International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2019): 7.583

Analysis on G-Hadoop for Big Data Computing Across Distributed Cloud Data Centers

Kartheek Pamarthi

Email: *kartheek.pamarthi[at]gmail.com*

Abstract: ***The computational demands for scientific data processing on a wide scale have recently increased dramatically. For example, in 2010, the data created by the Large Hadron Collider (LHC) in the domain of High Energy Physics (HEP) amounted to thirteen petabytes. In 34 different locations, 140 computing centres handle this massive volume of data. In recent years, the MapReduce paradigm has been the go-to programming approach for data-intensive, enterprise-level applications. Unfortunately, large-scale distributed data processing over several clusters is not yet possible with the MapReduce implementations that are available. These implementations were created for use with single cluster scenarios. One popular MapReduce implementation that uses a cluster to execute MapReduce jobs is the Hadoop framework. With G-Hadoop, the Hadoop MapReduce framework may be extended to conduct MapReduce workloads on several clusters. G-Hadoop, in contrast, only reuses the user login and job submission system of Hadoop (which is single-cluster). A novel G-Hadoop security model is suggested in this paper. The SSL protocol and public key cryptography are two of the many security technologies upon which this architecture is based; it was designed with dispersed environments in mind. Using a single-sign-on approach, this security framework streamlines the job submission and user authentication processes of the present G-Hadoop implementation. Furthermore, the G-Hadoop system is protected from conventional threats by means of a variety of security techniques provided by the built security architecture.***

Keywords: scientific data processing, MapReduce, G-Hadoop, security model, distributed environments

1. Introduction

The exponential expansion of the Internet and the WWW has meant that an enormous quantity of information is now accessible online. In addition, applications in the fields of social science, applied science, and engineering have resulted in the generation of substantial quantities of information, both structured and unstructured, that must be processed, analysed, and linked [1]. Computing that is data-intensive in today's world often makes advantage of contemporary data centre infrastructures and huge data processing paradigms of today. An investigation into the enormous data processing paradigm that is implemented across a number of data centres is the focus of this research. The need for distributing data-intensive studies of scientific data across several data centres has skyrocketed in the last several years.

The discipline of High Energy Physics (HEP) is a good example of an area that requires a significant amount of data analysis. Around thirteen petabytes of data were generated by the four primary detectors of the Large Hadron Collider (LHC) in the year 2010 [2]. Included in this group of detectors are ALICE, ATLAS, CMS, and LHCb. The 140+ computing centres distributed across 34 countries that make up the Worldwide LHC Computing Grid are responsible for storing these massive volumes of data. Data storage and first pass reconstruction are handled by the Tier 0 node of the Grid, which is located at the Centre for Energy and Nuclear Research (CERN). For the reasons of storage, extra reconstruction, and scheduled analysis, eleven sites that are categorised as Tier 1 receive a second copy of the data beginning with this Tier.

There are approximately 140 Tier 2 sites where simulations and user analysis are carried out [3]. Researchers frequently find themselves in the position of having to copy data from a number of different locations to the computer centre where the DATA ANALYSIS is going to be executed in order to carry out the latter. The copying process is laborious and wasteful due to the fact that different computing centres located all over the world are connected to one another through wide-area networks. Relocating the calculation instead of the data seems to be the most effective solution to this problem. Using data parallel processing paradigms on different clusters allows simulations to run on multiple computing centres at once, eliminating the need to replicate the data [4]. At present, data-intensive workflow systems are used for data processing that spans many data centres. Digital Asset Genie (DAGMan), Pegasus, Swift, Kepler, Virtual Workflow, Virtual Data System, and Taverna are some examples of such systems. Workflow paradigms can be used across different data centres, but there are several limitations to think about: (1) When it comes to high throughput data processing, which frequently requires massively parallel processing, the workflow system's coarse-grained parallelism falls short.

(2) Workflow systems built for data-intensive computing often necessitate the transfer of significant amounts of data between activities; in some cases, this might result in the movement of data blocks or data sets that are not necessary. (3) Workflow systems are required to ensure fault tolerance for the execution of tasks and the transmission of data, which is not an easy solution for computing that is data intensive. Data processing across distributed data centres using the MapReduce model seems like a no-brainer given its popularity. This would allow for the overcoming of the limitations that were previously highlighted in relation to workflow systems.

