The Hadoop Distributed File System: Architecture and Internals

V. Sajwan [1], V. Yadav [2], Dr.M.Haider [3]

[1,2] Final Year Students, [3] Professor
[1,2,3] Dept. of Computer Science
MGM's College of Engineering
and Technology, Noida Sector-62, INDIA

ABSTRACT: Hadoop is a popular for storage and implementation of the large datasets. Implementation is done by MapReduce but for that we need proper management and storage of datasets. This responsibility to store large datasets is taken by HDFS.In this paper, we describe the high overview of Hadoop Distributed File System architecture. We define different server roles, components of the HDFS. We describe small single rack implementation and multi rack implementation of the HDFS.This research paper also analyzes the performance of the HDFS and reveals the issues comes in performance of HDFS.

Keywords :HDFS,name node,data node,secondary name node

1. INTRODUCTION

The assimilation of the mobile data into our routine generates unpredictable amount of data . According to the IDC the digital world contain 480 exabytes of data [1]. The MapReduce came out as a scalable way to perform intensive data computation on cluster of computers.The success of MapReduce makes Hadoop as an open source implementation.There other main key component of Hadoop which stores large amount of data sets named as Hadoop Distributed File System which is used to store all inputs and outputs of the data for implementation.

2. HDFS ARCHITECTURE

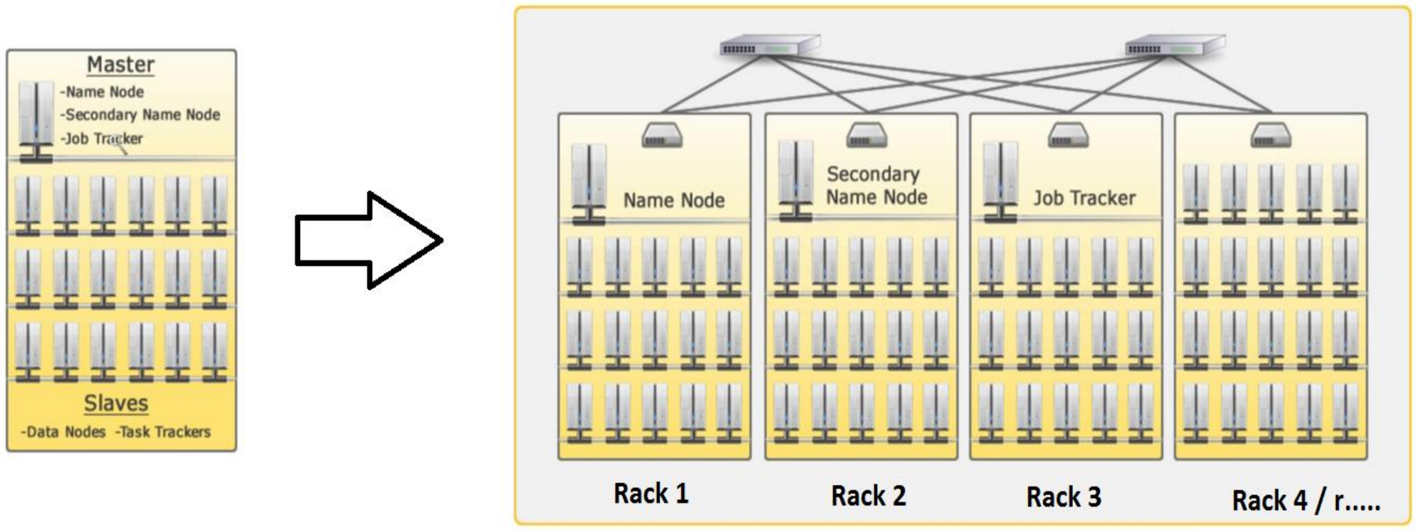

Fig 1: Right Hand Side Single Rack Cluster and Left Hand Side Multiple Rack Cluster

On the left hand side we have single node cluster and on the right hand side we have multi node cluster.In the single node cluster we have single machine which contain all the server roles .The HDFS architecture is the master slave architecture.We have name node,secondary name node,job tracker,data nodes and task tracker.Name node is the controller of data node.The name node controls the data node and the job tracker controls the task tracker.

