CHAPTER 1

INTRODUCTION

1.1 Background

Big data is an expensive term and is used for large amount of data sets or a combination of data sets which are very complexin nature [1]. The concept of big data has been in existence within digital communication and information science since the early days of computing. As data is created by everyone and for everything from various sources like mobile devices, call centres, web servers, and social networking sites, etc. [1], big data is growing rapidly. It is unstructured and semi-structured in nature which has been continuously generated by large organization or companies. It comes with enormous challenges including analysis, storage, searching, capturing, sharing, security, stealing issues, information exchange, data protection and visualization of data. Big data is very dynamic in nature, therefore conventional or traditional database management applications and techniques are not sufficient/inadequate to handle it properly [2]. It regularly alludes ordinarily to many on-going techniques to extract pattern or value from raw data and the use of predictive analytics. Sometimes, it is defined in terms of its characteristics including size and nature etc. In addition, its accuracy is very essential as our confident decision will lead us to achieve the goal after taking the results based on big data analysis.

Hardly five yearsago, personal computer stockpiling was ten to hundreds gigabytes. IDC's Digital Universe Study haspredicted that somewhere in between 2009 and 2020, computerized data or digital information will be increased by 44% from 0.8ZB to 35ZB [1]. Gartner forecasts also anticipated that 20.8 billion of IoT devices will grow by year 2020 [3]. Also, by some estimates, IoT devices will generate 507.5ZB of data per year by 2019 [3]. Furthermore, many surveys or

reviewers expect that volume of information will increase by 45% inthe next two years, and some says it will be doubled [1]. Figure-1 shows the exponential growth of big data volume with time. Thus, exponential growth of information or data is a moving target and requires more attention to capture, curate, handle & process and secure it.

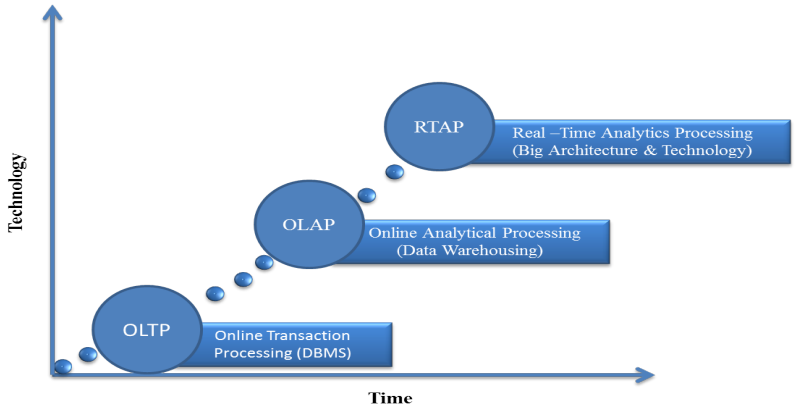

Figure-1.1: Growth of Data

Due to its dynamic nature, a better and modified model is required to handle and transfer the big data over the secure network [4]. Experts suggest that big data mining will provide the capability to take out the useful information from large datasets due to its characteristics likes volume, variability and velocity. The same was not possible before [5]. As big data is different from rest of data in terms of volume, velocity, variety, and value therefore, it's processing and moreover security & privacy protection have also become difficult for the government and the entrepreneurs. Because of its huge volume, the traditional methods cannot be used to extract and analyse it as they may not provide the accurate results for decision making etc. When the data was less in volume then itwas easily handled by RDBMS but RDBMS tools are not quite enough to manage when the data is large in volume i.e. big data [2]. Therefore, new and a capable processor is required which overcome the various problems arising during the processing of big data by traditional techniques. Apache Hadoop is developed as a distributed data processing platform for analyzing big data [6]. It is an

open source framework that can solve the big data's major issues like processing and managing within a tolerable time limit etc. along with extensive analysis. Many Enterprises can analyse big data containing user's sensitive information by using Hadoop and utilize them for their business purpose [4, 5, 7, 9].

The framework analyses and processes the big data. This is inspired by Google's MapReduce and Google File System (GFS) [13]. For distribution, processing and running applications for big data, this framework is used. Hadoop Distributed File System (HDFS) is a core component of Hadoop and it is used to store input and output data. It is the block-structured files system. Currently, the default size of the block is 128MB which was 64MB previously and default replication factor is 3 [6, 14]. The Block size and replication factors are configurable parameters. Hadoop MapReduce, a central module, is used to collect the data according to the requirement. Each job in MapReduce paradigm has user-defined map & reduce phase (that is processed in a parallel manner from one side to the other in a Hadoop cluster, by splitting the input data set into independent chunks and then making the data available for user consumption or additional processing) [14]. Earlier, it was designated for a single server but later it has scaled up to thousands of machines, each offering local computation and storage.

Big data analytics tools and technologies are rapidly adopted by many organizations and businesses [10, 11]. Unfortunately, the scientific community and researchers have given main focus to the advantages of big data analytics tools while less concern to its security & privacy protection. In today's environment, privacy & security protection are the major concerns and should be always on top of the priority before making the big data environment [12]. Moreover, security and privacy management have become the major concerns in respect of technical and social networks platforms. Big data need

security & privacy protection from unauthorized access [13]. Thus, big data security is a moving target and requires continuous monitoring.

1.2 Evolution of Big Data

Today, the world is wearing an entire gamut of digital patterns and interfaces and moving towards a future where even every single aspect of human being would be determined by ‘binaries’. With the continued bombardment of unimaginable technological innovations and breakthroughs, the concept that has been gaining currency across the globe is that of ‘big data’. The buzz around the big data is really for everybody, whether it is a national government or mega tech firms, entrepreneurs or academia. The term ‘big data’ has been unanimously accepted as the principle-driven of 21st century's socio-politic and economic landscapes. Hence, in the wake of all the curiosities around ‘big data’, it is important that the readers should be provided with a ‘synopsis’ of origins, initial developments and subsequent sophistication of the topic under discussion [16, 17].

The concept of big data started with a prediction in 1926 when Nicola Tesla presented with an idea that transactions and processing of data would be carried out by means of a handheld device [16]. After 11 years, a first major breakthrough came in the year 1937 when IBM successfully developed punch card reading machines for data tracking of millions of American employees and the public in general. The following decades saw the emergence of electronic computing on one hand and anxiety over storage and retrieval of ever-increasing data, on the other. At around the same time, Claude Shannon, often regarded as “Father of Information” discovered that Congress's Library had almost 100 trillion bits of data, a staggering figure [16].

Developments in the field of big data were then followed by a phase where impetus was laid upon how to make data warehousing more cost-effective and affordable endeavour. Eventually, the

breakeven point was reached when, in 1996, storing data on digital platform outscored paper usage for the same purpose. The very next year witnessed the birth of the term big data, although interpreted as a challenge while managing humungous unstructured data sets with existing computing systems. David Ellsworth and Michael Cox of NASA coined the term in one of their papers entitled Visualisation [16].

The invention of NoSQL, an open source database model, by Carlo Strozzi in 1998 played a pivotal role in processing large unstructured data sets for the next 10 years [16]. The year 1999 witnessed two important developments taking shape; quantification of information at UC Berkeley and the coning of term ‘Internet of Things’ by Kevin Ashton at P&G. To bring home the point, futuristic mobile technologies, 5G, is based upon the tenets of the Internet of Things. So, having a vision about the future, almost two decades ahead, speaks volumes about very significant digital paradigms. With the onset of the 21st century, dimensions of big data were configured by a Gartner Analyst Doug Laney, as 3v's namely Volume, Velocity, and Variety [16, 17].

The year 2005 saw quite a few interesting developments taking place in the area such as Hadoop, a tiny toy elephant. It is an open source platform for retrieval and processing of ‘big data’. With the National Science Board acknowledging the need to create right academic environment for churning out well, it takes care of the ever-increasing digital data. The three years that followed after, saw IDC carrying out first ever study showing estimates and forecast regarding the quantum of information growth, the introduction of first-ever Master's degree in Analytics by North Carolina University [11]. The year of 2008 was marked by a plethora of statistical calculations considering persistence onslaught of data that how US IP traffic will cross the figure of 1 zettabytes by the end of 2015 and how the CPU's across the world, have processed a mind-boggling 9.57 trillion

gigabytes of data until 2008. The tech titan, Google, announced that its servers were processing 20 petabytes of data every single day [17].

The trend of publishing reports on big data was continued. In 2009, Gartner report indicated how chief technology officers (CTOs) across the globe had only business intelligence in their mind while conducting the business [16]. According to the report, storage of analysts and managers in the US by 2018 would prove to be an impending crisis. The same facts were mentioned by one of the McKinsey's reports on big data in 2011. In the same year, IBM's supercomputer 'Watson' succeeded in processing nearly 4TB of data in seconds and that is how, 'Hadoop', the open source platform developed six years ago had begun to demonstrate its real potential by processing massive amounts of unstructured data. The year 2012 saw Obama administration paying heed to the concerns of big data companies that they would be having a tough time in finding analytics experts in coming years. The government came up with the announcement of big data centres and investments worth 200 million dollars on big data research programs. Further studies were conducted in the year 2013 highlighting an exponential increase in the volume of big data to 44 zettabytes by 2020 [1]. According to the study, the majority of the organizations were bullish about increasing their spending on big data analytics but were being pegged down by the lack of well-trained personnel.

1.3 Definitions of Big Data

Big data has become a dominant research topic in many fields, such as finance, business, international development, manufacturing, scientific research etc. Since the beginning era of computing, the approach of big data has been elemental to the world of digital communication and information science. It has got many definitions from technical practitioners, researchers and individuals. Few of them are as follows:

"Big data is a term used to refer to the study and applications of data setsthat are so big and complex that traditional data-processingapplication software are inadequate to deal with them" [18]. According to McKinsey & Company, "Big data refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyse"[7].

Both definitions emphasise on big volume of data but it can be seen that the volume of data is not the only criterion for Big Data. In addition to the large scale of data, big data has some other features that differentiate it from the other data. Gartner analyst Doug Lane and many other enterprises and researchers used the "3V" model to describe the big data [19] and in 2012, GartnerIT Glossary updated his definition as:

"Big data is high-volume, high-velocity, and high-variety information assets that demand cost-effective, innovative form of information processing for enhanced include advanced insights, decision making"[20].

Similarly, in 2012 TechAmerica Foundation defines big data as follows:*"Big data is a term that describes large volume of high velocity, complex and variable data that require advanced techniques and technologies to enable the capture, storage, distribution, management, and analysis of the information"[20].*

From Gartner definition, it can easily be inferred that big data is an information asset having high volume, velocity, and variety which require specific technologies and analytical methods to extract valuable information from it [4]. In the "3Vs" model, the word Volume is the size of the data set; Velocity indicates the speed of the data between the source and the destination, and Variety describes the range of diversity of data types and sources. These features that big data handles large amounts of data and uses different types of data,

including unstructured data and attributes that were never used in the database prior to big data.

Researchers and practitioners have explored big data by adding further V's according to their needs. SAS (Statistical Analysis System) added two extra dimensions i.e. Variability and Complexity [21]. In addition, Oracle has defined the big data in terms of 4 V's, Volume, Velocity, Variety, and Value [22]. The characteristics of large data can be summarized as 4V's, where 3V's are similar to 3V's models, and 4th V means Value i.e. the large data has great social value. The 4V model was widely recognized. Furthermore, A. Oguntimilehin, presented big data in terms of five V's and a Ci.e. Volume, Velocity, Variety, Variability, Value and a Complexity [1]. In 2014, Data Science Central, Kirk Born has defined big data in 10 V's i.e. Volume, Variety, Velocity, Veracity, Validity, Value, Variability, Venue, Vocabulary, and Vagueness [29]. All the V's has been listed and defined in next section. These characteristics provide various research horizons to the researchers and practitioners in order to effectively manage big data.

1.4 Big Data Characteristics

Over the time, researchers have explored big data in terms of its characteristics. These Characteristics make it easy to define the different aspects of big data and provide research horizon to the researcher and practitioners in order to effectively manage big data. All the characteristics has been listed and defined in table-1.

S.No.	Big Data Characteristics	Elucidation	Description
1	Volume	Size of data	Quantity of collected and stored data. Data size is in TB, PB [1, 20, 22-26, 30-31].
2	Velocity	Speed of data	The transfer rate of data between source and destination [4, 20-26, 30, 31].

3	Value	Importance of data	It simply represents the business value to be derived from big data [20-23, 29-31].
4	Variety	Type of data	Different type of data like pictures, videos, audio etc. arrives at the receiving end [4, 21-26, 30, 31].
5	Veracity	Data quality	Accurate analysis of captured data is virtually worthless if it's not accurate [22, 27, 30, 31].
6	Validity	Data Authenticity	Correctness or accuracy of data used to extract result in the form of information [23, 25, 29].
7	Volatility	Duration of Usefulness	Big data volatility means the stored data and how long is useful to the user [23, 29, 30]
8	Visualization	Data Process/Data act	It is a process of representing abstract [29, 30].
9	Virality	Spread Speed	It is defined as the rate at which the data is broadcast/spread by a user and received by different users for their use [22, 31].
10	Viscosity	Lag of Event	It is a time difference the event occurred and the event being described [22, 31].
11	Variability	Data Differentiation	Data arrives constantly from different sources and how efficiently it differentiates between noisy data or

			important data [21, 25, 29-31].
12	Venue	Different Platform	Various types of data arrived from different sources via different platforms like personnel system, private & public cloud etc. [22].
13	Vocabulary	Data Terminology	Data terminology likes data model, data structures etc. [22].
14	Vagueness	Indistinctness of existence in a Data	Vagueness concerns the reality in information that suggested little or no thought about what each might convey [22].
15	Complexity	Correlation of Data	Data comes from different sources and it is necessary to figure out the changes whether small or large in data with respect to the previously arrived data so that information can get quickly [1, 20, 21, 24].

Table-1.4: Big Data Characteristics

1.5 Importance and Applications of Big Data

In future, big data will become a basic key for growth of the individual firms and the basis for competition. In the environment of growing competition and the ability to capture the applicable information; all organizations are taking big data seriously. This will also lay the base work for complementary big data activities, like platforms development, infrastructure projects and techniques in settling complex and data-driven problems in sciences & engineering.

Thisis very helpful for exploitation of information aspects and creation of new facets for decision-makers [8, 32].

In the recent times, many US government agencies, such as the National Institute of Health (NIH) and the National Science Foundation (NSF) have ensured that the use of data-intensive decision has profound effect on their future development. Initially, the US government is trying to develop large datatechniques and technologies to ease their mission by big data [11]. According to the McKinsey Institute report [7], the effective use of big data has implicit benefits to transform economies and to give a new wave for productive development.

The valuable knowledge of the big data will be advantageous for the basic competition for today's enterprises. It is capable of creating new competitors as well as attracting skilled employees on big data. All policy makers, decision makers and researchers have to use the big data to highlight the next wave of development in their areas. With the big data many benefits can be obtained in the business segment, inform strategically to increase operational efficiency; develop better customer service; find and develop new services and products; new customers and markets identity etc. [8, 32, 34-39, 41]. Some applications of big data are explained as:

- **Understanding the Customers**

The requirement of every person/customer is unique in its own sense. They survey on many social media platforms before buying any product. Big data helps them for profiling these products into playful small 'jingles' in a far-reaching ways so that one-on-one, real-time interaction can be made with them. If they do not get what they really want, they will leave in short time. For example, if a customer who enters a bank, the big data tool permits the banking staff to offer him/her the exact service or product he/she wants. Big data also have an important role in digital and physical shopping areas: a retailer can

recommend a proposal to a company, which indicate a certain demand of consumers in the social media [32, 41].

- **Interest Based Recommendations**

Big data analytics allows the customer/person to personalise the contents on website or application in real time. For example, depending on age, gender and other physical parameters etc. of each consumer, the site or application can recommend the best desired product [8, 36, 41].

- **Customization of Products**

Big data also helps to understand that how other sites or companies comprehend your products so that you can enhance them accordingly. Analysis of unstructured social media allows you to differentiate your customer's needs and group them in different geographical locations or even in different demographic groups. On the top of it, big data allows you to test thousands of computer-aided designs within seconds so that you can check how different or minor changes can be done, e.g. affecting content costs, performance and lead times. Accordingly, you can change the process of production [1, 8, 32, 41].

- **Risk Analysis**

Success of your business not only depends on how you run it but other factors also play a key role for your achievements. With the help of big data analytics you can scan social media feeds and newspaper reports so that you can change the pace of latest development and environment accordingly [32].

- **Cost Effective Replacement**

In traditional industry environment, with the new advancements of technology certain type of machines or equipment's likely to become less effective or useless even though the machine or equipment has

useful life span. Big data tool helps to avoid such costly and unpractical expenses. As a result, the defective devices can be tracked without much delay for the better results [8, 32, 41].

- **Real Time Analysis**

Previously, if business personals lacked the technical issues required for large analysis, they had to ask IT professionals and as a result theyused to get the requested information. Presently, how they realize that the data given by them is valuable or right with the huge information instruments, specialized group working with quick calculations to assess the data given to them by their business consumers. They can create framework and introduce intuitive and dynamic permeability instruments that enable business consumers to analyse, view and make benefit from information [8, 36].

- **Personalised Healthcare**

Using big data tools and human genome mapping, one can cure the disease in a better and faster way, since everyone has their own medical record. It will overcome the generalised approach where for example a patient is suffering from cancer is treated with a medication, which if does not work is tied with another which takes time, money and increases the risk and fear in the patient. Big data tools can replace this and a patient can be treated as per his genes resulting in lesser fear and lower costs [35, 38, 39].

- **Data Security**

An organization will have the capacity to detect dangerous data and unsafe information which is conceivably sensitive. Since organization’s information can be mapped with huge information devices which permit penetrating and analysing the dangers that the association may confront inside.

Big data is being utilized in different managements and servicesand has huge scope of uses in various areas [32, 34-39, 41] as shown in figure-1.5. Big data prompts big time benefits for the analysts and numerous organizations.

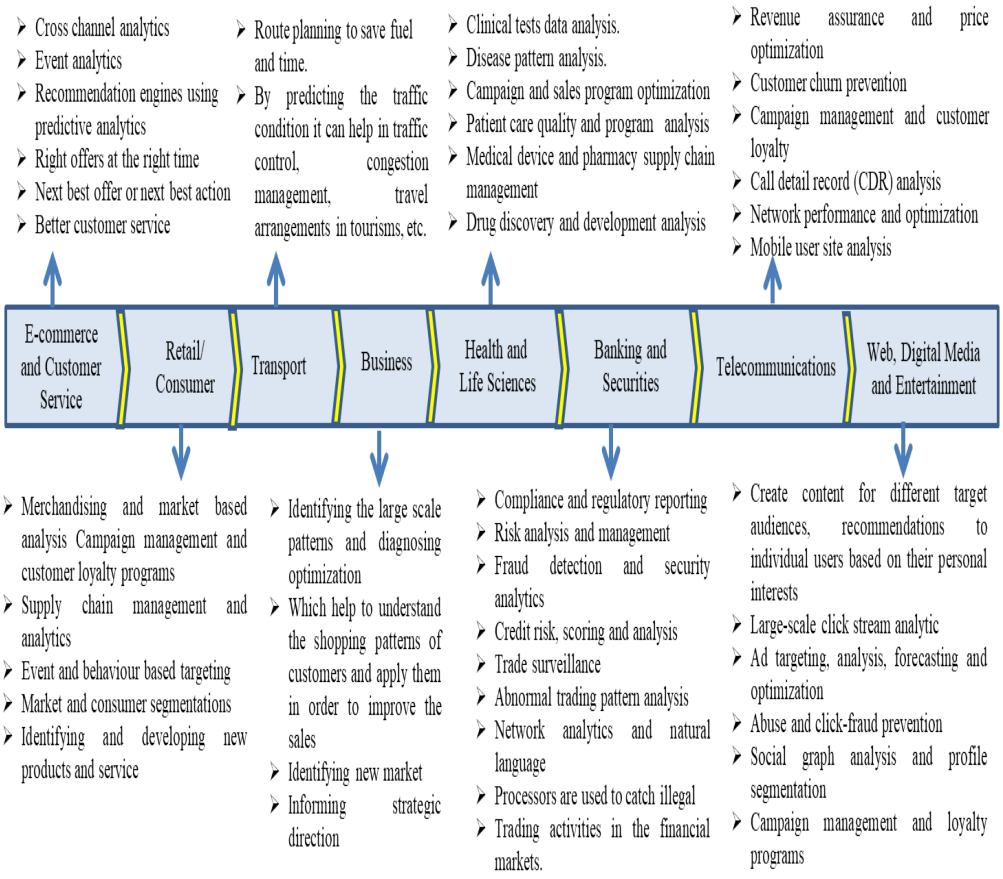

Figure-1.5: Big Data Applications

Electronic applications are regularly collecting lots of information, for example, network processing (counting long range interpersonal communication investigation, online correspondences, referral frameworks, reputation systems and anticipating markets), Internet content and documentation, Internet seek file and other hotspots. There are endless sensors around us that deliver little measures of sensor information that should be utilized, for example, Intelligent Transportation System (ITS), in the light of examination of substantial number of complex sensor information. Big data

innovation is turning out as a key IT part for the vast majority of the associations and business situations. There's developing spotlight on finding the business advantages of huge information investigation applications to legitimize the interests in them.

1.6 Big Data Challenges

Data volume will continue to grow and so are the possibilities of what can be done with so much raw data available. However organizations need to know just what they can do with that data and how much they can leverage to build insights for their consumers, products and services. While big data offers a numbers of benefits, it comes with its own set of issues. This is a new set of complex technologies, while still in the newest stages of development and evolutions.

Some of the issues include inadequate knowledge about the technologies involved, data privacy & security and inadequate analytical capabilities of organisations. A lot of enterprises also face lack of skills for dealing with big data technologies. As not many people are actually trained to work with big data, this has becomes an even bigger problem. This is not the only challenges or problem though. There are other challenges too. Some are identified after organisations begin to move into the big data space and some while they are paving the roadmap for the same. Following are the few challenges that are predominant in big data:

- **Processing Challenges**

Managing data in large and fast-growing volumes has been a daunting problem for many decades. Big data's storage and transmission issues have become significant and have magnified because of its characteristics like velocity, volume, and variety [37, 42]. When large datasets stored on the same infrastructure, controlling ownership and accessing becomes important issues. Majority of

generated data i.e., 80% of the world's total data is unstructured, and their sources are increasing rapidly, therefore, their alignment and sorting are creating more difficulties.

