International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 04, April-2018

Daemons of Hadoop: An Overview

Richa Vasuja[1]
Assistant Professor
Computer Science & Engineering
Department, Chandigarh University,
Punjab

Ayesha Bhandralia[2]
Assistant Professor
Computer Science & Engineering
Department, Chandigarh University,
Punjab

Kanika Chuchra[3]
Assistant Professor
Computer Science & Engineering
Department, Chandigarh University,
Punjab

***Abstract* - The term"Hadoop" is basically associated with new topic which is very much in trend i.e Big data. Big Data, as the name indicates itself that the data is of huge amount. So it depicts the trendy technologies to store and to manage large datasets which can be Structured, Un-Structured or Semi-Structured. Traditional technologies or databases were not able to handle this new type of data. Due to this inability of traditional databases, there was a need for some new technique i.e Hadoop. Hadoop is the foremost platform for handling the large volume of data that people are using now a days. Today the size of data ishugeholding petabytes, terabytes or somewhat larger than this size. Hadoop is based on work done by Google in the late 1990s/early 2000s. This paper will discuss about the daemons of key components of Hadoop: HDFS & MapReduce and how both these components are used to store the Big Data and also to process the Big Data respectively.**

Keywords: HDFS, MapReduce, NameNode, DataNode, SecondaryNameNode, JobTracker, TaskTracker.

I. INTRODUCTION

Hadoop, the most common term now a days, is an infrastructure software which is used for managing Big Data which is really a trendy topic now a days.[1]. The term ''big data'' itself indicates that the data which is big and therefore, it is also used to describe data' volume, data' variety, and data' velocity[2]. Big data is reviewed as a trendy technology [3] and also it is beneficial for various organisations which has huge data. Today we are having large amount of data which we first need to store and then process it for its retrieval. For this, there is a framework that used to do this thing and it is known as Hadoop[4]. Doug Cutting, Mike Cafarella at el took the solution which was given by Google and in the year 2005, he started working on an Open Source Project named HADOOP. The name Hadoop was given after the elephant toy of Doug's son. Now, Apache Software Foundation has Apache Hadoop as its registered trademark. Basically, Hadoop is based on work which was done by Google in the late 1990s/early 2000s. Specifically, on papers describing the Google File System (GFS) published in Oct 2003, and MapReduce published in 2004. This work takes an extensive new approach to the foremost problem of distributed computing. So it meets all the requirements we have for reliability and scalability. The key concept behind this is to distribute the data as it is originally stored in the system. Individual nodes can work on data local to those nodes. For processing at the beginning no data transfer over the network is required. Then development started on the Apache Nutch project, but was moved to the new Hadoop subproject in January 2006. This Hadoop was developed by Doug Cutting in 2006. He was working at Yahoo! at that time. He named it after his son's yellow toy elephant. It is an open source project under Apache sponsored by the Apache Software Foundation. It is based on Java Programming. Two fundamental functionalities of Hadoop are: "How to store this large volume of data?"and "How to process this data?". For this there are two most significant components in Hadoop i.e.:

1. HDFS: used to store Big Data
2. MapReduce: used to process Big Data

HDFS is an acronym for Hadoop Distributed File System. It is the foremost component of Hadoop Architecture. Hadoop is perfect for handling large amount of data and as its main storage systemit uses HDFS. It lets you connect nodes contained within clusters over which data files are distributed. Anyone can access and store the data files as one seamless file system. The accessing of data is done in a streaming manner i.e applications are executed directly using MapReduce. HDFS is an Apache Foundation Project and also a subproject of Apache Hadoop Project. It is a Distributed File System that is intended to run on Commodity Hardware[5].

MapReduce is made up of two separate words i.e Map and Reduce[6].Mapphase takes set of data and converts it into other set of data where individual elements are broken into tuples i.e key and value. Reducephase takes the output from MAP function as input and integrate those tuples into smaller set of tuples. MapReduce is the processing component of Apache Hadoop.It may also be referred as component that is used to process large amount of data. MapReduce processes data parallelly[7]. The MapReduce framework consists of single Job Tracker and Multiple TaskTrackers (one per DataNode). It is heart of Hadoop System as it processes and retrieves the stored data from HDFS. By using MapReduce, we can process the Big Data that is present on Hadoop or that is stored on HDFS. These days data is not stored in traditional way. Data gets divided into chunks which is stored on the miscellaneous data nodes. There is no complete data that is stored on a single place so a new framework was required that would have the capability to process the huge amount of data that is stored on different DataNodes. Also we need a framework that goes to the place where the data actually resides, processes the data there only and return the results.