"Big Data" refers to a database that contains an enormous quantity of relevant and historical data. If used properly, this data may help businesses make decisions based on facts rather than opinions, which is invaluable since it represents their most valuable asset. Charles Tilly coined the term "Big Data" in 1980 when he added it to the Oxford English Dictionary

[12]. There are a lot of sources that generate data right now. Some examples are social media, remote sensors, mobile phone GPS signals, transaction records, and log files. The Internet's socialisation causes the creation of gigabytes of data every day, which might be structured, unstructured, or semi-structured. There is a substantial amount of this data that has intrinsic commercial worth. Consequently, a great deal of important data will be lost if it is not collected and processed correctly. What we call "Big Data" is actually just a collection of old data stored in a more efficient way than before using technologies like the Hadoop framework and other analytical techniques.

According to reference [13], "big data" can also mean "hadoop. " Big Data is defined by four attributes: volume, velocity, variety, and veracity. Not only has the tremendous growth of data gave rise to problems about the amount, velocity, diversity, and accuracy of data, but it has also given rise to concerns regarding the privacy and security of data. Recently, it has been suggested that the inclusion of another "V, " which stands for vulnerability, is connected to it (see Figure 1 for more information).

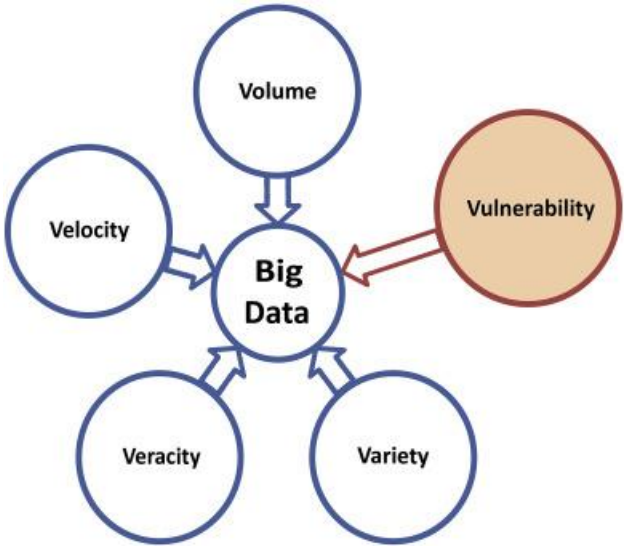

Figure 1: The several "V's" represent many facets of Big Data. The addition of one extra V is a result of system flaws.

Many countries have started major projects based on Big Data technology because of its ability to extract interesting value from data lakes. When it comes to capitalising on big data's prospects, the US is among the frontrunners. In March of 2012, with a budget of two hundred million dollars, the United States of America launched the Big Data Research and Development Initiative during the Obama administration. Launched on a global scale with new frameworks and technology, this Big Data endeavour has produced a plethora of new models. In order to facilitate higher storage capacity, parallel processing, and real-time analysis in a heterogeneous environment, new infrastructure has been designed. Big Data technology offers improved performance, scalability, and flexibility with little outlay of resources by leveraging common desktop PCs.

As a result of the implementation of cutting-edge environmentally friendly technology, the majority of storage and processing solutions are experiencing a steady decline in their prices.

2. Literature Review

Cloud computing

A electronic gadget The term "cloud" describes a group of services that are accessible through a network and provide on-demand access to scalable, guaranteed-quality-of-service (QoS), typically customised, and often affordable computer infrastructures. These networks are easily accessible to a large audience [14]. Cloud computing is based on the premise that users can access and utilise computer platforms, often called IT infrastructures, to host and execute their applications. An integrated computing platform is created as a service by providing users with services that enable them to access hardware, software, and data resources through computing clouds. One example of a technology that paves the way for cloud computing is the Hadoop framework, an open-source version of the MapReduce paradigm.

Distributed data-intensive computing

It is crucial to store, manage, access, and analyse massive volumes of data in order to meet the demands of data and information visualisation, analysis, mining, and searching. This is a huge problem that needs fixing. The development of data-intensive computing was a direct response to these needs. Since it relates to the study of data intensive computing, the research community is highly interested in the study on the paradigm of large data processing [15]. This paper presents the results of our work on the topic of implementing a data processing paradigm across multiple remote clusters. Several models and paradigms have been effective in data intensive computing, including All-Pairs, Sector/Sphere, DryadLINQ, and Mortar.

Notable among the technologies mentioned above is the MapReduce paradigm, which is a huge data processing approach that has gained widespread acceptance and will be discussed in the following section. Our work allows us to implement the MapReduce paradigm in a number of geographically distributed clusters.