- **Data Challenges**

Data Challenges in terms of its handling are often based on four V's i.e. volume (data at rest), Velocity, Veracity & Variety. Velocity (data in Motion); implies how quick the information is being produced and how quick it must be prepared to take care of the demand [33, 42]. Veracity (data in uncertainty); is to check its "exactness" or "honesty" i.e. regardless of whether the information which is being prepared and later utilized is dependable or not. Variety (Data in numerous structure); Data originates from different sources and such data from each source can't be dissected and prepared similarly.

- **Human Resources and Man Power Challenges**

The present workforce is not competent enough to use big data efficiently. According to a survey, 60% of the total companies don't have the required skill to handle big data. These skills should be extended to research, analytical, interpretative and creative. These skills are strongly needed to be developed in individuals with necessary training programs organized by different organizations [2].

- **Technical Challenges**

Only quality data will lead to the better results, other irrelevant or useless data will end up taking more storages & time and will result in no positive output. The structured data is always highly manageable. When a broad range of application data is being collected, it is very difficult to find out the valuable information and high-quality data.

- **Security & Privacy Challenges**

Organisations are facing trouble with data security. This happens to be a bigger challenge for them than many other data related

problems. The data that comes into enterprises is made available from a wide range of sources, some of which cannot be trusted to be secure and compliant with organizational standards. Also the need for privacy of information is increased in big data. For example, government has applied strict laws for electronic health records regarding the gravity of different types of data that can be exposed [37, 43, 44]. Technical and social networks have a major problem of security and privacy management. For example, consider the data gathered from location-based services, for which the user needs to share the location with their service provider. Attackers can guess a user's location from the query source.

In the modern era of digitization, big data will have an important role in developing the next level of generation in the world. Nowadays mobile devices are found in everyone's pocket and all information is available at the fingertips. It has been observed that majority of data privacy techniques and mechanisms are applicable for structure data only. And, around 80% of data which is being generated all across the globe is unstructured in nature therefore it requires to develop a solution which can be applicable for both i.e. structured as well as unstructured data [45]. Hence, there is a need to address the following challenges.

- Most distributed system computations are provided with only single level of protection, which is not recommended.
- Additional security measures are often not available for the automatic data transfer.
- When enormous information is received by a system, it needs to be reliable and accurate.
- By practicing information mining, unethical IT expert can collect personal data without asking permission or without informingthe concern.

- Recommended detailed audit is not regularly included in big data because it contains large amounts of information.
- Maximum utilization of data during data privacy.

1.7 Need For Security in Big Data

Due to regular increase of users in digital platform like Facebook, Instagram, Twitter, etc., the data is increasing significantly. Also, government is opting e-governance/digitization instead of their existing paper based system which leads the security and privacy issues. Digitization is the process of converting information like image, sound text etc. into digits or bits therefore it is of crucial importance to data processing, storage and transmission. Various companies belonging to different areas like banking, health, social media, etc. are very much concerned about securing customer privacy as well as the financial frauds. Following are the recent incidences worldwide which reflects the dire need to update the current security practices to cope up with the security requirements of big data:

- Facebook admitted that public data of its 2.2 billion users worldwide has been leaked. The revelation once again underlines the failure of social media giant to protect the privacy of esteemed users. Due to this, the company faces troubles and gave explanation to the government [49, 51].
- In October 2017, Yahoo confessed that one billion accounts were accessed by hackers and later admitted that the actual statistics three million [48, 49].
- The Guardian has reported that global consultancy firm Deloitte was also hit by a cyber-attack. Data compromised includes emails, usernames, passwords, health information, and details of Deloitte's clients. The intrusion into its networks was carried out in March, 2017 [46].
- A consumer credit score company Equifax had revealed that hackers accessed up to 143 million customer account details in July, 2017.

The details hacked includes names, social security numbers, driver's licences, and credit card numbers of around 200,000 people [47].

Above are the few incidents takenfrom worldwide strengthen that the field of big data security requires constant and regular research not only by growing or developing companies/countries but also by the giants companies or developed countries. Though there are many solutions offered to solve these issues but problems still exist [13] requiring urgent attention to ensure security and privacy of user's information.

1.8 Research Issues

Big data comprises of two categories; sensitive data and general data. Sensitive data is the one which contains information about any entity or person which needs to be protected whereas general data does not require any protection. Protecting sensitive data has become a big challenge in today's world. The reason behind is that current big data processing and storage tools treat all data with same importance and do not combine special action such as encryption or blind processing for the data. Though Hadoop has emerged as an efficient big data processing and managing tool, it does not have any provision for security of the data in rest as well as in motion. Hence, the data is always in danger as it is very easy to steal or make changes in the records if a hacker gains access to the clusters.

As per the discussion in the previous section, available research and encryption algorithms are not competent enough to protect the data. These technique suffers from the threat of brute force attack, etc. In addition, applicability of the available encryption techniques in big data environment is also questionable [12, 54-56]. Hence, while going through the literature about big data and its security, the following research questions come into mind and which need to be addressed:

- What is upto date status of the security of big data?
- Are available approaches for data security applicable for big data security?
- Are available security approaches for big data security sufficient enough to secure big data?
- Does Hadoop, big data processor, have any built-in security feature?
- Are available approaches for data security applicable for big data security?
- Can an approach be developed which is secure as well efficient in terms of encryption time, decryption time, throughput of encryption/decryption and power consumption.

As it is already discussed that the available research in the field of big data security and privacy is not sufficient to manage the security and privacy of the same because these researches have some limitations, it must be explored further to identify new security method. This inspires to develop in depth understandings of big data security and privacy. This will help to solve issues related to big data security and privacy. While going through the literature about big data security, the study present in the paper has dug out new security encryption approach of it. Implementing encryption in all data on a granular basis helps to ensure that, even if there is a system breach, the data itself remains protected.

1.9 Research Problem

No doubt, the modern society is fully dependent upon computers for creation, storage, processing, communication and transfer of information. This generates enormous data day by day. As the data is becoming huge day by day its management and security has become a challenging task [13]. Recently, in March, 2018 [50], a leading active-wear company's app, MyFitnessPal, found out that an unauthorized party had accessed user data a month prior. On investigating it was found that roughly 150 million user's information had been

compromised, including their usernames, email addresses, and hashed passwords. In the same month i.e. March 2018, it was reported that a data-mining company, Cambridge Analytica, had exploited Facebook to harvest up to 87 million individual's personal details [51]. This exploitation was accomplished through an app that paid users to take a personality test and give consent for data collection. However, not only was their data collected but also the data from the participant's 'friends' on Facebook was stolen.

These incidents impose the need of security of big data. Unfortunately, Hadoop, the big data handler, has no security features of its own. In addition, it has still not researched that the tools and techniques available for securing traditional data are sufficient enough to secure big data [13]. Hence, in order to find out the answers of the research questions posed in the previous section, the researcher has formulated the following research problem:

Hence, based on the above discussion the researcher has formulated ***"A Novel Approach to Secure Big Data Using Attributes Based Honey Encryption".***

1.10 Objectives of the Study

In order to achieve the most general goal of working out on security of big data by integrating security features in Hadoop, the researcher has set the following objectives:

- To review and critically examine the literature on big data and its characteristics, big data security and its current implementation.
- To appreciate the need, importance and significance of identifying big data processing, managing, capturing, creation, storage, searching, sharing, transferring, analysis and visualization.
- To carry out in depth study of big data processor i.e., Hadoop to find out its strength and weaknesses in terms of security and privacy.

- To propose an efficient algorithm to be implemented and integrated with Hadoop to improve security of the big data.
- To compare the performance of available encryption algorithms with the proposed one.
- To perform review and revision, if required.
- To carry out the validation of the proposed algorithm for proving its usefulness.

1.11 Methodology to be Adopted

The proposed research work encompasses the tasks of establishment, development and evaluation nature. The development of a security mechanism to provide security for big data in rest as well as in motion has been set the prime objective of the research. It is supposed to accomplish through several phases including the following:

- Conceptualization, Review and Requisite Specifications
- Development of a cryptographic algorithm to be integrated with Hadoop for improving big data security
- Review and revision of the proposed algorithm
- Implementation
- Review and revision, if required
- Assessment of Effectiveness
- Pre-tryout
- Tryout
- Documentation and Finalization

1.12 Limitation

The work presented in the thesis has the following limitations:

- Due to unavailability of industry data, the proposed encryption algorithm has been implemented on a small set of data.
- The proposed work has been implemented using only .txt file.

- Due to single node setup, the propose encryption algorithm is only validated with a small set of data (MB to GB).
- In the proposed work, mapper has only been used. Use of reduce has not been used.

1.13 Thesis Organization

A thesis of the research is a document that has been prepared to reflect the work done carried out during the research. It also covers the detailed study on research problem and their solutions. The present thesis includes six chapters whose details are mentioned in the subsequent paragraphs:

Chapter-2 Literature Review

This chapter presents state of the art review about big data, its applications and the associated issues related to it. Apache Hadoop and their architecture along with the components in detail have been discussed. The chapter also explains about the file read and writes operations at HDFS level in Hadoop. In this chapter, the researcher focuses on the security aspects of Hadoop. Therefore, it covers the various existing approaches/methodologies/framework for securing big data at HDFS storage. Some important existing approaches developed during 2010 to 2018 have also been described in detail.

Chapter-3 Proposed ABHE Algorithm for Big Data Security

In this chapter, the researcher proposes an encryption technique called ABHE algorithm. The proposed scheme secures big data during transmission and storage. The evaluation of proposed algorithm has been done on the basis of performance parameters like encryption-decryption time, encryption-decryption time throughput and power consumption. Furthermore, the performance of proposed algorithm has been compared with existing encryption algorithms namely; AES, DES, and Blowfish (without integrating with the Hadoop) with different sizes of text files varying from KB to MB.

Chapter-4 Implementation of the Proposed Encryption Algorithm

This chapter implemented the proposed encryption algorithm developed in chapter-3. The proposed algorithm is integrated with Hadoop environment. Performance of proposed encryption technique has been compared with that of the two previously available encryption algorithms namely; AES and AES with OTP (integrating on Hadoop) with different sizes of text files ranging from MBs to GBs (64MB, 128MB, 512MB, and 1GB). The criteria of the comparison have been taken as encryption-decryption time, throughput of encryption-decryption and power consumption. This has been clearly proven that the proposed algorithm performs better than the previously available cryptographic algorithms.

Chapter-5 Validation of the Proposed Encryption Algorithm

The proposed ABHE algorithm outperforms the previously existing algorithms namely AES, DES, Blowfish when compared (without integrating with Hadoop as presented in Chapter-3). Also on integration with Hadoop, it outperforms AES and AES with OTP as presented in Chapter-4. Though it performs better in all the aspect, its acceptability is doubtful without validation. Hence, for the purpose of validation, two data sets have been taken; one from the Hadoop environment and the other from non Hadoop environment. This chapter presents validation of proposed approach.

Chapter-6 Conclusion

Major research findings along with the other findings are presented. The research questions posed in the first chapter are addressed one by one. Significance of the research is discussed in detail. Future plans for extending the study are discussed.

CHAPTER 2

Literature Review

2.1 Background

A framework that allows the analysis and management of a larger amount of data than the traditional data processing technologies is known as Hadoop [58]. The traditional data differ from the big data by following three ways: the amount of data (volume), the rate of data generation and transmission (velocity), and the types of structured and unstructured data (variety). These are three fundamental V's of big data. Many new characteristics of big data have been added by authors [1, 4, 20-31]. The use of technologies to extract valuable information from data and the ability to combine data from different sources and different formats is one of the most vital parts of the big data world. Big data technologies have introduced a new and different way for organisations to store data [57], and have empowered these organisations to develop a more in-depth understanding of their businesses. It reaps more advantages [59].

Traditionally, structured format was used to store data, like relational database so it was more efficient to process and understand that data. Although, now a new trend has been introduced i.e. unstructured and semi-structured format is used to store data [60]. Now, majority of technologies such as cloud computing or ubiquitous network connectivity have provided a platform where it is easy to collect, store and process the data [61]. Rapid growth of big data technologies have been led by this set of characteristics. From the use of big data technologies, not only big organisations but also small ones reap benefits [62].

The utility of the big data technology has been largely favoured by the release of new software: Apache Hadoop. Apache has developed a framework that is Hadoop which allows the processing of large datasets

by dividing them among clusters of computers using programming models. It has been developed in such a way that it is scalable from a single server to thousands of them, where each of them holds computation and local storage.

Apache Hadoop is completely an open source which has introduced entirely new and effective method of storing and processing data [65]. Google's published documents first highlighted its attempt for handling the stream of data [13, 63-65]. From then, it is now the fundamental standard for storing, processing and analysing the huge amount of data in terabytes and petabytes [3]. Even Hadoop provides the same kind of efficiency of expensive hardware in affordable, industry-standard servers. Data forms the input for Hadoop framework that feeds the big data system. As mentioned earlier, these data comes from various sources.

Hadoop works on its own distributed file system (HDFS) [65] which helps in storing the data into different servers along with different functions, such as NameNode, which is used to store the metadata; the DataNode, which store the application data. MapReduce is the main characteristic of Hadoop. MapReduce mainly focuses on processing and generating large datasets [67]. Two different functions allows MapReduce paradigm to accomplish this goal i.e. Map and Reduce. The map function creates a set of intermediate key/value pairs. The reduce function merges them to produce a solution. Hadoop ecosystem is comprised with various projects apart from MapReduce framework i.e. Hive, Pig, Sqoop, Mahout, Zookeeper, Spark, and HBase. These tools with some other hundreds of choices offer a range of possibilities that allow organisations to meet their expectations [64, 68-80].

Hadoop framework provides the best and eminent technologies ever existed to manage and process the large amount of data [66]. Various enterprises are dependent on Hadoop with trust and confidence

for storing and processing their previous data. But Hadoop lacks when it comes to security and protection of the data which is a major drawback [65] in today's scenario.

In order to understand this, assume the case of storage and processing systems security. There is a risk of privacy and security because of the exchange of data in storage and processing systems. Sensitive information can easily be stolen while data is being transferred from one node to another which threatens the privacy where the right to privacy and protection of the personal data should be guaranteed. Thus the big data security and privacy protection is a moving target and requires more attention and focus which is not available in Hadoop [65]. This imposes the need of implementing high speed processing systems and improved security mechanisms.

2.2 Hadoop Architecture

Hadoop works on a master/slave architecture in which master is called a NameNode and slave is known as DataNode. NameNode and DataNode are the daemons which store data and metadata. These are used for systematic management of distributed storage. Hadoop architecture consists of a master node and many slave nodes. Under the MapReduce layer, a task is managed, monitored or executed as shown in figure-2.2. Hadoop architecture consists of two core components:

- Hadoop Distributed File System (HDFS): Data Storage Component [65].
- MapReduce: Distributed Data Processing Component [65].

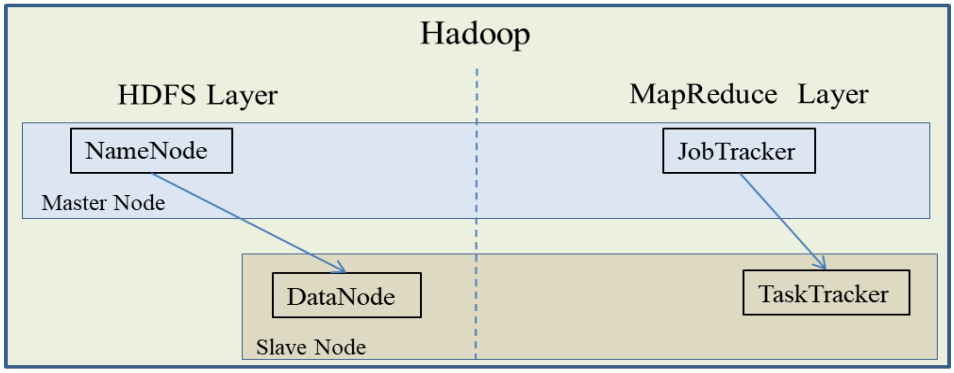

Figure-2.2: Hadoop Architecture

2.2.1 Hadoop Distributed File System

HDFS was developed with the use of distributed file system design. It can tolerate faults and hold huge amount of data while providing easier access to that data. HDFS works as a core component in Hadoop architecture. It is used to store various input and output data for the application. HDFS is the block-structured files system [3, 6, 15]. The current block size as default is 128MB which was 64MB earlier and default replication factor is 3. Block size and replication factors are configurable parameters. Various number of fixed size blocks are segregated in each file. They have the following characteristics:

- The blocks are stored in cluster on one or more machine hold adequate data storage capacity and their data storage is managed by the data node.
- One of the responsibilities of HDFS is to recover the data node. It also divides the data among the data nodes.
- HDFS manages adding or removing any node from a cluster is of HDFS.

a) HDFS nodes involve:

- Name Node
- Secondary Name Node
- Data Node

NameNode: The elemental part of the Hadoop is NameNode. So, complete Hadoop system goes down if the name node crashes. It controls the file system namespace and also stores the metadata information with the location of the data blocks.

Secondary NameNode: It is accountable for copying and merging the namespace image and editing log. In order to restart the name node if the NameNode crashes, the namespace image is stored in secondary name node.

DataNode: The storing and retrieving of the blocks of data is done by DataNode. It also reports the block's information to the name node periodically.

b) File Write Operation on HDFS

Following is the step by step process on how writing on a file is performed in HDFS:

1. HDFS client interacts with NameNode by calling the create() function on Distributed file system (DFS).
2. DFS sends a request to NameNode to create a new file.
3. NameNodes provides address of the DataNode i.e. Slave which is based on the availability of space and capacity in DataNode on which HDFS client is going to start writing data.
4. The HDFS client starts writing data through FSDataOutputStream to specific slave for a specified block.
5. The slave starts copying the block to another slave when HDFS client finished writing the blocks.
6. During the block copying and replication process, metadata of the file is updated in the NameNode by Datanode. (DataNode provides the periodically heartbeat signal to the NameNode).
7. After the successful completion of write operation DataNode sends the acknowledgement to HDFS client through DFS.
8. After that HDFS client closes the process.

9. Write operation is closed after receiving the acknowledgement from HDFS client.

The complete operation (with above steps i.e. 1,2,3...) is explained in figure-2.2.1(a)

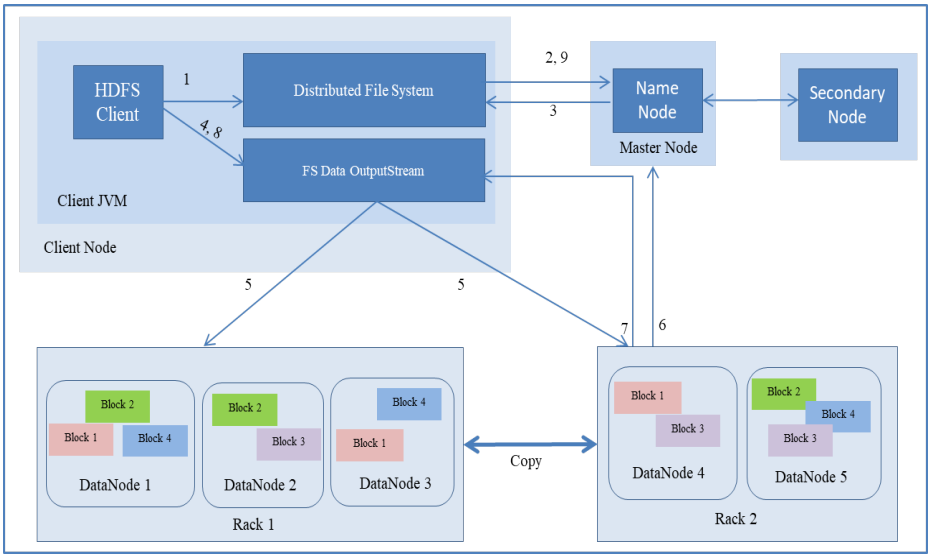

Figure-2.2.1(a): File Write Operation on HDFS

c) File Read Operation on HDFS

Following is the step by step process on how Read operation is performed in HDFS:

1. First of all HDFS client interacts with NameNode by calling the read() function on Distributed File System (DFS).
2. DFS sends a request to NameNode to read a file.
3. NameNode provides address of the DataNode i.e. Slave on which HDFS client will start reading the data.
4. The HDFS client starts reading the data through FSDataInputStream from specified slave from a specified block.
5. After a successful completion of read operation, HDFS client terminates read operation.
6. Read operation is closed after receiving the acknowledgement from HDFS client.

Step wise demonstration of Read operation is shown in figure-2.2.1(b).

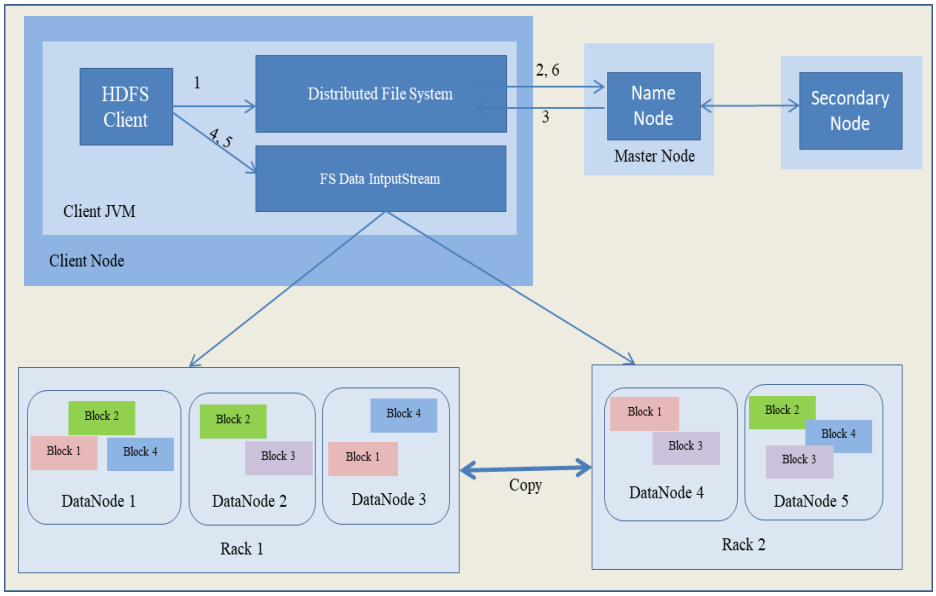

Figure-2.2.1(b): File Read Operation on HDFS

2.2.2 MapReduce

Hadoop MapReduce is a Java-based system developed by Google. The data from HDFS storage is processed by using MapReduce programming paradigm [66, 67]. A MapReduce program consist of two types of transformations that can be applied to data any number of times; a map transformation and a reduced transformation. A MapReduce job is an executing MapReduce program that is divided into map tasks that run parallel with each other and reduce tasks that run in parallel with each other as well.

a) **MapReduce** Nodes include the following:

- Job Tracker
- Task Trackers
- Job History

Job Trackers: It manages the jobs and resources in the cluster known as Task Trackers.

