IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 19, Issue 3, Ver. VI (May - June 2017), PP 80-85
www.iosrjournals.org

Security Issues in Hadoop Associated With Big Data

Dr. M Praveen Kumar[1], Sampurnima Pattem[2]
[1] *Malla Reddy Institute of Technology, Hyderabad, India*
[2] *Gitam University, Hyderabad, India*

Abstract. *Due to the advent of new technologies, devices, Now-a-days the amount of data produced by mankind is growing rapidly every year. Big data includes huge volume, high velocity, and extensible variety of data. The Big data is in the form of Structured data (Ex: Relational data), unstructured data (Ex: Text, PDF, Word) and Semi Structured data (Ex: XML data). It is really a tedious task to process such data through a traditional database server. Google solved this problem using an algorithm called Map Reduce, used in the technology Hadoop. In this paper we discuss some security issues in Hadoop associated with Big data. Big data applications are very much useful to Organizations, all type of Companies may be small or large scale companies and to the Industries etc.*

Keywords: *Big data, Hadoop, Map Reduce, HDFS (Hadoop Distributed File System).*

I. Introduction

The collection of large datasets that cannot be processed using traditional computing techniques is called "Big Data". Big Data is the word used to manage huge volumes of different structures of data (structured and unstructured data) that are too large that it is very difficult to use that data to Process, using traditional databases and software technologies.

There are five main terms that signify Big Data properties:

(a) Volume: Many factors contribute towards increasing Volume storing transactional data, live streaming data and data collected from sensors etc.,
(b) Variety: Today's the data is storing in all types of formats-from traditional databases, text documents, Emails, video, audio, transactions etc.,
(c) Velocity: This means how fast the data is being produced and how fast the data needs to be processed to meet the requirements.
(d) Variability: Variability refers to data whose meaning is constantly changing.
(e) Complexity: Complexity of the data also needs to be considered when the data is coming from multiple sources.

Big data is not suitable to work with using relational database management systems and desktop visualization/static package; it requires massively parallel software running on many servers. By using big data we can analyze large data set to find correlations between different data, spot business trends, determine quality of research, predict disease spread, combat crime and much more, thus the applications of Big data include banking ,financial services, retail, energy, healthcare, airlines, manufacturing, telecommunications, pharmaceuticals, life science and many more.

Table.1 Difference between Traditional Data and Big data

Components	Big data	Traditional Data
Queries	Largely Abandoned SQL	Traditional SQL
Architecture	Distributed	Centralized
Data Types	Structured,Semi-Structured and Unstructured	Structured
Data Model	No schema	Fixed Schema
Data Relationship	Unknown or complex Relationships	Known Relationship
Data volume	Petabytes or Exabytes	Terabytes
Data Traffic	More	Less
Data Integrity	Less	High

II. Hadoop

Hadoop is an open source, Java-based programming framework that supports the processing and storage of extremely large data sets in a distributed computing environment. It is part of the Apache project sponsored by the Apache Software Foundation.The open source Hadoop framework is based on Google'sMap-Reducesoftware and can process large data sets at a granular level. It offers analytics at a low cost and high speed that some analysts say can't be achieved any other way. Essential to the effectiveness of Hadoop is the Hadoop Distributed File System (HDFS), which allows parallel processing by spanning data over different nodes in a single cluster and provides fault tolerance. However, HDFS is the source of one of the main issues users see with Hadoop technology: expanded capacity requirements due to Hadoop storing three copies of each piece of data in case a DataNode fails or is taken offline. That failover setup is necessary because each NameNode that controls the copy and distribution process of data is a single point of failure. Other complaints point to the complicated technology stemming from Hadoop's Java framework.

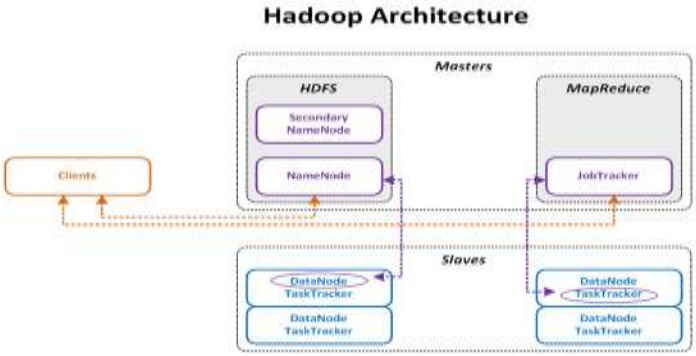


Fig.1. Hadoop Architecture

III. Hadoop Distributed File System (Hdfs)

HDFS is a file system that spans all the nodes in a Hadoop cluster for data storage. It links together file systems on local nodes to make it into one large file system. HDFS improves reliability by replicating data across multiple sources to overcome node failures.MapReduce is mainly used for parallel processing of large sets of data stored in Hadoop cluster.