MapReduce and Hadoop

MapReduce paradigm

A programming approach called MapReduce [16] was inspired by the Map and Reduce procedures, which are commonly used in functional programming. First, the Map function iteratively processes key-value pairings; second, the Reduce function combines a set of intermediate key-value pairs. There are numerous practical uses for this idea. Hadoop allows programmes to run on large clusters made of commodity hardware; it is the most popular framework for implementing the MapReduce model. Thanks to Hadoop, both data transmission reliability and dependability are made transparent. Multiple MapReduce implementations are available for various architectures, such as CUDA [17], multicore, FPGA, multiprocessor, cluster, large-scale shared-memory, streaming runtime, Grid, opportunistic, and mobile computing. Over a big physical cluster, virtual Hadoop clusters are made available with the use of Apache Hadoop on Demand (HOD) [18].

When allocating nodes, it makes use of the Torque resource management. myHadoop is a platform that allows users to use regular schedulers on HPC resources to provision on-demand Hadoop installations. As far as we are aware, the MapReduce concept has not been used to distributed multiple clusters yet.

The application of information technology to enhance traditional business intelligence is becoming more important in today's business landscape as a means to obtain a competitive edge. One of the more contemporary applications of IT is the analytical exploitation of client or customer data for the goal of trend forecasting and/or consumer behaviour tracking. The significant difficulty in gathering this large data is the sheer amount of user-generated data. If present trends continue, 35 zettabytes (35 x 1021 bytes) or 35 billion terabytes (TB) of data will have flowed across the internet by 2020. The Big Data phenomenon is also receiving a lot of attention, and with it comes an increasing demand for Big Data storage and analytics. International Data Corporation predicts that the Big Data sector would reach $32.4 billion in 2017. (IDC, 2013).

The growing need for Big Data and its anticipated impact on decision-making and ERP in the future highlight the significance of taking precautions to keep this information secure. Keeping client data secure throughout storage and transfer, as well as ensuring its confidentiality, integrity, and availability, is of the utmost importance because a security breach can have devastating consequences. Hadoop uses its own file system, HDFS, to manage the input/output data of the MapReduce applications.

The three most common kinds of vulnerabilities are those pertaining to infrastructure, data privacy, and data management [19]. The three aspects of data value chain, data life cycle, and architecture further classify these. Infrastructure, in the author's view, incorporates the architecturally-related vulnerabilities in both hardware and software. Protecting information while it is in use or while it is in transit is an important part of data privacy. According to an other source, there are five main areas of study concerning Big Data security and privacy concerns [20]: Hadoop encryption, auditing, and security; key management; and the ability to mask user identities. Data privacy and security in the Hadoop cloud environment is the focus of this article. Additionally, the author delves into different encryption methods that enhance Hadoop security, specifically focusing on Hadoop Distributed File System (HDFS) and the Kerberos network authentication protocol that is utilised for authentication in Hadoop. The Bullseye algorithm and the Name Node Security Enhance (NNSE) approach, which is employed between Hadoop's master and data nodes, are two further algorithms and methods that are mentioned for the purpose of monitoring and securing sensitive information.

Verizon published a document regarding cloud security [21]. The tiered security approach for cloud infrastructure is suggested in the study. The four sections of the paradigm are as follows: governance, value-added security, logical security, and basic security. Also detailed in that article was Verizon's sticky policy framework design. Big Data applications hosted in the cloud can be better protected with this plan. The author has previously discussed various types of attacks, including impersonation, replay, eavesdropping, man in the middle, and repudiation [22].

The author asserts that MapReduce processing is vulnerable to various attacks due to its distributed nature. Data confidentiality, availability for the mapper and reducer, and appropriate authorization and access control are essential for secure MapReduce computing. For authentication, we advise using the Kerberos protocol. A number of tokens are used, including those for delegation, block access, and jobs. Additionally, the article recommends and describes several security systems, including Apache Ranger, Apache Sentry, and Apache Knox. Additionally, the article covers how to secure data on a single public cloud from both advisory and multiuser cloud providers.

3. Proposed Method

Problem Statement

There is a process known as perturbation, which involves trading, including noise, or producing false information in order to replace distinctive traits with new ones. There is a decrease in the utility of information as a result of these anonymization efforts, which is a consequence of the overall data bad luck. Our proposal for productive big data performance is a combined method, which we present in order to address the challenges mentioned above.