Task Trackers: These are the slaves deployed on each machine.

Job History: Server is a daemon that provides historical information about completed applications.

b) MapReduce Work Process

Split phase, mapping phase, shuffle phase and reduce phase are the four phases that are included in the whole MapReduce process. Let's understand this with example as shown in figure-2.2.2:

1. Input to a MapReduce job is divided into fixed-size pieces called input splits is a chunk of the input that is consumed by a single map.
2. Mapping stage is first phase in the execution of MapReduce job in which map or mapper job is to process the input data. The input file is passed to the mapper function line by line to produce output values. In figure-2.2.2 job of mapping stage is to count number of occurrences of each word from input splits and to prepare a list in the form of <key, value>.
3. In shuffle stage output of mapping phase is consumed. Its task is to consolidate the relevant information from mapping stage output. In figure-2.2.2 same words are clubbed together along with their respective frequency.
4. Reduce phase is supported by the shuffle phase. The reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.
5. After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

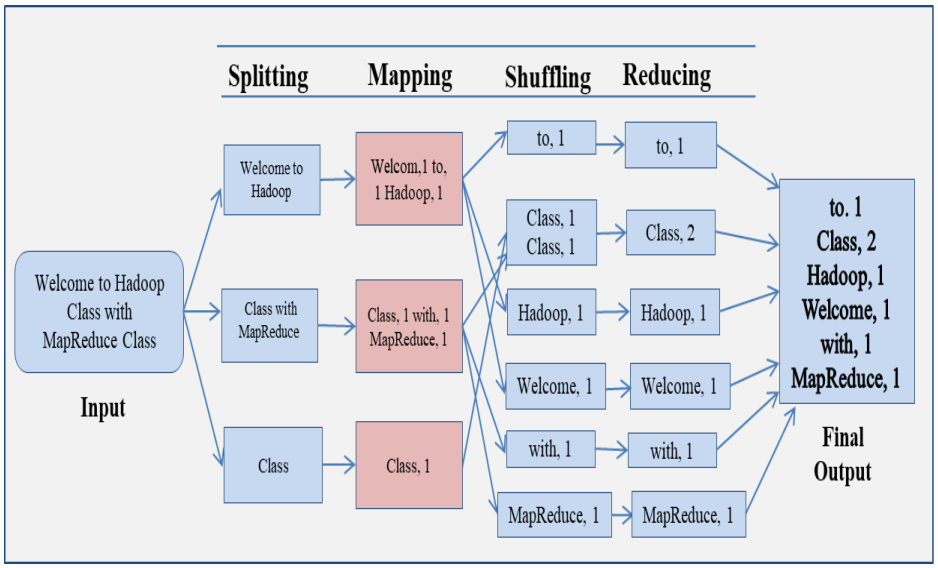

Figure-2.2.2: MapReduce Program Execution in Four Stages Process

c) MapReduce Input/Output Process

MapReduce process involves the following steps:

1. Input data in the form of image, video, text files is converted into < Key1 (K1) and Value1 (V1) > pair. This is done by input record reader.
2. Output (K1, V1) is again converted into Key 2 & Value 2 (K2, V2) by Mapper.
3. Second stage output i.e. K2, V2 is converted into K2 & list (Value2) with the help of shuffle and Sort techniques.
4. Reducer takes the values of Key 2, List (Value2) and generates the output Key 3, (Value 3).
5. Final output is generated by Output Record Writer which takes the output of Reducer (K 3, V3) as an input.

2.3 Hadoop Attributes

To overcome the problem of storing and processing of big data led to the invention of Hadoop. Hadoop has its own characteristics.

Following are the characteristics that are prominent in Hadoop environment:

- Cost effective: It is freely available in open market therefore, user don't need to pay for its subscription/licence.
- Flexible: It works in distributed manner. Hence it supports multiple operating systems and can store and process their data collectively.
- Fast: Data processing speed is very high.
- Resilient to failure: During a hardware failure, the system forwards the data to another node.
- Simple: It is easy to use and very handy.
- Reliable: It is very reliable because of its proven technology.
- Distributed process/computing:Various component of systems are shared with multiple computers giving better performance and efficiency.
- Time reduction: It has a parallel processing capability therefore, thereby reduced time during data processing.
- Customization: It is totally an open source therefore user can customize the source code as per their requirement.

2.4 Hadoop Ecosystem

Hadoop cannot be classified as a single tool or a programming language. It is a Java coded software library that is used to process large amount of data in a distributed environment. Hadoop standalone is not enough to provide all the services or facilities that are required to process big data. It is an ecosystem that consists of tools which help processing large amount of data ranging from gigabytes to petabytes. Hadoop falls under an Apache project which provides many facilities like MapReduce for parallel computing, etc. There is much more to do if one wants to create recommendation engine over big data, to run clustering engine over big data, and to get the nearly real-time access using big data itself. It is needed to add more and more components in

Hadoop to meet these requirements. This ultimately forms the Hadoop ecosystem [57].

Hadoop ecosystem comprises of Apache pig, Hive, HBase, MapReduce, Mahout, Oozie, Zookeeper, Sqoop, Ambari etc. These several components together when combined with Hadoop make the ecosystem much more scalable for a robust solution for big data. Figure-2.3 represents a typical architecture which comprises of the view of the overall Hadoop ecosystem.

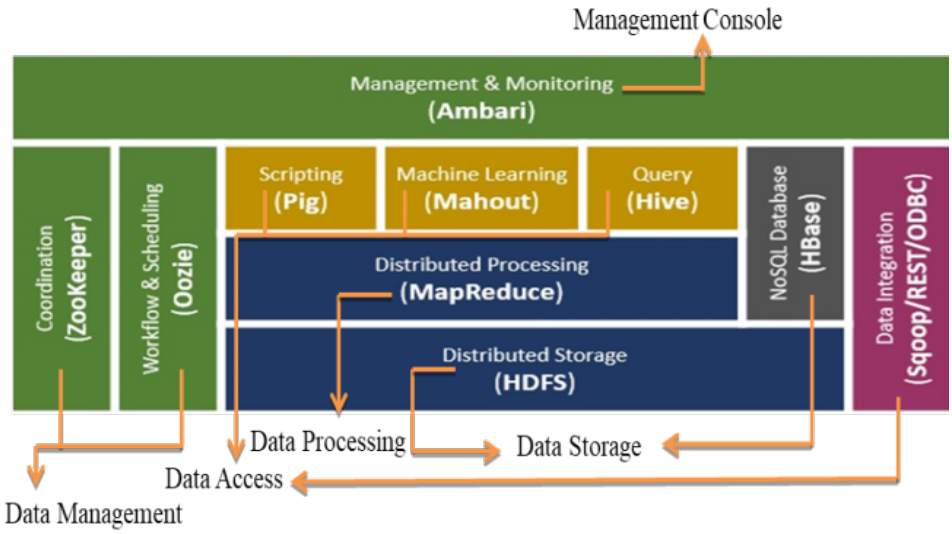

Figure-2.4: Apache Hadoop Ecosystem

2.4.1 Hadoop Distributed File System (HDFS): Data Storage

According to current standards, data is very enormous and a single computer isn't capable of storing it. So, a multiple networked system comes to play. HDFS has been developed using distributed file system design to meet the requirements. Its highly fault tolerant capability, cost effectiveness, and multi-platform support make it the perfect choice. HDFS is capable of storing very large amount of data which is stored on multiple machines. HDFS manages the application of huge datasets by having hundreds of nodes per cluster. Command line interface is used to interact with Hadoop. It can access the file and

system data over the browser and provides file permissions, authentication, etc. Its quick and automatic fault detection mechanism prevents any loss and recovers the data. HDFS has been discussed in detail in the previous section-2.2.1.

2.4.2 MapReduce: Data Process

It is a software framework that allows processing of large amount of data in parallel on the large cluster of commodity computer nodes to get faster results [66, 67]. MapReduce has been already discussed in detail in previous section-2.2.2.

2.4.3 Pig: Data Access

Pig has its own language called Pig Latin and provides a platform for computing the data flows in that language. It uses the basic data operations that include join, sort, filter, etc. It allows its users to create own function for read-write and process. It works on a Hadoop platform and uses HDFS along with Hadoop processing system called MapReduce. MapReduce is used to execute all Pig Latin scripts so that user can run series of MapReduce jobs instantly [64, 68, 70, 71, 73].

2.4.4 Hive: Data Access

Hive was developed by Facebook. It is a data warehouse infrastructure which defines a simple SQL like query language for querying and managing large datasets called Hive-QL (HQL). It also works in support of Hadoop. It is very much similar to SQL and easy to use. It frames various queries to be used on data using Hive-QL (HQL). It is used by Facebook to create faster queries, Netflix & FINRA CNET use this for data mining, internal log analysis, and ad-hoc queries [64, 68, 70-72].

2.4.5 HBase: Data Storage

HBase is a column-oriented key/value data store which is built to run over the Hadoop Distributed File System (HDFS). The wide-column store is inspired from the concept of Big Table and Apache

Hadoop. HBase performs exceptionally well in a random 'read or write' operation on the big datasets and batch computations with the help of MapReduce. HBase Master negotiates load balancing across all region servers and maintains the cluster state. It is not part of the actual data storage or retrieval path. Region server is installed on every machine and hosts data and processes I/O requests [68, 70, 71, 74].

2.4.6 Zookeeper: Monitoring and Management Component

Zookeeper is the centralized service used for maintaining configuration information, naming, providing distributed synchronization and group services which are very useful for a variety of distributed systems. HBase is not functional without Zookeper [76].

2.4.7 Mahout: Machine Learning

Mahout is a java library which provides the implementation of various machine learning techniques. It facilitates primary Recommendation (user base and item based collaboration & filtering), Clustering (group related text and documents), Classification (category using text and documents), and Frequent Item Set (individual items usually appears together). Mahout runs on top of Hadoop that is used for scalable and distributed machine learning [64, 75, 77].

2.4.8 Sqoop: Pluggable to RDBMS

It is a widely popular component because it can easily import data from many databases to HDFS and export data from HDFS back into RDBMS. It is a suite of tools that connect Hadoop and database system which generate code for using in MapReduce applications [78].

2.4.9 Oozie: Workflows Management Service

It is java web application which is based on data and time dependencies. It is a work flow management service system for Hadoop that allows to play and to connect lots of other essential components for instances, Pig, Hive and Sqoop [79].

2.4.10 Ambari: Managing Hadoop Clusters

Apache Ambari provides step by step procedure for installing Hadoop services across any number of hosts and also easy-to-use. Ambari provides central management for starting, stopping, and reconfiguring Hadoop services across and as monitors the entire cluster [80]. Amabari makes Hadoop management simpler by its Web user interface shown in figure-2.3.10. Most of the things are performed (configure/update) in a single click using ambari as a web interface.

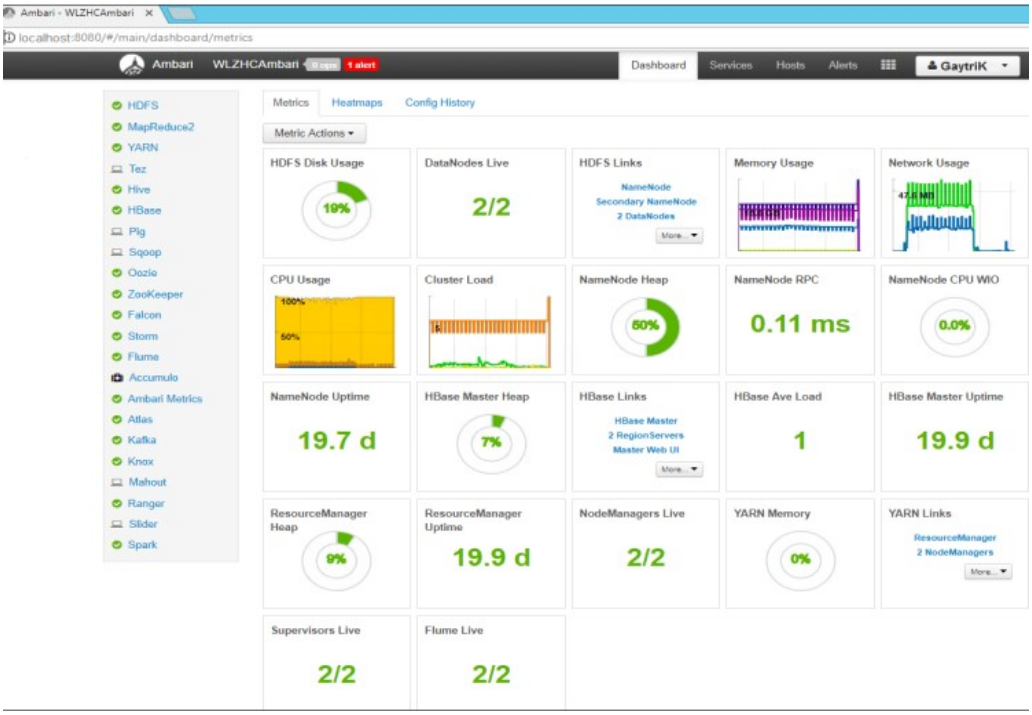

Figure-2.4.10: Ambari Web Interface View

It becomes very easy to organize, monitor and configure the Hadoop cluster, adding or deleting more nodes. Ambari is a completely open source tool and framework which provides the provision, to manage and monitor Hadoop clusters. It can also be used for simplified installation, configuration & management, and centralized security setup [80].

2.5 Hadoop Security

In 2008, when Hadoop was created, it intended to manage large amounts of data in a trusted environment. Hence, security issues weren't the topmost preference [82]. Taking into account the network as trusted, Hadoop uses local username. In default mode, there is no encryption between Hadoop and client host and in HDFS. All the files are stored in clear text and controlled by a central server called NameNode. So HDFS has no security mechanism against storage servers which may prove to be dangerous for the data stored on them. Besides, a strong security model is also lacking between Hadoop and HDFS. Matter-of-fact is that the communication between DataNodes and between clients and DataNodes is not encrypted. It had no authentication of users or service. In 2009, Yahoo focused on adding authentications. But Hadoop still had limited authorized capabilities [82]. In 2013, the Apache Software Foundation lunched project Rhino to add security features [82].

For many organizations Hadoop has evolved into an enterprise data platform that stores sensitive information. It has enabled data management that is massively scalable, agile, feature-rich and cost-effective. But as data that once was in silos is brought together in a vast data lake and made accessible to a variety of organisations, new security challenges arise. Big data that resides within a Hadoop environment may contain sensitive financial data, proprietary corporates information and personally identifiable information such as the names, address, and social security numbers of client's customers and employees. Due to the sensitive nature of this data and the potential damage that can occur if it falls into the wrong hands, it must be adequately protected [83-88].

2.6 Big Data: Hadoop Security Solutions

To elucidate the mentioned problems, some actions have been affixed to Hadoop to maintain the same [89, 90]:

Perimeter Security: Network Security firewalls, Apache Knox gateway

Authentication: Kerberos

Authorization: e.g. HDFS permissions, HDFS ACL3s, MR ACLs

Data Protection: Encryption at rest and encryption in motion. To provide security for data in HDFS few available mechanisms are as follow:

2.7 Authentication

Authentication implies your identification. Authenticators are answerable for gathering testimonials by the API (Application Programming Interface) consumers, authenticating them and publicizing the success or failure status to the clients or chain providers. Because of this primary check, uncertain users won't be able to access the cluster network and trusted network. Identification is regulated by client host. For strong authentication, Hadoop uses Kerberos [89, 90], and LDAP (Lightweight Directory Access), AD (Active Directory) integrated with Kerberos, establishing a single point of truth.

Kerberos is a computer grid authentication protocol which generates "tickets" to allow the nodes communicating over an unprotected network to prove their identity to one another. The reliable server authentication key is placed in each node of the array to achieve authenticity of the Hadoop cluster node communication which will develop the HDFS array. It can effectively prevent non-trusted machines posing as internal nodes registered to the NameNode and then process data on HDFS. These components are used throughout the cluster. Hence, from the storage point of view, the legitimacy of the nodes in HDFS cluster could be guaranteed by Kerberos.

It is completely entrusted by Hadoop for authentication between the client and server. Hadoop 1.0.0 version includes the Kerberos

mechanism. Client requests an encrypted token of the authentication agent. A particular service can be requested from the server by using this. Password guessing attacks remains inoperative to Kerberos and thus multipart authentication is not provided [92].

2.8 Authorization

Authorization or restrict access is the process of ensuring access within the data by the users as per the corporate policies or service provider. The authorization process is utilized by providersthat make decisions for giving permissions to the requested resources based on the user identity context. The authorization provider and their identity mapping rules determined the user identity. After evaluation of the user identity context, the authentication provider gives permission to the user for the requested resource.

Apart from this, authorization provider may also use an ACL (Access Control List) based authorization access called the Knox gateway [89, 90] which is based on the evaluation of rules that comprises username, groups and IP (Internet Protocol) addresses. These ACLs are pledged to prevent resources at the service level. Thus, they restrict entry to the Hadoop services based on user, group and remote IP address. The aim of Hadoop's developer is to provide a common authorization schema for the Hadoop platform to manage all the authorization policies for Hadoop components.

2.9 Data Protection

Data protection is a process to protect the data at rest or store and during transmission with the help of encryption and masking [89, 90]. Encryption is a technique which acts as an added layer in security in which data is encrypted (unreadable) during transmission or at rest. Hadoop is using the existing capabilities of data protection i.e. encryption during transmission, while bringing the solutions for encrypting data at rest, data discovery and data masking. However the

security in Hadoop is at an early stage, the work done by the researchers and practitioners are very commendable. There are various literatures aiming to improve the security of the sensitive data at rest as well as during transmission and some pertinent & relevant approaches are discussed below:

2.9.1 Achieving Secure, Scalable, and Fine-grained Data Access Control (2010)

The work combines techniques of attribute-based encryption (ABE), proxy re-encryption, and lazy re-encryption [93]. That simultaneously achieves fine grainedness, scalability, and data confidentiality during data access control in cloud computing. In this work, data files are encrypted using symmetric DEKs (symmetric data encryption key of a data files) and later, encrypted DEKs with KP-ABE (public key cryptography primitive for one-to-many communications). This dual encryption technique is called hybrid encryption.

The KP-ABE technique is used for fine-grained data access control and basic operations like files creation or deletion and allocation of new user. To achieve this major issue of user allocation, the authors combined proxy re-encryption with KP-ABE and distribute some tedious computational tasks to cloud servers. The cloud server stores secret key comprises key components and one dummy attribute corresponding to each user. Whenever data owner does some changes in set of attributes during user allocation, the corresponding proxy re-encryption keys are generated and transferred to cloud servers. Cloud servers, update their secret key components on the basis of proxy re-encryption keys received and re-encrypt data files accordingly. Due to this, data owner is free from computation load during user allocation and do not need to stay online all the time since cloud servers have already taken over this task after having the pre keys. Further, the burden of secret key updating and re-encryption of data file tasks are merged as single task using lazy re-encryption technique to save computation overhead of cloud servers during user revocation.

2.9.2 A certificate less proxy re-encryption scheme for secure data sharing (2012)

This research work presented a Certificate-Less Proxy Re-Encryption scheme (CL-PRE) [98] for secure data sharing with public cloud. This scheme uniquely integrates identity-based public key into proxy re-encryption technique. It eliminates the key escrow problem in traditional identity-based encryption, and does not require the use of certificates to guarantee the authenticity of public keys. CL-PRE is utilized to decrease the figuring and correspondence cost for information proprietor.

2.9.3 Fully Homomorphic Encryption (2013)

This research [90] proposed a design of trusted file system by using two emerging techniques i.e., cryptography fully homomorphic encryption technology and authentication agent technology for Hadoop which ensures the reliability and safety from the three levels of hardware, data, users and operations. In homomorphic encryption technology, usercan protect the security of data along with the efficiency of the application by making the encrypted data operable. On the other hand, authentication agent technology offers the various access control techniques which are the combination of different mechanisms like access control, privilege separation and security audit mechanisms and ensure the security of data stored in Hadoop file system.

Fully homomorphic encryption technique givesability to various users to carry out any operation on encrypted data with same results provided nature of the data remains same i.e., encrypted form throughout the operation. The data remains in encrypted form when processed with map reduce technique and stored safely on HDFS.

2.9.4 A Novel Data Encryption in HDFS (2013)

A new method for encrypting a file while uploading in HDFS has been proposedin this research work [94]. The upload process is done

along with the encryption process before uploading data to HDFS. In this method [94], user selectsa file to upload and provide a unique secret key to encrypt that file. In this approach, user can feel the same experience when uploading a normal (without encryption) file to HDFS since, encryption is done transparently. Also, this method uses the characteristics of read/write operation to reduce the total time in HDFS. The author experimented on 32MB file and observed that the encrypting upload and decrypting download process is usually 1.3 to 1.4 times faster than the conventional method. The major drawback of this approach is key management because keys are increasedwith respect the usersand to deal with them is quite challenging. Additionally, encrypting file sharing issue is also not possible with this approach. This proposed approach is lagging due to these two major issues and need some attention from new researchers and practitioners.

2.9.5 Secure Hadoop with Encrypted HDFS using AES encryption/decryption (2013)

Security in Hadoop architecture is proposed in this paper by introducingencryption and decryption functions in HDFS [95]. In Hadoop, it is achieved by adding AES encrypt/decrypt function to Compression Codec. Experiments on Hadoop proved that the computation overhead is reduced by less than 7% when representative MapReduce job on encrypted HDFS.

2.9.6 Triple Encryption Scheme for Hadoop-Based Data (2013)

The blooming of cloud computing in past years is due to its ability to provide users with on-demand, flexible, reliable, and low cost services. Therefore, data protection is a vital concern in the cloud data storage with the increasing demand of cloud applications day by day. A method called novel triple encryption has been introduced in this paper [96] to achieve data protection at cloud storage. The Triple Encryption approach uses DEA (Data Encryption Algorithm) for HDFS files encryption, RSA for the data key encryption and finally IDEA for encrypting the user's RSA private key. In this approach, DES and RSA

based hybrid encryption technique is used to encrypt HDFS files and IDEA (International Data Encryption Algorithm) to encrypt the RSA based user's key.In Hybrid encryption, DES algorithm is used to encrypt the files and get the Data key. This Data key is again encrypted by using RSA algorithm to provide additional security. The Data key can only be decrypted by using private key therefore; user has to always keep this private key always. This method uses both symmetrical and asymmetrical encryption techniques, so called as hybrid encryption. This approach is tested and implemented in Hadoop based cloud data storage.

