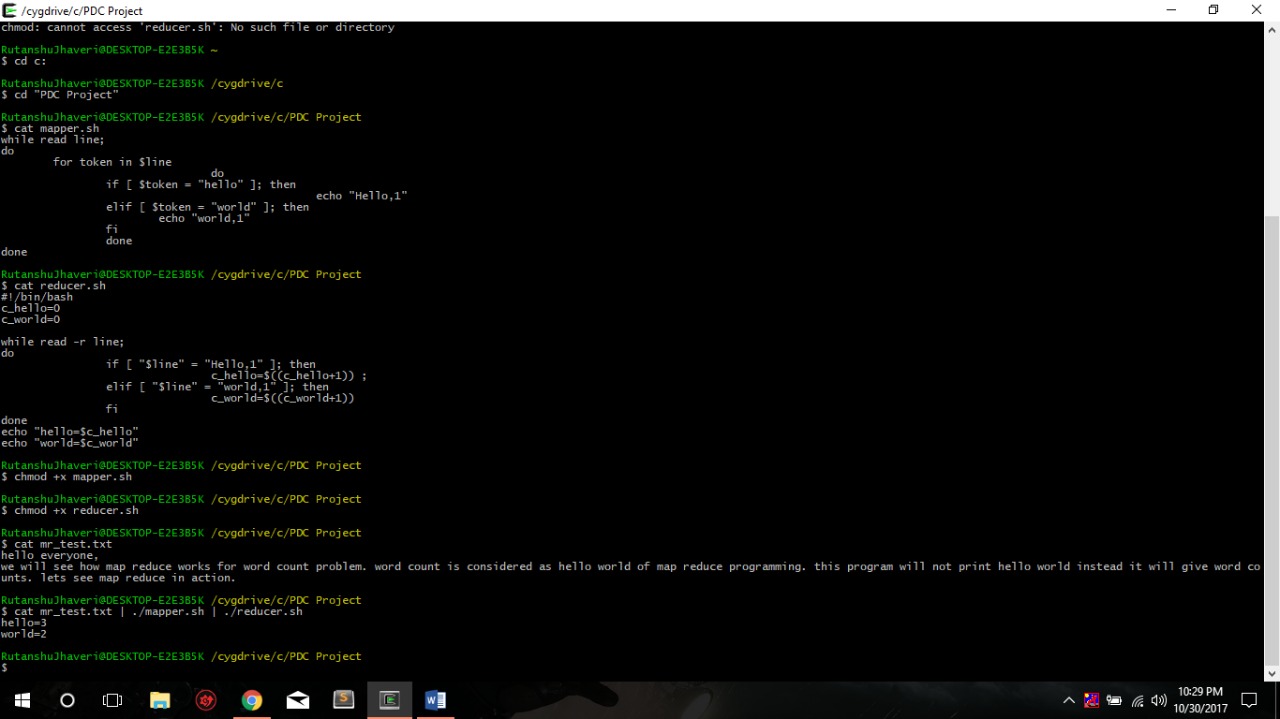


Mapper and reducer shell scripts to count the words “hello” and “world”



Giving permission to file of shell scripts





**Project is on Virtual Box**

We can show the code pig and hive files.

Therefore, there is no directory to show, The final project would be shown in video of Review 3.

https://github.com/rutanshuj/PDCProject