DESIGN AND ANALYSIS OF ENCRYPTION SCHEMES BASED ON SOME ADDITIONAL PROPERTIES

SUBMITTED TO THE DELHI TECHNOLOGICAL UNIVERSITY
FOR THE AWARD OF THE DEGREE OF

DOCTOR OF PHILOSOPHY

NITIN JAIN

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
DELHI, INDIA
2023

DESIGN AND ANALYSIS OF ENCRYPTION SCHEMES BASED ON SOME ADDITIONAL PROPERTIES

By

NITIN JAIN
(2K13/PHD/IT/04)

SUBMITTED IN FULFILLMENT OF THE
REQUIREMENT OF THE DEGREE OF

DOCTOR OF PHILOSOPHY

Under the Joint Supervision of

Supervisor
Dr. O.P.Verma
Professor and Head, Department of
ECE, DTU, New Delhi -110042

Co-Supervisor
Dr. Saibal K. Pal
Scientist- 'G' SAG, DRDO
Metcalfe House, Delhi-110054

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
DELHI, INDIA

2023

DECLARATION

I, Nitin Jain, PhD student (Roll No. 2K13/Ph.D/IT/04), hereby declare that the thesis entitled "**Design and Analysis of Encryption Schemes Based on Some Additional Properties**", which is being submitted for the award of the degree of Doctor of Philosophy in Information Technology is a record of bonafide research work carried out by me in the Department of Information Technology, Delhi Technological University. I further declare that this work is based on original research and has not been submitted to any university or institution for any degree or diploma.

Nitin Jain

(2K13/PHD/IT/04)

CERTIFICATE

This is to certify that the work embodied in the thesis entitled "**Design & Analysis of Encryption Scheme with Some Additional Properties**" submitted by Nitin Jain to the Department of Information Technology, Delhi Technological University, Delhi, for the award of the degree of Doctor of Philosophy is a record of bonafide research work carried out by him under our guidance and supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. The results contained in this thesis have not been submitted to any other university or institute for the award of any degree or diploma.

Dr. O.P.Verma
Professor and Head, Department of ECE,
DTU, New Delhi -110042

Dr. Saibal K. Pal
Scientist- 'G' SAG, DRDO
Metcalfe House, Delhi-110054

Dedicated to My Father……

ACKNOWLEDGEMENT

First and foremost, praises and thanks to God, the almighty, for Divine showers of blessings on me for successfully completing my research work.

I would like to express my heartfelt gratitude to my supervisor, **Dr. O P Verma**, Professor D/o of ECE, DTU, Delhi. I immensely thank Prof. Verma for his motivation, support, and guidance. He always encouraged and helped me during the ups and downs of my PhD journey. I am also thankful to my Co-Supervisor, **Dr. Saibal K. Pal**, Scientist-'G' Scientific Analysis Group, DRDO, Metcalfe House, Delhi. He boosted my spirits and ignited my passion for learning.

I would like to extend my special thanks to **Prof. Dinesh K. Vishwakarma**, Head, D/o Information Technology, who always encouraged me to move further. I am grateful to the members of my Department's Research Committee for reviewing my work and for their valuable comments. I would also like to thank the faculty and staff of the department for their assistance and encouragement during my research work.

I extend my special thanks to **Dr. Lalita**, Assistant Librarian, Delhi Technological University. She is always ready to help and extends her full support. I am thankful to all my friends for their support and motivation.

I am thankful to my dear wife, Mrs. Khyati, my elder brother Mr. Pradeep and my younger brother, Mr. Naveen, who keeps motivating me in my PhD journey. I am also thankful to my sister-in-law, Mrs. Priya Jain, and my nephews, Daksh and Arham Jain, for their constant love.

Last but not least, I wish to thank my Grandparents and my parents **Sh. Dharampal Jain** and **Smt. Krishna Jain** for giving their unconditional love and relentless encouragement.

Nitin Jain

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

IoT	:	Internet-Of-Things
SS	:	Secret Sharing
VSS	:	Verifiable Secret Sharing
LFSR	:	Linear-Feedback Shift Register
PVSS	:	Publically Verifiable Secret Sharing Scheme
GCD	:	Greatest Common Divisor
SIMD	:	Single Instruction Multiple Data
DGHV	:	Dijk-Gentry-Halevi-Vaikutanathan
FHE	:	Fully Homomorphic Encryption
FPGA	:	Field Programmable Gate Array
PoW	:	Proof-of-Work
DAG	:	Directed Acyclic Graphs
CNSS	:	Committee On National Security Systems
RSA	:	Rivest, Shamir, Adleman
SaaS	:	Software-as-a-Service
PaaS	:	Platform-as-a-Service
IaaS	:	Infrastructure-as-a-Service
FaaS	:	Function-as-a-Service
IP	:	Internet Protocol
GN	:	Gateway Nodes
LoWPANs	:	Low-Power Wireless Personal Area Networks
BC	:	Blockchain
SC	:	Smart Contracts
PoS	:	Proof-of-Stake
PAN	:	Personal Area Networks
LM	:	Local Manager
WSN	:	Wireless Sensor Network
MI	:	Manufacturing Industry
ECC	:	Elliptic Curve Cryptography
DLP	:	Discrete Logarithm Problem
CRT	:	Chinese Remainder Theorem

JRSS	:	Joint Random Secret Sharing
GMP	:	Gnu Multiple Precision Arithmetic Library
NTL	:	Number Theory Library
DLR	:	Dealer Leakage Resilient
CSP	:	Cloud Service Providers
IDSs	:	Intrusion Detection System
DLPSs	:	Data Leak Prevention Systems
DES	:	Data Encryption Standard
AES	:	Advanced Encryption Standard
GNFS	:	General Number Field Sieve
IS-SDLC	:	Integrated-Secure Software Development Life Cycle
PAHDFS	:	Preference-Aware Hadoop Distributed File System
OECC	:	Optimal Elliptic Curve Cryptography
MGOA	:	Modified Grasshopper Optimisation Algorithm
FCM	:	Fuzzy C-Means
GOA	:	Grasshopper Optimization Algorithm
OBL	:	Oppositional Based Learning
KDD	:	Knowledge Discovery Dataset
PIR	:	Private Information Retrieval
CNN	:	Convolutional Neural Network
MNIST	:	Modified National Institute of Standards and Technology Dataset
TFHE	:	Fast Fully Homomorphic Encryption Scheme Over the Torus
IND-CPA	:	Indistinguishability under Chosen-Plaintext Attack
MPC	:	Multiparty Computation
API	:	Application Programming Interface
CC	:	Computation Cost
CCM	:	Cipher Block Chaining-Message
CH	:	Cluster Head
DAG	:	Directed Acyclic Graphs
DpoS	:	Delegated Proof-of-Stake
JSON	:	Javascript Object Notation
LM	:	Local Manager
LpoS	:	Leased Proof-of-Stake
PoA	:	Proof-of-Authority

PoB	:	Proof-of-Burn
PoEEC	:	Proof-of-Energy Efficient Consensus
PoI	:	Proof-of-Importance
PoSpace	:	Proof-of-Space
PoSV	:	Proof-of-Stake Velocity
PoWeight	:	Proof-of-Weight
PSN	:	Proximity Sensor Nodes
SC	:	Smart Contracts
VM	:	Virtual Machine
ABE	:	Attribute-Based Encryption

PUBLICATIONS

Journals

[1] Verma, O. P., Jain, N., & Pal, S. K. (2020). A hybrid-based verifiable secret sharing scheme using Chinese remainder theorem. *Arabian Journal for Science and Engineering*, *45*, 2395-2406, doi: https://doi.org/10.1007/s13369-019-03992-7

(SCIE) Impact Factor- 2.9

[2] Verma, O. P., Jain, N., & Pal, S. K. (2019). Design and analysis of an optimal ECC algorithm with effective access control mechanism for big data. *Multimedia Tools and Applications*, 79, 9757-9783, doi: https://doi.org/10.1007/s11042-019-7677-2

(SCIE) Impact Factor- 3.6

Under Review

[3] A paper entitled "A blockchain-based lightweight encryption scheme for constrained IoT environments" is under review in SCIE indexed journal.

Conference Proceedings

[4] Verma, O. P., Jain, N., & Pal, S. K., Manjwani B.(2015). Practical Application of Homomorphic Encryption, in *Bilingual International Conference on Information Technology: Yesterday, Toady, and Tomorrow*, ISBN: 9678-81-86514-73-3, 19-21 February 2015, pp. 92-96 © Defence Scientific Information & Documentation, Defence Research and Development Organisation, Available at: https://www .drdo.gov.in/sites/default/files/publcations-document/artifical-intelligence.pdf

[5] Sawhney, M. S., Verma, O. P., Jain, N., & Pal, S. K., (2015). Recent Developments in Homomorphic Encryption, in *Bilingual International Conference on Information Technology: Yesterday, Toady, and Tomorrow*, ISBN: 9678-81-86514-73-3, 19-21 February 2015, pp. 101-105 © Defence Scientific Information & Documentation, Defence Research and Development Organisation. Available at: https://www. drdo.gov.in/sites/default/files/publications-document/artifical-intelligence.pdf

Poster Paper Presentations and Summer School

Presented "*Design and Analysis of Homomorphic Schemes*" and participated in a workshop on Real World Crypto-2015 in London, UK (Jan 8-11, 2015)

Summer school on real-world crypto and privacy Šibenik (Croatia), Europe (June 5-9, 2017)

ABSTRACT

The objective of this research is to cater for the emerging need to provide privacy or confidentiality to the magnanimous digital communications happening in the modern world. Encryption and access control are two primary tools helping achieve confidentiality. Existing literature reveals that researchers have been working in the theses areas, but new challenges are faced regularly with recent advancements. New developments like Big Data, Cloud Computing, IoT etc. and increased use of distributed environments pose new hurdles in achieving desired levels of privacy in vulnerable communications. In this thesis, schemes to address the privacy needs of modern-day communications are proposed.

Traditional encryption methods include symmetric-key encryption schemes like DES, AES, IDEA, as well as asymmetric-key encryption schemes like RSA, Diffie Helman Key exchange etc. However, they do not accommodate the need for modern-day communications, which are happening at an unprecedented rate. Secure key management, including secure exchanges for such high-rate electronic transactions, is a significant research area today. To safeguard the cryptographic keys and other possible applications, a hybrid verifiable secret sharing is proposed in this thesis to fulfil the need for multilevel and multi-secret environments. Results are verified to check the performance of the scheme mathematically and experimentally. Secret sharing provides more trustful management of sensitive data, including key launch codes for missiles and many such types of information. However, the challenge is to store and compute the necessary shares. Compared to the existing methods, homomorphic encryption is a more powerful way to handle this challenge. It allows computation on encrypted data (shares) stored on the cloud. It obviates the need for secure (private) storage and paves the way to allow computation of a certain depth. This would help to provide many more practical applications such as Electronic voting, multiparty computation, and many more. Another important aspect is that data and transactions on the cloud need to be trusted. However, such information can always be leaked without appropriate security measures because cloud resources are shared by multiple users and managed by third-party cloud service providers. Thus, there is a need to have a mechanism that can offer distributed trust, which can be

achieved using Blockchain. Likewise, with the advancement of smart activities via the Internet of Things, the data moves in the air. This data is being exchanged via sensor nodes installed in various sensor-based applications, including healthcare, agriculture, the manufacturing industry, and many more. These applications may be susceptible and require appropriate security provisions to be put in place to establish trust in such human-less communications. For handling the challenges mentioned above in trust management, Blockchain acts as a potential technology. Exploiting its potential, a Blockchain-based lightweight scheme is proposed to cater the low-powered sensor devices. This scheme also ensures performance in terms of transactional throughput and mining latency. Another buzzword that emerges hand to hand with IoT and Cloud is Big Data. Thereby, systematic management of Big-Data is a matter of concern while storing and accessing it on the cloud, so, a cloud-based Big-Data management scheme is proposed.