2.9.7 Accessing HDFSBased on Attribute-Group (2014)

Due to various security issues in the cloud storage, development of cloud storage and its use has been decreased. To gain users confidence, author proposed an attribute group based data security access scheme to protect the data during network and data sharing features in cloud storage services. In this scheme [97], data owner has limited user rights and re-encryption on the data node reduces the computational cost along with the management of the clients. It also reduces the complexity of property and rights management. Also, the author uses cipher text CP-ABE encryption algorithm to provide cloud storage data security. The centralised management of key distribution and NameNode based CP-ABE algorithms have advantages like easier to manage keys and more transparent to the user as compare to the data owner directly distributed key distribution technique.

2.9.8 Towards a trusted HDFS storage platform (2014)

The concept of creating a trusted HDFS and the protected mechanisms for combating the security threats to a Hadoop infrastructure are explained in this research paper [99]. Also, the researchers have tried to figure out the relation between security mechanisms and their effect on its performance [99]. In the discussion, author implemented trusted computing concepts on a Hadoop considering a threat based environment. This framework is based on

theTrusted Platform Module (TPG) and implemented into a base environment. Also, the authors have designed an encryption scheme for Hadoop utilizing hardware key protections and AES-NI for accelerate the encryption and compared results after their implementation. Further, author claimed 16% overhead reduction on encryption and 11% overhead reduction while decryption during experiment on simulated 128MB block data with the AES-NI instructions. This approach provides a concrete layered security design in Hadoop.

2.9.9 Security Framework in G-Hadoop (2014)

An approach has been introduced where Hadoop's MapReduce task run simultaneously on multiple clusters in a Grid environment called G-Hadoop [90, 91]. G-Hadoop reuses user authentication and job submission mechanism of Hadoop in a Grid environment. But initially, Hadoop's user authentication and job submission mechanism have been designed for a single cluster in a non Grid environment. Therefore, G-Hadoop is an extended version of Hadoop MapReduce Framework and uses the Secure Shell (SSH) protocol to establish a secure link between a user and the target cluster. In this approach, a dedicated (single) connection is required to each participating cluster and users have to log on to only those clusters for which they are authenticated. Hence, a new security Framework has been designed for G-Hadoop. This new model is based on various security solutions like public key cryptography and the SSL (Secure Sockets Layer) protocol, or GSI (Globus Security Infrastructure).

Some basic concepts like proxy credentials, user session, and user instance, are applied in this security framework to provide the functionalities in Grid environment [90, 91]. This security framework introduced a single-sign on approach during user authentication and job submission process of the G-Hadoop. Also, this security approach protects the G-Hadoop system during threat environment i.e., traditional attacks, abusing and misusing. The designed security model is based on the concept of Globus Security Infrastructure (GSI) and use

SSL for communication between the master node and the CA (Certification Authority) server. GSI is based on single sign on process and use asymmetric cryptography to provide a secure communication. GSI is also known as standard for grid security and adopt different techniques to meet various security requirements. This includes authentication, integrity of the messages and delegation of the authority from one entity to another in a grid environment. The user can only log-in into the master node after providing his authentication in the form of user name and password and submit jobs to the cluster. SSL handshaking is used in the security framework to establish a secure connection between master and a slave node.

2.9.10 Distributed File System for Data-at-Rest Security for Hadoop-as-a-Service (2015)

In this paper the security of data at rest or stored in Hadoop when it is offered as a cloud servicehas been explained [100]. Also, the researchers have introduced a new distributed file system called SDFS for Hadoop. It has the capability to read/write secured input data on behalf of MapReduce applications by applying enterprise's controlled distributed keys [100]. It provides the same security as the secure cloud gateway using the same configuration parameters (e.g. type of encryption, strength, secret sharing scheme, etc.) through the common configuration file of these two components provide during the installation of the service. In this approach, MapReduce uses SDFS file system instead of traditional file system i.e. HDFS to read/write confidential data by usingsdfs:/ as a prefix in the URL of the files or directories. In result, SDFS fetched the keys responsible for the security function from a keycaching server in the cloud service provider. Using those keys, SDFS collect data from the share, decrypts and checks authenticity during read operation and encrypts, signing and secret-share splitting during a writeoperation before sending to the MapReduce applications. The key-caching server maintains synchronization with the master key management server of the enterprises premises while obtaining and update the keys.

The SDFS system uses a configurable security design which is a combination of secret sharing and information dispersal to providing security along with high data availability. During performance analysis of SDFS, the authors found that SDFS introduces the minimal computational overhead as compared to default HDFS. Also, with the use of secret sharing scheme, we get the security and data availability at low computational cost as compared to default HDFS.

2.9.11 Security scheme for Hadoop distributed file system suing elliptic curve cryptography (2015)

In this paper [65], to protect sensitive data stored in HDFS against security threats like breach and impersonation attacks, a token based authentication scheme has been introduced. In this scheme, HDFS client authentication is done by the DataNode through block access token and acts as an additional layer of security along with the existing security i.e. symmetric key HDFS authentication. Also, use of ECC technology in the proposed scheme makes authentication keys anonymous and provided protection against external threats like security breaches or accidental exposures. This scheme adopts the hash chain of keys approach instead of a public key exchange approach which is a very common HDFS authentication protocol. Apart from providing protection to the sensitive HDFS data, it also improves the performance as compared topublic key-based HDFS authentication protocols.

2.9.12 Secure Data Transmission for Multi sharing in Big Data Storage (2016)

A method of privacy preserving security by using different mechanisms i.e. anonymity, multiple receiver and conditional sharing is explained in this paper [102]. In this approach, to get the maximum security, Advanced Encryption Standard (AES) with Message Digest (MD5) & Data Encryption Standard (DES) are used to encrypt the data and authentication of data has been done using the DSA. Also, security

and privacy preserving approaches are used for the big data processing in the proposed framework. In this approach, owner uploads the data in cloud storage and after encryption, data is stored in HDFS. Thereafter, the data is shared among the multiple receivers. Cipher text is used to hide the identity of the sender and receiver whereas Anonymization mechanism is used to hide information of a particular receiver. A mechanism based on user and their received data category called conditional sharing starts working after receiving the receiver details. And, if user's category is matched with receiver's data category, then receiver get authenticated and the transmission is started. Once conditional sharing is complete, receiver retrieves the cipher text. The big data is shared with the cloud only if, the result is secured. This proposed algorithm is verified for small data sets only.

2.9.13 Towards the Development of Best Data Security for Big Data (2017)

In this paper, author described about the big data and its security issues [103]. Also, he has described about the existing ways to improve the big data security like security hardening methodology with attributes relation graph, attribute selection methodology, content based access control model and a scalable multidimensional anonymization approach. The author of this paper [103], have proposed an intelligent security approach based on real time data collection and threat analytics to detect the threat before the security breach takes place.

2.9.14 Design and Implementation of HDFS Data Encryption Scheme using ARIA Algorithm on Hadoop (2017)

In this paper [55], the author presented an encryption scheme based on South Korea's ARIA encryption scheme to protect the HDFS data in Hadoop. The ARIA algorithm uses 128-bit block for data encryption. In this approach, data is divided into HDFS blocks which support encryption and decryption in Hadoop. The authors also introduced a scheme which supports the encryption of variable length

data (not necessary 128-bit data). ARIA based algorithm supports the encryption of variable length data and provides the same level of data security at cost of only 23 % performance degradation (during query processing) compared to AES algorithm. This proposed scheme can be used in various applications like word counting, sorting, k-Means, and hierarchical clustering. In addition, the researchers explained the future of ARIA based encryption scheme in real word applications like location-based services and financial information processing.

2.9.15 Time domain attribute based encryption for big data access control in cloud environment (2017)

In this paper [81], author proposed a new technique of encryption called as a time domain attribute based encryption to achieve end to end security of big data in cloud. The authors proposed a time domain attribute based access control (TDAAC) algorithm which shows how ciphertext policy attribute based encryption (CP-ABE) with time domain and access rights can strengthen the data security against security breach or deliberate unauthorized access to data stored in the cloud. In addition, the researchers presented a method based upon on-demand user/attribute revocation for data access permissions which enhanced the dynamic modification of access policies for stored data in cloud server. Also, the authors have planned to implement TDAAC algorithm on real cloud based big data systems and explored the time-domain access control scheme in standard model whichmay enhance the security of TDAAC during threat.

2.9.16 Chaos-Based Simultaneous Compression and Encryption for Hadoop (2017)

This paper [104] introduced aframework based on a masking pseudorandom key stream to increasethe encryption quality and provide robust encryption security & compression during read and write operation when integrated in HDFS. Also, the researchers have proposed a scheme for Hadoop using simultaneous compression and encryption to solve the implementation issues. The enhancement

consequently improves speed and efficiency of the algorithm. Likewise, a chaotic pseudorandom keystream generator controlled PWLM mode without decreasing its compression capability. In this, a secret key is generated by using a cyclic shift operation in PWLM. The proposed algorithm is very much compatible with Hadoop which provides a robust encryption security and compression during storage of data. Various experimental results concluded that the performance of the cluster in Hadoop gets reduced when compression and encryption operations are done separately because they need a significant volume of data for both the operations. This proposed algorithm is capable to do compression and encryption of data simultaneously during MapReduce which reduces the required data space with minimum network resources. The proposed algorithm has passededits security analysis test with a 99% confidence interval. Since, all NIST SP800-22 as says are successfully passed on ciphertext generated from the plaintext.

2.9.17 Data Encryption Based on AES and OTP (2018)

This research paper [105], has highlighted a method to improve the upload and download time with reduction of encrypted file size by integrating HDFS files with AES and OTP algorithms. They performed encryption and decryption by two different ways where first is using AES algorithm and second is using AES and OTP algorithms. The researchers chose cipher block chaining with the ECB mode of AES algorithm for handling HDFS blocks and OTP algorithm is used as a stream cipher which keeps length of the plaintext remains same.For decryption, a private key is required which is always in the custody of user. In this method, when user has requested to upload a file to HDFS, the application server generate a random key which is then divided into 2 keys for doing multi encryption and decryption by using AES and OTP algorithms. Moreover, the authors have compared the file encryption time among Generic HDFS, encrypted HDFS by AES and HDFS encrypted file by AES with OTP. The results show that the AES with OTP algorithm increased the encrypted file size that by 20% of

the original file. The researchers also executed parallel decryption processing in Map Task to improve performance.

2.9.18 Privacy preservation techniques in big data analytics (2018)

In this paper [45], authors described about the various privacy threats and preservation techniques and models along with their limitations. The authors also proposed a Data lake based method for privacy preservation of unstructured data. Data lake is a repository to store raw format of the data either structured or unstructured, coming from different sources. Apache Flume is used for data ingestion from different sources and for their processing; data is transformed to HIVE tables. Also, Hadoop MapReduce using machine learning or vertically distributed can be appliedto classify sensitive attributes of data whereas tokenization is use to map the vertically distributed data.

2.10 Major Findings

After a cautious and focused study of various methodologies/approaches on big data and Hadoop security, the following observations have been made:

- Hadoop stores the data on multiple nodes in a distributed manner while metadata and edit logs are stored on name nodes. The communication of data happens between client node and data node. Hence, multiple nodes are used in the Hadoop framework. The data is vulnerable to the hacks during the transmission, as it is not encrypted by default. Various communication protocols such as Remote Procedure Call (RPC), Transmission Control Protocol over Internet Protocol (TCP/IP), and Hypertext Transfer Protocol (HTTP) are used for internode communication. The available solutions for securing the data transmission include Kerberos, Simple Authentication and Security Layer (SASL) etc. Being the traditional approach for securing big data, these approaches are not sufficient enough to secure it.

- Data that is stored in fixed storage is known as data at rest or at storage level. Initially the stored data is prone to security attacks being not encrypted. Since, Hadoop works on the principal of spreading data across multiple nodes, consequently it is exposed to all insecure entry points. There are numerous methods available for data encryption in Hadoop. As Hadoop deals with large volume of data, it takes time in the encryption/decryption process. In order to maintain the performance, it is important to use an encryption technique that is fast enough to encrypt/decrypt. According to the studies the encrypted data increases in the size almost by one and half times of the original data so the file upload time also gets affected.
- Cloud providers need to design a cost effective infrastructure that understands customer's needs at all levels. To meet the requirements, it is needed to share the storage devices amongst the multiple users, which is known as multi-tendency. But sharing of resources results in security vulnerability. The attacker is able to get easy access to customer data if he is using the same physical device, if proper security measures are not implemented.
- Companies would never know if the data is being used by someone else or not, because they don't have direct control over their data. The lack of resources monitoring mechanism creates many security issues.
- Customers have to rely on the cloud providers using trust mechanism as an alternate they get transparent control over their data and cloud resources. Cloud providers build confidence over their customers by assuring them the provider's operations are certified in compliance with organizational safeguard and standards.
- The same security capability which are for "non big data" are needed for big data which provides solutions that address user authentication and access control, management, data masking and encryption.

2.11 Conclusion

Many organizations and corporations are using Hadoop because of its capability to store and process enormous data at high speeds. There is a need of strong security mechanism to secure the sensitive information in the Hadoop clusters. Many researches and studies began to overcome the issue of protecting information from the eyes of unwanted people. A study which introduced the user authentication which is based on network authentication protocol known as Kerberos. Apart from this, many other techniques have also been used to gain authentication. Hashing is the other example. In the hashing technique, user authentication is done for NameNode, and then it generates a hash function and compares it with the user. It employs a one-time pad algorithm in a simple authentication. Also, the need of transferring passwords between the servers is skipped.

To protect big data stored at HDFS, end-to-end cryptography and data encryption have been done to prohibit data disclosure in HDFS. There are drawbacks in the discussed methods which is easier and less expensive to implement. The attacker will gain all access to the data in the case of system violation. The existing methods such as password based encryption (PBE) are vulnerable to brute-force attacks. It is because of the fact that an incorrect key that is random guessed in the attack leads to invalid looking plain text message, and hence confirming the invalidity of the key. The attacker will exploit this loophole by continuously guessing random keys until a valid looking text message appears. Some other approach or methods must be explored to use big data in a secure and managed way.

It can be concluded that the techniques explored in the researches are not sufficient enough as huge volume of big data is now gradually involving everywhere in various fields. To identify importance of security in big data location, it should be explored more as more privacy and security approaches are involving. There is a need of

balance among data usage, privacy protection and data protection. This requires solutions to increase the security and reliability of a distributed system. More research on identified security issues of big data may secure big data in Hadoop environment.

CHAPTER 3

Proposed ABHE Algorithm for Big Data Security

3.1 Background

To serve billions of users worldwide, the internet is a global system of interconnected computer networks that use standard Internet Protocol Suite (TCP/IP). Internet is network that consists of millions of private, public, academic, business, and government networks, of local to global scope, that are linked by a broad system of electronic, wireless and optical networking technologies. In this inter connected environment, the need to protect sensitive data from unauthorized access has seen a drastic change in recent years, since internet is a network that is growing rapidly day by day.

Various commercial organizations store their confidential data to provide massive information to users using their services. Some of them store this data in a plaintext format and some employ password-based encryption (PBE) [107]. Since the information is stored in messaging centre and is visible to the network provider staff, the content can be easily modified and moulded. Therefore messaging is not an appropriate communication medium for secure communications, since a user's private data is prone to hacking.

The Hadoop, one of the recent trends in big data technology is used as a framework for the storage and processing. It is an open source distributed computing framework implemented in Java and consists of two modules that are, MapReduce and Hadoop Distributed File System (HDFS). Hadoop is usually used in a large cluster or a public cloud service such as Yahoo!, Facebook, Twitter, and Amazon. The scalability of Hadoop has been shown by the popularity of these applications, but it is designed without security for stored data.

The inherent security in Hadoop is simple file permission and access control mechanisms. So, encryption may prove to be the best solution for securing HDFS files that are stored in DataNodes and for transferring files among DataNodes while executing MapReduce jobs. By using cryptography in Hadoop, some targets for data protection like data confidentiality and data integrity can be achieved. There are two types of cryptography key: secret key cryptography which is also known as symmetric key cryptography and public key cryptography which is also known as asymmetric key cryptography [119]. Secret key cryptography schemes are generally categorized as stream ciphers e.g. RC4, password based encryption, and OTP algorithm or block ciphers e.g. AES, DES, 3DES, and BLOWFISH algorithms [120].

Encryption is mainly used to ensure secrecy. Encryption actually means secret writing which was initially used by ancient human desiring to keep and store secrets. Incident days, encryption was available only to Generals and Emperors, but today it is nearly used by everyone, every day, every time whenever a credit card transaction, data storage and node to node communication is done, phone call is made, secure website is used; encryption techniques are used. Robustness of an encryption algorithm depends on the key length [108]. However, the available encryption algorithms are considered to be secure, but given enough time and computing power, they will be vulnerable to brute-force attacks [109]. Also, the existing encryption mechanisms have vulnerability; that is, when decrypting with a wrongly guessed key, they yield an invalid looking plaintext message, while decrypting with the right key, they give a valid-looking plaintext message, confirming that the cipher-text message is correctly decrypted [109].

In the same row, the honey encryption has been proposed by Jules and Ristenpart [110]. It is a concept which addresses vulnerability discussed in the previous paragraph and makes the password based encryption (PBE) more difficult to be broken by brute-force.

Traditional encryption methods would show random text with no meaning at all when tried to decrypt with wrong key and hence confirming its invalidity. On the contrary, honey encryption shows plausible looking text even when the key is wrong so the attacker won't know if the guessed key is the right one. However, it is possible to brute force honey encryption if the attacker has crib that must match with it to confirm its legitimacy [111]. To overcome the same, there is a need to propose a new approach to solve the issue and focus on this vulnerability and making password based encryption more secure.

3.2 Cryptographic Technologies for Big Data

The evolution of big data comes with the challenges to secure massive, streaming and increasingly private data. Also, big data imposes to industries numerous data security threats. In addition, when public cloud is used as big data storage for analytics, the risk of breach may increase. Hence, the focus should be to protect private data from unauthorized access. To achieve this task, various sophisticated techniques are available across the industry and statistical result gives additional confidence to the user to migrate data and computations to the big data storage [101]. Various cryptography techniques which are available in the market are depending upon basic encryption process. In the chapter-2 (section 2.9), the researcher have highlighted some of the new research directions for cryptography techniques for securing big data summing up the work of various researchers and practitioners.

The available security techniques and approaches have some limitations like high storage and computation cost as well as complexity whereas some encryption techniques also suffer from brute force attack and crib matching [13, 55, 56]. In this thesis, researcher has tried to overcome these issues. The basis of the proposed research is Honey encryption and the aim is to overcome existing issues in Honey encryption. The proposed research work has designed and implemented an attribute based Honey encryption algorithm which is

an expansion of public key encryption that would allow the users to encrypt and decrypt messages based on user attributes. This will help to put attackers in the blacklist. The users who are whitelisted and the attributes provided by them are matched with the given attributes will be able to decrypt the encrypted message.

The researcher has also evaluated the performance of proposed encryption algorithm as well as compared the performance of three different encryption algorithms namely; AES, DES, and Blowfish [112, 113] with the proposed one. The criterion of performance evaluation includes encryption time, decryption time, encryption time throughput, decryption time throughput and power consumption. The following section briefly discusses about honey encryption.

3.3 Honey Encryption

Though vulnerable to brute force attacks, password-based-encryption (PBE) [107] is commonly used by many systems. The honey term in the information security terminology describes a false resource. For example, to attract attackers to probe and penetrate, honeyspot [115] is a false server. Honeyword [114] is a false username and password in the database. Once used for login, an intrusion is detected. Honey Encryption (HE) is a scheme to enhance the security of password based encryption. When an attacker attempts to access the encrypted data with the incorrect password, instead of rejecting their access, the HE algorithm generates an indistinguishable bogus data that closely resembles the actual data. It deters hackers by serving fake data for every incorrect guess of the key code. This is a unique approach to slow down the attackers by potentially burying the correct key. Therefore, it increases the complexity of password guessing as well as cracking process.

Honey encryption tricks the attacker by making him believe that guessed key is valid. There are many technologies that share the same

term honey. For example, honeywords [90] are passwords that are used as decoy and raises alarms generally if a login attempt is made using it. Honeypots [115], Honeynet [116], and Honeyfarm [117] are some examples of luring systems to fool the attacker. Honey encryption is related to Format-Preserving-Encryption (FPE) [118] and Format-Transforming Encryption (FTE) [119]. The plaintext message is same as the cipher-text message in the FPE whereas the plaintext message space is different from cipher-text message space in FTE. Honey encryption stores a message to a seed range in the seed space. Since both seed space and message space are different, the cipher-text message space is different from the message space.

Vinayak and Nahala [120] have applied the honey encryption mechanism in MANETs to safeguard ad hoc networks from the brute-force attack. Tyagi et al. [121] have used honey encryption technique to protect simplified text messages and credit card numbers. In those applications, most of the data are uniformly distributed message spaces. Hoyul Choi et al. [122] have worked on the issue of human error, typos. If a legitimate user mistakenly types wrong password, he would get different result than expected and may get confused. They have proposed schemes to effectively solve typos problem while providing message recovery security. Edwin Mok et al. [123] have proposed an eXtended Honey Encryption (XHE) scheme by adding an additional protection mechanism on the encrypted data. However, Honey encryption is still vulnerable to brute force attack if the attacker has access to the copy of some information regarding the message. Attacker can match that information with the decrypted result to validate if the guessed key is correct.

As it has already been discussed that the available research in the field of honey encryption security and privacy is not sufficient to maintain the security and privacy of the data because these studied have some limitations. It must be explored further to bring out new security methods. This persuades the researcher of the thesis to

develop in depth understanding of data security and privacy to solve issues related to honey encryption. While going through the literature about securing through Honey encryption the researcher has dug out new security encryption approach. The idea is that a sender encrypts data that can only be decrypted and read by the authentic recipient.

3.4 Proposed Encryption Algorithm

The proposed encryption algorithm is a more secured version of honey encryption. The encryption algorithm provides two tier securities so that it can overcome the limitations prevailing in existing encryption techniques. The proposed algorithm is termed as Attribute-Based Honey Encryption (ABHE). Its 128/256 bits encryption algorithm will perform two layer of encryption in order to enhance security and effectiveness.