3. 1 HDFS Architecture

The HDFS is the Java portable file system which is more scalable, reliable, distributed in the Hadoop framework environment. A Hadoop cluster contains the combination of single Name node and group of Data nodes. Using Commodity Hardware it provides redundant storage of large amounts of data with low latency where it performs the operations like "Write Once, Read Many Times". The files are stored as Block with default size of64MB. The communication between the nodes occurs through Remote Procedure calls. Name node stores metadata like the name, replicas, file attributes, locations of each block address and the fast lookup of metadata is stored in Random Access Memory by Metadata. It also reduces the data loss and prevents corruption of the file system.Name node only monitors the number of blocks in data node and if any block lost or failed in the replica of a datanode,the name node creates another replica of the same block. Each block in the data node is maintained with timestamp to identify the current status. If any failure occurs in the node, it need not be repair immediately it can be repaired periodically. HDFS allows more than 1000 nodes by a single Operator. Each block is replicated across many data nodes where original data node is mentioned as rack1 and replicated node as rack2 in Hadoop framework and never supports Data Cache [19][20] due to Large set of data. The architecture of HDFS is shown in Figure 4.

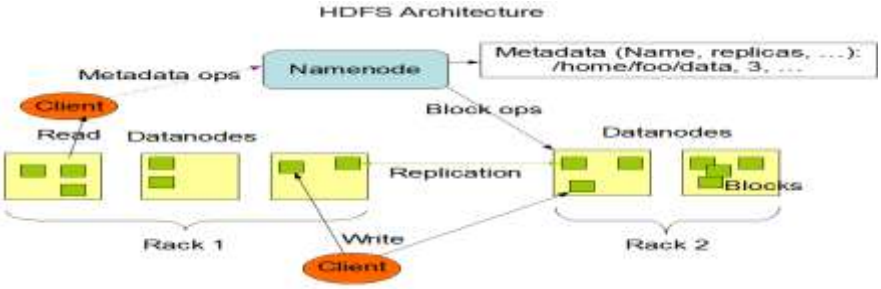


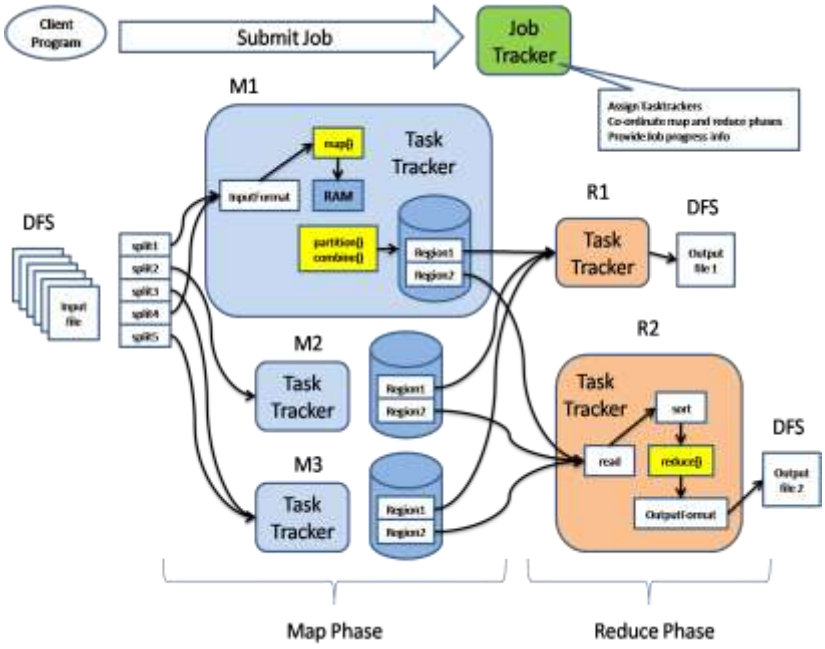
Fig.2. HDFS Architecture

3.2 Map Reduce

Hadoop MapReduce is a framework used to write applications that process large amounts ofdata in parallel on clusters of commodity hardware resources in a reliable, fault-tolerant manner. A Map Reduce job first divides the data into individual chunks which are processed by Map jobsin parallel. The outputs of the maps sorted by the framework are then input to the reduce tasks.Generally the input and the output of the job are both stored in a file-system. Scheduling, Monitoring and re-executing failed tasks are taken care by the framework. MapReduce is mainly used for parallel processing of large sets of data stored in Hadoop cluster. Initially, it is a hypothesis specially designed by Google to provide parallelism, data distribution and fault-tolerance. MR processes data in the form of key-value pairs. A key-value (KV) pair is a mapping element between two linked data items - key and its value.