The proposed Schemes are implemented and analysed as part of the thesis work. Experiments show promising results with respect to their efficiency, communication cost, security, and other parameters suitable for their respective applications compared to the state-of-the-art.

CHAPTER 1

INTRODUCTION

This chapter offers the various concepts of Information security more precisely, i.e., cryptography and some additional concepts and related properties, which go hand in hand to cater the today's needs. In this league, several ideas concerned with information security are discussed to formalize the further discussion for the remaining chapters. Several other properties, protocols, and concepts are also discussed, which are used along with cryptography to strengthen the idea and respective solution.

1.1 The Security

Information Security is a vast field that deals with different concerns and functions. Its prime concern is to deal with the security of original information from leakage and ensure that unauthorized users have no accessibility to read or secretly alter the messages, which are intended for other receivers. Besides, it also curtails unauthorized users who are not permitted to access the information through remote services. In short, the term 'security' is concerned with the problems of meaningful messages, which are received and replied accordingly [1].

In general, there are many branches of security. In a hierarchal pyramid, security is considered the highest level, followed by information and network security. The lower level of the pyramid is divided into various branches, including the security of the database, computer, device, and application security. All of these branches may be considered as subsets of the entire discipline of security. The present study deals with network security as a subset of information security, as shown in Figure 1.1, which includes data, hardware, and software protection on computer security. However, the main interest of information safety is to prevent leakage and to protect its resources, such as computer data and voice communication [2], [3].

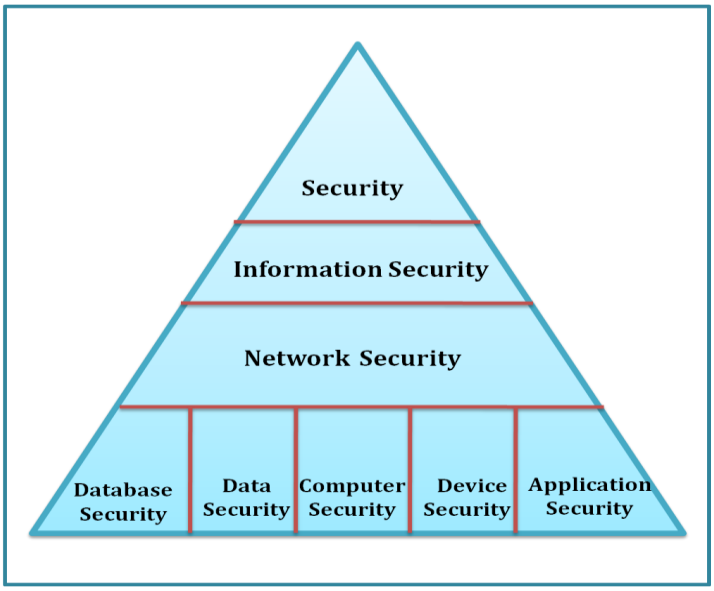

Figure 1.1: Hierarchal Pyramid Representing Different Branches of Security

1.2 Introduction to Information Security

Hardware, software, data, users, operations, and networks are the six basic components of an information system. The National Security Systems committee (CNSS) has defined Information Security as a field that deals with data management and its essential features, along with the systems and hardware. It stores, utilizes, and helps in the transmission of information. This also protects the information from education, policymaking, and technology threats. Figure.1.2 depicts a visual representation of the components of Information Security.

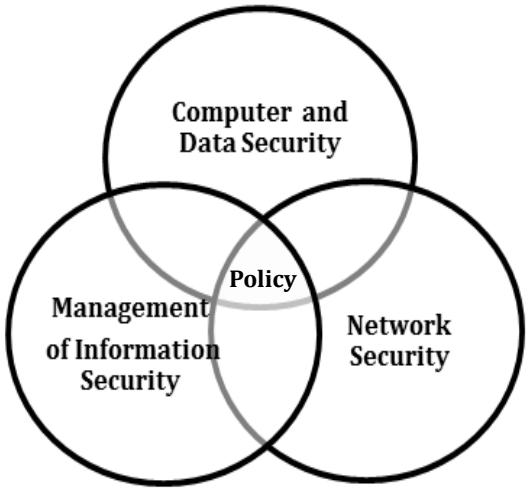

Figure 1.2: Elements of Information Security

Information security refers to the protection of data from unauthorized users in terms of access, use, reading, modification, disclosure, inspection, destruction, or recording, as well as in terms of storage, transport, or processing, and refusal of services to the authorized personnel. It also includes the provision of services to illegal users, as well as the actions required to detect, document, and combat such risks [3], [4]. It is concerned with the confidentiality, reliability, and accessibility of data, irrespective of data form, i.e., electronic, print, or any other form. Multiple regulations, educational and technological techniques are used to provide this form of protection [5].

1.3 Importance of Cryptography

After human resources, information is an organization's most valuable asset. Information is protected by techniques like Cryptography [6] and Steganography [7]. These are the two most essential strategies for securing data transfer via networks.

Cryptography is a method of scrambling secret communication that is unreadable and incoherent. While the former is an ancient skill of hiding information, digital technology allows for masking information in digital media. In the case of cryptography, sender A shares some data or secrets with B over some insecure channel, which may be an unsafe network or a telephone line. The whole idea of security develops due to the insecurity of networks. The message, which is being transmitted, may be intercepted by an eavesdropper who sometimes just reads the message and does not do any harm. In the worst case, one may modify the message, which is not detected by recipient B. It was also used in times of political tension and war to communicate securely, guarding secret information against the enemy. Despite its history of about 4000 years, cryptography only came of age in the 1800s with the invention of technologies such as the telegraph (for rapid communication over great distances) and manual rotary machines, followed in the early 1900s by electrical rotary machines. Cryptography changes its shape from art to science.

It was inevitable at some point; that someone would try to formalize the principal aims of a cryptographic system. Claude Shannon [8] was among the first to do so in 1948. He argued that a cryptosystem designer should assume that the system may be

attacked by someone who has access to it. This happened during the two world wars when machines were stolen and reverse-engineered.

Nowadays, communication is done digitally, with the advent of smart cards, electronic voting, ATMs, online monetary transactions etc. Indeed, these things become a necessity of our day-to-day activities. Thus, modern cryptography has entered the household and become an inseparable part of life in contrast to the past scenarios when it was only used by the military and government. The security of modern cryptography primarily relies on mathematical primitives, which is very much inspired by the classical mathematical toolbox.

Inspired by the discussion, the classification of the art and science of coding, in addition to break the secret writing, is depicted in Figure 1.3. Cryptology is defined as the study of writing and breaking secret writing/codes. Broadly it can be divided into Cryptography and Cryptanalysis [9]. Cryptanalysis is the study of breaking a particular code or scheme. It can be done using standard mathematical assumptions and computing capabilities.

The symmetric cipher is defined by the same key on both sides, i.e., the sender and the receiver. On the transmitter side, the key is used to encrypt the data, and on the receiver side, the key is used to decrypt the data. The convenience of using the same key for both encryption and decryption is dictated by the fact of ensuring that both parties have the correct keys in a tense situation, even when people are widely available and dispersed geographically. Also, for instance, during the war period, the keys must be physically presented to the staff in remote and hazardous areas. Therefore, the drawback of these private key cryptosystems (also called symmetric key cryptography) is that sender and receiver need to share the secret key before they actually begin [10].

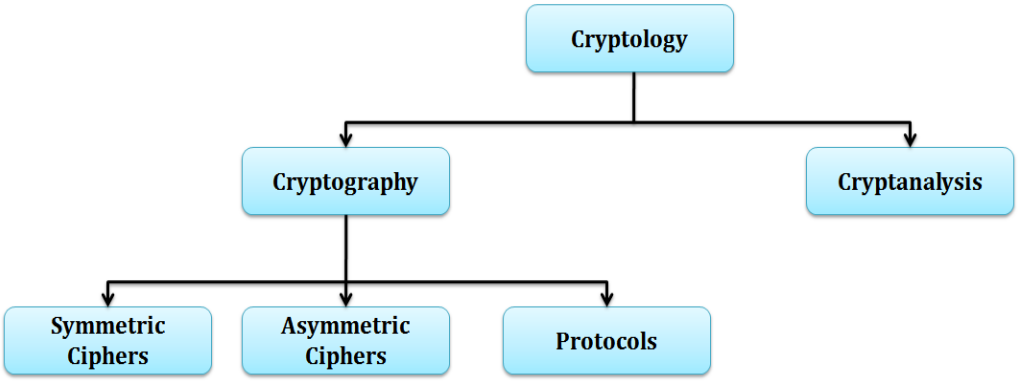

Figure 1.3: Classification of Cryptology

In the 1970s, the revolutionary theory of public-key cryptography was explored. The fact that encryption and decryption keys were different from which are used in public-key or asymmetric cryptography. As a consequence, it can bypass the need to share a key between the sender and the receiver [10].

Another crucial aspect of cryptography is cryptographic protocols, also called communication protocols. It is brought into the conception to render various security assurance using some cryptographic mechanisms. An advance cryptographic protocol furnishes top-level security services such as voting schemes, secure user identification, and digital cash. Various cryptographic primitives, such as Encryption algorithms, hash functions, random number generators, and digital signatures, are used to construct the various protocols. A protocol must involve more than two participants and should be mandatorily distributed in nature. Several defined rules have to be followed by all participants involved in the protocol. There are a number of communication protocols [6]. To name a few, includes Key Exchange and Entity Authentication, identification Schemes, Commitment Schemes, Secret Sharing(SS), Verifiable Electronic Elections, Mix Nets and Shuffles, Receipt-Free, and Coercion-Resistant Elections. SS schemes are discussed in brief in section 1.5 and subsequently in detail in chapters 2 and 4, respectively.

Till now, only the utilization of cryptography in the transmission of classified messages has been discussed. To complete this discussion, it should be noted that providing privacy is not the main target of cryptography. Cryptography is additionally used to give answers to different issues:

a) **Data integrity**: The receiver should be able to check whether the message has been updated, either inadvertently or intentionally, during transmission. Nobody should be able to replace a false message with the original message or part of it.

b) **Authentication:** The receiver should be able to check the source of a message. No one should be able to deliver a message claiming to be the sender (authentication of the data origin). Both sender and receiver will be able to recognize each other when initiating a contact (entity authentication).

c) **Non-repudiation**: the sender will not be able to refuse the receipt of a post.

1.4 Development of Public Key Cryptography

Until 1970s, traditional cryptography was known as a dark art practised by confidential government agencies and a handful of egotistic amateurs. The introduction of public-key cryptography changed the prevailing scenario and attracted researchers worldwide to the domain of cryptography. It is motivated by the need for computerized financial transactions. A fundamental problem in cryptography is that it doesn't know how to safely communicate over an insecure medium, which is being monitored and sometimes controlled by some attacker. Lately, this issue has gained attention and intrigued people. This is primarily due to the widespread usage of computers and communication networks. These communication systems are now being used for several different purposes, from simple exchanges of messages among friends to transactions made with credit cards and also to the transmission of sensitive documents. However, the initial icebreaker was with the Diffe-Hellman Key [11] in 1976. The next breakthrough was with RSA [12] in 1978, which is named after the name of three prominent researchers, Ron Rivest, Adi Shamir, and Leonard Adleman.

Another integral cryptosystem was ElGamal [13] in 1985. These cryptosystems were sufficient to initiate a systematic analysis of the subject; however, the actual push is missing in the community, which led to security notions and formal definitions. It was also well said by Karl Marx that "Necessity is blind until it becomes conscious." So, the community should work hard to develop a secure cryptosystem, and it should be corroborated by defined mathematical and computational proofs rather than heuristic arguments.

A formal security notion is the beginning of the next step, until things are pen down: what we are trying to achieve, how we can accomplish them. The need for a proper security notion was realized when some standard systems were compromised by systematic cryptanalysis. Now, this is the time to formalize the notion in terms of Provable security and other security definitions. After that, many new directions have emerged in the community, not only considering the strength of mathematical primitives, but also dealing with the adversary computing capability [14].