In proposed algorithm, Cipher text Policy- Attribute Based Encryption (CP-ABE) [124, 125] has been used. A user's private-key is associated with an access policy over a set of attributes from the defined universe of attributes within the system. A user will be able to decrypt a cipher-text, if and only if his attributes satisfy the policy of the respective cipher-text. For instance, the universe of attributes is defined to be {A, B, C, D} and User-1 receives a key having attributes {A, B} and user 2 to attribute {D}. If a cipher-text is encrypted with respect to the policy (A∧C)∨D, then user 2 will be able to decrypt, while User 1 will not be able to decrypt. The whole idea of the proposed encryption technique has been depicted in figure-3.4.

Firstly, a set of attributes are chosen from the file to be encrypted; then a set of rules/policies are created for these attributes. On the basis of these rules, the given file is encrypted. Further, for more security the encrypted file is again protected by password. As this password is based on honey encryption, it creates a set of honey words. The encrypted file is passed on and may be received by different users.

Now according to the proposed algorithm, only the user having the desired set of <attributes, password> will be able to decrypt the data. If someone wants to decrypt the encrypted file, he/she will have to enter the correct password. If password does not match, the user will be treated as intruder and previously set honey words will be displayed to him. If the password matches, the genuine user has to enter the private key which has been already created while encrypting the file. Again, if private key does not match, the user will not be allowed to access the file. On matching, the user will be able to successfully decrypt the file. The overall process will provide better security for files. The algorithmic representation for the proposed algorithm ABHE has been given in figure-3.4.

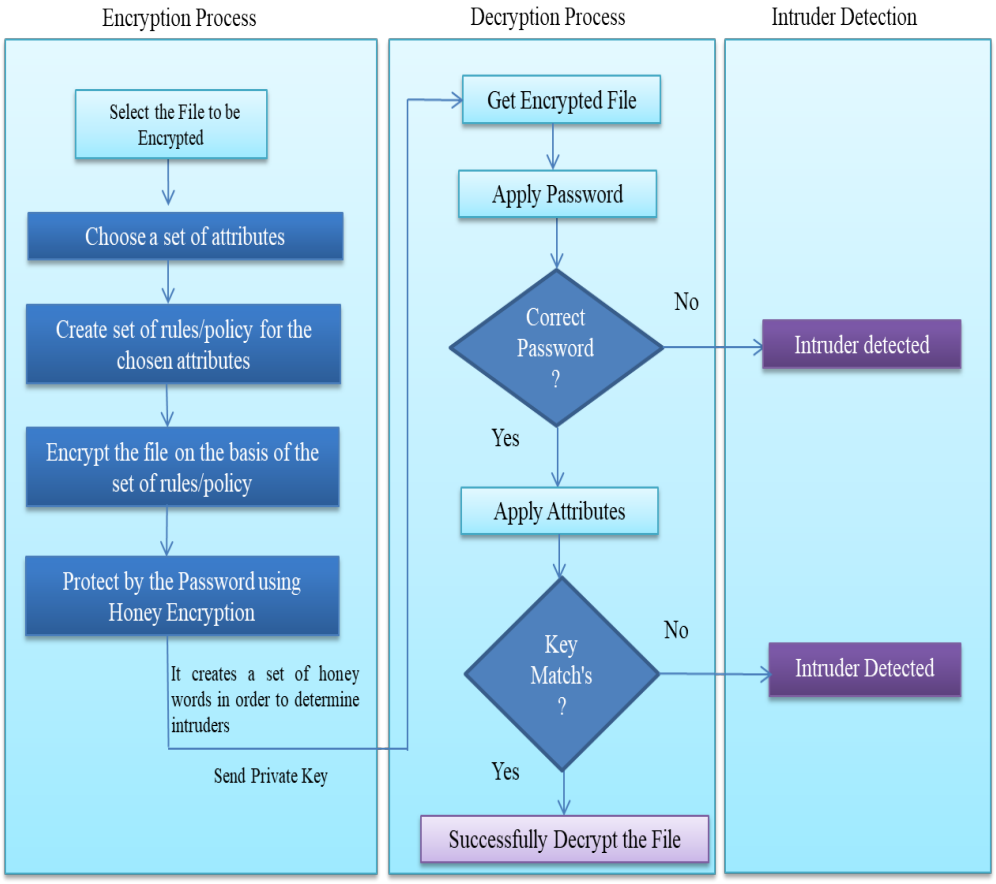

Figure-3.4: Data Flow Diagram for Encryption and Decryption

ABHE Algorithm for Data security

Input: Plain Text file
Output: Encrypted file
Step 01: **Generate Private Key**
Step 01.a: Set of attributes is specified that describe the key.
Step 01.b: Output private key 'P'
Step 02: **Encryption:**(The algorithm encrypts File 'F' with policy 'P' and outputs the cipher-text)
Step 02.a: Selects the file to be encrypted and set of attributes.
Step 02.b: Encrypt a file F using a set of attributes occurring in the policy 'P'
Step 03.c: Generate cipher-text CT
Step03: **Encrypted file (in step-02) is protected again by the password**
Step 04: **Generate honey words and present it to user.**
Step 05: **Decryption:**(Decryption algorithm gets as input an encrypted file which is protected by the password. Cipher-text CT is produced by the encrypted algorithm, an access policy 'P' under which CT was encrypted.)
Step 05.a: Input is encrypted file
Step 05.b: Enter the password; if password matches, the cipher text CT is decrypted, otherwise intruder is detected.
Step 05.c: User applies y number of attributes to compute private key
Step 05.d: If key matches, file is decrypted and output the corresponding original file 'F', otherwise it outputs NULL.

3.5 Simulation Results and Analysis

For analysing the performance of the ABHE algorithm, the same has been simulated. The simulation was performed on a laptop with Windows 10, 64bit, processor Pentium and CPU 1.90GHz with 2GB of RAM. Random text files of different sizes ranging from KBs to MBs (49KB, 59KB, 100KB, 247KB, 321KB, 1MB, 2MB, 5MB, 10MB and 20MB) were generated. Java 1.7.0_65 (64-Bit) has been used as the language of choice for implementation. Encryption time and throughput as well as decryption time and throughput of ABHE are presented in the last column of table-3.5(a) and table-3.5(b) respectively. Performance of ABHE has been compared with that of the three previously available encryption algorithms namely; AES, DES, and Blowfish [112, 113] with different sizes of text files. The criteria of the comparison have been taken as encryption time, decryption time, throughput of encryption and throughput of decryption. The performance matrices for the purpose have been discussed as;

3.5.1 Encryption Time: Total time taken to produce a cipher-text from plain-text is known as the encryption time [112, 113].

3.5.2 Decryption Time: The total time taken to produce the plain-text from cipher-text is known as decryption time [112, 113].

3.5.3 Throughput of Encryption: The speed of encryption depends upon the throughput of the encryption scheme, as it defines the speed of encryption. In the case of encryption scheme throughput is calculated as the summation of sizes of all plain texts in k bytes divided by total encryption time [112, 113].

$$\text{Throughput of Encryption} = \frac{\text{Total Plain Text in Kbytes}}{\text{Total Encryption Time}}$$

3.5.4 Throughput of Decryption: The speed of decryption depends upon the throughput of the decryption scheme, as it defines the speed of decryption. In the case of decryption scheme throughput is calculated as the summation of sizes of all cipher-text text in k bytes divided by total decryption time [112, 113].

$$\text{Throughput of Decryption} = \frac{\text{Total Cipher-Text in Kbytes}}{\text{Total Encryption Time}}$$

3.5.5 Power Consumption: Power consumption decreases with decrease in throughput of encryption as well as decryption [126].

Comparison between Proposed Scheme and Existing Schemes

The AES/DES/Blowfish and proposed algorithms have been compared on the basis of their encryption-decryption time and throughput with different file sizes (49KB, 59KB, 100KB, 247KB, 321KB, 1MB, 2MB, 5MB, 10MB and 20MB) as shown in table-3.5(a) and table-3.5(b). The values for each criterion was logged and graphically plotted to represent the results as shown in figure-3.5(a), figure-3.5(b), figure-3.5(c), and figure-3.5(d).

Figure-3.5(a) shows the time taken (in msec) during the encryption process by different algorithms i.e. AES, DES, Blowfish and Proposed Algorithm (ABHE). From the figure-3.5(a), it is clear that the proposed algorithm is taking less time as compared to other existing algorithms. Figure-3.5(b) is showing the comparison among encryption throughputs of different algorithms i.e. AES, DES, Blowfish and Proposed ABHE Algorithm. Figure-3.5(c) shows the comparison of decryption time of AES, DES, Blowfish and Proposed Algorithm (ABHE). Figure-3.5(d) shows the comparison of decryption throughput of AES, DES, Blowfish and Proposed Algorithm (ABHE) in normal system environment.

Size of Text File	AES	DES	BLOWFISH	Proposed Algorithm(ABHE)
49Kbytes	56	29	36	39
59Kbytes	38	33	36	34
100Kbytes	90	49	37	34
247Kbytes	112	47	45	39
321Kbytes	164	82	45	56
1 Mbytes	80	136.2	133.2	62
2 Mbytes	154.7	269.6	192.6	203
5 Mbytes	376.1	665.4	373.6	230
10 Mbytes	683.7	1236.2	702.5	591
20 Mbytes	1350.5	2356.5	1355.2	1202
Encryption Time (msec)	**3105**	**4903.9**	**2956.1**	**2490**
Throughput (Kbytes/msec)	**12.78**	**8.09**	**13.42**	**15.93**

Table-3.5(a): Encryption Time for Different Size of Text Files

The proposed ABHE scheme has shown considerable improvements over the previously existing schemes namely AES, DES and BLOWFISH. In almost every case it has shown lesser encryption

and decryption time. As a result encryption throughput and decryption throughput is increased which can be clearly depicted from table-3.5(a) and table-3.5(b).

Size of Text File	AES	DES	BLOWFISH	Proposed Algorithm (ABHE)
49Kbytes	63	50	38	45
59Kbytes	58	42	26	56
100Kbytes	60	57	52	55
247Kbytes	76	72	66	59
321Kbytes	149	74	92	60
1 Mbytes	118.9	144.6	50.2	76
2 Mbytes	197.6	269.6	126.3	99
5 Mbytes	457.7	690.9	210.7	154
10 Mbytes	897.5	1294.6	575.4	243
20 Mbytes	1844.5	2744.2	1025.5	615
Decryption Time (msec)	3922.2	5438.9	2262.1	1462
Throughput (Kbytes/msec)	10.11	7.29	17.54	27.14

Table-3.5(b): Decryption Time for Different Size of Text Files

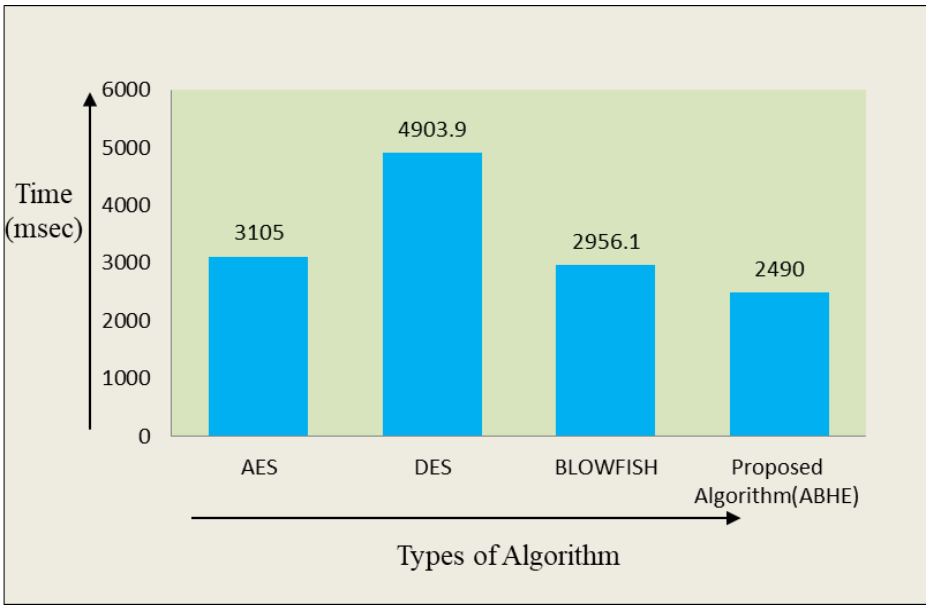

Figure-3.5(a): Encryption Time of AES, DES, Blowfish and Proposed ABHE Algorithm

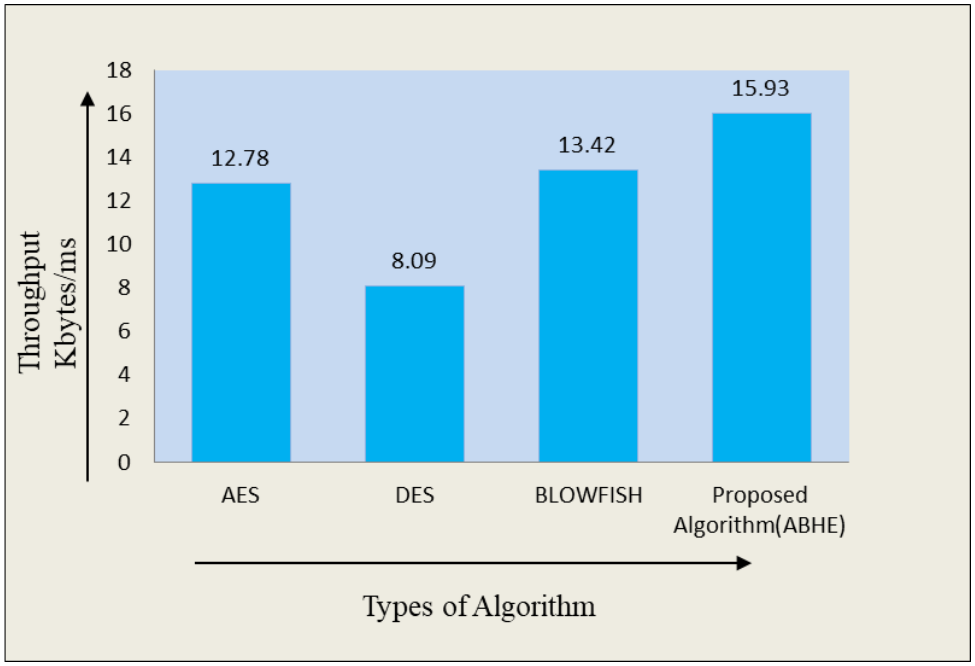

Figure-3.5(b): Encryption Throughput (in Kbytes/msec) of AES, DES, Blowfish and Proposed ABHE Algorithm

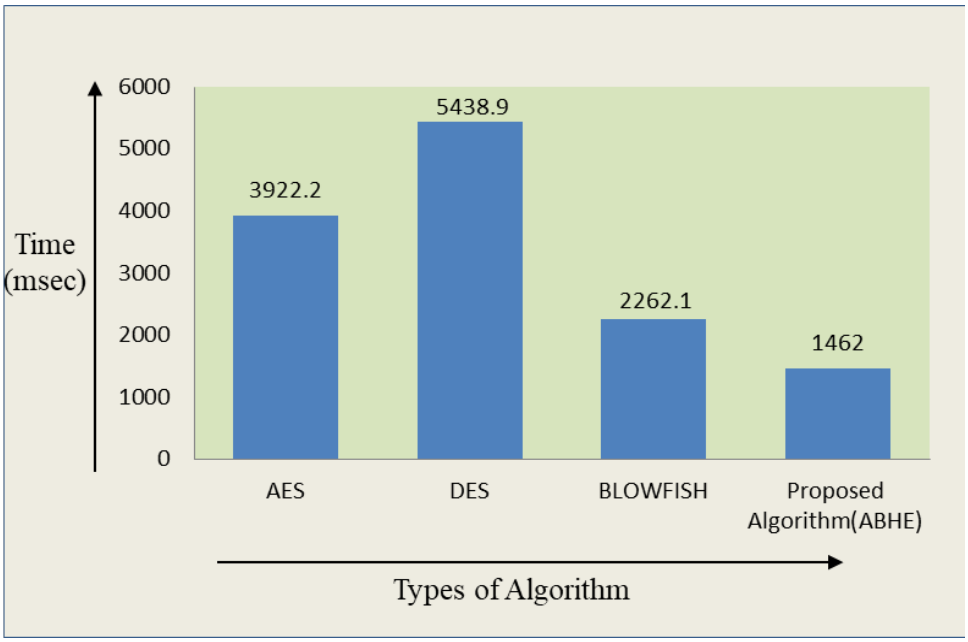

Figure-3.5(c): Decryption Time (in msec) of AES, DES, Blowfish and Proposed ABHE Algorithm

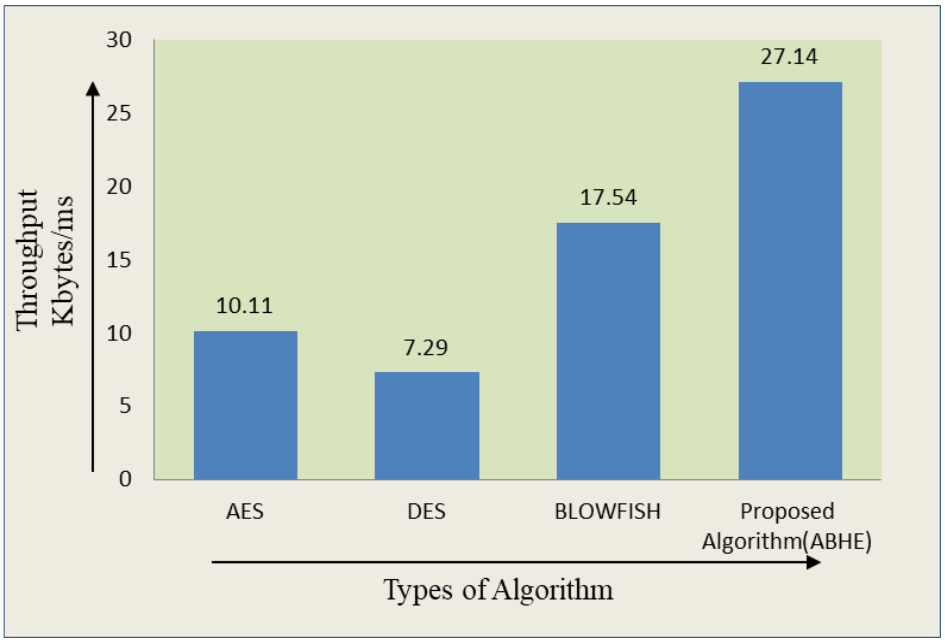

Figure-3.5(d): Decryption Throughput Kbytes/ms of AES, DES, Blowfish and Proposed ABHE Algorithm

Results of simulation from table-3.5(a) and table-3.5(b), it is clear that:

- The proposed algorithm ABHE is taking less time to encrypt and decrypt text files than the AES, DES and Blowfish.
- The throughput of ABHE is very high as compared to the AES, DES and Blowfish algorithms.
- As the throughput increases, the power consumption decreases hence the power consumption of ABHE is low than that of the AES, DES and Blowfish.

3.6 Significance of the Proposed Algorithm

The attributes selection scheme in the proposed algorithm is quite flexible. There is no restriction on the number of attributes to be taken in the proposed algorithm and authority can add any attributes into the proposed algorithm. In addition, it has already been seen in the previous section that the ABHE performs better than available encryption techniques including AES, DES and Blowfish. The significance of ABHE has been summarized as:

- It will help to improve various features during communication including security, efficiency over conventional encryption.
- It will provide dual layer of security during storage and motion of data by imposing double access control. Firstly the file is encrypted by the ciphertext policy based attribute encryption and furthermore it is protected by password based encryption i.e. honey encryption.
- The access policy in our proposed algorithm is a propositional formula which is quite simple and easy to be computed. The policy is generated from the attributes using operations performed in random order in order to make it more secure.
- It is easy to implement the ABHE algorithm but at the same time it is very difficult for intruder to revert it back.
- It will generate cryptography key which is faster and more efficient.

Undoubtedly, the proposed algorithm will help to provide strong access control because in the big data context, access to data requires a

strong access control system so that any malicious party must be denied from getting access to storage servers. With the proposed mechanism, only the node which possesses sufficient administrative rights could have the possibility to manage and process any content.

3.7 Conclusion

Encryption is a process wherein a message is decoded using a key and who so ever has that key can only read that coded message. It is used to protect the message from an unauthorized person. Encryption of data ensure that only an authorize person who has an authorize key can decrypt the data and read message. Data encryption algorithm has always been a key player in modern information applications and simultaneously it also plays an important role in network security. While evaluating security of a network, the major parameters including execution time, throughput and power consumption play a key role. Though there exist numerous encryption algorithms which claim to be secure as well as robust. During literature survey, it has been found that the existing algorithms come out to be weaker against brute force attacks.

The idea of the proposed ABHE algorithm is to overcome this drawback of the existing algorithms without adding complexity in the process of communication. To support the claim, performance of proposed ABHE algorithm has been evaluated. Encryption-decryption time, encryption-decryption throughput and power consumption of the proposed ABHE algorithm has been compared with the existing algorithms in the area including Blowfish, AES, and DES with the help of simulation. The simulation results conclude that the proposed ABHE algorithm has better performance than the existing algorithms.

CHAPTER 4

IMPLEMENTATION OF THE PROPOSED ENCRYPTION ALGORITHM

4.1 Background

In this digital era, the most challenging task for any individual is to store data securely. The big data is handled through Hadoop. The ability to provide reliable, flexible, demanding low cost services of huge data is through HDFS storage. The continuous growth of huge data and the applications used by Hadoop tremendously increase the possibility of malicious attacks. In this scenario, it has become important to secure the data related to an organization or individual stored in HDFS storage. Security is at top priority for several organizations. To secure data stored at HDFS, researchers and practitioners are paying considerable attention to provide safety for user's software resources, storage, and massive computing.

Big data security and privacy is the most significant issue for the Hadoop storage. Day by day large amount of data are uploaded through digital applications which require lots of storage space, computing resources and active system performance. On processing, big data identifies the risks through data analytics that increase the chances of attacks. The exponential use of smart phones, social networking sites, downloaded app, web sensor are generating huge amount of data and thus leads several issues in big data storage customization, security, cost-effectiveness, smooth performance, vendor lock-in, and compliance; Each issue has its own importance in Hadoop. In this thesis, the researcher is mainly focusing on security and privacy issues of big data storage.