The proposed approach on HDFS based on GDP Technique

Assuring data confidentiality prior to transmission is one of the key objectives of GDP. The grouped information is obtained by using the steps of the Geometric Data Perturbation (GDP) method.

These methods are employed to the clustered information, which makes it impossible for the unapproved clients to comprehend the information. At that point, the perturbed database had already been utilised to store the information that had been perturbed.

Geometric Data Perturbation (GDP) Technique

Randomization of the data is accomplished by the use of the perturbation approach, which takes the data from any one of the tables. An equation that can be used to represent the Geometric data Perturbation (GDP) is listed in below figure 2 is:

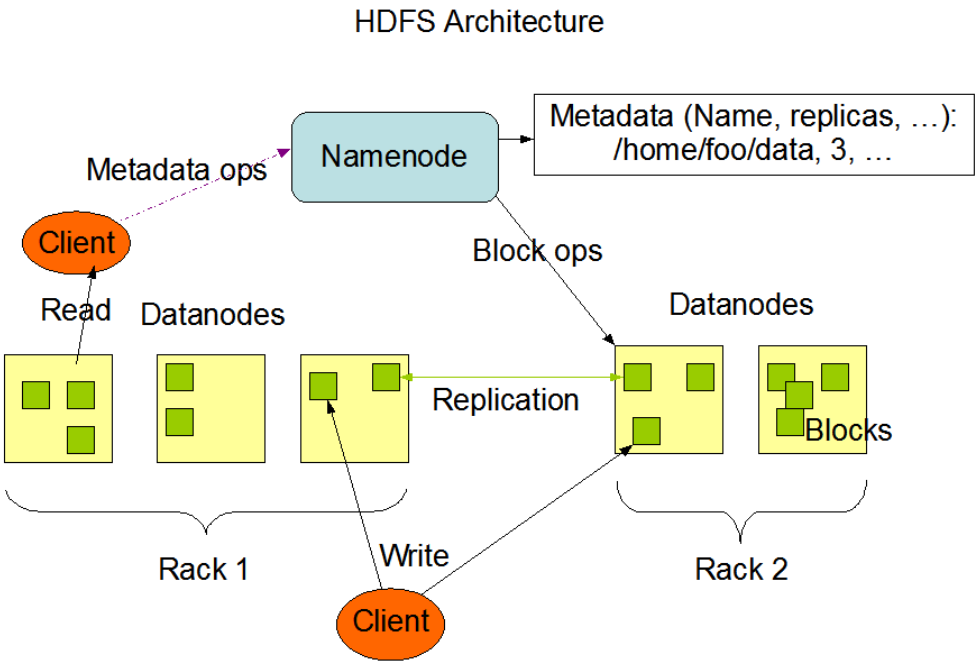

Figure 2: Proposed HDFS Based GDP framework

The software packages NameNode and DataNode are both optimised for use on common consumer electronics.

Operating systems (OS) that are frequently used by these machines are GNU/Linux computers. Both the NameNode and the DataNode software can be executed on any computer that is capable of running Java. HDFS is constructed using the Java programming language. HDFS is able to be launched on a broad variety of computers since it is written in Java, which is a language that is extremely portable. A dedicated workstation that is solely responsible for running the NameNode software is typically included in a deployment. A single instance of the DataNode software is executed on each of the other machines that are part of the cluster. It is possible to operate many DataNodes on the same system by virtue of the design; but, in actual deployments, this is not very common.

The design of the system is very much simplified by the presence of a single NameNode within a cluster. All of the HDFS metadata is stored in the NameNode, which serves as both the arbitrator and the repository. Information pertaining to users is never allowed to pass through the NameNode because the system was developed in such a way.

The File System Namespace

In addition, HDFS is compatible with the conventional hierarchical file organisation. The ability to create folders and store files within those directories is available to both users and applications. Users can create and remove files, move data across directories, and rename files; the file system's namespace hierarchy is similar to other popular file systems; and so on. Currently, HDFS does not have user quotas in place. In HDFS, neither hard nor soft links are supported. In contrast, these features are not totally out of the question due to the HDFS architecture. The file system namespace is maintained by NameNode. Any time the file system's namespace or any of its properties are modified, the NameNode must keep track of the change. Applications have the option to tell HDFS how many copies of a file it should maintain while running. One measure of a file's replication factor is the total number of existing copies. The NameNode is responsible for remembering this information.