1.4.1 Formalizing the Security Notions

Public-key cryptosystems have been made and broken simultaneously since their origin. The major reason behind this is unconditionally safe or perfectly safe cryptosystems are difficult (intuitively impossible) to construct. The next best thing to do is to minimize the complexity of breaking the cryptosystem to a believed-to-be-hard "real problem P, either computationally or decisively. Thus, one of the critical tasks in creating a secure cryptosystem is to look for an acceptable weak issue to which the protection can be reduced.

Definition: Let's assume there are two difficulties, X and Y. If an algorithm exists that is polynomial in time as a function of X's input length and solves X using an oracle that solves Y, then X P Y (X poly time reduces to Y).

Goldwasser and Micali [15] work originates the concept of provable security. The goal of provable security is to provide a reduction from the security of the cryptosystem to some well-studied hard problem (the underlying problem). But if we make the assumption that the underlying problem cannot be solved by algorithms with a reasonable probability and at a cost less than any limit, then the reduction say something about the minimum cost of breaking the cryptosystem. Provable security generally consists of three steps: the first one is the rigorous notion of security, the second is the construction of a scheme, and the third one is the mathematical proof that the scheme satisfies the security notion given in the first step.

Reductionist Proof: The scientific approach to verifying safety is to link a way to split the scheme to a solution to a problem that is supposed to be complicated (for example,

factoring a large number that is a combination of two primes). A reduction proof works as follows: We take an arbitrary effective "algorithm A (i.e., the opponent) that attempts to disrupt our scheme. Then, it can be demonstrated that another powerful algorithm, B, using A, solves a difficult problem. The key point is that algorithm B is able to solve the problem as long as A succeeds in cracking the scheme. Since it is assumed that the problem is difficult, algorithm A splits the scheme with only a small probability. We point out that for any efficient algorithm attempting to crack the scheme, and not just some unique ones, such proof must hold.

Security Notions: Security notions are formal models or methods for analyzing the goal and power of an adversary in breaking the protocol. In this process, allow the adversary a certain privilege or power P and ask him to achieve a goal G. If someone succeeds, the cryptosystem is said to be G-insecure under the attack P. Else, it is called G-secure under P.

In real-life scenarios, it may not be possible to capture or model all possible privileges or power that an adversary may get during the actual execution of the cryptosystem and design a security notion accordingly. One of the possible reasons for this is that, as history shows, security is not proactive; it is always reactive. Once a successful attack is launched, then try to design another cryptosystem and corresponding security notions to resist that attack. But this modified primitive may be susceptible to some attacks of a different type, which are not modelled in the security notion or not known till date. Moreover, it is curious to observe that if a cryptosystem is known to be secure under an attack P, then it gives the adversary an implicit hint to try something other than P to break the system, i.e., security proofs, apart from giving its user, assurance of safety, also allows the adversary to exclude those modelled attacks and try something else to maximize its own motives. Apart from this, even poor or faulty implementation may lead to weak systems, which can be exploited by the adversary. On the other hand, the absence of any security model always leaves the system to be exploited in many ways when used in a larger setup. In fact, it has been evident from cryptographic literature that it took quite a long to and a realistic attack on some well-established cryptosystems with no security proofs. Thus measuring the odds in both

cases, it seems that it is better to have some security notion and proofs than to rely on heuristic security arguments without proofs.

Safe communication has different favours, depending on the security factor and interested in knowing how powerful adversaries may be. We are mainly concerned in this study with the confidentiality and reliability element of the communications.

The development and utility of another very different but closely related cryptographic primitive called the secret sharing schemes are discussed in the next section.

1.5 Secret Sharing Schemes

In public key cryptography, certain operations like encryption of a message using public credentials or verifying a signature can be performed by any user in the network. However, only the user who knows the corresponding secret/private credentials can perform the relevant decryption of ciphertext or sign a message.

In certain cases, the associated private information is very sensitive information, maybe a key for a secret mission etc. So, it becomes very difficult to rely on a single user or machine. A logical solution is to make shares of the secret key and distribute these shares among participants in a semi-trusted environment. Now, some authorized subset of the participants can cooperate to get the secret key. This approach leads to either distributed decryption or distributed signature schemes, depending on the feature being considered secret.

To accept the challenge of distributed nature of the scheme, SS has gained much attention since its inception. In addition to this, SS schemes act as primitive for many cryptographic protocols [6], such as multiparty computation, oblivious transfer etc.

A SS scheme allows for the splitting of a secret S into different n parts, called shares. These shares are distributed to a given set of participants P, so that only certain eligible subsets of participants can recover the secret with their respective shares. The subset of that eligible group of participants forms access structures, which can be joined to reconstruct the secret.

In 1979, Blakley [16] and Shamir [17] independently came up with their respective schemes, known as (t, n) threshold schemes. Later, as interest in this area started growing, SS schemes with features like general access systems (where eligible subsets are not all the same size), multiple secrets (when more secrets are to be shared), flexibility, and multi-usability (reconstruction of one secret does not jeopardize the security of the other secrets) were developed [18], [19], [20], [21].

However, as the number of secrets to be exchanged and the number of eligible sets in the access system for sharing multiple secrets with the general access structures grows, so does the size of the shares. Thus, these issues are discussed in detail in Chapters 2 and 4, respectively.

1.6 Cloud

Cloud or cloud computing [22] can be easily heard in day-to-day life, even by non-specialists. Applications like Gmail, Instagram, Dropbox, Google Drive, and many more are cloud-based. Cloud becomes the saviour when someone wishes to restore his Instagram or Facebook account when the old mobile breaks or is lost. All data which gets restored is all due to the cloud services only. In today's world, almost everybody is using its services. Cloud is a new technology that has emerged to satisfy the computing resource needs of users who cannot afford high-end hardware/software. The hardware/software services over the cloud are offered to users via the internet. Cloud has gained so much popularity in the recent past because the internet and related hardware have been very fast in the last few years. Also, cloud computing leverages cloud managers and administrators to use technology by minimizing human intervention efficiently.

Majorly the cloud provides four types of services, viz. Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), Infrastructure-as-a-Service (IaaS), and Function-as-a-Service (FaaS). Another classification in terms of deployment is - Private, Public, and Hybrid Cloud. Private cloud set-ups are owned by particular organizations, such as clouds owned by specific banks etc. On the other hand, the Public cloud is being shared by multiple organizations or individuals by means of virtual machines by paying the cost as a tenant, known as multi-tenancy, whereas the hybrid cloud act as a

combination of both. However, every technology comes with certain challenges. To name a few are cloud security & its privacy, access management, performance, compliance issues, multiple cloud management, and many more. The security aspect of data on the cloud as a major concern has been addressed.

1.7 Homomorphic Schemes

Maintaining the privacy of outsourced digital data on the cloud while doing computations is a major challenge. These issues can be addressed by encrypting data before outsourcing it and then computing over the encrypted data; this is the underlying idea behind homomorphic encryption [23]. Homomorphic Encryption strives to preserve privacy while also providing the ability to compute over encrypted data, search encrypted data, and so on. This additional infrastructure (computation over encrypted data) undoubtedly leads to a wide range of practical applications. Homomorphic Encryption has expanded into new fields as interest and inclination toward cloud computing have grown. A similar approach can be used to solve a variety of real-world issues. A systematic literature survey is conducted to address the mentioned issues and also explained (with the help of implementation results) in chapter 3. Homomorphism can be achieved in E-Voting, E-Auction, Secret sharing, and other for other real-world problems explained in the subsequent chapters.

1.8 Big Data

The last decade has shown mountainous growth of digital data. It is anticipated that the data produced from mobile and wireless devices will share 66% of the total IP traffic by the end of 2023 [24], [25]. It is also anticipated that two-third of the total population will be using the internet, i.e., the social media sites such as Facebook, Instagram, Twitter, Youtube etc. Further, IoT devices also contribute to the magnanimity of the data being processed, stored, and transmitted today. IoT devices are basically resource-constrained devices capable of intercommunicating without human intervention, which finds extensive applications, such as communicating sensory information in sectors like healthcare, agriculture, manufacturing industries, etc. With such applications and the amount of data generated by the people, efficient data analysis is inevitable, which

stimulates the requirement of parallel and distributed methods for the analysis of big datasets. Big Data is described as a dataset that is too vast and complicated for typical database management systems and algorithms to handle. Generally, big data is used to analyze unstructured data, which means that data for which schema is not adequately defined and is heterogeneous in nature. However, contrary to this notion, structured data can also be treated as big data if the traditional algorithms, which are sequential in nature, cannot process it within a reasonable time. Gartner and Laney [24] introduced the 3Vs concept defining big data as high volume, velocity, and variety. These three forms make it unsuitable for processing via traditional methods. However, till date, 8Vs of the big data are presented here, which are explained as follows:

1. **Volume:** Volume refers to the huge size of the data that becomes beyond the abilities of the conventional hardware and methods to handle it. The size of the data determines whether it should be considered big data or not.

2. **Velocity:** It refers to the speed at which the data is generated. For example, the streaming data such as Twitter tweets, Facebook comments, and customers are generated for e-commerce products. The traditional database techniques are not able to handle such fast streaming real-time datasets.

3. **Variety:** It means the data can be in any format, such as structured, unstructured, or semi-structured. The unstructured data does not have a proper schema, and it may consist of text, audio, or video. For example, call records, sales records, and spreadsheets are examples of unstructured data. Further, semi-structured data are like structured data, but it is not organized in RDBMS format but rather separated by tags or with some other markers.

4. **Veracity:** Veracity deals with the reliability or the trust of the source for the dataset. It also refers to, how meaningful it is to analyze the data with confidence.

5. **Volatility:** It refers to the life of the stored data, which means for how long the data is valid. For example, the one-time password (OTP) of the banking applications is valid for a few minutes.

6. **Value:** It refers to the importance of the data from the business perspective. The value of big data deals with revenue generation, customer satisfaction, profit, and the relationship of the businessman with the customer.

7. **Visualization:** Traditional data visualization tools face difficulty due to poor scalability response time. Traditional graphs are not appropriate for plotting billions of data points. Thus, modern tools and techniques are required for representing big data sets.

8. **Vulnerability:** It refers to how vulnerable your data is in terms of security. Thus, the security of big data is also a prime concern and should be handled carefully.

As already mentioned, it is not easy to analyze with the help of traditional methods. The increasing growth of digital data has raised new challenges to the existing data analysis methods. Various tools and techniques have been developed for analyzing big datasets. The primary sources of the big datasets are social media, sensor devices, satellites, and mobile devices, and generally, the labels of such datasets are not available. Clustering is one of the popular data analysis techniques that group data items into different clusters. In contrast, data in the i^{th} cluster differ from data in the k^{th} cluster, whereas i is not equal to k [26]. Over the past several years, several methods have been proposed for solving clustering problems. K-means is used to cluster for many of the clustering problems that are being used due to its simplicity [27], [28]. However, it does not meet the practical requirements of managing big data, and hence fuzziness is introduced in this algorithm leading to fuzzy c means clustering. Further, Hadoop and MapReduce [29] are open-source tools that are being used to keep a check on the performance. It allows distributed and parallel processing, thus helping to improve performance. MapReduce is a parallel programming model which works on the top of the Hadoop [30], [31], [32]. The proposed work has successfully utilized the architecture of Hadoop for distributed processing and the MapReduce model for parallel programming.