When using the Hadoop storage service, users have no need to store data locally and carry along all the time. Instead, data is kept on

Hadoop storage server so that users can access anytime and anywhere. Therefore, the Hadoop storage server is responsible for all the related works, from hardware allocation to data security protection. In terms of users, with a device connected to the internet like mobile, computer, they are able to get their data in hand. The service is especially attractive for the one who has little knowledge about data security.The Hadoop is one of the recent trends in technology which is used as a framework for the big data storage. It is an open source implementation of the framework based on java. Hadoop is used in a large clusters or a public cloud service such as Facebook, Yahoo, Twitter and Amazon, it is considered as a standard distribution parallel processing system [106]. The scalability of Hadoop has been shown by the popularity of these applications, but it is designed without security for stored HDFS data.

As discussed in previous chapter-2 (section 2.9), there are numerous methods and approaches to security for stored HDFS files explored by the various researchers and practitioners. Here, encryption may prove to be one of the best solutions for securing data in HDFS that is stored in DataNodes. Also, for transferring files from one DataNode to another DataNode while executing MapReduce jobs. By using encryption techniques, confidentiality and integrity of data can be achieved in Hadoop environment. Studies were executed utilizing different encryption algorithms. The outcomes showed that file sizes expanded to one and half of the original file. In addition, file uploading and downloading time also expanded. The main purpose of this research is to address the big data security issue in Hadoop. In order to secure big data on Hadoop, the researcher has proposed a new algorithm ABHE in chapter-3. It has been shown considerable improvements in terms of encryption time, decryption time, encryption throughput, decryption throughput etc. in comparison to previously available algorithms including AES, DES and BLOWFISH [112, 113]. In the present chapter, the proposed algorithm is integrated with Hadoop environment. Performance of proposed encryption technique

has been compared with that of the two previously available encryption algorithms namely; AES [105] and AES with OTP [105] with different sizes of text files ranging from MBs to GBs (64MB, 128MB, 256MB, 512MB and 1GB). Encryption-decryption times in Hadoop are usually influenced by input data size, block size (Hadoop distributes data to data nodes in blocks), cluster size. It has been seen that by using the proposed ABHE algorithm the uploading-downloading time has been improved as the file size of original file has almost been same.

4.2 Data Classification Based on its Sensitivity

On the basis of sensitivity of data and access limitations, data can be divided into four major categories i.e. public data, private data, confidential data and restricted data. This categorization helps to determine how to transfer data in and out of the Hadoop cluster and identify the schema for its protection and processing.

Public Data: This type of data is publicly available and can be used by anybody from anywhere as per the requirements [127]. Therefore, there is no need to restrict its access and no need to secure it. This data is easily available on the Internet but it is stored on the Hadoop cluster for faster fetching. Weather report about different cities of a world is a best suited example for this category.

Private Data: This type of data is meant for specific users and not related to common public. Such data may not have any sensitive details, but it be kept restricted since it may affect the business of a particular firm [127]. An example of private data is a dataset purchased by the company from external sources. It has a limited access and control by someone.

Confidential Data: This type of datasets contains the elements which should be kept confidential [127]. An example would be the employee's personnel information such as an email address, a phone

number, etc. Access to this dataset may be restricted and needs to be encrypted or masked at sender as well as recipient end.

Restricted Data: This type of dataset contains data that should be meant for a particular user like head of a company etc. or approved limited users [127]. A dataset containing financial information of customers or health records fall into this category. Access to these datasets should be strictly restricted and provided with a secret key. These data is also need to be encrypted so that only authorize user may read it.

4.3 Data Centric Security

Data centric security is an approach that emphasizes on the security of data itself rather than the security of networks, servers or applications etc. [128]. Data can be secured by two different approaches; by controlling its access and second by applying a cryptographic [128] technique over it. First method is easy and less expensive to implement than the second. But if the system breach happens, an attacker may get an access to all the data and may harm it. Therefore, implementing the encryption technique on all the data while in storage is always giving an additional protection to it and even if there is a system breach, the data itself remains protected.

After configured, data read and write to HDFS directories are encrypted and decrypted without changing in the application code of the user. This provides an end-to-end protection, which means that data can only be encrypted and decrypted by the HDFS client. This fulfils the two major requirements for encryption: Encryption at rest (data during store) as well as in traffic or during move (for example, when data is traveling over the network).

4.4 Cryptography Technique

Cryptography is the branch of information security which deals with the protection of information by transforming the plaintext (ordinary text) into cipher text and then back again in the original form of text (known as decryption). In symmetric cryptographic technique, the encryption and decryption keys are similar, i.e. encryption key can be calculated from the decryption key and vice versa but in the asymmetric cryptographic technique, these keys are independent but correlated. The encryption key is called a public key and the decryption key is known as the private key which is to be kept secret. These keys work as a key pair. In today's world, privacy and security are the major concern for the technologies including e-commerce, personal information over the Internet, confidential Govt. reports etc. Therefore, to solve the security issue cryptographic technique is a very useful as well as a very popular tool for an authorized user.

Cryptography can be applied at different levels in a traditional data management software and hardware stack. It comes with different advantages and disadvantages. Encryption at application level is the safest and most flexible approach among all whereas encryption at file system level is easy to implement. Encryption at database level has similar properties like application level but has some performance issues (i.e. indexes cannot be encrypted). Furthermore, encryption at disk level has high performance and easy to implement but quite inflexible and protects against physical theft only. Encryption at HDFS level fits between database level and the file-level encryption in this stack. HDFS cryptography provides good performance and existing Hadoop applications can be transparently run on encrypted data. Cryptography at HDFS level also prevents attacks at the file system level and below (so-called "operating system-level attacks"). The operating system and disk only interact with encrypted bytes, since the data is already encrypted by HDFS. The next section discusses about

the implementation of the proposed cryptographic ABHE algorithm in Hadoop environment.

4.5 Implementation

To find out its suitability and applicability for large data, the proposed ABHE algorithm is integrated with Hadoop. The performance is evaluated by encrypting-decrypting different sizes of files (increasing in gigabytes). The performance of the same is also compared with the existing algorithms including AES and AES with OTP.

4.5.1 Implementation Environment

All implementations and experiments are based on Hadoop cluster. The Hadoop cluster consists of one host that runs on the laptop with Intel core i3-2330M processor 2.20GHz with Turbo Boost up to 2.93GHz and 4GB RAM. One of the hosts is tagged as NameNode and the other one is used as DataNode. NameNode plays a role of centre point in cluster and all information about stored data is saved on it. For simplicity, there is only one NameNode and one DataNode in a Hadoop cluster. DataNode provides the physical space for storing data.

The operating system of the host is Linux with CentOS 6.4, Ubuntu-14.04 (GNU/ Linux 3.13.0-24-generic x86-64). On top of the operating system is Hadoop with version 2.7.3. Single node architecture of Hadoop is used in which all components of the Hadoop are on the same node. Implementation and stand-alone applications are written in Java. As Hadoop requires JDK for its working so Java Open Java Development Kit (JDK) is installed on the system using the <*apt-get*> command. The running component of the Hadoop can be checked using the <*jps*> command. HDFS distribution process is meant to avoid outages, provide back up and redundancy of the services and make them highly available.

4.5.2 Implementation Setup Process

In order to install Hadoop, Java needs to be installed on each host. Hadoop is downloaded and installed from its official site. Before running the script, some configuration needs to be done. The three main configuration files are conf/core-site.xml, conf/hdfs-site.xml, conf/mapred-site.xml is used to specify the address of NameNode. The process of the implementation is given in flow chart as shown in figure-4.5.2(a).

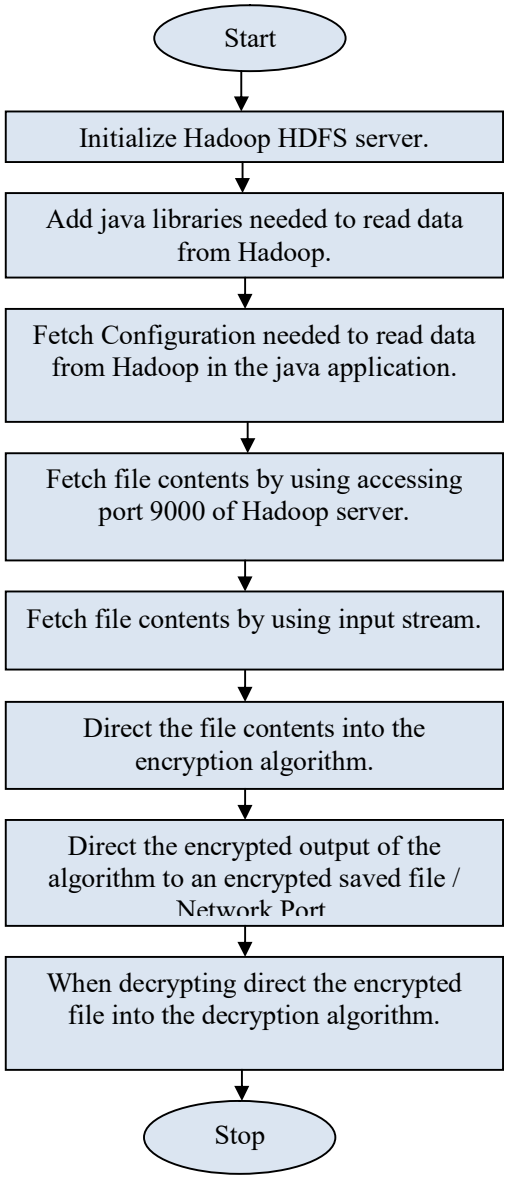

Figure-4.5.2(a): Flow of Process for Integration of Proposed ABHE Algorithm with Hadoop

In this process firstly the Hadoop HDFS server is installed then, all the Java libraries required to read the data from Hadoop are added. Further, the configuration required to read data from Hadoop in Java application is taken. Now file content is fetched by accessing port 9000 of Hadoop server and content of file is read by using input stream. Then the file content is fed as input to the proposed ABHE algorithm. After the encryption process the encrypted output generated from the proposed algorithm is directed to save file/network port. For decryption process the encrypted file is given as input to the proposed decryption algorithm. For making this integration more clear, the researcher has presented in detail the encryption-decryption process of the proposed ABHE algorithm in the following subsections.

Encrypting (Writing) Files on HDFS

Encryption is performed while writing the file on Hadoop so that the stored data can be saved from various attacks. This process involves a number of steps which has been shown as follows:

1. HDFS client interacts with NameNode by calling the create() function on Distributed file system (DFS).
2. DFS sends a request to NameNode to create a new file.
3. NameNodes provide address of the DataNode i.e. Slave which is based on the availability of space and capacity in DataNode on which HDFS client is going to start writing encrypted data.
4. The HDFS client starts entering the attributes to encrypt the file. After that for more security, it applies the password which is based on Honey encryption. Now the HDFS client starts writing data though FSDataOutputStream to specific slave for a specified block.
5. The slave starts copying the block to another slave when HDFS client finished writing the blocks.
6. During the block copying and replication process, metadata of the file is updated in the NameNode by Datanode. (DataNode provides the periodically heartbeat signal to the NameNode).

7. After the successful completion of write operation DataNode send the acknowledgement to HDFS client through DFS.
8. After that HDFS client closed the process.
9. Write operation is closed after receiving the acknowledgement from HDFS client.

The complete operation (with above steps i.e. 1, 2, 3...) is explained in figure-4.5.2(b). Steps 1, 2, 5, 6, 7, 8 and 9 have already been discussed in chapter-2 while discussing about file writing (without encryption) in Hadoop.

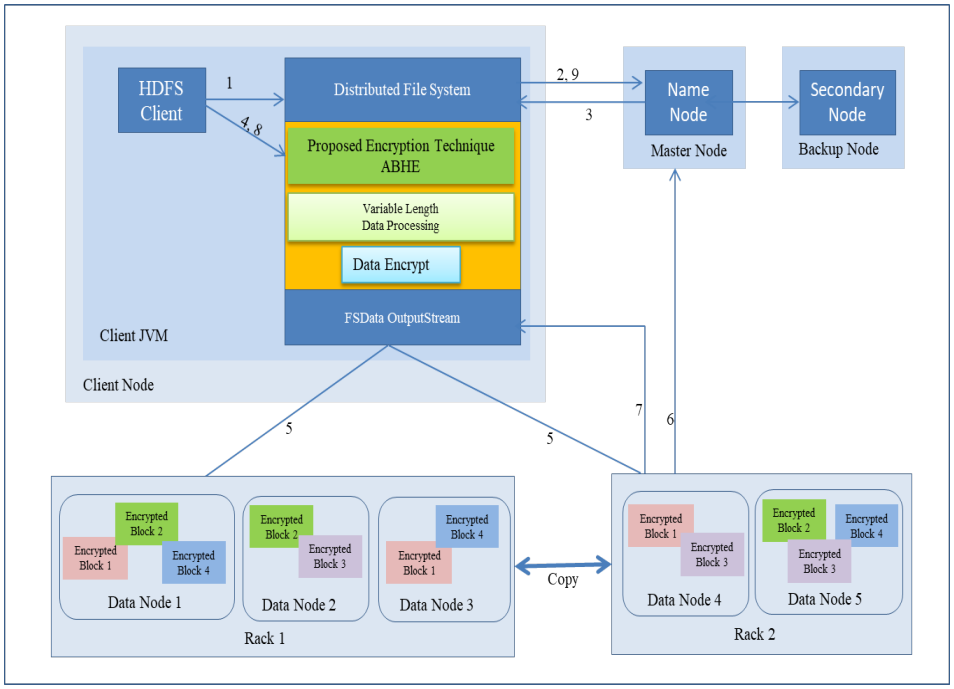

Figure-4.5.2(b): Writing a File with Encryption in HDFS

Decrypting (Read) Files from HDFS with Proposed Algorithm

With our proposed cryptographic scheme, whenever a node will try to read a file on HDFS it will first has to decrypt the file. Then only it will be allowed to perform reads operation. This has been done in the proposed approach to filter out the intruders or unauthorized access. Following is the step by step process on how Read operation is performed in HDFS with the proposed approach:

1. First of all HDFS client interacts with NameNode by calling the read() function on Distributed File System (DFS).
2. DFS sends a request to NameNode for reading a file.
3. NameNode provides address of the DataNode i.e. Slave on which HDFS client will start reading the data.
4. For HDFS client to start reading data through FSDataInputStream from specified slave and from a specified block, firstly it has to enter the correct password. If password does not match, the user will be treated as an intruder. If the password matches, the genuine user has to enter the private key which has been already created while encrypting the file. Again, if private key does not match, the user will not be allowed to access the file. On matching, the user will be able to successfully decrypt the file.
5. After a successful completion of read operation, HDFS client terminates read operation.
6. Read operation is closed after receiving the acknowledgement from HDFS client.

As it has been shown in step 4, the proposed approach provides dual layer security to the data stored in HDFS. Other steps except step 4 have already been discussed in chapter-2. These are included here to make the description more clear. Step wise demonstration of decryption operation is shown in figure-4.5.2(c).

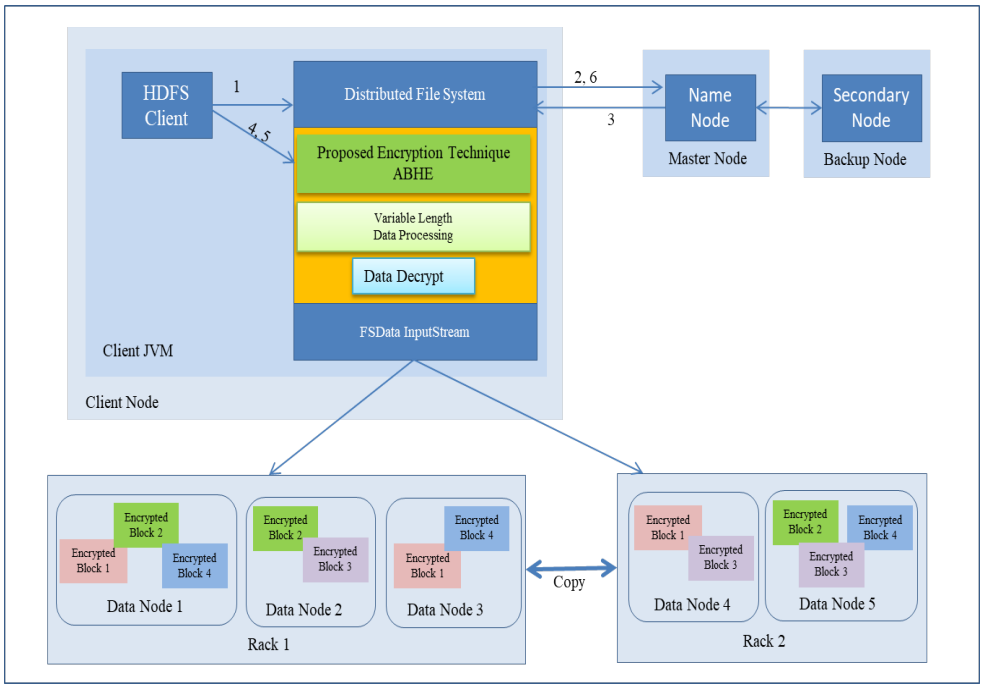

Figure-4.5.2(c): Reading a File with Decryption in HDFS

4.6 Experimental Results and Analysis

For analysing the performance of the ABHE algorithm in Hadoop environment, the same has been simulated with random text files of different sizes ranging from MB to GB (64MB, 128MB, 512MB, 256MB, and 1GB). The working of proposed algorithm has been demonstrated with screen shots as shown in figure-4.6(a) and figure-4.6(b). Performance of ABHE has been compared with that of the two previously available encryption algorithms namely; AES and AES with OTP with different sizes of text files. The criteria of the comparison have been taken as encryption time, decryption time, throughput of encryption and throughput of decryption.

As shown in table-4.6(a), it took 12.9751 minutes for the encrypted HDFS using AES algorithm, whereas it took 11.2511 minutes for the encrypted HDFS using AES with OTP algorithm. On the other hand the proposed approach took only 6.08 minutes to encrypt 1GB file in HDFS. This clearly shows the better performance of the proposed algorithm over the previously available cryptographic algorithms. To

make claim more concrete, the researcher has compared proposed approach with the other two (AES and AES-OTP) on the basis of size of the files after encryption. The same is shown in table-4.6(b). Here, with AES, file of size 1GB has become 1.5 GB after encryption and with AES-OTP the size has become 1.19 GB. On the other hand, with the proposed approach 1GB file was almost 1GB even after encryption.

Figure-4.6(a): Working of the Proposed ABHE encryption Algorithm when Attribute and Password are entered for 64 MB file size. (For both encryption and decryption)

Figure-4.6(b): Woking of Proposed ABHE Encryption Algorithm when Right and Wrong Password are entered for 64 MB File Size.