II HADOOP DAEMONS OVERVIEW

HDFS is responsible for storing huge volume of data on the cluster in Hadoop and MapReduce is responsible for processing this data. To better understand how HDFS and MapReduce achieves all this, lets first understand the Daemons of both. Basically, daemons in computing term is a process that runs in the background. So Hadoop framework is having five such daemons that runs in the background to store this huge amount of data and process it. Out of these five, three daemons are of HDFS and two daemons are of MapReduce. Each daemon runs separately in its own JVM.

Daemons are:

1. NameNode
2. DataNode
3. SecondaryNameNode
4. JobTracker
5. TaskTracker

HDFS is the building block of Hadoop.Its architecture is relies on Master-Slave Model. A cluster is having thousands of nodes which are interconnected with each other. There is one Master node within the cluster that is also referred to as Head of the cluster. All other nodes are the Slave nodes or also known to as the Worker nodes.

It is having a single Active NameNode and multiple DataNode. At any given

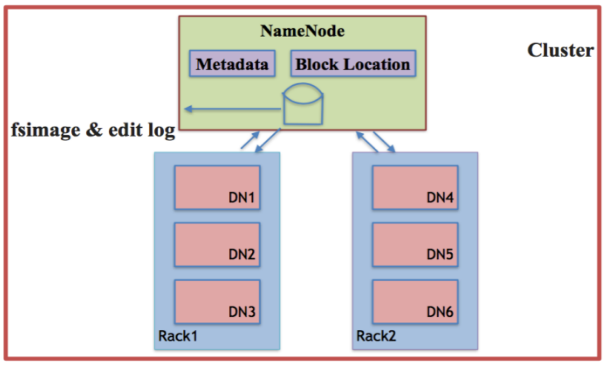

Figure 1: HDFS Daemons

point of time, there will be only one active NameNode. NameNode and DataNode are basically the server machines. Together they form a cluster of machines as shown in the Figure 1.

III DAEMONS DESCRIPTION

1. NameNode: Master Node
NameNode is the master node of the cluster. It manages and maintains the file system namespace i.e it stores the Metadata of the files as well as the directories of the filesystem. Also it stores the exact location of all the blocks where the actual data is placed. As shown inFigure 1, NameNode includes Metadata, Block Location and fsimage & Edit Log.

A.Metadata

File name	abc.txt
File Size	300MB, 150 MB
No. of Blocks	3
Block ID	[1, 2, 3]
User	Hduser
Group	Hadoop
Permission	rw
Replication	3
Block size	128MB, 64MB

Table 1: Metadata

Metadata in general means data about data. This metadata is stored on both the main memory and the hard disks. The information that it contains is in the form of table as shown in Table 1. MetaData is stored on the main memory to serve the client request.If the client wants to access the file named abc.txt, then the client should contact the NameNode.NameNode will look for metadata in the main memory and serve the client request accordingly.It may allow access or deny it according to the file permission.So, for fast access of data in HDFS, metadata is stored in the main memory. If the MasterNode i.e NameNode smash then whole data present in the main memory is completely lost, this is when metadata stored on the hard disk comes to rescue. Only metadata is recovered using this but not the block location. Basically this NameNode stores the metadata, it does not stores the actual data.

B. Block Location
When DataNode start up it will send Block Report to the NameNode.DataNode will scan through its hard disk, generates a list of all the blocks on it and sends this report to the NameNode. This report is referred to as Block Report. This report is send periodically.NameNode use Block ID from the block report that is send by all DataNodes for Location Mapping.This mapping is usually known as Block Location.In Metadata the filename is mapped with the Block IDs and in Block Location the Block IDs are mapped with the DataNodesaddress.
As a result of this, for a given file NameNode knows on which DataNodes the Blocks are located. So, the NameNode knows that the blocks of abc.txt file is stored on DN1, DN2, DN3, DN4 and DN5. Using Metadata and Block Location, NameNode has a complete knowledge of the entire cluster.

C. fsimage and edit log
fsimage represents the complete snapshot of the entire HDFS. Its format is very efficient to read. The complete file system image means the complete details of files and blocks on HDFS and also includes Namespace image and Edit Log[8].Any modification done on HDFS is recorded in edit log. Edit log is nothing but the log of activities on HDFS performed by the clients. It keeps on piling up and grows as the activities on the HDFS keeps on happening. e.g Let us consider copying of abc.txt file on HDFS. This operation is recorded in edit log and the data stored on the main memory is also updated. as a result client always sees the updated fs namespace. Each operation in the edit log is called the record or transaction. Each transaction is identified by the transaction ID. Transaction ID is incremented sequentially. Now consider if the NameNode crashes, then the whole file system would go down, this is also known as Single Point Of Failure(SPOF). Due to SPOF, all operation done till now are erased from the main memory. So when the NameNode restarts, fsimage is first loaded into the main memory. By doing this NameNode will restore all the previous state but to restore the recent operation, it is done by replaying all the operation from current edit log. There are many problems with this edit log as overtime edit log will accumulate lot of records and the size become very very large and in extreme cases it can fill up the disk space of NameNode. Also replaying lot of records from edit log would take a lot of time, as a result NameNode's boot up time is increased. This is when Secondary NameNode comes into pictures.