1.9 Industry 4.0 and Related Technologies

In Industry 4.0, the manufacturing industry (MI) has seen a paradigm shift from traditional manufacturing processes towards smart manufacturing through simplification in three aspects- communication through ubiquitous connectivity, decision-based analytics, and responsive automation. To leverage an efficient solution for MI stakeholders, modern manufacturing units involve integrating control systems, communication channels, and computational devices that communicate with each other through embedded sensor units. With the tremendous rise in sensor devices, the massive generated data needs to be efficiently and responsively analyzed for decision analytics at server nodes through low-latency open channels. Thus, the exchanged data packets are vulnerable to security and privacy attacks, leading to mission-critical applications' failures. Also, centralized server nodes induce high latency in channel communication; thus, to efficiently model a smart and responsive solution, decentralization is applicable. However, an inherent notion of trust, auditable and immutability is necessary to secure data among decentralized edge nodes. This facilitates energy-efficient data exchange modalities that leverage the triple benefits of mass-scalable production, portability of sensor components, and improved processing latency [33]. In the Internet of Things (IoT), internet-connected devices are becoming increasingly smart, in the sense that most of the functions (such as collecting environmental data and sending it to edge or cloud servers) may be completed with minimum human interaction. IoT devices or smart gadgets/objects are examples of these products, which can be physical or virtual. A device ID or an IP address assigned to smart devices is uniquely identified. Setting up an IoT ecosystem comes with various issues, one of which is security. Communication between users, smart devices, and GNs, for example, is usually done across insecure channels. In other words, the communications may be intercepted, hijacked, destroyed, or manipulated (for example, by introducing forged messages), among other things. This comprises designing and implementing a secure and efficient user access control mechanism in the IoT system, ensuring that only authorised personnel have access to critical information and/or services. To address the issues of decentralized trust in open

channels, Blockchain-based solutions for edge-IoT nodes can be employed in smart MI to offer trust, immutability, and reliability of stored data in those nodes.

Blockchain is a distributed ledger that has immutability and traceability as some of its core characteristics. Hence, there is no scope for communicating entities to distrust each other, and any attempt to violate the integrity of the transactions on the blockchain will get immediately detected. This is because multiple participating nodes maintain a copy of the entire blockchain, and any new block to the blockchain is added through a consensus algorithm, which is run by miner nodes. The consensus algorithm ensures that only valid blocks are added to the blockchain. Furthermore, each block in the blockchain contains a hash of the preceding block, which is used to maintain the integrity of the transactions recorded in the blocks, ensuring the necessary immutability and traceability characteristics.

In IoT ecosystems, sensor nodes forward data to server nodes through low-powered IoT protocols, which are not compatible with resource-intensive consensus protocols. Thus, consensus protocols like Proof-of-Work (PoW) and Proof-of-Stake (PoS) are not scalable with IoT devices, as services through such protocols suffer from bottlenecks in large-scale ecosystems [34]. Thus, an energy-efficient consensus mechanism is needed to be deployed that ensures scalable IoT ecosystems. To address the mentioned challenges, researchers have explored the possibility of securing low-powered and wide-range IoT Personal Area Networks (PAN) [35]. Similarly, other works are proposed to minimize the packet header structures over 6LowPAN to mitigate network overheads [36]. The mentioned approaches rely on the availability of the coverage range of wireless devices to reduce computations. Moreover, reduced packet structure may adversely affect security measures, which otherwise are added, and this may further be exploited by a malicious entity. Thus, proper security measures are required to be deployed for end IoT devices that ensure authentication through reliable signing procedures. To explore the same, researchers have proposed registration schemes for end devices that grant access only to authorized users. Once the user is authenticated, secure data exchange occurs through shared secret keys [37].

Registration-based schemes for end-IoT devices suffer from limitations of high bottlenecks and bandwidth issues at server nodes. Due to this, proximity sensor networks elect a local manager (LM) in each wireless sensor network (WSN). The role of LM is to identify packet headers and, based on destination addresses, form necessary signing and encryption of sensor data in the local coverage range [38]. Thus, consensus among LMs can be formed in BC to ensure secure data exchange through efficient signing and encryption procedures [39]. As discussed above, to effectively leverage smart and responsive MI through constrained IoT ecosystems, efficient signing and encryption procedures for elected LMs are required. To design an effective security mechanism that ensures sensor confidentiality and data integrity, the signing and encryption schemes cannot perform in isolation. Signcryption schemes facilitate an effective and lightweight scheme for real-time responsive automation in MI that provides a scalable solution in Industry 4.0 ecosystems.

1.10 Research Gaps and Thesis Objective

As discussed, security is an integral aspect of day-to-day communications. In addition to the standard encryption process (cryptography), there is a need to have additional properties in the security provisions to cater different challenges posed due to the emerging technologies and requirements of Cloud computing, extensive deployment of IoT devices, and handling of Big Data etc. A comprehensive literature review of recent security solutions to different security challenges reveals some gaps; thereby, thesis objectives are discussed:

- The very first aspect which was looked upon is Homomorphic Encryption for securing cloud computing applications. Fully homomorphic schemes are practically hard to describe, and the universal homomorphic scheme is practically impossible, which can provide all operations of indefinite depth. However, the concern is to have a sturdy scheme which can resist computation of a certain depth.

- Sharing of Cryptographic keys [40], [41], [42], [43], [44], [45] for modern-day communications is very important and happening at unimaginable high rates. These communications either require sharing keys or distributing the secret value among the shareholders, which later can combine to reconstruct the secret. No such scheme is available in the literature which can share multi-secret in a multilevel environment. Sometimes as per practical requirements, it is required to share the secret in multiple levels, and shareholders from a higher level may contribute in secret reconstruction. Another chllange is the security of the scheme. Some schemes are theoretically secure because of strong mathematical primitive; however, to rely upon these number-theoretic problems or to furnish any other additional information, which in turn, may add to the computational overhead of the scheme.

- Further, recent communications include mass-scale transactions happening without human intervention, including IoT devices. Whenever there is some human-less interaction, a kind of distrust is there, i.e., the participating entities do not trust each other. In such situations, Blockchain, as a technology with inherent distributed trust, can provide a suitable solution. Blockchain (BC) as a technology is being used in almost every application, varying from healthcare, payment transfer, industrial IoT, and many more. However, the BC-based solutions are struggling with security and efficiency challenges which are required to be addressed to make them practically suitable for real-time on-ground applications. BC also allows automation in MI stakeholders through smart contracts (SC) that are executed when specified, and both parties meet agreed functionalities. Thus, it eliminates intermediaries at every point in the MI flow cycle. Although BC offers a trusted exchange of data among open channels, IoT-based schemes still suffer from the limitations of resources, minimal storage in end devices, low bandwidth, and variable packet delays due to poor wireless connections [46]. Thus, an energy-efficient mechanism is needed to be deployed that ensures scalable IoT ecosystems.

- Also, existing Big Data processing methods face many scalability and efficiency issues. For security, traditional encryption schemes cannot efficiently accommodate the huge volume of big data. Additionally, updating the access policy for different users is a big challenge that needs to be handled in view of meeting the requirements of prevailing conditions.

1.11 Contribution and Thesis Layout

This section mentions the contribution. The thesis is composed of seven chapters in all. The chapters are structured as follows:-

Chapter 1 introduces the work in the thesis, stating the basic terminology and concepts. It provides a fundamental idea about the concepts of Information security. It also comprises the basics of Secret Sharing Schemes, Big data, ECC, Block Chain, and Homomorphic Encryption.

Chapter 2 briefly discusses the literature available. This chapter starts with homomorphic encryption; three possible construction for homomorphic schemes are discussed, followed by a bit of detail about Gentry's scheme. A few other techniques are also discussed in this league. Next to that, different types of secret sharing, i.e., verifiable secret, multi-secret, and multi-level schemes, are discussed. Apart from this, lightweight security mechanisms for resource constraint environments (IoT) have been discussed. Lastly, the access control mechanism in reference to Cloud has been talked about. This chapter concisely discusses the literature review. The detailed literature review has been addressed in the respective chapters.

Chapter 3 presents homomorphic scheme constructs: lattices, approximate GCD, Learning with error, and Integer factorization. These three constructs are used to develop a lot of homomorphic schemes. A number of homomorphic schemes are implemented and analyzed with respect to security parameters. Encryption and decryption time are also analyzed for different inputs. It is shown that fully homomorphic computation can be applied to a certain depth in some practical

applications. Here homomorphic schemes are extensively discussed, which form the necessary background for the rest of the chapters.

Chapter 4 presents a verifiable secret-sharing scheme that can work in multi-level and multi-secret environments. A detailed outline is also depicted with the help of a complete block diagram. The proposed scheme is compared with the existing work, and the result shows that the proposed scheme outperforms in terms of security, efficiency, and communication costs.

Chapter 5 discusses the advancing security provisions that are required to automate the manufacturing industries under the hood of Industry 4.0. Such automation requires frequent humanless communications in resource-constrained IoT-based environments, posing serious security challenges. A BC-envisioned lightweight scheme is presented that sends data through elected CH from PSN. CH uses a secure signcryption scheme that minimizes computational overheads. An energy-efficient consensus mechanism is proposed to improve latency and storage overheads. This chapter addresses the security concerns by proposing a blockchain-based solution that includes an energy-efficient, lightweight consensus mechanism.

Chapter 6 discusses the scalability and efficiency issues mainly due to the increased volume of data. The increased volume of data occupies more space for storing the data in the cloud, making it difficult to process. Traditional schemes have many flexibility issues, and therefore they cannot provide complete security to the data. The chapter proposes a solution integrating existing methods like Fuzzy C Mean clustering, the Elliptic Curve cryptography encryption algorithm that is optimized with the Grasshopper optimization algorithm. The chapter demonstrates the implementation and comparative results that use the Hadoop MapReduce framework in the CloudSim environment.

Chapter 7 provides a summary of the proposed methods. It also delivers the conclusion and presents some technical recommendations for future research directions.

References: This section gives the reference details of the thesis.

CHAPTER 2

BRIEF LITERATURE REVIEW

Information Security is an integral aspect of digital communications. In addition to the standard encryption process, there is a need to have additional properties in the security provisions to cater the different challenges posed due to the emerging technologies, Cloud computing, extensive deployment of IoT devices, handling of Big Data etc. In view of the above discussion, there are four different areas: secret sharing schemes, homomorphic encryption, secure big data mechanisms, and lightweight solutions for blockchain. These four fields seem different, but every area has some associated application or other integral correlation. For example, secret sharing is a critical application of Homomorphic schemes. Another aspect to consider is that cloud security and its discussion would be incomplete without homomorphic encryption, which is essential in cloud security management. Further, big data and the cloud are also closely related to each other, and the latter functionalities help handle the big data management. Lastly, lightweight security management may be achieved using blockchain to ensure cloud safety mechanisms.

In this chapter, the literature review is briefly discussed, and their extensive literature review has been discussed in the respective chapters.

2.1 Secret Sharing

The concept of secret sharing (SS) schemes was coined by Shamir [17] and Blakley [16] in 1979. Since then, it has attracted the interest of several researchers. SS has been found valuable in several applications, such as witness encryption [17], secure communication [47], and access control [48]. Some drawbacks of the SS schemes presented in [16], [17] which may act as a constraint for practical usage. Some of them may include fake shares distribution by the malicious dealer; deceitful shareholders may submit a fake/invalid share, need of mutually trusted dealer, the requirement of private channel for share distribution etc. An advancement of SS schemes, known as verifiable secret sharing (VSS) schemes, came into the picture to handle the dishonesty of shareholders mainly or sometime dealer too. To make SS

verifiable, some auxiliary information is to be added that helps the share-holders to verify their respective shares. The shareholders do not accept their respective share, if they find them inconsistent or invalid. With the help of VSS schemes, it is possible for the shareholders to verify their shares without having access to the secret and without revealing their shares. Other flavors of SS schemes include multiple [49] multilevel [50], weighted [51], and protected SS (PSS) schemes [52].