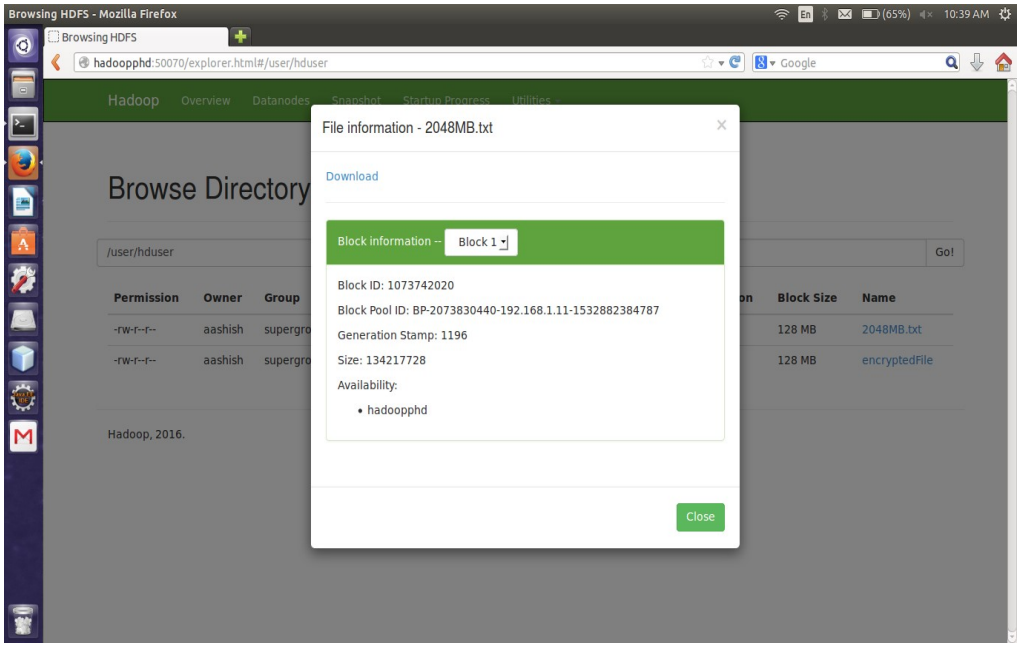

Figure-4.6(c): Size Block one before Encryption

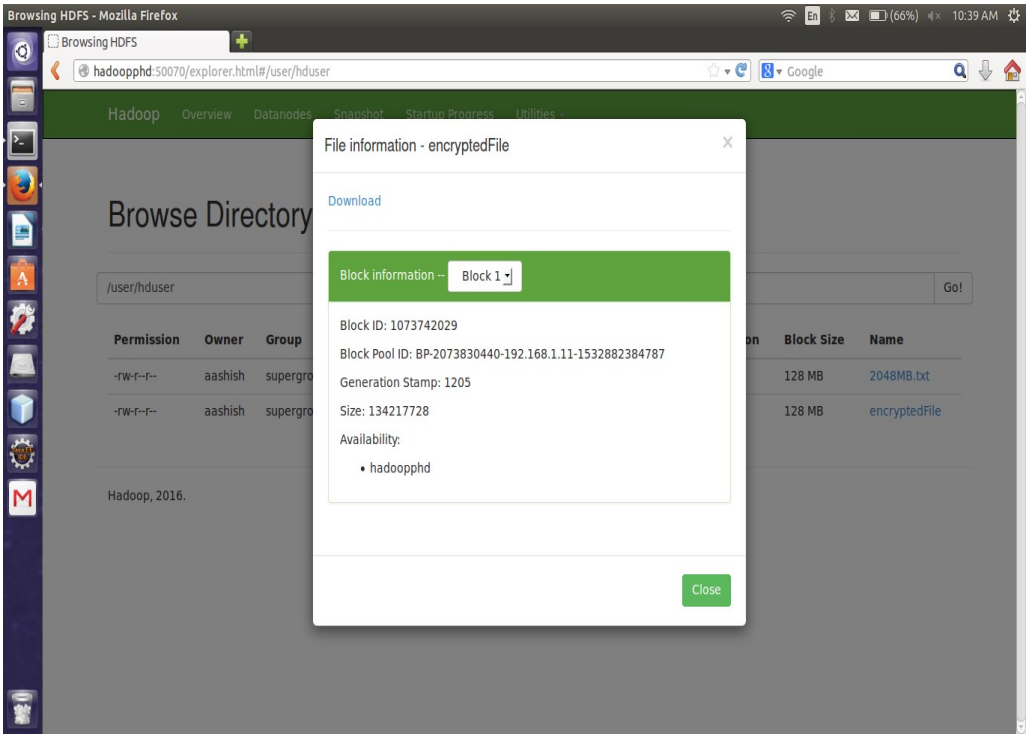

Figure-4.6(d): Size of Block one after Encryption

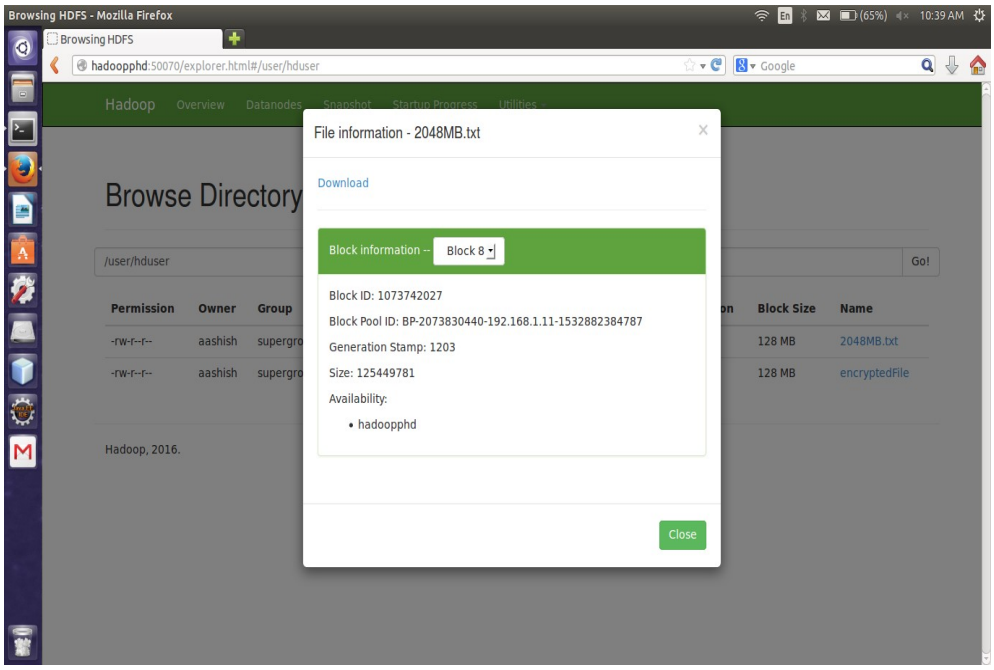

Figure-4.6(e): Size of Block eight before Encryption

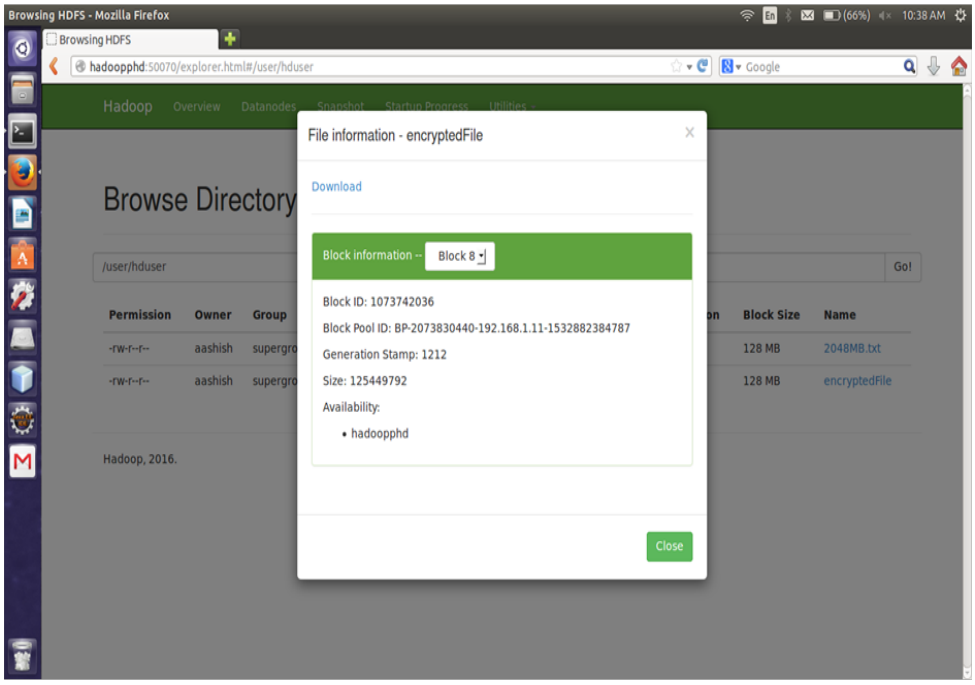

Figure-4.6(f): Size of Block eight afterEncryption

When using the proposed approach for decryption of 1GB file on Mapper job, it took 6.73 minutes. On the other hand the existing

algorithms i.e. AES and AES with OTP respectively to 14.0841 and 12.2115 respectively for decrypting 1 GB file as shown in table-4.6(c).

File Size (MB)	AES Algorithm (Minutes)	AES with OTP Algorithm (Minutes)	Proposed ABHE Algorithm (Minutes)
64	0.8704	0.7311	0.026
128	1.8216	1.3820	0.168
256	2.7396	2.5484	0.45
512	6.6682	4.8780	1.35
1024	12.9751	11.2511	6.08
Total Decryption Time	25.0749	20.7906	8.074
Throughput of Encryption (MB/Minutes)	79.12	95.42	245.72

Table-4.6(a): File Encryption Performance Comparison among AES and AES with OTP Algorithms and the Proposed ABHE Algorithm

File Size (MB)	AES Algorithm (MB)	AES with OTP Algorithm (MB)	Proposed ABHE Algorithm (MB)
64	96.0	74.7	64
128	192.8	149.3	128
256	384.0	298.7	256
512	768.0	597.3	512
1024	1536	1228.3	1024

Table-4.6(b): Data Size Comparison among AES and AES with OTP algorithmsand the Proposed ABHE Algorithm

File Size (MB)	AES Algorithm (Minutes)	AES with OTP Algorithm (Minutes)	Proposed ABHE Algorithm (Minutes)
64	1.3056	1.0950	0.03065
128	2.1859	1.6560	0.168
256	2.8641	2.6554	0.45
512	8.9494	6.5361	1.35
1024	14.0841	12.2115	6.73
Total Encryption Time	29.3891	24.154	8.72865
Throughput of Encryption (MB/Minutes)	67.50	82.13	227.29

Table-4.6(c): File Decryption Performance Comparison among AES and AES with OTP Algorithm and the Proposed ABHE Algorithm

The values for each criterion was logged and graphically plotted to represent the results as shown in figure-4.6(g) and figure-4.6(h). Figure-4.6(g) and figure-4.6(h) show the comparative time taken (in minutes) during the encryption and decryption process by different algorithms i.e. AES, AES with OTP and Proposed Algorithm (ABHE). From both the figures, it is clear that the proposed algorithm is taking less time for encryption and decryption as compared to other existing algorithms in Hadoop environment.

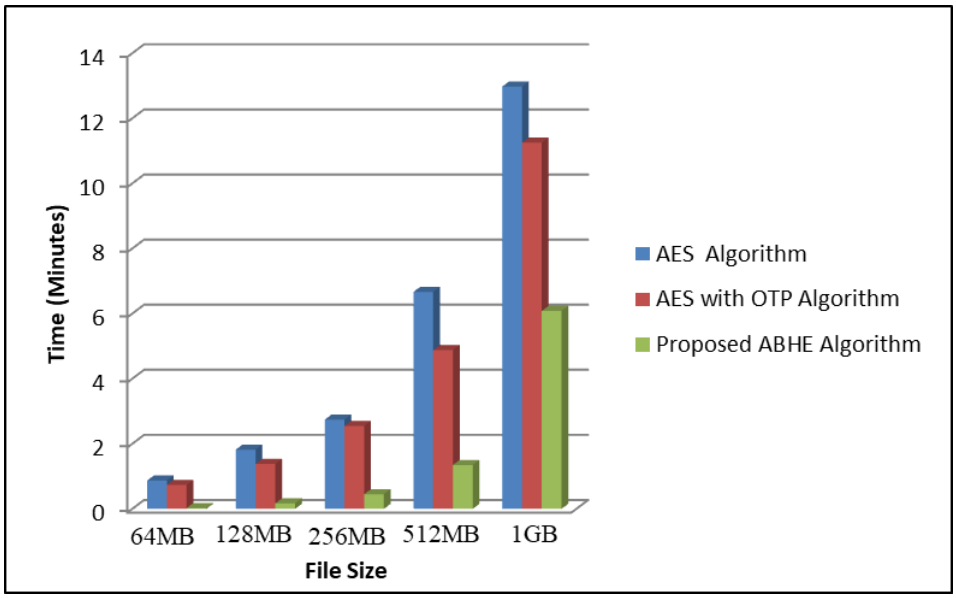

Figure-4.6(g): Encryption Time (minutes) of AES, AES with OTP and Proposed ABHE Algorithm

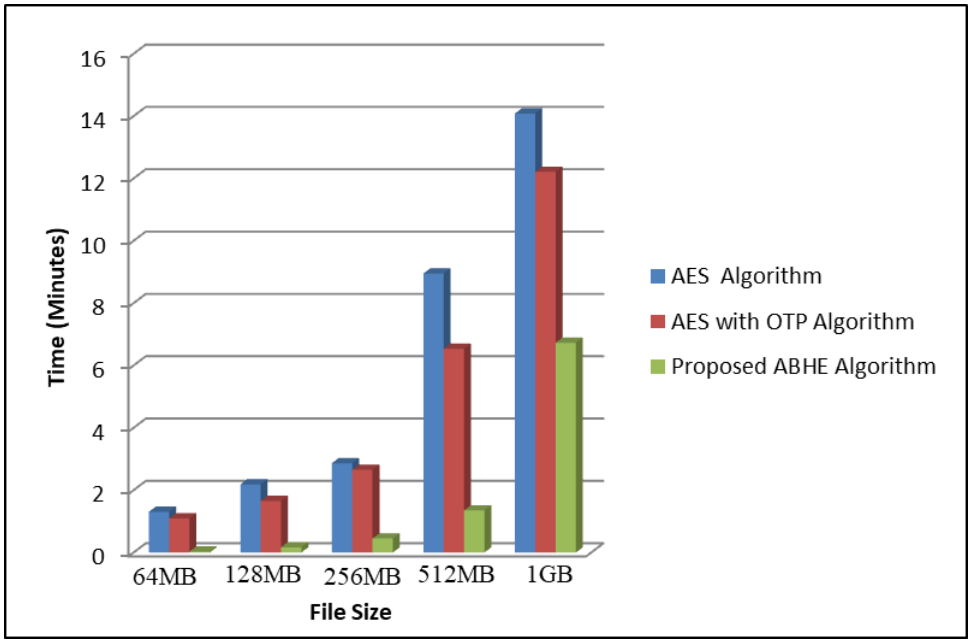

Figure-4.6(h): Decryption Time (minutes) of AES, AES with OTP and Proposed ABHE Algorithm

When the proposed ABHE algorithm is integrated with Hadoop, it showed better performance than the previously available cryptographic

algorithm. From the results of table-4.6(a), table-4.6(b) and table-4.6(c), it is clear that:

- The proposed algorithm ABHE is taking less time to encrypt and decrypt text files than the AES, AES with OTP algorithms.
- The throughput of ABHE is very high as compared to the AES, AES with OTP algorithms.
- As the throughput increases, the power consumption decreases hence the power consumption of ABHE is low than that of the AES, AES with OTP.

Proposed encryption algorithm improves uploading-downloading time and does not expand the size of original file. Comparison of file size 1.12GB (in terms of block size) before and after encryption has been shown in figure-4.6(c), figure-4.6(d), figure-4.6(e) and figure-4.6(f). Here, the file 2048MB.txt is of size 1.12GB. This file contains 0 to 8 blocks. The size of blocks 0 to 7 remains unchanged after encryption. The change in size of block 8 is 125449781 bytes to 125449792 bytes. The change in block 8 is insignificant.

4.7 Conclusion

In this chapter, the researcher has focused on protection of big data stored in HDFS by integrating the proposed ABHE algorithm with Hadoop. Big data is stored in various nodes belonging to many clusters which are distributed around the all over world. The entire infrastructure between clusters and nodes are spawned through public and private networks. If someone can modify the inter-node communication it would be very easy to extract valuable information. Therefore, it is a big challenge for big data tools to adopt new secure network protocols in order to protect interactions between different parties. Hence, in order to increase the security of big data a new encryption technique has been developed. From the simulation results, it has been observed that the proposed encryption technique successfully enhanced the file encryption-decryption process in HDFS

and also provided two layers security for the stored HDFS data as well as protected the data during transmission from one DataNode to anther DataNode.

In a nutshell, for ensuring data security in Hadoop through the proposed encryption technique, HDFS files are encrypted by using attribute based honey encryption through the proposed ABHE algorithm. Results have shown that the proposed encryption technique outperforms the existing encryption algorithm like AES and AES with OTP when integrated with Hadoop. The parameters for comparison include size of the file after encryption, Encryption-decryption time, throughput of encryption-decryption time and power consumption.

CHAPTER 5

VALIDATION OF THE PROPOSED ENCRYPTION ALGORITHM

5.1 Background

To scrutinize the performance and usefulness of a proposed approach/methodology, validation techniques are the most appropriate among all mechanisms. Validation is termed as an authentication which fully satisfied objective of an evidence as proposed by ISO/IEC. The first requirement that is needed to be fulfilled for any specific use or implementation is collecting data from various identified sources. Practical testing presents a scientific foundation to the engineering of anticipated learning and from which statistical data is collected to perform the validation process of models or approaches. There is a chance that different sections of models have different level of validity to get the whole boundary of system behaviour. Assumption, input parameter values and distributions, output values and conclusions are the features in which the model validation has been categorized [130].

In other words, the procedure which is used to estimate the comparison between the two results, one from the simulation i.e. computational result and other is actual (hypothetical) data is called validation [131]. The major objective of validation is to identify and quantify the error. It is the process to validate the actual and expected output of the software activities. It is the dynamic procedure of testing the real approach/model verification. Validation is the process; to validate the approach/model which we have developed is right or not [130].

Validation is one of the very important parts of the research work which prove that the hypothetical assumptions which we assumed have existence in the living environment. This also defines the work in

terms of consistency, accuracy and robustness of a pre assumed research work. In addition, it is also a method to prove that our assumptions are real and can be used to fulfil the predefined output. It also gives an approach which converts the hypothetical pre assumed work into real output. Some characteristics of validation are given below:

5.2 Statistical Analysis

For validation of the proposed algorithm,t-Test: Paired Two Samples for Means is carried out [129]. The paired two samples for means t- Test, sometimes called the paired t-test, is a statistical procedure used to determine whether the mean difference between two sets of observations is zero [129]. In a paired t-test, each subject or entity is measured twice, resulting in pairs of observations [129]. The rejection or acceptance of a null hypothesis is based upon either (0.05 or 5%) alpha (α) or (0.01 or 1%) alpha (α) level of significance for one tailed or two tailed test [134].

The proposed ABHE algorithm outperforms the previously existing algorithms namely AES, DES, Blowfish when compared (without integrating with Hadoop as presented in Chapter-3). Also on integration with Hadoop, it outperforms AES and AES with OTP as presented in Chapter-4. Though it performs better in all the aspect, its acceptability is doubtful without validation. Hence, for the purpose of validation, two data sets have been taken; one from the Hadoopenvironment and the other from non Hadoop environment.

5.2.1 Data Set-1 (Without Hadoop Integration)

Since AES is claimed to be the better compared to the other two i.e. DES and Blowfish [112, 113], hence for the purpose of validation AESand proposed algorithms havebeen compared with different file sizes (49 KB, 59KB, 100KB, 247KB, 321KB, 1MB, 2MB, 5MB, 10MB, and 20MB). Encryption time for different size of text fileswith AES and proposed algorithm (ABHE) is shown in table-5.2.1(a).

Size of Text File	AES (in msec)	Proposed Algorithm(ABHE) (in msec)	Difference (D) of Encryption Time
49Kbytes	56	39	17
59Kbytes	38	34	4
100Kytes	90	34	56
247Kbytes	112	39	73
321Kbites	164	56	108
1 Mbytes	80	62	18
2 Mbytes	154.7	203	-48
5 Mbytes	376.1	230	146
10 Mbytes	683.7	591	92
20 Mbytes	1350.5	1202	148

Table-5.2.1(a): File Encryption Performance Comparison between AES and the Proposed ABHE Algorithm

Hypothesis Testing

For the purpose of validation, hypothesis testing is performed on the performance of AES and proposed one i.e. ABHE in normal system environment (without Hadoop integration). Following are the hypothesis formulated for validation:

Null Hypothesis (H0): The proposed Algorithm ABHE takes more time than that of AES while encryption.

Alternative Hypothesis (H1): The proposed Algorithm ABHE takes less time than that of AES while encryption.

Assumptions

The key assumption relates to normality or non-normally of the data [132, 133]. One of the reasons for the popularity of the t-test is its robustness in the face of assumption violation [132, 133]. Unfortunately, in practice it often happens that more than one

assumption does not meet. It is a good practice to evaluate the degree of deviation from these assumptions in order to assess the quality of the results. In a paired t-test, the observations are defined as the differences between two sets of values, and each assumption refers to these differences, not the original data values. The paired t-test has the following four assumptions [132, 133]:

- The dependent variable must be continuous (interval/ratio).
- The observations are independent of one another.
- The dependent variable should be approximately normally distributed.
- The dependent variable should not contain any outliers.

The continuity of dependent variable and independence of observations are usually not testable [132, 133], but can be reasonably assumed if the data collection process was random without replacement. In our observation data, it is reasonable to assume that the contributing algorithms are independent of one another and dependent variable (difference) assumes the continuous interval.

To test the assumption of normality, as usual, for the results to be valid, a variety of methods are available, but the simplest is to inspect the data visually using a Shapiro-Wilk test shown in table-5.2.1(b). The majority of data in real world is never perfectly normal, so this assumption can be considered reasonably met if test significant value isgreater than our alpha value. As it comes to be .773, the data set is a better candidate for paired two samples for means. For application of t-test in the scenario, test significant value .773 is greater than alpha, 0.05.

Shapiro-Wilk Tests for Normality

	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	Df	Sig.
Difference	.151	10	.200*	.959	10	.773

Table-5.2.1(b): Shapiro-Wilk Tests of Normality

Outliers are rare values that are very far away from the majority of the data. They can influence our result and lead towards incorrect values if not handled properly. The simplest way to avoid these outliers is to remove them from our data [132, 133]. But, removing these data points may lose some critical information. As it can be seen in box plot in the figure-5.2.1(a), outliers are not present therefore, there is no need to remove and our result is uninfluenced by them. All assumptions have been checked. This shows that our data is normally distributed.

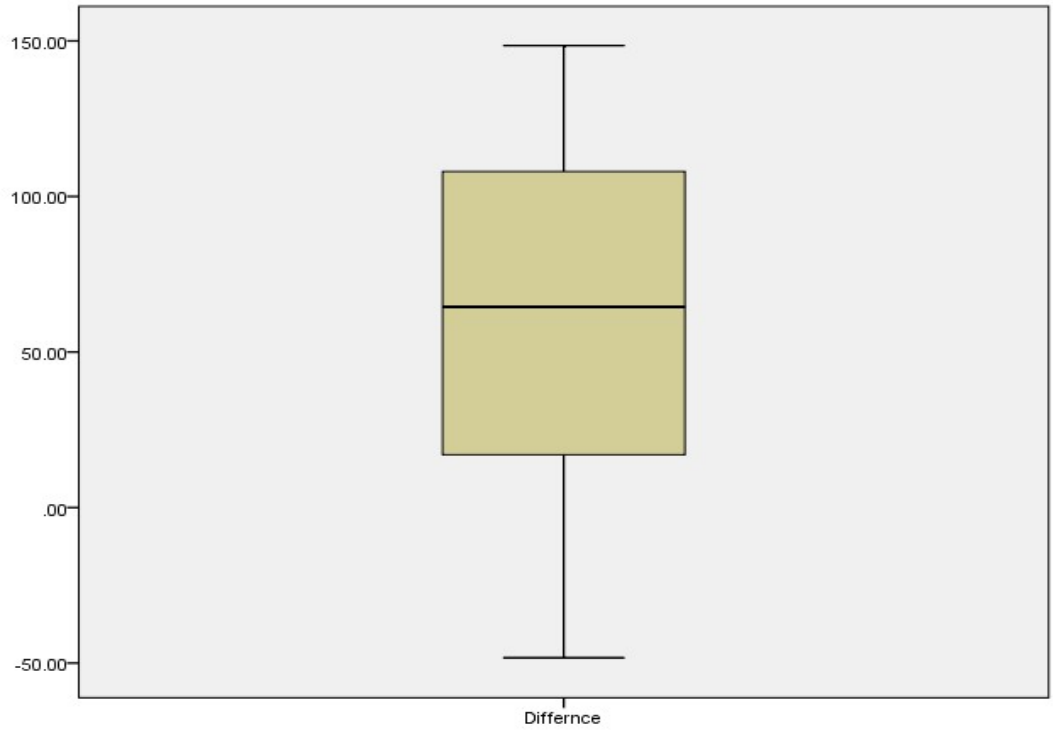

Figure-5.2.1(a): Box Plot for difference measures (column Difference (D) of Table-5.2.1(a))

Level of Significance of the Proposed Encryption Technique ABHE

To find out the significant difference between run time of AES and proposed algorithm (ABHE) mean; the mean of AES and proposed algorithm (ABHE) are shown in table-5.2.1(c). Pearson's correlation coefficient (r) test is used to measure the strength of a linear relationship between two quantitative variables. Pearson correlation is .96 (0.957872) which indicatesthat the two variables are rather closely correlated.The degree of freedom is 9 (10-1). Statistical observation is P(T <= t) and two tail (0.014419) gives the probability that the absolute value of the t-Statistic (3.022595) would be observed that is larger in absolute value than the Critical t value (2.262157). Since the p (0.014419) value is less than our alpha, 0.05, we strongly reject null hypothesis and alternate hypothesis is accepted. Hence it is validated that performance of proposed encryption algorithm is better than the previous algorithm AES.