2. DataNode: Slave Node
DataNode acts as a workhorses of HDFS. The actual data is stored in the DataNode. The Files are divided into blocks and then these blocks are stored in different DataNodes across the cluster[9]. For e.g: If the file size is 150 MB and the block size is 64MB(default) then:

64+64+22=150 MB and no. of blocks are 3. After dividing Blocks are stored one by one.

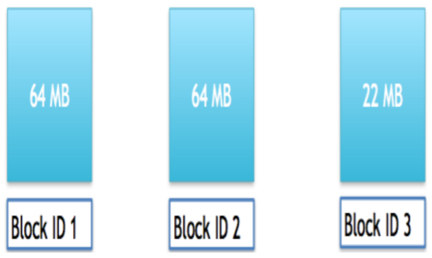

Figure 2: Blocks

While it is being stored, it is also replicated[10]and stored in different DataNodes as shown: Block ID 1 is of 64 MB, it replicates and stored in DN1, the second and third replica of Block ID 1 is stored in DN4 and DN5 respectively. Similarly Block ID 2 is of 64 MB, and it is stored in DN2, DN4 and DN5. Block ID 3 is only 22 MB, and is stored in DN3, DN4 and DN5. Last Block Size<=Block Size. All it depends on the file size.DataNode does not have any information about the files but they do have the information of the blocks.

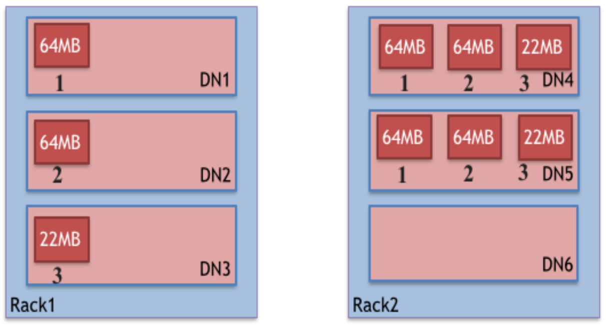

Figure 3: Blocks in respective DataNodes

3. Secondary NameNode
Secondary NameNode does not function like NameNode. If NameNode crashes then Secondary NameNode is used as the Master Node of the cluster is a wrong statement. It does not replace the NameNode. The main function of Secondary NameNode is to combine the fsimage and the edit log to produce a new fsimage(compressed) so that NameNode's main memory does not fills up due to ever increasing edit log. Also it creates checkpoints of the NameNode. Also known as Checkpoint Node. Basically Secondary NameNode is a Java program that just merges the fsimage and the edit log and creates checkpoints. As this merger is complex too so Secondary NameNode should be running on a good hardware configuration as it requires good computing resources. Due to merging, less time will be taken to replay all the records and because of this the boot up time of the NameNode is also reduced. Creating new fsimage is too complex. Sometimes it takes more time, during this time client access to HDFS is restricted. At this point checkpoint comes, now NameNode does not restricts access to the client, it will be active all the time. And Checkpointing is triggered after 1 hour or 1 million transactions in the edit log whatever is early.

4. JobTracker: Master Process
JobTracker is basically a MapReduce Daemon. It is a Master Process. Each cluster can have a single JobTracker and multiple TaskTrackers[11].The foremost function of the JobTracker is Resource Management i.e tracking the TaskTrackers, tracking its progress and fault tolerance.JobTracker creates and run the Job on the NameNode and allocates the job to the TaskTracker. Whenever the client submits the job to the JobTracker, it divides the job and splits it into the tasks. After this the JobTracker decides what tasks should run on which worker node. The process of assigning tasks to the worker nodes is referred to as Task scheduling. JobTracker is responsible for monitoring the tasks assigned to the worker node. The JobTracker carries out the communication between the client and the TaskTracker by making use of Remote Procedure Calls(RPC). RPC can be considered as a language that is used by the processes to communicate with each other. JobTracker keeps track of all the jobs and the associated tasks within the main memory. The memory requirement is huge, it depends on the number of tasks and tasks differs from one job to another.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 04, April-2018