In multiple SS schemes, there are multiple (say p) secrets instead of a single, as in traditional SS schemes. To share *p* secrets, one approach is to run *p* instances of the simple scheme. However, this seems to be a very naïve way and not a desirable solution due to high computational complexity. So a scheme is desirable if a single run [49] can share all p secrets. Recent work in this direction is done by Amroudi *et al.* [53], where authors obviate the need for a secure channel by encrypting the shares with the NTRU cryptosystem, which is a lattice-based and reasonably fast approach. A multivariate polynomial coefficient is used to share the multi-secret with the verification of shares performed using the hash function. Another work in this league is a scheme by Meng *et al.* [54] that uses cellular automata and the hash function for verifiability. Trust management without the dealer is achieved with the help of linear and parallel computations to increase efficiency. Another multiple SS scheme is proposed by Tentu *et al.* [55], where multiple secrets are distributed using discrete logarithms and quadratic residue problems. This scheme is used for the level-ordered access structure. Hu *et al.* [56] proposed a verifiable multi-secret sharing based on the Lagrange polynomial and the public key cryptosystem. They used a linear feedback shift register (LFSR)-based cryptosystem to enhance the scheme's efficiency. The scheme provides reasonably good security with added efficiency. In the recent past, Giri *et al.* [57] proposed a multi-scheme whose assumption is based on the geometry in the finite field. The scheme is claimed to be secured as the distributed shares are not the actual ones but their shadow values. Liu *et al.* [58] proposed a multi-secret scheme that proved the failure of asynchronous reconstruction of shares given by Harn and Hsu [59]. They also proved that by reconstructing any secret, the rest of the secrets could be obtained illegitimately. They improved the abnormality of the scheme by taking the common pairwise key for a pair of shareholders. In multi-level

or hierarchical SS schemes [50], participants are divided into m different levels, and a threshold is associated with each level. For secret recovery, a participant from the targeted level or higher level can contribute in the secret reconstruction. Zhong *et al.* [40] extended the idea of giving a shadow number to images. A shadow image is a share that stops the cheating of shareholders before the actual image is recovered. They extended the idea of a weighted scheme by giving higher priority to the shareholder at a higher level; i.e., priorities should be decided as per the capabilities of shareholders at different levels. In weighted SS schemes [51], a weight with a positive value is assigned to each participant. Secret reconstruction is possible only when the sum of weights of the authorized subset is equal to or greater than the threshold. In previously described schemes [16], [17], [47], [48], [60], [61], each shareholder has unity weight. However, in this, different shareholders may be assigned different weights. The concept of such schemes can be applied directly when there is a need to give more rights to higher rank officials. In SS schemes, traditionally, to avoid the chances of recovery of a secret by non-shareholders, secure pairwise channels are established among the shareholders via a shared key. To reduce this computational inefficiency, Harn *et al.* [52] coined PSS scheme. In addition to the secret reconstruction, the shared key is also established with the help of shares possessed by the shareholders in a pairwise manner. Though this scheme is computationally less efficient than Shamir's SS scheme, it can be used even if the adversary has unlimited computational power. Another class of VSS schemes is known as the publically VSS (PVSS) scheme [41], [42]. Such VSS schemes possess a unique property that anyone can verify that distributed shares are valid or not, i.e., the maliciousness of the shareholders can be handled by this type of scheme. Shareholders receive a valid share but do not submit a valid one during reconstruction. A remarkable PVSS scheme was presented by Behnad *et al.* [43], where members were selected in order to avoid illegal member participation. With the proliferation of big data and cloud computing technology and its associated requirement, homomorphic secret sharing schemes were proposed in the recent past. Though the concept of cryptographic homomorphism is ancient, it has been touched by various researchers from time to time. Li *et al.* in 2018 discussed the various cryptographic primitives which can be used for privacy preservation requirement of

various online applications [44]. A scheme by Rajabi *et al.* [45], whose security is based upon the approximate shortest polynomial problem, exploits homomorphic as well as collision resistance property by taking appropriate Knapsack.

2.2 Homomorphic Schemes

In 1978, the concept of Privacy homomorphism was introduced by Rivest, Adleman, and Deatouzous [62], and some security flaws were observed. Since then, the search for encryption schemes with special properties (for ex., homomorphism) was set out. Homomorphic encryption schemes allow computing operations directly on the cipher text. A scheme allowing a fairly limited number of operations is said to be somewhat homomorphic, and if it allows unlimited operations, then it is said to be fully homomorphic. Three directions were identified, i.e., Ideal Lattice, Approximate Greatest Common Divisor (GCD), and Learning with Errors. In the Lattice-based direction, Gentry's [63] scheme is a popular fully homomorphic scheme and is considered a blueprint for further improvements. However, it has high computational requirements, making it impractical for practical applications.

In Gentry's scheme, the pair of keys and the ciphertext is represented by a mesh of ideal lattices in the Gentry scheme. Many researchers and companies like Google, IBM, and Microsoft worked on Gentry's approach, addressing the issues was making it practical for industrial applications. The challenges regarding the efficiency and effectiveness of lattice as primitives were addressed, and new primitives like approximate GCD and learning with error were also explored to address the aforementioned challenges [63], [64], [65], [66]. Though lattice is proven to be secure and efficient primitive for many cryptographic schemes, however, it is not well suited for Gentry full homomorphic construction for practical usage. Bootstrapping was the approach that was used to convert a scheme from somewhat to fully homomorphic. It was used to reduce the noise generated in the ciphertext during operations. The number of operation that is being performed depends upon the depth of the circuit taken for the appropriate operation [63], [67]. Gentry's bootstrapping step is only allowed for low-depth decryption operations. As

a result, certain "methods" are used to reduce the decryption's complexity. Squashing was introduced to handle the above complexity. Gentry's technique requires selecting a group of vectors whose sum equals the secret key's multiplicative inverse. When the ciphertext is multiplied by the set of elements, the circuit's polynomial degree is reduced to a level that the algorithm can handle. The ciphertext should be "bootstrappable" in this case. Sparse Subset Sum Problem (SSSP) [68] demonstrates the algorithm's provable security. Bootstrapping, in a general sense, is a decryption technique that generates a clear ciphertext from noise-based encryption that matches the plaintext. An algorithm is said to be "bootstrappable" if it can evaluate its own decryption circuit [63]. The whole idea is to get a noise-free ciphertext from the defined procedure. The bootstrapping is done in two steps. The first step generates a pair of secret and public keys. The private (secret) key is kept with the user, and the public key is shared on the server. Then, the secret is sent to sever in the encrypted form, where all the ciphertexts are present. The encrypted secret key is used in decrypting the ciphertext in a homomorphic way. The result is having all the desired operations in the evaluate function as well. The operated text is in the plain form and encrypted using public key and can be decrypted by the respective user [67]. As the algorithm is semantically secure, an adversary cannot tell the difference between encrypting the secret key and encrypting an arbitrary number. Since, the technique is regarded semantically secure [67], it can encrypt any number. To summarize, the noisy ciphertext is homomorphically decrypted to remove the noise, and then unique homomorphic encryption introduces little noise into the ciphertext. The ciphertext is now just data that has been encrypted. On this "clear" ciphertext, more homomorphic calculations can be done until a threshold point is achieved. Fully homomorphic encryption schemes allow a third party to fully run arbitrary computations on encrypted plaintext as needed without having to learn any part of the inputs or calculation outputs. While fully homomorphic encryption methods have advantages, they also have drawbacks. Gentry's bootstrapping strategy, for example, greatly increases the computing cost and represents a significant barrier to

completely homomorphic encryption viability. IBM released a library for the implementation of homomorphic encryption in 2013. Apart from the software-based issues, Moore *et al.*[69] investigated hardware-based problems in handling multiplication algorithms and hardware architecture for large integer multiplication. For this, they used FPGA technology to develop the building blocks of partially and fully homomorphic encryption methods to test their practicality. In 2013, Cheon *et al.* [70] extended the DGHV into a batch fully homomorphic encryption scheme, i.e., a scheme that supports encrypting and homomorphically processes a vector of plaintexts as a single ciphertext. Apart from the above-mentioned direction, there are other directions, like schemes based on integer factorization [71] and schemes based on NTRU [66], [72]. A more detailed chronological development is discussed in chapter 3.

2.3 Lightweight security solution applicable in resource-constrained IoT environments

Another major area contributing significant proportions of data over the cloud and hence to Big Data is in the form of IoT. There is significant research that is ongoing in the area of security solutions in the IoT environment. To address the issues of secured lightweight data exchange through sensors in constrained IoT environments, Mohanty *et al.* [73] proposed an effective and lightweight encryption model employing blockchain through an aggregated overlay network. However, the security evaluations for key exchange are not presented. Dorri *et al.* [74] proposed distributed consensus model through CH election to reduce processing overheads of mining blocks. For the same, the authors have computed BC- throughput from cumulative transaction loads. However, the security aspects of data exchange are not explored. Zhang *et al.* [75] suggested a lightweight data exchange algorithm for secured consensus among participating stakeholders in Industrial IoT applications. For the same, the authors have computed average hop count values of data exchange, with energy computations at hop counts. Ali *et al.* [76] proposed a symmetric encryption scheme for metering operations in smart grids with

lightweight key exchange phases. Huang *et al.* [77] proposed a credit-based proof-of-work (PoW) consensus scheme in IoT that facilitates a secure system and transactional efficiency with the inherent notion of data confidentiality. The model is proposed on directed acyclic graphs (DAG) with simulations performed over raspberry PI in smart factory eco-systems.

2.4 Big Data Management

Nowadays, data security and access control mechanism has become the topmost priority. Security of data is essential for individuals and also for the organization. This has become an important matter of concern due to the excess usage of data. The amount of data that is being created is enormous in size, and their mode of applications is also vast such as in mobile phone applications, social networking applications, and various types of sensor nodes and IoT devices. Along with the applications, many challenges are also associated with such a large size of data which is also termed big data. Of many challenges, some of the major issues with big data are its storage security management and its verification, system customization (as per the requirement), cost-effectiveness, seamless and secure access management. Few technologies that came as a solution to the aforementioned challenges are distributed computing, i.e., cloud computing, access control mechanisms that may be based on the attribute of user identity, and encryption techniques to handle the security aspect, i.e., confidentiality, integrity, and authentication. Some of the recent studies [78], [79], [80] suggested that the area of cloud computing and its associated techniques can cater the big data-related problems. Another field, homomorphic encryption [81], [82], is also growing popular in handling the computation in the cloud over encrypted data, which would definitely help the community in terms of privacy preservation and industry requirements. Singh *et al.* [83] and a few other authors [84], [85] in their studies have proposed an access control mechanism with respect to the user's attribute to access the cloud. There is always a need for practical and efficient solutions to modern-day applications.

CHAPTER 3

ANALYSIS AND IMPLEMENTATION OF HOMOMORPHIC SCHEMES

The need to outsource the data is increasing at an alarming rate. The prime concern is to protect the privacy of digital data and to perform mathematical computations on it. These issues can be handled if data is encrypted before outsourcing it and then performing computation over the encrypted data; this is the basic concept of Homomorphic encryption. The core purpose of homomorphic encryption is to assure the confidentiality of encrypted data for performing mathematical computations. Certainly, this additional capability (performing computation over encrypted data) leads to many practical applications. The rising interest of users in cloud computation has unlocked various fields for Homomorphic Encryption. Many real-world problems can be addressed by the same. This chapter explains how homomorphism can be useful in various practical applications such as E-Voting, E-Auction, Secret sharing, and other real-world problems.

3.1 Asymmetric Encryption and its Way Forward

Asymmetric encryption is the major area under which homomorphic encryption is investigated. Asymmetric encryption uses two different keys for encryption and decryption, unlike symmetric encryption. The data is encrypted with one key and decrypted using the other. The term "public key cryptography" is frequently used interchangeably. The encryption and decryption keys are termed public and private, respectively. These two are linked via some mathematical function, and it is almost impossible to get any information regarding the private key with the known public key. This is something very integral regarding public key cryptosystems.