T -Test: Paired Two Sample for Means

Statistical Observation	AES	Proposed Algorithm (ABHE)
Mean	310.5	249
Variance	172344.1	142279.8
Observations	10	10
Pearson Correlation		0.957872
Hypothesized Mean Difference	0	
Degree of Freedom (df)		9
t Stat		3.022595
P(T<=t) one-tail		0.007209
t Critical one-tail		1.833113
P(T<=t) two-tail		**0.014419**
t Critical two-tail		**2.262157**

Table-5.2.1(c): T -Test: Paired Two Sample for Means

5.2.2 Data Set-2 (on Integration with Hadoop)

When the proposed algorithm have been integrated with Hadoop and have been compared with the existing available algorithms i.e. AES and AES with OTP [105] (on Hadoop integration), the proposed algorithm outperformed both of them. It clearly reflects that the proposed one is better. But acceptability of the proposed ABHE algorithm (while integrating with Hadoop) cannot be determined without validation. Hence validation is performed with another set of data with Hadoop integration. This set of data is obtained from the implementation of the proposed algorithm with Hadoop. In addition, the data from AES with OTP algorithm (with Hadoop integration) have been used for comparison and for the purpose of validation as AES with OTP is claimed to be the better one when compared with AES algorithm [105]. The comparative data with different file size (64MB, 128MB, 256MB, 512MB and 1024MB) has been obtained. Encryption time for different size of text files with AES-OTP and proposed algorithm (ABHE) is shown in table-5.2.2(a).

File Size (MB)	AES with OTP Algorithm (Minutes)	Proposed ABHE Algorithm (Minutes)	Difference (D) of Encryption Time
64	0.7311	0.026	.71
128	1.3820	0.168	1.22
256	2.5484	0.45	2.09
512	4.8780	1.35	3.52
1024	11.2511	6.08	5.17

Table-5.2.2(a): File Encryption Performance Comparison between AES with OTP and the Proposed ABHE Algorithm

Hypothesis Testing

For the purpose of validation, hypothesis testing is performed on the performance of AES with OTP algorithm and proposed one i.e.

ABHE in Hadoop environment. Following are the hypothesis formulated for validation:

Null Hypothesis (H0): The proposed Algorithm ABHE takes more time than that of AES with OTP while encryption.

Alternative Hypothesis (H1): The proposed Algorithm ABHE takes less time than that of AES with OTP while encryption.

Assumptions

For testing the hypothesis, firstly normality of the data sets has been checked (more detailed discussion has been given in section-5.2.1). As it comes to be .669 (shown in table-5.2.2(b)), the data set is a better candidate for paired two samples for means. For application of t- test in the scenario, normality can be obtained by Shapiro-Wilk test significant value .669 that is greater than alpha value, 0.05.

Shapiro-Wilk Test for Normality

	Kolmogorov-Smirnov			**Shapiro-Wilk**		
	Statistic	Df	Sig.	Statistic	Df	Sig.
Difference	.198	5	.200*	.940	5	**.669**

Table-5.2.2(b): Shapiro-Wilk Test of Normality

As it can be seen from the box plotted in figure-5.2.2(a), outliers are not present therefore, there is no need to remove them and our result is uninfluenced by them.In the box plot there's no points plotted above the top fraction or blow the bottom fraction so it clear there's no outliers in this distribution. Now that all assumptions are checked and satisfied it means that our data is normally distributed.

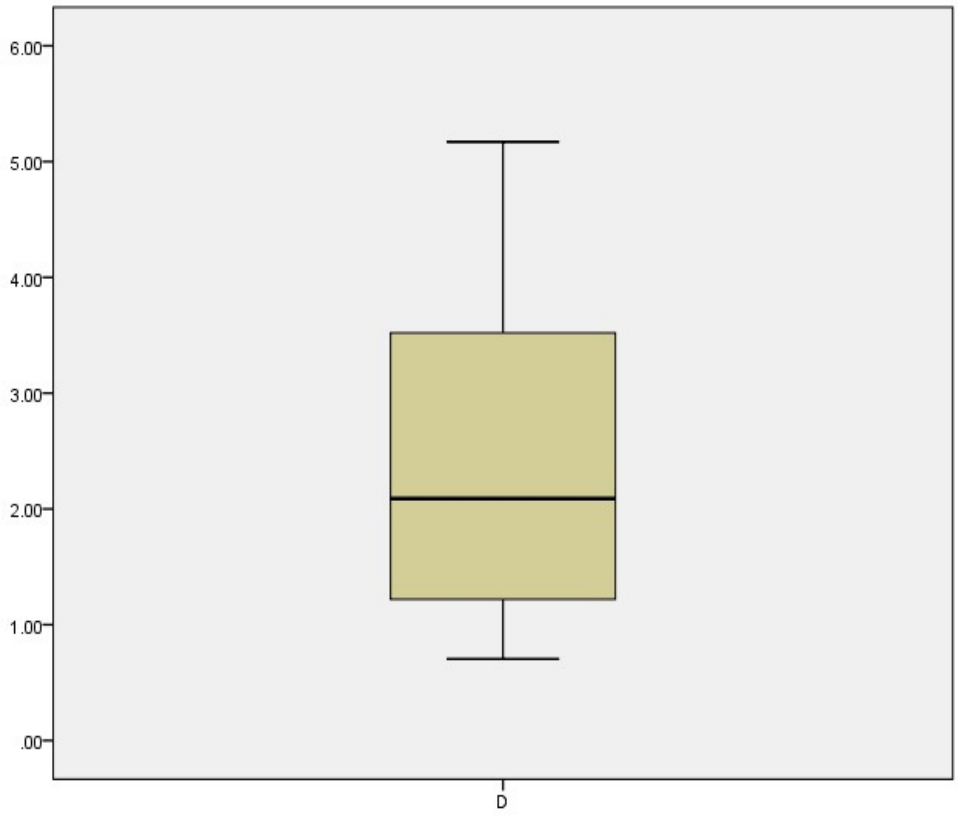

Figure-5.2.2(a): Box Plot for difference measures (column Difference (D) of Table-5.2.2(a))

Level of Significance of the Proposed Encryption Technique ABHE

To find out the significant difference between mean run time of AES with OTP algorithm and proposed algorithm (ABHE); the mean of AES and proposed algorithm (ABHE) have been calculated which is shown in table-5.2.2(c). Pearson correlation is .98. The degree of freedom is 4. Statistical observation t-Statistic is 3.128369 that is larger in absolute t critical value (2.776445).Since the p = 0.03 (.035241) value is less than alpha, 0.05, we strongly reject null hypothesis and hence alternate hypothesis is accepted. Hence it is validated that performance of proposed encryption algorithm is better than the previously available algorithms AES and AES with OTP.

T -Test: Paired Two Sample for Means

Statistical Observation	AES and OTP Algorithm	Proposed Algorithm (ABHE)
Mean	4.15812	1.6148
Variance	18.21777	6.495601
Observations	5	5
Pearson Correlation		0.984015
Hypothesized Mean Difference	0	
Degree of Freedom (df)		4
t Stat		3.128369
P(T<=t) one-tail		0.01762
t Critical one-tail		2.131847
P(T<=t) two-tail		0.035241
t Critical two-tail		2.776445

Table-5.2.2(c): T-Test: Paired Two Sample for Means

5.3 Conclusion

Any new method or approach becomes acceptable by society or industry only after its validation. Validation approves the applicability of the proposed approach at larger level. For testing the usefulness of the proposed approach (with Hadoop integration and without Hadoop integration) for securing data stored at HDFS, a t-test: paired two sample for means is carried out.

The results of validation show a strong level of acceptance for proposed algorithm.Togain confidence in the usefulness of the proposed approach, thevalidation is carried out on the two data sets; one from the Hadoop environment and the other from non Hadoop environment. Furthermore, for the purpose of validation, hypothesis testing is performed on the performance of previously available algorithms and proposed one i.e. ABHE (integrating with Hadoop and without Hadoop environment). It is found that the t-values obtained by computation performed on available algorithms and proposed ABHE algorithm values are exceeding the t-critical values in both environments i.e. integration with and without Hadoop. Hence, the null

hypotheses formulated at the beginning of statistical analysis are rejected one by one and alternative hypothesis are accepted. Hence it is validated that performance of proposed encryption algorithm (integrating with and without Hadoop environment) is better than the previous available algorithm. Researcher's claim that theproposed ABHE encryption algorithmis able to provide effective and more efficient and better performance other as compared with the existing algorithms.

CHAPTER 6

CONCLUSION

6.1 Background

Big data security is a major concern in the software field due to its exponential growth which include personnel as well as private details. This happens due to the regular use of social networking sites likes Facebook, Instagram, Twitter etc. as well as banking and healthcare services by the users. This data is increasing significantly and stored in HDFS & cloud storages. Companies dealing with the various sectors like banking, healthcare, social media etc. are very much concerned about securing their customer's privacy as well as the financial frauds with them. As a recent security breach cited in July 2018 wherein 1.5 million private details of patients including the Prime Minister Len Hsien Loong records from Sing Health patients Singapore [38] were leaked.

Privacy and security breach risk arises, as distributed data repositories can be accessed from any node all over the world. Hence, security of sensitive information is affected. However, it can also threaten the privacy of a person or organization and in such cases; the right to privacy and the protection of personal data should be guaranteed. Thus, the big data security and privacy protection is a moving target which requires more attention and focus. For securing the stored data in HDFS, overcoming the security issues in the Hadoop environment are much advocated. But, reducing attacks on Hadoop HDFS storage and improving the same is a challenging task.

There are various mechanisms and approaches available to secure the data stored in HDFS. These include AES data encryption for HDFS, and AES with OTP encryption for HDFS. These approaches ensure the confidentiality of data stored at HDFS. However, a comprehensive review stated that there are not enough security mechanisms and

approaches available to protect data from attacks at storage level. In addition, it is also evident from the literature that implementation of encryption techniques for HDFS has received a little attention when it comes to consider the file size. It affects transmission speed. There are plenty of literatures available to improve the security of data stored at HDFS in Hadoop environment [107-127]. As encryption is one of the easy and simple techniques. It is feasible to integrate encryption with Hadoop for securing data at rest as well as in transit. The proposed research has been carried out to fulfil therequirement of data security while it is processed in Hadoop. As a result, an encryption technique has been proposed for securing the data during storage as well as transmission at HDFS level. The proposed encryption technique supports attributes based Honey encryption which provides two layers security for data stored at HDFS and also, supports the encryption of the variable length data files. The proposed encryption technique is implemented and integrated with Hadoop. The results have shown the enhanced security for big data. The proposed encryption technique encrypts the contents of each file which has been stored in HDFS. The encrypted data can only be retrieved by applying the corresponding proposed key to fulfil the objective of providing security to the data.

To check the performance of introduced encryption algorithm, the parameters namely encryption-decryption time, encryption-decryption throughput, and encryption-decryption power consumption are computed. Analysis of the performance parameters yields that the proposed algorithm is better than the previously available algorithms. Also, the validation is performed by two sets of data (with and without integrating on Hadoop). For the validation of ABHE approach, the student paired t-test is carried out to determine its level of significance. Statistical interpretation has shown that the proposed ABHE encryption technique is acceptable at all aspect.

6.2 Research Contribution

In depth study of the existing literature on securing data at HDFS storage and achieving the research objectives, the researcher has made the following contributions:

- **Explored Existing Big Data Characteristics and Proposed Three New Characteristics:** In this digital world, many organisations are dealing with enormous data and due to this various issues have come up.Although, there are many solutions available to solve the issuesas discussed in [135], problems continue. Due to this, need for in-depth knowledge of big data has arisen which has helped to find out the solution of big data issues effectively. During the study about big data, three new characteristics Verbosity, Versatility, and Voluntariness have been proposed in addition to the previously explored characteristics. The contribution same is published in [135].

  The new characteristics are named on the basis of features of big data which could also correlate with the various real world applications and organisations like urban planning model after considering the environment and traffic patterns.With the help of proposed characteristics, manufacturers can optimize their productivity and can improve the performance of their equipment's by analysing the product issues and customer performances, energy companies can increase the production to meet out the demands of energy during peak hours, and also, the field of life science and medical science to improve the patient health and many more.These proposed characteristics are useful for students and data analytic departments. The proposed characteristics will definitely create a future scope for the developers to work upon.

- **Study of Big Data Processing Solution: Hadoop:** The utility of big data technology has been largely favoured by the release of new

software: Apache Hadoop also, discussed in chapter-2. Apache software foundation has developed a framework called Hadoop which allows the processing of large datasets by dividing them among clusters of computer using programming models. Hadoop is developed as a distributed data processing platform for analyzing big data. Many Enterprises are analysing big data containing user's sensitive information by using Hadoop and utilize them for the marketing and processing their previous stored data. The researcher has conducted in depth study on the same. The contribution has been published in [136, 138].

Even Hadoop provides the same kind of efficiency of expensive hardware in affordable, industry- standard servers. Data forms the input for Hadoop framework that feeds the big data system. As mentioned earlier, these data comes from various sources. Although, Hadoop can process the big data efficiently but lacks in the prospects of security and protection which is a major drawback in today's scenario as discussed in chapter-2.

- **Study of Various Technology and Fundamental Tools for Big Data:** Various tools and file systems have been developed for the use of storage, managementand analysis of big data. These tools help in the processing of large amount of data ranging from gigabytes to petabyte. Apache pig, Hive, HBase, MapReduce, Mahout, Oozie, Zookeeper, Sqoop, Ambari etc. components together when combined with Hadoop make the ecosystem much more scalable for a robust solution for big data. The detailed description is already presented in chapter-2 and the related study has been published in [137].

- **Literature Survey on Securing Big Data:** An exhaustive review has been done on secure storage of big data at HDFS during the initial phase of research work. There are many researches and studies to overcome the issue of protecting information from the eyes of

unauthorized persons. The contribution has been published in [138]. Available approaches on securing data stored at HDFS shows that the research on big data security is still immature and has many limitations.

In addition, the techniques available are not sufficiently solving the purpose as the huge volume of big data is now gradually increasing everywhere in various fields. Thus, it requires more privacy and security approaches and needs to be explored further to identify the importance of security of big data at both levels i.e. at storage level and at the transit. Big data security usually involves theimplementingthe security solutions to enhance the security, reliability and security of a distributed system environment; it needs a balance between data usage & data protection. The proposed research may provide the security in big data environment.

- **Proposed a Cryptographic Approach (ABHE Encryption Algorithm) for Big Data Security at Rest as well as at Transit:** To secure big data while stored at HDFS as well as in transit the researcher has proposed anattributebased Honey encryption technique. The same is discussed in chapter-3 and accepted for publication in [139]. The proposed encryption algorithm provides double layer security to the data stored at HDFS and alsoduring transmission. In this proposed encryption technique, firstly we apply the attribute based encryption (which is based on ciphertext policy based attribute encryption) on a particular record and then again secure the document by password (which is based on Honey encryption). The attributes in the proposed encryption technique are very adaptable. The access policy in the proposed approach is a propositional equation which is very straightforward. We set forth the policy from the attributes utilizing activities performed in arbitrary request to make it more secure.

Evaluation of the performance of proposed algorithm has also been done. The parameters for performance include encryption–decryption time (rate of encryption is given by encryption time and rate of decryption is given by decryption time), throughput of encryption-decryption where throughput is calculated as the size of plain text (in Kbyte or Mbyte) is divided by total encryption-decryption time. The speed of encryption-decryption process depends upon the throughput of the encryption-decryption scheme, as it defines the speed of encryption-decryption. Power consumption of encryption-decryption schemes depend upon the throughput of encryption-decryption. In case of encryption-decryption, as the throughput increases, power consumption decreases.

The proposed ABHE algorithm has been compared with previously available encryption algorithms on the basis of the performance parameters i.e. encryption-decryption time, throughput and power consumption with different file sizes (increases KB to MB).The results show that the proposed ABHE scheme considerably improves over the previously existing schemes namely AES, DES and BLOWFISH (without integrating with Hadoop). The detailed description is presented in chapter-3.

- **Integration of Proposed ABHE Encryption Technique with Hadoop for Securing Big Data:** To prove the usefulness of the proposed ABHE encryption algorithm for big data security, the researcher has integrated the same with Hadoop to secure stored HDFS data as well as data in transmission. The proposed ABHE algorithm with Hadoop environment is also evaluated on the basis of same performance parameters i.e. encryption-decryption time, throughput and power consumption. Also, the researcher has compared the results with previously available HDFS encryption algorithms namely AES and AES-OTP with different file sizes (increasing from MB to GB). The performance parameters results have shown that the proposed ABHE scheme with Hadoop

environment has provide the considerable improvements over the previously existing schemes namely AES, AES with OTP (Integrating with Hadoop). Also, the proposed algorithm provides dual layer security for data stored at HDFS. The detailed description is given in chapter-4 and same is presented in [140].

- **Validation of the Proposed ABHE Encryption Technique:** The researcher has validated the proposed HDFS encryption approach provides better performance when compared with the existing algorithms. The validation of proposed ABHE approach is performed on the basis of two data sets (with and without integration on Hadoop) which are described in detail at chapter-5. For the validation of proposed algorithm, paired t-test has carried out. The successful validation of the proposed encryption technique has also reflected its acceptability.

6.3 Significance of the Work

The proposed study has been able to successfully solve the weaknesses present in the security approaches available for big data security. The significance of the proposed work is as follows:

- It is expected that the newly identified characteristics may provide the simple and effective approach towards big data management which in turn will add values in applications and research environment.
- The proposed big data characteristics may be used to develop the various techniques to solve the problem of massive while managing it.
- Proposed encryption technique which uses the concept of Attributes Based Honey Encryption (ABHE) may help to securing sensitive information stored at HDFS in insecure environment such as the internet and cloud storages.

- With the proposed algorithm, the security of data is not only dependent on the secrecy of encryption algorithm but also on the security of the key this provides dual layer security for the data.
- As with the proposed ABHE algorithm, the execution time (a function of encryption time) is less as compared to the other approaches this will prove to be the fast enough to secure the data without delay.
- The proposed ABHE algorithm has higher throughput. It again proves its applicability on big data.
- It provides a feasible solution for secure communication from one DataNode to another DataNode.
- As the proposed encryption technique does not increase the file size, therefore,it saves memory and bandwidth, and hence reduces traffic in a network.
- Proposed ABHE encryption approach has the ability to encrypt structured as well as unstructured data under a single platform.
- Only HDFS clients can encrypt or decrypt data with accurate attributes and password.
- Proposed ABHE technique provides dual layer security for all DataNode as data is not confined to a single machine. Clients are able to access system and data from anywhere.
- The proposed encryption approach may form a basisfor the development of new, modified or refined approaches.

6.4 Research Findings

During the course of study, the aim of the researcher has been to find out the answer of the research questions raised in chapter-1. In this row, the researcher has also tried to solve the security issues identified during literature survey. This section systematically presents the answers of the research questions formulated in chapter-1.

Research Question: What is upto date status of the security of big data?

Research Finding: As big data is a new concept there are very few approaches especially meant for big data security? It is generally treated by traditional security approaches.

Research Question: Are available approaches for data security applicable for big data security?

Research Finding: Big data security is a different from data security thereforethe existing approaches and techniques of data security are not much capable and useful in case of big data security.

Research Question: Are available security approaches for big data security sufficient enough to secure big data?

Research Finding: No, available security approaches are sufficient enough to handle big data security issues as they have several limitations in terms of speed, file size, power consumption etc.

Research Question: Does Hadoop, big data processor, have any built-in security feature?

Research Finding: With the time, users are more concerned with security. Due to this, Hadoop open source community has added some security features like authentication, access control, and data encryption. Initially, at the time of development of Hadoop,a little attention was given to security. This is the reason, no security features were introduced against external threats.

Research Question: Can an approach be developed which is secure as well efficient in terms of encryption time, decryption time, throughput of encryption/decryption and power consumption.

Research Finding: During the course of study, the researcher found that it is feasible to develop an approach which is secure as well efficient in terms of encryption time, decryption time, throughput of encryption/decryption and power consumption. The proposed ABHE algorithm attains the purpose.

6.5 Future Direction

Research is a continuous approach in which reaching at one milestone open door foranother. When considering the future research plan, following tasks are needs to be addressed:

- In future, we may apply the proposed encryption technique inthe real world applications, such financial information processing and Healthcare foundations etc.
- The proposed AHBE algorithm has been used on single DataNode. In future as per requirements we can add multiple DataNode in Hadoop cluster.
- To keep pace with modern cyber criminals proposed algorithm may be implemented with AI (artificial intelligence) that can help to provide the rapid detection and response capabilities.
- Researcher is planning to conduct more experiments on IoT devices in order to protect digital transformation and maintain rigid security posture.

6.6 Conclusion

Big data is becoming highly prevalent in businesses and organizations day-to-day activities. The researcher has addressed the burning issue of security for the data stored in HDFS. For the purpose, ABHE encryption technique is proposed. The aim of the ABHE algorithm is to provide security on stored data in HDFS as well as data in transmission. To test the usefulness of the proposed ABHE algorithm, its implementation has been performed by the researcher. The proposed ABHE algorithm is integrated with Hadoop. It has been observed that the proposed ABHE encryption approach encrypts the content of files stored in HDFS thus by securing it from the attacker. Hence, the data or files now can bestored in Hadoop without worrying about security issues by applying the proposed ABHE encryption algorithm on the files which are stored in Hadoop.

Results of the experiment show that our proposed ABHE encryption approach is better than the existing algorithms namely AES and AES with OTP on the basis of size of the files after encryption, throughput and power consumption. It has been seen that with AES, file of size 1GB becomes 1.5GB after encryption and with AES-OTP the size becomes 1.19GB. On the contrary, it has been observed that with the proposed approach 1GB file remains almost 1GB even after encryption. Throughput and power consumption are much higher than the AES and AES with OTP algorithms. The proposed approach has been validated by rejecting the null hypothesis and accepting the alternate hypothesis.