Data Replication

The original intent of HDFS was to provide a reliable way to store extremely big files across a large cluster of machines. With the exception of the last block, every block in a file has the same size, and each file is kept as a succession of blocks. Duplicating a file's blocks allows for fault tolerance. Each file can have its own configuration options for block size and replication factor. Applications have the option to set the number of file copies. You can change the replication factor later on, but you can choose it when you're making a file.

There is only ever one person who can write to a file in HDFS because it is a write-once file system. Each and every decision pertaining to the replication of blocks is made by the NameNode. During regular intervals, it is provided with a Heartbeat and a Blockreport from every single DataNode that is part of the cluster. When a Heartbeat is received, it indicates that the DataNode is operating and functioning correctly. Any and all blocks that are present on a DataNode are included in a Blockreport and it was shown in figure 3.

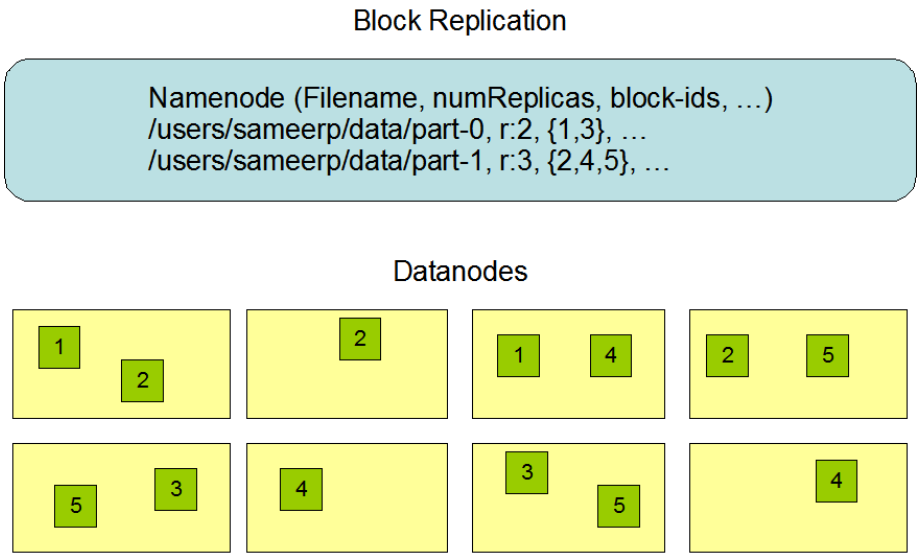

Figure 3: Block Replication analysis.

Replica Placement: The First Baby Steps

Reliability and performance in HDFS are greatly affected by the placement of replicas. What sets HDFS apart from the majority of distributed file systems is its optimisation of replica placement. A great deal of expertise and fine-tuning is required for this feature. For better data availability, dependability, and utilisation of network capacity, a rack-aware replica placement policy should be implemented. As a first step in this manner, the replica placement policy is currently being implemented. Validating the policy on production systems, understanding its behaviour, and laying the groundwork to test and study more complex policies are the short-term objectives of implementing it. Many racks of computers form a cluster that runs large HDFS instances. To enable connectivity between nodes situated in different racks, switches are necessary. As a general rule, bandwidth between computers on the same rack is usually greater than bandwidth between computers on different racks.

According to Hadoop Rack Awareness, the NameNode finds out which rack each DataNode is a part of. Placing copies on separate racks is an easy but inefficient strategy. This makes it possible to access data from numerous racks at once and avoids data loss in the event of a complete rack failure. This strategy ensures that replicas are distributed uniformly throughout the cluster, making it easier to handle load balancing in the event of component failure. The cost of writes goes up under this arrangement though, since they have to move blocks to more than one rack. As a general rule, when the replication ratio is 3, HDFS will typically distribute copies to three different nodes: one in the local rack, one in a separate remote rack, and one more within the same distant rack. Overall, this approach improves write performance by reducing inter-rack write traffic. Rack failure is much less likely than node failure, and this strategy has no effect on data availability or reliability guarantees. The overall network bandwidth needed to read data is decreased due to the reduction from three racks per block to two. This guideline states that the copies of a file will not be evenly distributed throughout the shelves. Every node has one third of the replicas, every rack has two thirds, and every other rack has one third of the replicas distributed evenly.

Without affecting data dependability or read performance, this policy enhances write performance. The protocol outlined here for the default placement of replicas is still under development.