3.2 Group Homomorphism

Group Homomorphism means homomorphism between groups. Let us define two groups $(G,*)$ and $(H,+)$,where * and + are respectively operations defined on the defined groups. A homomorphism from G to H is a function $\varphi: G \rightarrow H$ that satisfies

for all $a, b \in G$, $\varphi(a * b) = \varphi(a) + \varphi(b)$. The operations taken for the defined groups are arbitrary operations, as explained in Figure 3.1. If φ is a one-to-one function, then it is said to be monomorphic, and if it is onto, then it is said to be epimorphic, and if it is bijective, then it is said to be isomorphic. This isomorphic property of the groups ensures the correct computation on the encrypted data. An example showing group homomorphism is presented here:

Let two groups be $(\mathbb{R}, +)$ and $(\mathbb{R} - \{0\}, *)$, where + is known as addition, and * is known as multiplication. φ is defined as a function from $\mathbb{R} \rightarrow \mathbb{R} - \{0\}$ by $\varphi(x) = e^x$. Take two elements p and q that belong to the first group. So $\varphi(p) = e^p$ and $(q) = e^q$, which can be easily verified that $\varphi(p + q) = \varphi(p) * \varphi(q)$.

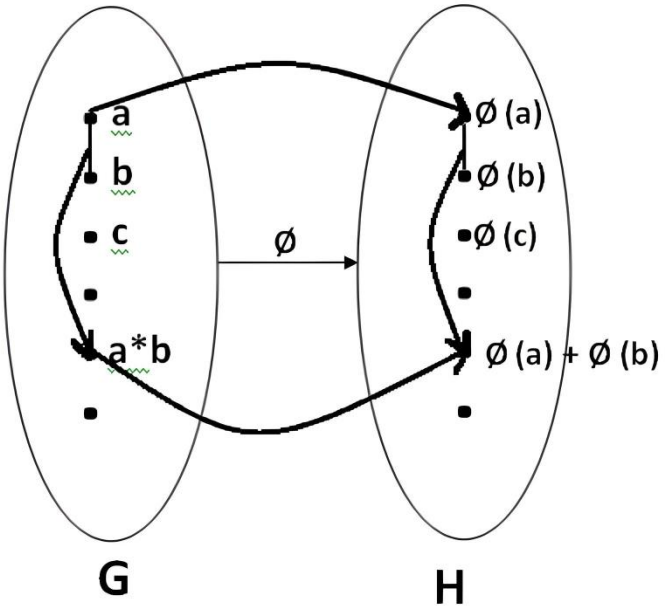

Figure 3.1: Schematic Diagram of Group Homomorphism

3.3 Homomorphic encryption

In recent decades, cryptographic schemes, particularly homomorphic schemes, have been studied extensively because of their essential property of performing mathematical operations on encrypted data and getting the same operation done on actual plaintext. For a given plaintexts *P1* and *P2* and the ciphertext *C1* and *C2*, then homomorphic encryption allows computation directly on ciphertext $C1\ operation\ C2$ without revealing $P1$ or $P2$, and decryption can be done thereafter.

Homomorphic encryption schemes consist of four types of algorithms, which are discussed below:

Keygen (λ)

- Input – security parameter λ.
- Output – pair $(sk, pk) \in$, where sk, pk, K denote a secret key, the public key, and Key Space, respectively.

Encrypt (pk, π)

- Input – a public key pk and a plaintext π.
- Output – a ciphertext ψ.

Decrypt (sk, ψ)

- Input – secret key sk and ciphertext ψ.
- Output – the corresponding plaintext π.

Evaluate (pk, C, ψ)

- Input – a public key pk, a circuit C with t inputs and a set ψ of t ciphertext $\psi 1, \psi 2, \dots, \psi t$.
- Output – a ciphertext ψ.

If ψ_i is the generated ciphertext for $i = 1 \dots t$ and $\psi = (\psi_1, \dots\dots, \psi_t)$, then Evaluate function (pk, C, ψ) return ciphertext ψ, which is an equivalent form of operated plaintext, i.e., $C(\pi_1, \dots\dots, \pi_t)$ for a circuit C with t inputs.

A homomorphic scheme works efficiently (a set of circuits) when it correctly evaluates the desired operation on the plaintext that holds for all circuits $C \in C$.

The additional algorithm Evaluate will follow the following steps as depicted in Figure 3.2.

If ψ_i is a ciphertext corresponding to the plaintext π_i for $i = 1 \dots t$ and $\Psi = (\psi_1, \dots\dots, \psi_t)$, then **Evaluate** (pk, C, Ψ) shall return a ciphertext ψ corresponding to the plaintext $C(\pi_1, \dots\dots, \pi_t)$ for a circuit C with t inputs.

The scheme will evaluate $\mathcal{C}$ (a set of circuits) when the correctness-condition on the algorithm **Evaluate** holds for all circuits $C \in \mathcal{C}$.

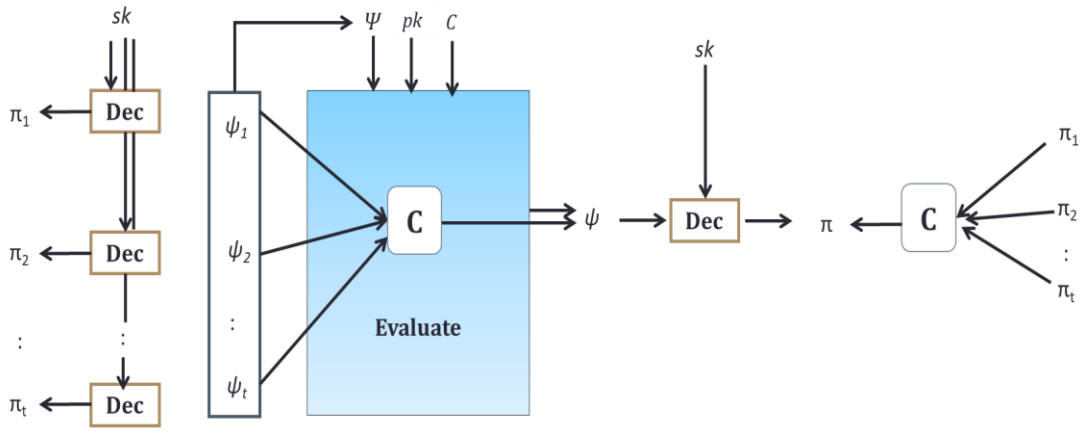

Figure 3.2: An Illustrative View of the Evaluate-Algorithm

3.3.1 Somewhat Homomorphic Encryption

Any scheme is termed as somewhat homomorphic if it can manage with the limited number of operations. Such operation may be addition, multiplication, or any other desired operations, i.e., the depth of the decryption circuit can go up to a limited extent. Present schemes in this category are RSA, which is multiplicative homomorphic, and Paillier, which is additive homomorphic. Both of the schemes show homomorphism for single addition and multiplication, respectively.

3.3.2 Fully Homomorphic Encryption (FHE)

In contrast to the partly homomorphic system, an encryption scheme is said to be fully homomorphic if it permits an endless number of operations. All of the circuits and the size of its decryption method (as a circuit) that are bound by a polynomial in the security parameter are accurately evaluated. Patrick [86] demonstrated the FHE using Figure 3.2 and argues that the size constraint of the decryption algorithm excludes trivial schemes in which **Evaluate** merely outputs (C, Ψ) and the decryption algorithm **Decrypt,** every single section of π is decrypted first and then consequently circuit C can be applied to the decrypted part.

3.4 Development of Homomorphic Schemes

Whitfield Diffie and Martin Hellman [11], [87], [88] were the first ones who invent Public key cryptography in 1976 due to this reason, and it is also termed as Diffie-Hellman encryption. These techniques are only secure if a difficult mathematical issue remains difficult and cannot be solved in polynomial time. However, the major shortcoming is the encryption and decryption speed, which is quiet less as compared to symmetric schemes due to mathematical computations, which are non-trivial in nature. This is the core reason that public key schemes are primarily used to exchange small data or key management.

The notion of Privacy homomorphism was coined in 1978 by Rivest, Adleman and Deatouzous in 1978 [62], and there are some security flaws have been found in it. Since then, the search for encryption schemes with special properties (for ex., homomorphism) was set out. The development of homomorphic schemes started from 1978 to 2007, as depicted in Figure 3.3. Some of the integral schemes are discussed in detail.

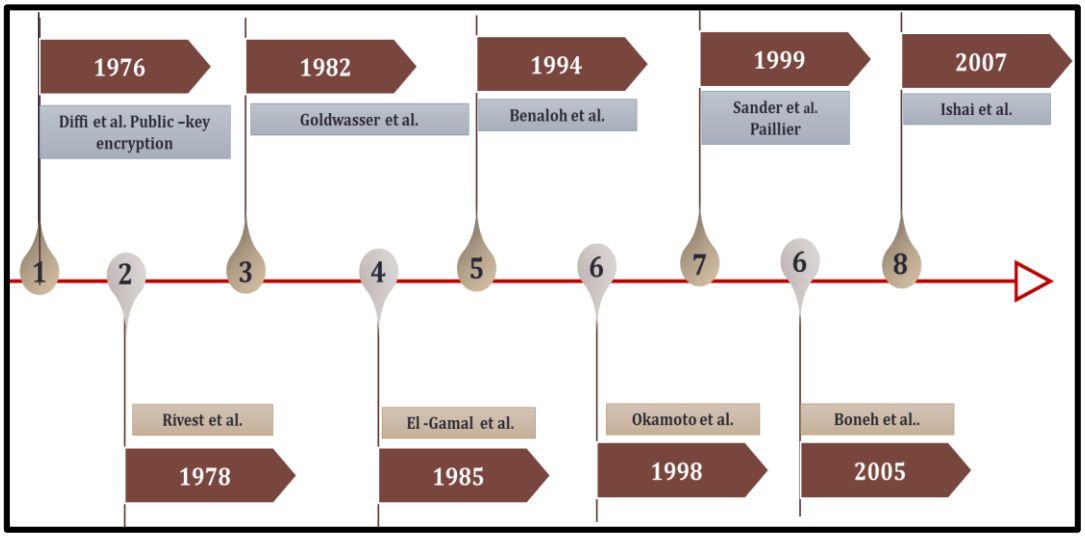

Figure 3.3: Development of Homomorphic Encryption Algorithms from 1976 to 2007.

Goldwasser-Micali Scheme in 1982 [15], [89] is considered as first probabilistic public-key encryption scheme. It is proved that it can be secured under the assumptions of cryptography. In the Goldwasser–Micali cryptosystem, if the public key is the modulus m and quadratic non-residue x, then the encryption of a

bit b is $\varepsilon(b) = x^b r^2 \, mod \, m$ for some random $r \in \{0, \dots, m-1\}$. The homomorphic property is then

$$\varepsilon(b_1).\varepsilon(b_2) = x^{b1} r_1^2 x^{b2} r_2^2 = x^{b1+b2}(r_1 \, r_2)^2 = \varepsilon(b_1 \oplus b_2) \quad (3.1)$$

where $\oplus$ denotes addition modulo 2, (i.e., exclusive-or).

In 1985, ElGamal *et al.* [13] developed public key cryptography using elliptic curve cryptography. It is the most widely used cryptographic scheme.

Benaloh cryptosystem [90] is an extension of the Goldwasser-Micali cryptosystem with nearly the same encryption cost but with an increased decryption cost.

In the Benaloh cryptosystem, if the base is *g* and the public key is modulus *m* with a block size of *c*, then the encryption of a message x *is*, $\varepsilon(x) = g^x r^c \, mod \, m$ for some random $r \in \{0, \dots, m-1\}$. The homomorphic property is then

$$\varepsilon(x_1).\varepsilon(x_2) = (g^{x1} r_1^c)(g^{x2} r_2^c) = g^{x1+x2}(r_1 \, r_2)^2 = \varepsilon(x_1 + x_2 \, mod \, c) \quad (3.2)$$

Okamato & Uchiyama (1998) [91] proposed to change the base group *G* in order to improve the performance of earlier homomorphic encryption schemes. Taking $n = p^2 q$, where *p* and *q* are two large prime numbers and group $G = Z^*_{p^2}$, they achieved $k = p$. The security of this scheme rests on the hardness of determining whether a number x in $(Z/nZ)*$, also belongs to subgroup of order p. However, it has been proposed that an attack of ciphertext can break the factorization scheme, due to this reason, it is not a widely used scheme.