2.1 Name Node

It is the controller that controls all the file system. So any request that comes in to the file system like to create a directory,to create a file,to read and write a file is gonna go through the name node.So name node is essentially manages the file system.It holds n memory of snapshot of the filesystem works like.It holds block wrappers. Whenever we put a file in HDFS it gonna break the file open the blocks and spread across the data nodes. The name nodes knows all the blocks are into the cluster .

2.2 Data Nodes

The data node are the workhorse of the system.They perform all the block operation.They gonna be receiving instruction from the name node of where to put the blocks and how to put the blocks.If we want data out of the cluster we give command to the cluster for data.The client actually communicate with name node . We give command to the name node for getting the blocks where data sits.The name node sends the information to the client.Than client directly communicates the name node. Where the data nodes serves those blocks directly to the client.

Data nodes are also responsible for the replication.As name node is the controller sending the information where to replicate and data node is the one that do physical replication.

2.3 Secondary Name Node

It is used to take snapshot of the everytime every now and than. It likes system back up.Its like system restore for the name node.
In Figure 1 there is also multi rack cluster setup on the right side .In this we broke out the server roles into separate server and even separate server has their own rack.

3. CHALLENGES AND ITS SOLUTIONS

3.1 Data Loss Prevention

HDFS is reliable because it uses replication.It uses software solution enter the multiple copies across the cluster rather than hardware solution.It does by ensuring there are multiple copies of data, the blocks across our cluster.If we will loose a single node it is not a big deal. Name node has a map where all the blocks and data nodes are in the cluster.So if we will loose a node the name node recognize where the blocks on that node and it re-replicate those blocks across the cluster .If we loose entire rack .

It is where Rack Awareness come into play can help us out. It does this by understanding our network apology. We need describe our network apology to the name node but ones it understands it and it is rack aware. Anytime data comes into our cluster the name node gonna ensure the data node ensure multiple copies sit on multiple racks.But if loose whole rack the name node knows what data on that node it re-replicate across the remaining racks in the cluster.

3.2 Network Performance

Bandwidth is the scarce resources. Now we gonna make a function here that in a rack communication is much higher bandwidth lower latency than cross rack communication . So that function would may be rack awareness can keep our bulky flows inside of the rack . If it get all the data from single node it will do it.It uses core switches and ensure optimal network utilization.

4. FEATURES of the HDFS

4.1 Rack awareness

We are aware about the fact the HDFS divides the data into multiple blocks and stores them on different machines .HDFS can be rack aware by the use of a script which allows the master node to map the network topology of the cluster and the default implementation with in the HDFS allows you to provide an executable script which returns the "rack address" of each of a list of IP addressees.

To set the rack mapping script specify the key *topology.script.file.name* in *conf/hadoop site.xml* .This provides a command to runto return the rack id ; it must executable script .By default,Hadoop will attempt to send a set of IP addresses to the file as several separate command line arguments.You can control the maximum acceptable number of arguments with *the topology.script.number.args* key.

4.2 Reliable storage

The file store in HDFS provides scalable,fault-tolerant storage at low cost .The HDFS software detects and compensates for hardware issues,including disk problems and server failure.

HDFS stores file across the collectionof servers in a cluster.Files are decomposed into the blocks and each block is written to more than one of the servers.The replication provides both fault-tolerance and performance .

4.3 High throughput

HDFS ensures data availability by continually monitoring the servers in a cluster and the blocks include checksums.When a block is read,the checksum is verified ,and if the block is damaged it will be restored from one of its replicas .If a server or disk fails, all of the data it stored is replicated to some other nodes or nodes in the cluster ,from the collection of replicas. As a result,HDFS runs well on commodity hardware.It tolerates and compensate for,failures in the cluster . As cluster get large ,even very expensive fault-tolerant servers are likely to fail .Because HDFS expects failure, organizations send less on servers and let software compensate for hardware issues.

5. HDFS INTERNAL

Name node is the single most important node in our cluster because it is the single point of failure. The name node everything goes through communicate_ controller. It contains the file system metadata and holds n memory of map of entire cluster.So if name node goes down our entire cluster goes down.