Replica Selection

HDFS tries to minimise read latency and global bandwidth utilisation by fulfilling read requests from replicas that are geographically closest to the reader. If two replicas are on the same rack as a reader node, the one closest to the reader will be given priority when processing a read request. It is recommended to utilise a replica physically located in the local data centre instead of any distant replica when an angg or HDFS cluster is dispersed over multiple data centres.

Safemode

The NameNode enters a protected mode when it starts up. When a NameNode enters Safemode, data block replication stops. The NameNode is notified of status changes by the DataNodes through Heartbeat and Blockreport messages. A DataNode's Blockreport details all the data blocks it is hosting. There must be at least one copy of each block. Safe replication of a data block occurs when at least one copy of the block has communicated with the NameNode. Once a certain percentage of safely duplicated data blocks check in with the NameNode, plus an additional 30 seconds, it will exit Safemode. Data blocks that do not have the required amount of replicas will be included in the list.

Next, the NameNode sends copies of these blocks to every DataNode in the network.

The Persistence of File System Metadata

Nodes in the NameNode hierarchy hold the HDFS namespace. Every time metadata in the file system is modified, the NameNode keeps a note of it in a transaction log named the EditLog. Take HDFS as an example; whenever a new file is created, the NameNode will make a note of it in the EditLog. Also, whenever the replication factor of a file is changed, the EditLog is updated accordingly. Within the file system of its local host operating system, the NameNode stores the EditLog. The complete file system namespace, including the mapping of blocks to files and characteristics, is

stored in a file called the FsImage. The local file system of the NameNode additionally keeps a copy of the FsImage. A complete copy of the file system's namespace and Blockmap is kept in memory by the NameNode. The small design of this critical information item means that a NameNode with 4 GB of RAM may easily handle a large number of files and directories.

To begin, the NameNode loads the FsImage and EditLog from disc. It then updates the FsImage in memory by applying all the transactions from the EditLog. Finally, it saves this updated version to disc as a new FsImage. After applying its operations to the persistent FsImage, it can truncate the old EditLog. It is referred to as a checkpoint procedure. At now, a checkpoint is only triggered when the NameNode boots up. Plans are afoot to implement periodic checkpointing soon.

In its native file system, the DataNode keeps HDFS data in files. When it comes to HDFS files, the DataNode is clueless. It uses its native file system to store each HDFS data block in its own file. Not every file that the DataNode creates ends up in the same folder. Rather, it makes use of a heuristic to establish subdirectories based on the ideal number of files for each directory. Putting all of your local files in one directory isn't the best idea because your file system can crash under the weight of all those data. A DataNode's Blockreport is a report it delivers to the NameNode after starting up. It lists all the HDFS data blocks that are associated with each local file and traverses over the file system.

4. Result and Discussion

Table 1 and Figure 2 display the timestamps for the suggested perturbation technique. Our suggested and existing K-means clustering algorithms are shown to be correlated.

Table 1: Time taken to Perturb Health Data

File Size (MB)	10	20	30	40	50
Existing Approach (MS)	5923	8115	35561	19112	26550
Proposed Approach (MS)	5115	6545	11445	15855	22456

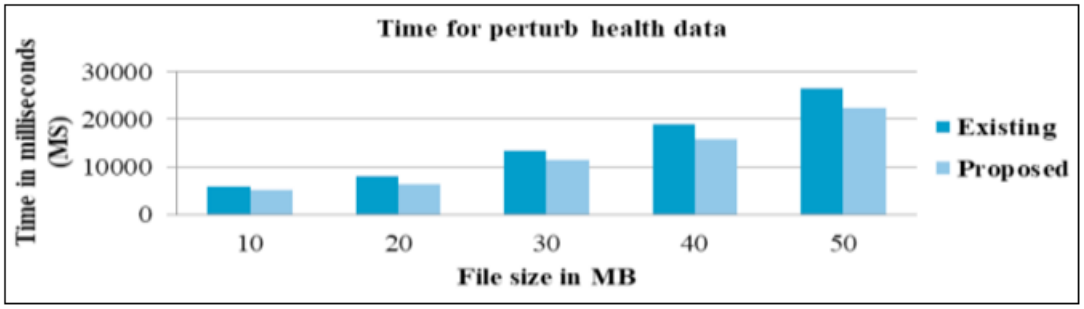

Figure 4: Comparative Analysis of Proposed and Existing method during Perturbation

Figure 4 displays a time-related comparison of the proposed and existing methods during the perturbation. With a file size of approximately 10 MB, the current method takes around 5923 MS to perturb the file, whereas the suggested approach takes around 5115 MS. This shows that the proposed approach is 86% better than the existing method.