Nacche & Stern (1998) [92] presented an improvement to Benaloh's (Benaloh, 1987) scheme. This scheme was much more efficient when the parameter *k* used in Benaloh's scheme was chosen to be of greater value. The proposed encryption method was nearly the same as in Benaloh's scheme, but the decryption method was different. The improvement reduced the cost of decryption.

Paillier (1999) [93] proposed an efficient, additive, scalar, and probabilistic scheme based on an arithmetic ring of N^2 where *N* is the product of two large primes numbers. The author extended his proposal to the elliptic curve Paillier scheme. The elliptic

curve Paillier scheme is much slower than the original Paillier scheme as it computes on an elliptic curve modulo large numbers. However, the cost of decryption is too high in this scheme as it requires exponentiation modulo N^2 to the power $\lambda(N)$ and multiplication to the modulo N. This scheme had a smaller expansion in comparison to other encryption schemes and thus had great acceptability.

Damgard- Jurik [94] proposed a generalized form of Paillier's probabilistic scheme to groups of the form Z_n^{s+1} *for* $s>0$. They achieved lower values of expansion by choosing larger values of *s*. This scheme was computationally more expensive than Paillier's scheme. It was also proved that the semantic security of this scheme depends upon whether the two given elements are in the same coset or not.

Boneh-Goh-Nissim public key cryptosystem that follows homomorphism. It uses a bilinear map built on various composite order finite groups. This structure follows Palliier [93] and accomplishes additive homomorphism with additional functionality enabled by the use of a bilinear map. Using this strategy, one may perform multiplication followed by arbitrary additions on encrypted data. As a result, encrypted values can be analyzed in multivariate polynomials of total degree 2. The security of this technique is based on the hardness assumption of the subgroup decision problem. The primary distinguishing feature of this scheme is that, unlike the preceding case of the slightly homomorphic scheme with the exception of Sander *et al.*, both multiplication and addition are permitted [95]. Computation on encrypted values is possible by means of NC^1 [95] circuits. The weakness is that their construction uses 2-DNF [90] formula, which increases the length of ciphertext [90].

Ishai and Paskin defined a compact technique for evaluating branching algorithms in 2007 [96], whereas security is addressed using the N^{th} residuosity problem.

3.5 Recent developments in Homomorphic Schemes

After 2007, there was a major shift in the development of homomorphic schemes. The need for such schemes arises due to the popularity of the cloud and its usage in day-to-day applications. This is also due to the development of internet-related services in the last 20 years. Figure 3.4 represents the development of schemes after 2009; in fact, a revolutionary interest began with the Gentry [23] PhD thesis in 2009.

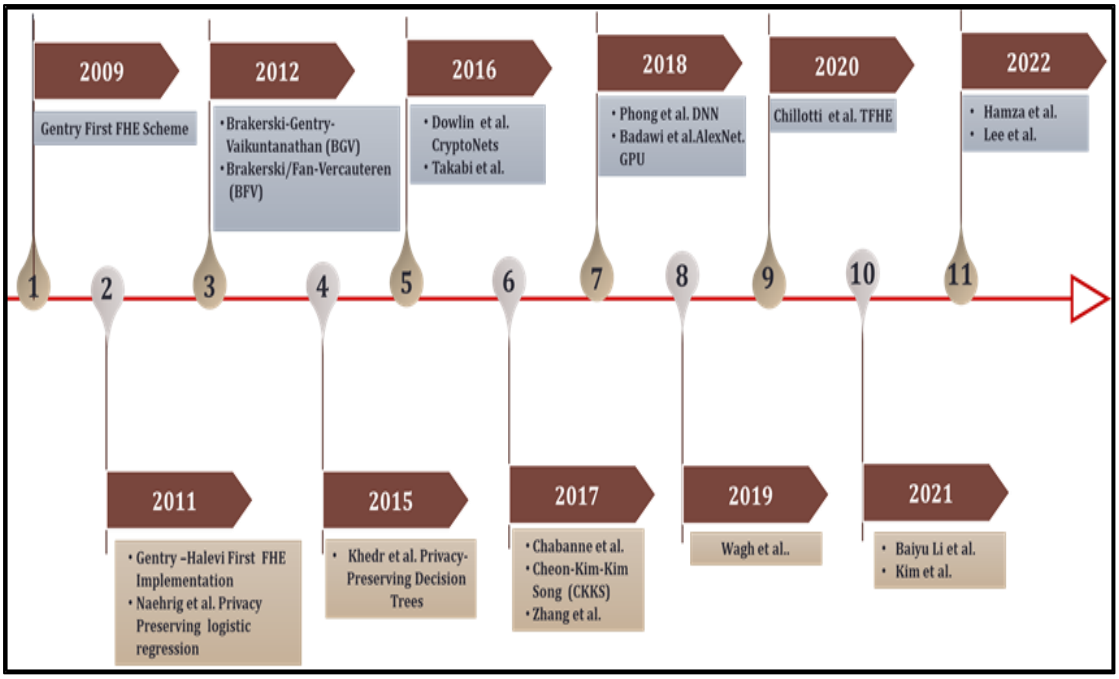

Figure 3.4: Recent Advancement in the Homomorphic Schemes from 2009 to 2022

In the recent developments in the homomorphic schemes, there are three directions, the first is Ideal Lattice, the second is Approximate Greatest Common Divisor (GCD), and the last is based upon Learning with Errors. In the Lattice-based direction, Gentry made a breakthrough in 2009 [23]. Gentry's scheme is fully homomorphic and is considered a blueprint for further improvements with some high computation requirements. A few of the facts and detailed descriptions, along with several proofs, were published in Gentry's thesis [63] in 2009.

Gentry's scheme is a fully homomorphic scheme and is considered as a blueprint for further improvements with some high computation requirements. A few facts, detailed descriptions, and proofs were published in Gentry's thesis [63] in 2009. In 2010 Smart and Vercauteren [97] took the challenge to implement Gentry's scheme and found it to be impractical. After this, Gentry and Halevi [64] in 2010 tried to remove the impracticality of the scheme and also eliminate the essential property of the determinant to be prime.

In 2011, Gentry and Halevi Showed that SIMD (Single Instruction Multiple Data) styles could improve the performance by an overall factor of 2.4 and cipher text size can also be reduced by a factor of 1/72. The same authors manifested how to get rid of

squashing as well, though the underlying construction is ideal lattice only. In 2012, they improved the bootstrapping [98] (the basis for fully homomorphism) using a modulus that was near to the power of two in comparison to the generic binary circuit approach, which can also be combined with SIMD operations.

The next direction is on the underlying assumption of finding approximate GCD. This approach came in December 2009, later slightly revised [64] in June 2010(DGHV). The main theme for this direction is conceptual simplicity because of using integer modulo 2 instead of using ideal lattice; however public key size for this scheme is of order $\tilde{\mathcal{O}}(\lambda^{10})$, which is not a suitable size for any practical scheme.

In 2010, Coron and Mandal reduced the size of the public in a Fully Homomorphic Encryption system from $\tilde{\mathcal{O}}(\lambda^{10})$ down to $\tilde{\mathcal{O}}(\lambda^{7})$ by reducing the public key size of the Homomorphic scheme. Furthermore, Coron *et al.* [99] made some improvements by reducing the public key size from $\tilde{\mathcal{O}}(\lambda^{10})$ to $\tilde{\mathcal{O}}(\lambda^{7})$ for somewhat homomorphic part. This scheme also describes an optimized implementation of fully homomorphic schemes in comparison to [100]. In 2012, Chunsheng proposed a heuristic attack [101] on schemes due to integers using a lattice reduction algorithm. In addition to this, the author also showcased some improvements to handle the described attacks. In addition to the above schemes by Cheon *et al.* [70] with some other authors above scheme processes the vector of plaintext bit as a single ciphertext. In 2012, Kim et al. combined the idea of [62] and [64]. This not only improved efficiency but also opened new ways for practical schemes.

The third major direction is schemes based on learning with errors. The major checkpoint with Gentry's blueprint is the larger key size and per-gate evaluation time for the practical FHE. To eliminate this, in 2011, Brakerski and Vaikuntanathan (BGV) [102], despite using Squashing, suggested a new dimension-modulus reduction technique, which was effective in reducing the ciphertexts and the complexities associated with the decryption process by not introducing any additional assumptions. In 2011, BGV improved the scheme by repetitively switching the modulus, thereby helping to keep the noise level almost constant. Furthermore, in

2011, Lauter, Naehrig, and Vaikuntanathan [103] proposed a number of application-specific optimizations of the BGV scheme.

Hee *et al.* [70] developed the DGHV into a batch completely homomorphic encryption system in 2013, i.e., one that can encrypt and homomorphically process a vector of plaintexts as a single ciphertext.

Khedr *et al.* [104] use an FHE scheme to create Bayesian filters and Decision Trees (DT) for encrypted data. The categorization model allows you to multiply ciphertexts without having to swap keys.

Dowlin *et al.* [105] proposed CryptoNets to solve the problem of blind non-interactive categorization. The NN employs the SHE approach for inputs and homomorphically propagates signals throughout the network. The computational overhead and the replacement of the sigmoidal activation function limit its performance. Several following research has focused on ways to enhance the mentioned constraints.

Takabi *et al.* [106] investigate decentralized scenarios with several participants and scattered datasets. As an activation function, the NN employs a polynomial approximation. Nonetheless, because the implementation takes an interactive approach, the client performs the recryption process directly.

By adding a normalized layer before each activation layer, Chabanne *et al.* [107] overcome the constraints of CryptoNets. This is the first time a homomorphic evaluation of Deep Neural Networks has been possible (DNN). Through an FHE technique, the NN attained accuracy comparable to the best non-secure versions.

In a cloud computing setting, Zhang *et al.* [108] presented a deep learning model for big data applications. The model is trained using the BGV homomorphic technique [102] and a back-propagation algorithm.

CryptoDL was developed by Hesamifard *et al.* [109] to demonstrate the possibility of finding the lowest degree polynomial approximation of an activation function within a

given error range. Polynomials are used to approximate the Rectified Linear Unit (*ReLU*), Sigmoid, and hyperbolic Tangent (*Tanh*) functions.

On a DNN with an additive HE, Phong *et al.* [110] develop an asynchronous Stochastic Gradient Descent (SGD). When compared to traditional deep learning systems, the technique maintains accuracy with a reasonable increase in overhead.

On Graphics Processing Units, Badawi *et al.* [111] offer a Convolutional Neural Network (CNN) for image categorization with FHE features (GPU). The AlexNet improves classification speed while maintaining security and accuracy; it classifies the MNIST dataset in 1% of the time it takes CryptoNets.

Wagh *et al.* [112] build the foundations for a unique NN protocol with a secure three-party system. The approach allows for the training and inference of many NN designs without the need to learn about the data.

Using the TFHE technique, Chilloti *et al.* [113], [114] built a deterministic weighted automaton that can compute the maximum function.

Baiyu Li *et al.* [115], in their study, majorly available homomorphic libraries. A few of them include HEAAN, SEAL, HElib, and PALISADE. They also provided theoretical findings for the security evaluation of homomorphic schemes particularly related to approximate numbers. They came up with alternatives based on indistinguishability and simulation, as well as constrained definitions that limit the sequence and quantity of adversarial queries.

Hamza *et al.* [116] offered a complete evaluation of several homomorphic encryption tools for Big Data research and their applications in their article published in 2022. They have considered employing homomorphic encryption methods to create a security framework for Big Data analysis while maintaining privacy. In addition, they compared the implementation results and performances of various homomorphic encryption toolkits. Their goal was to see how homomorphic encryption technology could be used to increase the usability and efficiency of privacy-preserving machine learning in Big Data processing.

3.6 Analysis of Homomorphic Schemes and its Interpretation

A number of homomorphic schemes are implemented and analyzed. The analysis is done under variable parameters, and a time-based comparison is also carried out. The implementation is done in C/C + + using GMP (GNU Multiple Precision) and NTL (Number Theory Library) libraries and tested on a 3-GHz third-generation system. GMP is an open source that is multi-precision and can be used for various types of operations on signed integers, floating point numbers, and rational numbers. The richness of function, friendly interface, and free availability make it so popular and useful. The limit of precision just depends upon the machine, not on the library. There are a variety of homomorphic schemes that are implemented and also analyzed under different security considerations. Various other schemes are also implemented to cater various applications in different scenarios.