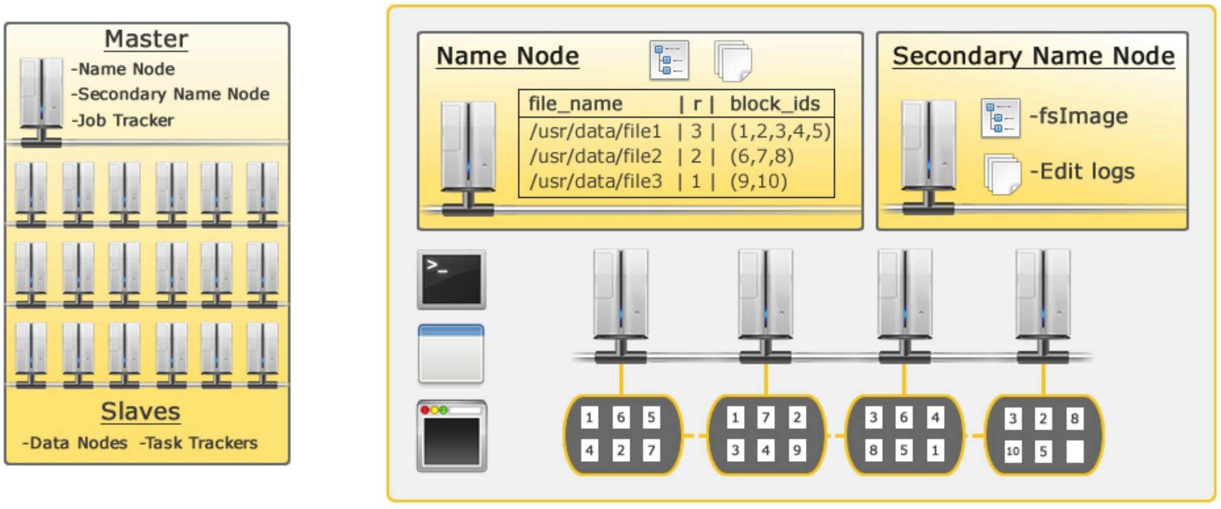

Fig 2: Internal structure of the HDFS and its components

There are no are no high availability solution out of the box for name node. There are things we can implement that will ensure if disaster does occur we can get back up running in relatively short amount of time.

In figure 2 the name node has three files (i.e. file 1,file 2,file 3).The replication factor r of file 1 is 3 which means it gonna replicate 3 times of every blocks (1,2,3,4,5) spread across the where ever data nodes they are sit on.

The name holds n memory snapshot of the file system (as shown in figure 2). It checks all the file,tracks replication value which by the way default of 3 but we can overwrite the replication factor when we put a file in HDFS.That determine how many blocks of each file out there and it also hold mapping of the block ids to the data node and it something like figure 2. The name node also has edit log or journal. They keeps track all the transaction. It doesn't put data into here but any client that requests information or change information and put something in HDFS it can perform this function with the help of edit log.

The name holds the snapshot of the filesystem in memory and all the file system goes into the edit block. If we want to use data from sdit block to memory that's gonna take reboot of the name node or check point process. At the time of Reboot name node merge the memory with edit block and forms fsImage and when name node reboots it reloads the fsimage into the memory .

If we reboot the name node very often it means the edit block keep growing and if we loose name node at time of this event we just lost all the changes to our system.So that's why it nice to have secondary name node. A secondary name node takes all the responsibility of merging the log in the file system image. Esentially, it is our house keeping blocks for the name node.It periodically head over the name node it takes more recent fsImage . It merges the two together and reloads the image into the name node.Next time the name node starts it gonna have latest file system .If catastrophic event occurs we have backup over file system image at the secondary name node .So we can reconstruct the name node with this information.

Basically,name node the most important block in the cluster it contains the snapshot of the entire file system (where the files are,where the blocks associate with the file and what data they are sit on and how many blocks they should be base on the replication factor).The client communicate the name node and name node instruct those clients where they can find stuff on the data nodes.

