**Introduction to Big Data**

* **IBM gives a definition for BIG DATE that any data that has 3 V's that can be called as BIG DATA. Those 3v's are Volume, Variety, and Velocity.**
* **Usually in traditional data systems we previously deal with only structured data when it comes to Big Data it is more of Structured, Semi Structured and Unstructured Data as well.**
* **Volume: When it comes to volume the Data is very huge now a days that traditional system couldn’t be able to handle. In general, around 2.5 quintillion bytes of data is being generated every day.**
* **Variety: the data could be structured, semi structured and unstructured.**

**Structured (RDBMS Data bases (Oracle and MySQL)**

**Semi structured (CSV, XML, JSON)**

**Unstructured (Audio, Video, Image and Log Files)**

* **Velocity: The Systems should be able to tackle huge traffic that comes in to.**

**The Whole goal of BIG DATA is to process huge amount of data which traditional systems are not capable of processing which can be difficult by using traditional data systems.**

**Before processing, there is another huddle called storage, as to process huge amount of data we first need to store it! Traditional systems are not capable of storing large or massive amount of data.**

**STORE 🡺 PROCESS 🡺 SCALE**

* **Store (store massive amount of data)** 🡺 **PROCESS (process it in a timely manner)**🡺 **SCALE (Scale easily as data grows)**

**These are the three things required to design a good big data system.**

* **When it comes to scalability, there are two ways to design system 1. Monolithic and 2. Distributed.**
  + - * **1.Monoloithic** 🡺 **One powerful system with lot of resources.**
      * **2. Distributed**🡺 **Many smaller systems come together.**
* **In monolithic as it is a single powerful server, it is hard to add resources after a certain limit. Resources could be RAM, HARD DISK, CPU.**
* **Monolithic is not scalable that means if monolithic system becomes 2X resources != 2X Performance. So that’s why monolithic system cannot be scalable after certain limit.**
* **On the other hand, imagine we have 6 nodes which is a distributed system, and it is scalable so in distributed systems, 2X resources = 2X speed. That is why all the big data systems are distributed system because of this scalability after a certain point. Which is no in the case of Monolithic systems.**
* **Monolithic architecture is based on Vertical scaling (Not True Scaling), Distributed scaling is based on Horizontal scaling.**
* **So in summary here we are trying to solve the issues we face in case of Big data, as initially we need to deal with store after that processing and then scalability (for this we use distributed systems).**

**Hadoop is a Big data Framework to solve problems related to Hadoop.**

**Hadoop 1.0 🡺 It consists of Map Reduce, HDFS**

**Hadoop 2.0 🡺 It consists of Map Reduce, YARN, HDFS**

**YARN:**

**It is Yet Another Resource Negotiator.**

**It helps to mainly responsible for resource management, it is just like an operating system over multiple machines (cluster).**

**That means if some node in the cluster needs resources it will connect YARN for it.**

**As Hadoop 2.0 consists of HDFS, Map Reduce, YARN**

**HDFS for Distributed Storage**

**Map Reduce for Distributed Processing**

**YARN for Resource Management.**

**There are some eco system technologies around this Hadoop such as Hive, HBase, Sqoop, OOZIE, PIG, Spark together it is called Hadoop ecosystem.**

**Code written in Map reduce is Java, Hive is like a wrapper over Map Reduce like if we don’t know the Java using Hive, we can write queries in SQL in which it can hard code those queries into Java to Map Reduce. The Query language which we used in Hive is not SQL it is HQL (Hive Query Language).**

**Sqoop: Sqoop is tool used to transfer Data/Data Ingestion from Traditional Database to Hadoop using Sqoop Commands. It is a command-line interface application for transferring data between relational databases and Hadoop. Sqoop is also a Map Reduce Job only. Only Mappers do this job in real time.**

**PIG: 1. It helps to clean data 2. It helps to convert unstructured data to Structured data. Now a days PIG is not used in Industry as these things can be done by using Apache spark.**

**HBase: It is a column Oriented NoSQL database that runs on top of HDFS.**

**OOZIE: It is a Workflow Scheduler system to manage Apache Hadoop Jobs.**

**SPARK: A Distributed general purpose in-memory compute engine.**

**So, as we seen that in Hadoop cluster there are mainly three components**

1. **HDFS is like Storage Unit, MapReduce is like compute Engine, Yarn is like Resource manager.**

**That means we can say HDFS🡺HDFS | MapReduce🡺SPARK | YARN🡺 YARN**

**From this we can say owe can’t say Hadoop is same as spark, but we can say MapReduce is same as SPARK. SPARK is much better than MapReduce.**

**So current industry uses HDFS|SPARK|YARN**

**SPARK is like plug and plaything, but it needs additionally two things Storage system and Resource Manager**

**SPARK as Compute Engine**

**Plug it with any storage system (Local Storage/HDFS/Amazon S3)**

**Plug it with any Resource Manager (YARN/MESOS/Kubernetes)**

**And the options among them can be decided by the Hadoop admin.**

**SPARK is written in SCALA, however, spark officially support Java, Scala Python and R.**

HDFS Architecture:

HDFS is like distributed system.