5. TaskTracker: Slave Process

TaskTracker is again a MapReduce Daemon. It is a Slave Process. Within a cluster there can be multiple TaskTracker. It is the responsibility of the TaskTracker to execute all the tasks theatre assigned by the JobTracker. Each DataNode/SlaveNode can have a single TaskTracker running under them. Within each TaskTracker there are number of Map and reduce slots. These slots are referred to as a Task slots. The number of Map and Reduce slots determine how many Map and Reduce task can be executed simultaneously. The TaskTracker is pre-configured with the number of slots indicating the number of tasks it can accept. When a JobTracker tries to schedule a task, it looks for an empty slot in the TaskTracker running on the same server which hosts the DataNode, where the data for that task resides. If not found, it looks for the machine in the same rack. TaskTracker sends the HeartBeat signal to the JobTracker after every 3 seconds and if the JobTracker doesn't receive this signal, it will consider that TaskTracker as in dead state.

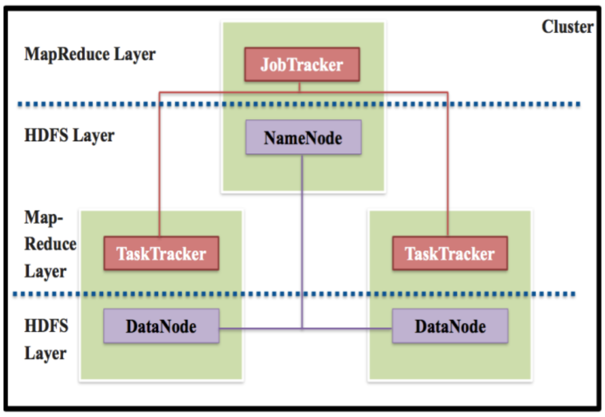

Figure 4: JobTracker and TaskTracker

IV SUMMARY

Big Data is one of the trendy topic now a days in the market. We are living in the era where we all are surrounded by huge amount of data which is difficult to handle. But with the help of Hadoop, the data is handled very nicely and efficiently. Hadoop is highly fault-tolerant and can be deployed on low cost hardwareIn this paper we have studied how all the daemons plays an important role in handling this huge amount of data. The daemons of HDFS i.e NameNode, DataNode and Secondary NameNode helps to store the huge volume of data and the daemons of MapReduce i.e JobTracker and TaskTracker helps to process this huge volume of data.All these daemons together makes Hadoop strong for storing and retrieving the data at anytime.

V REFERENCES

[1] M. A. Beyer and D. Laney, “The importance of ‟big data‟: A definition,” Gartner, Tech. Rep., 2012.

[2] Chen, C. L. P., & Zhang, C.-Y. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. Information Sciences, 275, 314–347. doi:10.1016/j.ins.2014.01.015.

[3] McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D., & Barton, D. (2012). Big Data. The management revolution. Harvard Bus Rev, 90(10), 61–67.

[4] Harshawardhan S. Bhosale1, Prof. Devendra Gadekar, JSPM’s Imperial College of Engineering & Research, Wagholi, Pune,a review on Big Data

[5] Yu Li; Wenming Qiu; Awada, U. ; Keqiu Li,,(Dec 2012),” Big Data Processing in Cloud Computing Environments”

[6] K. Bakshi, "Considerations for Big Data: Architecture and Approach", Aerospace Conference IEEE, Big Sky Montana, March 2012

[7] R. Abbott and H. Garcia-Molina. “Scheduling I/O requests with dead-lines:” A performance evaluation. In Proceedings of the 11th Real-Time Systems Symposium, pages 113–124, Dec 1990.

[8] G. Candea, N. Polyzotis, and R. Vingralek. “A scalable, predictable join Operator for highly concurrent data warehouses”. In 35th International Conference on Very Large Data Bases (VLDB), 2009.

[9] Garlasu, D.; Sandulescu, V; Halcu, I. ; Neculoiu, G. ;,(17-19 Jan. 2013),”A Big Dataimplementation based on Grid Computing”, Grid Computing

[10] Vishal S. Patil,PravinD.Soni” Hadoop Skeleton & Fault Tolerance In HadoopClusters”,IJAIEM Vol.2,Issue no.2,February 20

[11] MohommadAsif Khan, Zulfiqar A. Menon, SajidKhan,”Highly Available Hadoop Namenode Architecture”, International Conference on Advanced ComputerScience Applications and Technologies 2012,Confrence Publishing Services, DOI 10.1109/ACSAT.2012.52.