Table 2: Time taken to de-perturb health data

File Size (MB)	10	20	30	40	50
Existing Approach (MS)	6113	8224	14177	20661	28885
Proposed Approach (MS)	5333	7124	10177	17661	22885

There is a time-based comparison of the current method with the proposed one in Table 2 and Figure 5, which show the results of the de-perturbation/information recovery process. Here, at a record size of roughly 20 MB, the current methodology takes around 6133 MS to retrieve the information, whereas the new solution takes about 5333 MS, an improvement of 84.95%. The strategies' time-to-action varies with the data's amount and scope.

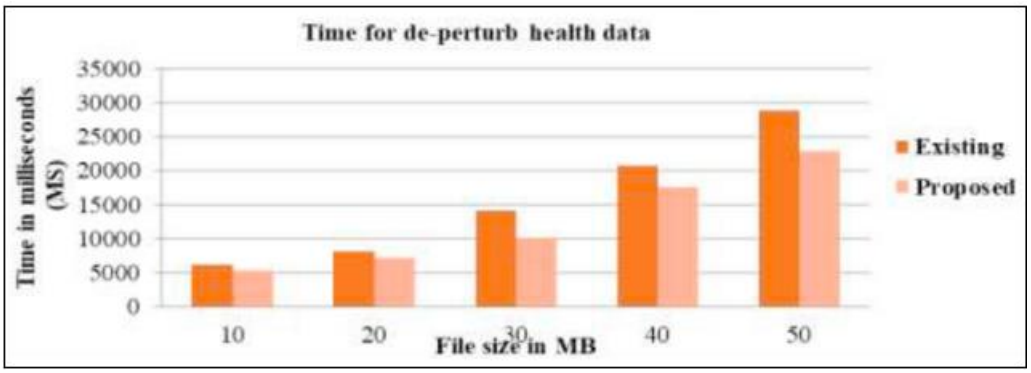

Figure 5: Comparative Analysis of Proposed and Existing method during Deperturbation

With regard to file size, Table 3 and Figure 6 display the accuracy attained by the proposed and existing traditional K-means clustering algorithms. Here, the accuracy rate for the conventional approach is approximately 86.32% at a file size of 30 MB, while the new approach achieves an accuracy rate of approximately 90.344 %. Without compromising cluster

accuracy, this proposed strategy guarantees that the entire clustering process runs on time.

Table 3: Accuracy

File Size (MB)	10	20	30	40	50
Existing K- Means Clustering Algorithm (%)	82.12	83.76	86.92	87.67556	91.678
Proposed GDP Algorithm (%)	84.776	86.88	90.344	91.776	93.66543

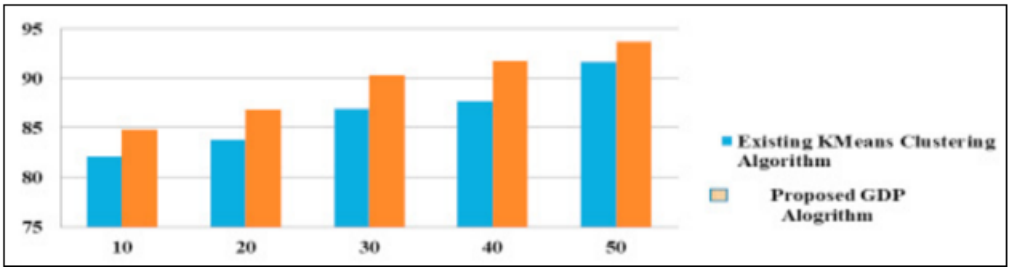

Figure 6: Accuracy comparison on proposed and existing method

5. Conclusion

Various data types and data volumes were used to test the suggested technique. This GDP algorithm compares the suggested and existing methods for maintaining health care information exchanges using metrics such as the time it takes to perturb health data and the time it takes to de-perturb health data. Before sending data back to the server, we used GDP to make it stronger. We plan to enhance the efficiency of data recovery processing and implement therapeutic image and sound information exchanges in GDP in further studies. G-Hadoop is built around an architecture that is based on a master-slave communication mechanism. To enable globally distributed data-intensive computation across many administrative domains, we combine the traditional HDFS file system with the Gfarm file system, which can manage large data sets across dispersed clusters. Distributed clusters can now manage enormous data collections with the help of Hadoop distributed file system (HDFS) and the Gfarm file system (GFS). The suggested security framework protects GHadoop from typical attacks like replay, delay, and man-in-the-middle attacks, allowing it to communicate securely across public networks. To further safeguard G-Hadoop resources from misuse, it implements many safeguards. Taken as a whole, it offers a reliable and comprehensive solution to the user's problem of logging into G-Hadoop with a single sign-on. Phase V task execution, encryption techniques, and keys will all be designed to be flexible, making cryptographic analysis more difficult for attackers.