To understand this let us consider we have 4 node cluster, and we have file of size 500MB for example. Initially we divide the whole file into n parts for n number of nodes, those parts called Blocks so in the above example we divide the whole file into 4 blocks as we have n Nodes, and then each block will be saved into each Node separately.

Node 1 Node 2 Node 3 Node

File

* Now the problem is which block is stored in which node, to track this we have a Master Node, which consists of meta data. Meta Data gives which block is saved in which node.
* Master Node is also called Name Node and Slave nodes are also called Data Node.

request

Name Node

Data Node

Data Node

Data Node

Data Node

Meta data

Meta Data

* The Flow is like this , Client node is the user node it request the the Name node for the Particular file. As the name node knows which data is in which place name node gets the meta data and gives it to the client and tell that to search the nodes based on the meta data it gave. Then client uses meta data to check In which nodes those blocks are there and reads them to complete the file and gets it.
* Now What if any Data node gets down/fails?

To Tackle this there is a concept called Replication Factor, Default Hadoop Replication Factor is 3. Replication Factor means each block of the file will usually becomes multiple copies and those copies are shared among the Data Nodes so that if any node got fails we can retrieve that from other Data Node.

B1 B2 B3 B2 B3 B4 B3 B4 B1 B4 B1 B2

* But How Hadoop Knows when a Data node is failed?

Hadoop follows a concept called Heartbeat, Each Data node sends heart beats to Name node in every 3 seconds, if a Name Node doesn’t receive 10 consecutive heart beats it assumes that the data node is dead or running very slow.

If data node goes down the replication factor comes down to < 3, Name node will create one more copy to maintain the replication factor.

* What if Name node gets failed how to tackle that issue?

In Hadoop Version 1 the Name node was a single point of failure, at this time if name nodes get failed there will be down time and need time to make the name node re work. But in Hadoop version 2 Name node is no longer a single point of failure. In Usual case if name node gets failed means no access to metadata, No meta data means No access to cluster.

The Problem When the Name Node is failed is that we will be loosing the block mapping information. If we have the latest block mapping information(metadata) then we can make sure that there is no downtime involved.

So for this, Hadoop work as follows.

So understand how Hadoop is managing to get the latest metadata we need to understand about two files by which this meta data is made.

There are two important meta data files we have and also we have Secondary Name node here to make this possible.

1. The two important meta data files are fsimage and edit logs(edits),

fsimage: snapshot of inmemory filesystem at a given moment.

Edit logs: all the new changes or transactions that happen after the snapshot is taken will come to the edit logs file.

The Merging of fsimage + editlogs will gives you the latest fsimage.

fsimage+editlogs= latest fsimage.

But merging of fsimage + editlogs is a compute heavy process so here Main name node shouldn’t able to take the activity of merging these 2 files as name node is already busy doing other computing here we use secondary Name node for this task .

As we seen name node failure means no access to Meta data.

No meta data means no access to cluster.

Name node contains FSimage and edit logs

So from name node every 30 sec Hadoop gets fsimage and edit logs after that secondary name node takes out that fsimage and edit logs and merge them once fsimage+edit logs happens we get newly created fsimage from here.

After the merging edit logs becomes empty to get newly generated logs so that they not over written.

In general if name node is not failed then Name node will be active and secondly name node become passive. Once the main name node gets failed then secondary name node becomes active or primary name node. Once secondary name node becomes active it is the Hadoop responsibility to get new secondary name node so that it can do check point thing(merging fsimage and edit logs).

The default block size in Hadoop 2?

It is 128 MB , and can be changed which is usually done by Hadoop admins.

What will happen if we increase the block size?

We will have less number of blocks, if number of blocks are less then this makes less redundant because it takes only few data nodes  to save them.

What happens if we decrease the block size?

If we decrease the block size we will have more number of blocks which takes more data nodes, due to this huge amount of meta data will happens which is in the end need to be managed by name node and can be over burden to it.

So the block size should neither be too high or too low.

What is Rack awareness mechanism?

Rack means group of systems placed in different geographical locations.In general all the hardware systems won't be placed in the same location because if any natural calamities happen then we loose whole data at once. So that's why usually we split the whole systems or hardware and place those in different geographical locations.

As usually rack consists data nodes too when the blocks are being shared of a particular file, let us assume we have three racks loc 1 ,2,3 and let assume we are in loc 1 it is easy for us to send those blocks to the loc 1 rack as it is in same place where we are but if we send all the blocks to the same nearest location even though it requires small amount of network bandwidth and involves less input output operationns if that particular rack gets failed there is huge chance that we might loose our data.  So it is not recommended.

So it is advisable to choose multiple racks so that if one rack is gone which consists of replicas there won't be a problem.

In other way if we place replicas in different racks in different locations it requires huge bandwidth and time to retrieve this so to overcome this hadoop usual place one block of replica in one rack and other in one rack so that it is some what better compared to individual one.

This is the default approach but Hadoop admins can change this in any time.

What is block report?

Each data node sends a block report to the name node at a fixed frequency, indicating if any blocks are corrupted. If so Hadoop tries to create another copy of that block using meta data.

There are two ways to achieve high availability of name node

1. Using Shared location, as we seen usually to avoid name node failure primary name node shares fsimage and edit logs to a particular location from where the secondary name node takes this and merges them and captures newly created fsimage.