The key (K) acts as an identifier to the value. An example of a key-value (KV) pair is a pair where the key is the node Id and the value is its properties including neighbor nodes, predecessor node, etc. MR API provides the following features like batch processing, parallel processing of huge amounts of data and high availability. For processing large sets of data MR comes into the picture. The programmers will write MR applications that could be suitable for their business scenarios. Programmers have to understand the MR working flow and according to the flow, applications will be developed and deployed across Hadoop clusters. Hadoop built on Java APIs and it provides some MR APIs that is going to deal with parallel computing across nodes. The MR work flow undergoes different phases and the end result will be stored in hdfs with replications. Job tracker is going to take care of all MR jobs that are running on various nodes present in the Hadoop cluster. Job tracker plays vital role in scheduling jobs and it will keep track of the entire map and reduce jobs. Actual map and reduce tasks are performed by Task tracker.

Hadoop Map Reduce architecture:



Map reduce architecture consists of mainly two processing stages. First one is the Map stage and the second one is Reduce stage. The actual MR process happens in task tracker. In between map and reduce stages, Intermediate process will take place. Intermediate process will do operations like shuffle and sorting of the mapper output data. The Intermediate data is going to get stored in local file system.

Mapper Phase

In Mapper Phase the input data is going to split into 2 components, Key and Value. The key is writable and comparable in the processing stage. Value is writable only during the processing stage. Suppose, client submits input data to Hadoop system, the Job tracker assigns tasks to task tracker. The input data that is going to get split into several input splits. Input splits are the logical splits in nature. Record reader converts these input splits in Key-Value (KV) pair. This is the actual input data format for the mapped input for further processing of data inside Task tracker. The input format type varies from one type of application to another. So the programmer has to observe input data and to code according. Combiner is also called as mini reducer. The reducer code is placed in the mapper as a combiner. When mapper output is a huge amount of data, it will

require high network bandwidth. To solve this bandwidth issue, we will place the reduced code in mapper as combiner for better performance. Default partition used in this process is Hash partition. A partition module in Hadoop plays a very important role to partition the data received from either different mappers or combiners. Petitioner reduces the pressure that builds on reducer and gives more performance. There is a customized partition which can be performed on any relevant data on different basis or conditions. Also, it has static and dynamic partitions which play a very important role in hadoop as well as hive. The partitioner would split the data into numbers of folders using reducers at the end of map reduce phase. According to the business requirement developer will design this partition code. This partitioner runs in between Mapper and Reducer. It is very efficient for query purpose.

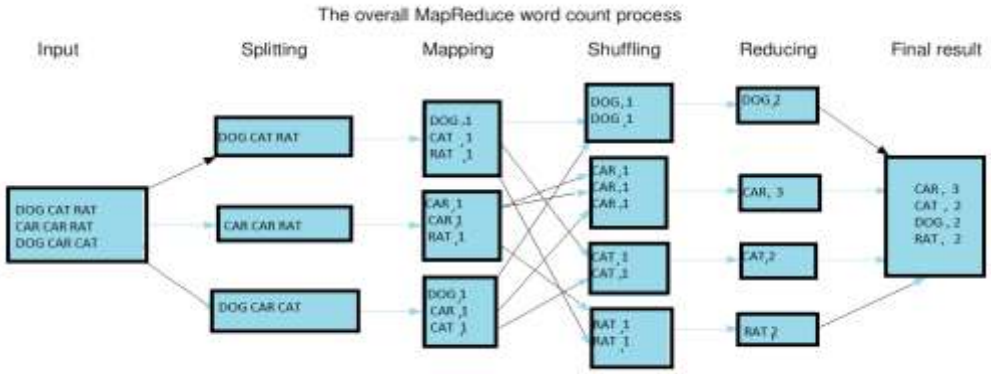
Intermediate Process

The mapper output data undergoes shuffle and sorting in intermediate process. The intermediate data is going to get stored in local file system without having replications in Hadoop nodes. This intermediate data is the data that is generated after some computations based on certain logics. Hadoop uses a Round-Robin algorithm to write the intermediate data to local disk. There are many other sorting factors to reach the conditions to write the data to local disks.