References

[1] Dontha R. The origins of big data. Available: https: //www.kdnuggets. com/2017/ 02/origins-big-data. html. [Accessed 2 January 2019].
[2] Hashem IAT, et al. The rise of "big data" on cloud computing: review and open research issues. Inf Syst 2015: 98–115.
[3] Hurwitz JS, Nugent A, Halper F, Kaufman M. Big data for dummies. USA: wiley; 2013.
[4] Sharma S. Vulnerability – introducing V of big data. Available: https: //www.datas ciencecentral. com/profiles/blogs/vulnerability-introducing-10th-v-of-big-data. [Accessed 2 January 2019].
[5] Weiss R, Zgorski LJ. Obama administration unveils "big data" initiative: announces $200 million in new R&D investments. Washington, DC: The White House; 2012.
[6] Brauna TD, Siegel HJ. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. J Parallel Distrib Comput 2001: 810–37.
[7] Purcell BM. Big Data using cloud computing. Holy Family University Journal of Technology; 2013.
[8] Rouse M. Apache Hadoop YARN. Available: https: //searchdatamanagement. techtar get. com/definition/Apache-Hadoop-YARN-Yet-Another-Resource-Negotiator. [Accessed 4 January 2019].
[9] Scaling. Available: https: //stackoverflow. com/questions/11707879/difference-bet ween-scaling-horizontally-and-vertically-for-databases/12349220. [Accessed 4 January 2019].
[10] XingWang. Powered by Apache Hadoop. Available: https: //wiki. apache. org/hadoo p/PoweredBy. [Accessed 4 January 2019].
[11] Columbus L. Forecast of Big Data market size, based on revenue. may 2018. p.23. Available: https: //www.forbes. com/sites/louiscolumbus/2018/05/23/10-chartsthat-will-change-your-perspective-of-big-datas-growth/#1a9db88e2926. [Accessed 5 january 2019].
[12] SteveLoughran. Products that include Apache Hadoop or derivative works and commercial support. Available: https: //wiki. apache. org/hadoop/Distributions% 20and%20Commercial%20Support. [Accessed 5 January 2019].
[13] Gurjit singh Bhathal ASD. Big data solution: improvised distributions. In: Proceedings of the second international conference on intelligent computing and control systems. Madurai, India: ICICCS 2018; 2018.
[14] Fujitsu. FUJITSU technology solutions. Munich: Fujitsu; 2017.
[15] Amal Dahbur BMABT. A survey of risks, threats and vulnerabilities in cloud computing. Int J Cloud Appl Comput (IJCAC) 2011: 11–5.
[16] Ye H, Cheng X, Yuan M, Xu L, Gao J, Cheng C. A survey of security and privacy in big data. In: 16th international symposium on communications and information technologies. ISCIT); 2016.
[17] Terzi DS, Terzi R, Sagiroglu S. A survey on security and privacy issues in big data. In: 10th international

conference for internet technology and secured transactions. ICITST); 2015.
[18] Sharif A, Cooney S, Gong S. Current security threats and prevention measures relating to cloud services, Hadoop concurrent processing, and big data. In: IEEE international conference on big data; 2015. Washington, DC, USA.
[19] Derbeko P, et al. Security and privacy aspects in MapReduce on clouds: a survey. Comput Sci Rev 2016: 1–28.
[20] MITRE Corporation. Apache vulnerability statistics. Available: https: //www.cvede tails. com/vendor/45/Apache. html. [Accessed 7 January 2019].
[21] Yoder M. Cloudera's process for handling security vulnerabilities. Available: https: //blog. cloudera. com/blog/2016/05/clouderas-process-for-handling-securi ty-vulnerabilities/. [Accessed 8 January 2019].
[22] Bekker G.2019 thales data threat report – global edition. Available: https: //www.thalesesecurity. com/2019/data-threat-report. [Accessed 8 January 2019]