Secondary node is basically a housekeeping .It is the backup of the name node metadata and something disaster occurs it is secondary name node which is used to reconstruct the name node.

Name node doesn't communicate with the secondary name node and data node.The data node and secondary name node communicates with the name node.Data nodes seconds sends heartbeat every 3 seconds to the name node .This is how name node knows that data node is on line.If data doesn't send heartbeat by 10 minutes the name node gonna considered a dead node that when its gonna take all the blocks that were on it and rereplicate them across the cluster.

HDFS by default stores the our data in 64 MB chunks of disk.

In Block placement uses rack awareness we know that name node is smart enough to ensure that reliable across the rack in multi cluster environment because it replicates that file and blocks of that file across rack.So if we loose rack there is no problem.In a rack communication it will calculate the distance network bandwidth between our data node and it will place the blocks close to each other possible.There is where network utilization come into play.It utilizes the network which enale high performance because we always have short distance possible to a file without sacrificing the reliability.

Two main note in the HDFS is :-

(a) Rebalancing :- As the times goes on the disk start to fill up and it need to add more node into the cluster. We add more nodes into the cluster but how do we rebalance the data that alsmost full full across the rest of our cluster without new nodes add in.We have Rebalancing tool,really easy to setup and configure in use and ones we run it makes some time to paying size our cluster .It rebalance and evenly distribute data across the cluster.

(b) Replication Management :- Every 10th heartbeat to the name node is the block report . With that block report the name node configure out if we were under replicated or over replicated . If we were over replicated it gonna mark block to remove and if it is under replicated it gonna create priority queue and things with least amount of blocks in the cluster it gonna be higher in that list and files with one block is on the higher priority as compare to file having two or three blocks.

Every name node has the block scanner on it so it can check out the integrity of all over the blocks and it directly imacts on the block report. So,if we have a corrupted block report in the block report the name node recognize that it gonna remove it and re-replicate the good block to that node .

6. CONCLUSION

We have learnt several this from this work.First,we have learnt is what HDFS is all about.Second,we have learnt about the architecture of HDFS.Third,we have learnt about the components of the HDFS.Fourth,we go through different challenges in HDFS and its solutions.Then, we go through different features of HDFS.In other words the reason why we should use HDFS over relational database machine.Then,we explained the internals of the HDFS and also explains how data stores in the HDFS,working of the HDFS components digramatically.

7. ACKNOWLEDGMENTS

The author are grateful to the participants who contributed to research and our guide Dr.M.Haider.

8. REFERENCES

[1] Shipa, Manjit kaur, “*Big Data and Methodology”,* 10 Oct, 2013

[2] Pareedpa, A.; Dr.Antony Selvadoss, “*Significant Trends of Big Data*”, 8 Aug, 2013

[3] Gurpeet Singh Bedi, Ashima, “*Big Data Analysis with Dataset Scaling in Yet another Resource Negotiator (YARN)”,* 5April, 2013

[4] *Hadoop-The Definitive Guide*, Tom White, Edition-3, 27Jan, 2012

[5] Mrigank Mridul, Akashdeep Khajuria,Snehasish Dutta,Kumar N, “*Analysis of Big data using Apache Hadoop and MapReduce”*,Volume 4, May 2014

[6] IBM 2012, What is big data: Bring big data to the enterprise,http://www.01.ibm.com/software/data/bigdata/, IBM

[7] Sam Madden, *“From Databases to Big Data”,*IEEE computer society,2012

[8] *“Data Mining with BigData”* ,Xindong Wu, Xingquan Zhu, Gong-Qing Wu, Wei Ding , 1041-4347/13/$31.00 ©
2013 IEEE

[9] Russom, *“Big Data Analytics”* , TDWI Research,2011

[10] An Oracle White Paper, *“Hadoop and NoSQL Technologies and the Oracle DataBase”,*February 2012

[11]Apache Hadoop. http://hadoop.apache.org [last accessed 10th February 2015]

[12]Cloudera. http://cloudera .com/blog/2009/02//the-small-files-problem.[Last accessed: 12th April 2015]