Reducer Phase

Shuffled and sorted data is going to pass as input to the reducer. In this phase, all incoming data is going to combine and same actual key value pairs are going to write into HDFS system. Record writer writes data from reducer to HDFS. The reducer is not so mandatory for searching and mapping purpose.

MapReduce word count Example



Explanation:

Suppose the text file having the data like as shown in Input part in the above figure. Assume that, it is the input data for our MR task. We have to find out the word count at end of MR Job. The internal data flow can be shown in the above example diagram. The line splits in splitting phase and gives a key value pair to input by record reader. Here, three mappers are running parallel and each mapper task is going to generate output for each input row that comes as input to it. After mapper phase, the data is going to shuffle and sort. All the grouping will be done here and the value is passed as input to Reducer phase. The reducers then finally combine each key-value pair and pass those values to HDFS via record writer.

IV. Security Issues In HDFS

The HDFS is the base layer of Hadoop Architecture contains different classifications of data and it is more sensitive to security issues. Also the risk of data access, theft and unwanted disclosure takes place when embedded a data in single Hadoop environment. The replicated data is also not secure which needs more security for protecting from breaches and vulnerabilities. Mostly Government Sector and Organisations never using Hadoop environment for storing valuable data because of less security concerns inside a Hadoop Technology. They are providing security in outside of Hadoop Environment like firewall and Intrusion Detection System. Hadoop environment is prevented with security for avoiding the theft, vulnerabilities only by encrypting the block levels and individual file system in HadoopEnvironment.Even though other authors encrypted the block and nodes using **encryption technique but no perfect algorithm is mentioned to maintain the security in Hadoop Environment**. In order to increase the security some approaches are mentioned below.

4.1) HDFS Security Approaches

The proposed work represents different Approaches for securing data in Hadoop distributed file system. The first approach is based on Kerberos in HDFS.

a. Kerberos Mechanism

In early versions of Hadoop **the restricting access w**here designed to prevent accidental data loss, rather than prevent unauthorized access to data. The file permission system in HDFS prevents one user to accidentally whole file system from a program. But does not prevent a malicious user from assuming root's identify to access or delete any data in cluster. HDFS file permissions provide only a mechanism for authorization. Only authorization is not enough for security purpose, because the system is still open for abuse**.** In year 2009 the first Hadoop authentication mechanism was implemented in Yahoo! In their design, the Hadoop itself does not manage the user authentication, instead it relies on Kerberos. Kerberos is a mature open-source network authentication protocol to authenticate users. Kerberos does not manage the user permissions. It just performs the user authentication process.It's the job of Hadoop to determine whether authenticated user has permissions to perform a given action. The first component, *primary*, is a string and may be operating system username of user or a name of a service. The *instance* is an optional section that follows the primary component. The instance may define a user role or a host name, on which the service is running. The instance is separated by primary by using slash. The third component, *realm*, is similar to a domain in DNS. The realm in Kerberos defines the group of principals.

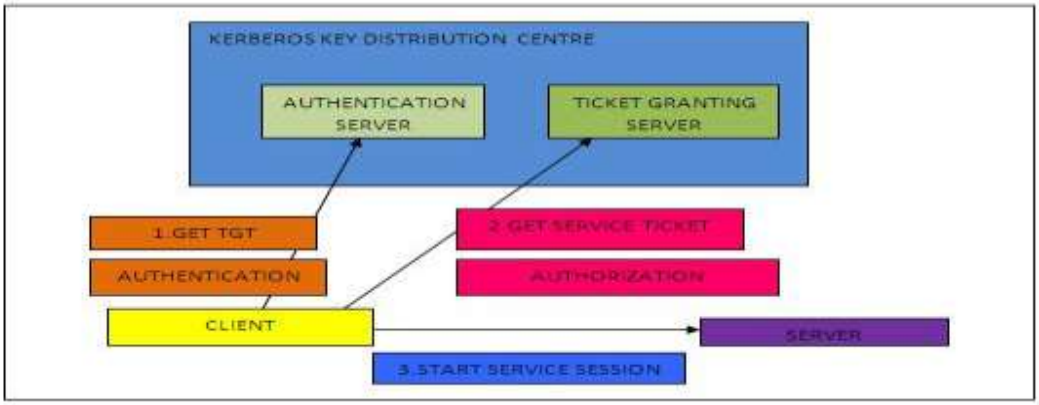


Fig.5 Kerberos Key Distribution Centre

b. Namenode Approach

In HDFS if there is any problem in Name node event and becomes unavailable, it makes the group of system service and data stored in the HDFS make unavailable so it is not easy to access the data in secure way from this critical situation. In order to increase the security in data availability, it is achieved by using two Namenode. These two Name nodes servers are allowed to run successfully in the same cluster. These two redundant name nodes are provided by Name Node Security Enhance (NNSE), which holds Bull Eye Algorithm. It allows the Hadoop administrator to run the options for two nodes. From these name node one acts as Master and other acts as a slave in order to reduce an unnecessary or unexpected server crash and allows predicting from natural disasters. If the Master Name node crashes, the administrator needs to ask permission from Name Node Security Enhance to provide a data from a slave node in order to cover a time lagging and data unavailability in secure manner.

Without getting permission from NNSE admin never retrieves the data from slave node to reduce the complex retrieval issue. If both Name node acts as a master there is a continuous risk occurs, reduces a secure data availability and bottleneck in performance over a local area network or Wide Area Network. Thus in future we can also increase security by using *Vital configuration* that provides and ensures data is available in secured way to client by replicating many Namenode by Name Node Security Enhance in HDFS blocks between many data centres and clusters.

V. Discussion

The proposed work represents different Approaches for securing data in Hadoop distributed file system. The first approach is based on Kerberos in HDFS, it is used to access a data blocks correctly and also only by an authorised user. Here Ticket Granting Ticket and Service Ticket playing a major role in providing a security in name node.The Second approach is based on Bull Eye Algorithm Approach explains about the security method from node to node and also scan the nodes in all the angles to prevent from attacks. The third

approach is based on Name node where the security is achieved by replicating [17] a name node to reduce the server crashes for future references.

VI. Conclusion

This paper shows the big data information and characteristics used in world wide. The issues are also mentioned to give idea about the big data issues in real time. The security issue is pointed more in order to increase the security in big data. We can improve security in big data by using any one of the approach or by combining these three approaches in Hadoop Distributed File System which is the base layer in Hadoop, where it contains large number of blocks. These approaches are introduced to overcome certain issues occurs in the name node and also in Data node. In Future these approaches are also implemented in other layers of Hadoop Technology.

References

[1]. Prof. Dr. Philippe Cudré-Mauroux, "An Introduction to BIG DATA", June 6, 2013
[2]. Fremont Rider, "The future of the Research Library",
[3]. http://www.gartner.com/newsroom/id/2 848718, STAMFORD, Conn., September 17, 2014.
[4]. "Leveraging Massively parallel Processing in an Oracle Environment for White Paper, November Big Data", an Oracle.
[5]. Jeffrey Dean and Sanjay Ghemawat,"Map Reduce: Simplified Data.
[6]. Datguise protect, http://www.dataguise.com/?q=dataguise- dgsecure-platform.
[7]. ParvizDeyhim, "Best Practices for Amazon EMR", August 2013.
[8]. Al-Janabi, Rasheed, M.A.-S., "Public-Key Cryptography Enabled Kerberos Authentication", IEEE, Developments in E-systems Engineering (DeSE), 2011.
[9]. Heindel L.E, "Highly reliable synchronous and asynchronous remote procedure s", Conference Proceedings of the IEEE FifteenthAnnual International Phoenix.
[10]. The journey to big data, EMC2 Publications.
[11]. Horton Technical Preview for Apache Spark, Horton works Inc.
[12]. Shay Chen, "Application Denial of Service", Hack tics Ltd, 2007.
[13]. Daniel J. Bernstein, "Understanding brute force", National Science Foundation, Chicago.
[14]. Introduction to Pig, Cloud era, 2009.
[15]. P.Victer Paul, N. Saravanan, S.K.V. Jayakumar,P. Dhavachelan and R. Baskaran, "QoS enhancements for global replication management in peer to peer networks".
[16]. AshishThusoo, JoydeepSenSarma, "Hive –A Petabyte Scale Data Warehouse UsingHadoop, Facebook Data Infrastructure Team.
[17]. P. Victer Paul, D. Rajaguru, N. Saravanan, R. Baskaran and P.Dhavachelvan, "Efficient service cache management in mobile P2P.
[18]. N. Saravanan, R. Baskaran, M. Shanmugam, M.S. SaleemBasha and P. Victer Paul, "An Effective Model for QoS ssessment in Data Caching in MANET.