A Homomorphic Scheme by Ramaiah *et al.* [117], suggested that three more parameters to be used in addition to those used by DGHV [64] scheme. The three more parameters used are the symbol p, which denotes the size of the plaintext, with the complexity of order O(n); parameter e', which denotes the size of another secret key integer, R. For schemes to be fully homomorphic the size of e' is taken as ≥ $p.\theta(n \lg^2 n)$. e is the bit length of the secret key integer P. The scheme is as follows:

$KeyGen^{L}(n)$: This step pertains to the key generation process. Firstly, two random numbers (P and R) are being selected of size e and e' respectively.. i.e., $P \xleftarrow{\$} (2\mathbf{Z}+1) \cap [2^{e-1}, 2^{e})$ and $R \xleftarrow{\$} (2\square+1) \cap [2^{e'-1}, 2^{e'})$. Choose two g – bit random integers Q_0, Q_1. For this, sample $Q_i \xleftarrow{\$} \square \cap \left[0,\ 2^{g}/P\right)$, for i = 0,1. Another random integer $R' \xleftarrow{\$} \square \cap \left[2^{r'-1},\ 2^{r'}\right)$. Compute $X_0 = PQ_0$, $X_1 = PQ_1 + RR'$. Output the secret key, $SK = (P, R)$ and the public key, $PK = (X_0, X_1)$.

$Encrypt^{L}(PK, M \in \square \cap [-2^{p-1}, 2^{p-1}))$: In this step, the range of the plaintext M is defined as: M belongs to $[-2^{p-1}, 2^{p-1})$. Choose two s-bit random integers N1, N2, so that N2 > N1, and N2 is an even number. For this, sample $N_i \xleftarrow{\$} \square [2^{s-1}, 2^{s})$, for i =

1, 2. Compute $X_2 = [N_1 X_1] \bmod X_0$. The ciphertext can be calculated as: $C = [M + N_2 X_2] \bmod X_0$.

Note: The reason for choosing N_2 as a random even number is to make it easier to reduce the scheme's security to the two-element PAGCD problem.

Decrypt^L (SK, C): The plaintext integer $M = [C \bmod P] \bmod R$

Evaluate^L $(PK, CKT, (C_1, ..., C_k))$: Here, CKT is the circuit, by which arithmetic operations are to be carried out. C_1,…..,C_k are the ciphertext corresponding to the plaintext $M_1, ..., M_k$. Requires operations can be carried out as per the operations below:

Add^L: *Compute* $C = \left[C_1 + C_2\right] \bmod X_0$, and

Mul^L : *Compute* $C = \left[C_1 \times C_2\right] \bmod X_0$

The ciphertext resulting after the complete evaluation of the circuit is decrypted using the Decrypt^L algorithm.

3.6.1 Implementation Results

Several homomorphic schemes are implemented in Linux using GMP and NTL libraries on a 2 GHz system having 4 GB of RAM. Implementation results are presented below with results on the plaintext.

Ramaiah et al. [117]***:*** Timing analysis is done for different values of security parameter, and a comparison has been made. The following calculations based on timing are taken on encrypting and evaluating functions like x^2+xy+y^2 and $x^4+x^3+x^2y+y^2x+y^3$. Tables 3.1 and 3.2 indicate the significant findings:

i) Result for Evaluation function $x^4+x^3+x^2y+y^2x+y^3$

Table 3.1: Comparison of time for evaluation circuit for plaintext and ciphertext

Security Parameter (no. of bits)	**Evaluation (Plaintext)** (Time in Seconds)	**Evaluation (Ciphertext)** (Time in Seconds)
5	0.015	0.017
10	0.074	0.105
15	0.085	0.106
25	0.106	0.109

Table 3.1 can be analysed, as when the security parameter increases, the time taken to evaluate the function using plain text increases initially, then it reaches a constant where to evaluate the same function using encrypted data time increases initially, followed by a constant and in the last again increases but now at a slower rate.

ii) Result for Evaluation function x^2+xy+y^2

Table 3.2: Comparison of time for evaluation circuit for plaintext and ciphertext

Security Parameter (no. of bits)	**Evaluation (Plaintext)** (Time in Seconds)	**Evaluation (Ciphertext)** (Time in Seconds)
5	0.011	0.015
10	0.012	0.027
15	0.045	0.078
25	0.059	0.106

From the above two tables, it can see that, the time required to evaluate $x^4+x^3+x^2y+y^2x+y^3$circuit is more than x^2+xy+y^2. The first circuit needs more computation as compared to the first, so it takes more time.

Xiao et al. [71]**:** The security is based on integer factorization problem. It uses matrix similarity transformation to attain homomorphism. Below are implementation results for a given plaintext Table 3.3.

Table 3.3: Comparison of Encryption and Decryption time with respect to input key size

Size (no. of bits)	Keygen (microseconds)	Encrypt (microseconds)	Decrypt (microseconds)
64	73996	3373	118
128	61060	3551	216
256	70767	4067	666
512	110733	5618	1757
1024	351096	11785	5551
2048	4 sec, 256570ms	24396	16198

The importance of such scheme lies in the fact that such type of scheme may be used for data with a faster evaluation process as compared to other homomorphic schemes.

Further analysis of homomorphic schemes is demonstrated in the subsequent section (Applications-based Implementation and Analysis of Homomorphic Encryption)

3.7 Application based Implementation and Analysis of Homomorphic Schemes

Some of the applications are listed below. Few of them are implemented and analyzed in serve to day-to-day applications.

3.7.1 E-Voting

Electronic voting is also termed E-voting. It uses an electronic management system for casting and counting votes. In E-voting, votes are digitized and Confidentiality of the voter is threatened. If his vote is decrypted by the election consultants who are actually counting the votes. To overcome this issue, the concept of homomorphic E-voting came into existence. In this scheme, votes of the voters are counted before decryption. There can be ‘ N ’ number of candidates, and voters must vote for one and only one candidate. A vote can be represented by a vector where 1 and 0 respectively denote the voting in favour and against, respectively. Thus there will be n entries in the vector equal to the number of candidates in the election. Each voter encrypts his vote and submits it to the election authorities, and they continue to count the vote in

its encrypted state with any additive homomorphic encryption algorithm. It is discussed below:-

Let V be the set of voters and X be the set of candidates. Suppose there are 4 voters and 3 candidates.

$$V = \{ V_1 , V_2 , V_3 , V_4 \} \text{ and } X = \{ X_1, X_2, X_3 \} \quad (3.3)$$

Using the Paillier Cryptosystem [93]

we took p =5 and q =7 as two primes, then n =35 and n^2=1225 and λ =12. g is chosen to be 141. Assume that V_1 voted for X_1 and random value(r) chosen by him for encryption is 11. The first voter's vector is [01 00 00], which shows he voted for X_1 ; therefore, its corresponding value is 01, and the value corresponding to X_2 and X_3 is 00. Now convert this binary value into its decimal equivalent (say X), which is equal to 16. Then the value of the encrypted vote is calculated as follows:

$$Enc(X, r) = g^X \times r^n mod\ n^2$$

$$Enc(16,11) = 141^{16} \times 11^{35} mod\ 1225 = 541$$

Similarly, encrypted values of other votes are computed, which are shown in Table 3.4

Table 3.4: Implementation results of E-voting

Voters	Random No.	X_1	X_2	X_3	Binary equivalent of a vote	Decimal equivalent of a vote	Encrypted vote
V_1	11	01	00	00	010000	16	541
V_2	2	00	01	00	000100	4	298
V_3	3	00	01	00	000100	4	202
V_4	6	00	00	01	000001	1	741

In order to sum the votes, we multiply the encrypted vote values modulo n^2 and calculate the cipher text C as

$$C = 541 \times 298 \times 202 \times 741 mod 1225 = 1101 \tag{3.4}$$

Now the decryption is done as follows:

$$Dec(C) = \left(L\left(C^{\lambda} mod\ n^2 \right) \times \left(L\left(g^{\lambda} mod\ n^2 \right) \right)^{-1} \right) mod\ n \tag{3.5}$$

$$L\left(1101^{12} mod\ 1225 \right) = (351 - 1)/35 = 10 \tag{3.6}$$

$$L\left(141^{12}\ mod\ 1225 \right) = (456 - 1)/35 = 13 \tag{3.7}$$

$$Dec(C) = 10 \times (13)^{-1}\ mod\ 35 = 25 \tag{3.8}$$

The binary equivalent of 25 is 01 10 01 (01 02 01), which shows 2 votes are cast for X_2, Therefore X_2 , is the winner.

3.7.2 Multiparty Computation

Multiparty computation (MPC) is a branch of cryptography that allows many parties to jointly calculate a function over their inputs, even if the participants are untrustworthy. Therefore, their inputs must be kept private. MPC protocol is required for communication and for preserving the privacy of data so that the party who wants to compute a function will have no information about the inputs provided by all other parties and have access only to the final value computed. Assume there are m number of parties $(P_1, P_2, P_3 \ldots ..P_m)$ providing their inputs $(X_1, X_2, X_3 \ldots X_m)$ to the server for computation such that inputs must be kept private from each other and from the server too. The server should not have access to these values. The topology of the network is depicted in Figure 3.5.

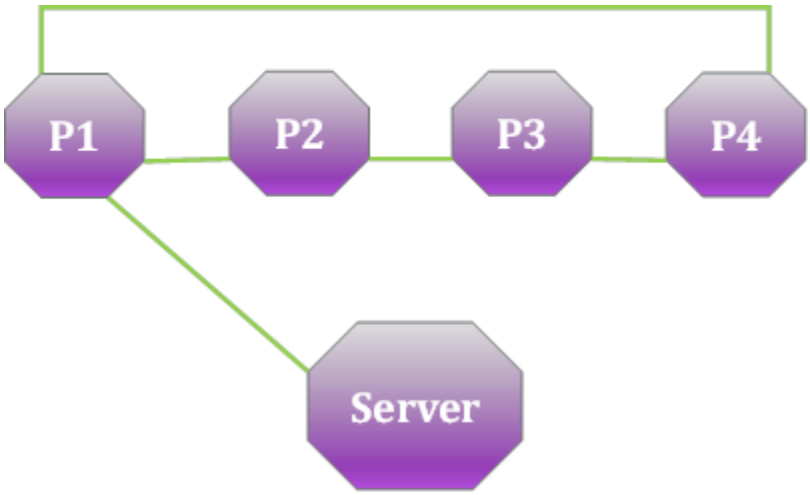

Figure 3.5: Topology of the Network

The server generates a public-private homomorphic key pair for computation. P_1 encrypts its input X_1 using the public key of the server and then multiplies it with a random value 'a' known only to P_1, then encrypts this value with the public key of P_2 and forwards it to P_2. P_2, then encrypts its input X_2 with the public key of server and decrypts the value sent by P_1 with its own private key and then multiplies that value with encrypted X_2 and gets $Enc(a \times X_1) \times Enc(X_2)$, encrypts this with the public key of P_3 and forwards it to P_3 and this way it continues till P_m gets the encrypted value of $a \times X_1 \times X_2 \times X_3 ... X_{m-1}$. P_m then multiplies this value with its input X_m which is also encrypted with server's public key and then encrypts $Enc(a \times X_1 \times X_2 \times \ldots X_{m-1}) \times Enc(X_m)$ with P_1's public key and transmit it to P_1, P_1 divides the whole value by 'a' and forwards it to the server where he decrypts the value of the function with its private key. Thus, individual inputs are kept private from the server, and all other remaining parties, and the value of the function is computed. Figure 3.6 illustrates the whole process considering only 3 parties, and the function to be computed by the server is taken as $F(X_1, X_2, X_3) = X_1 \times X_2 \times X_3$

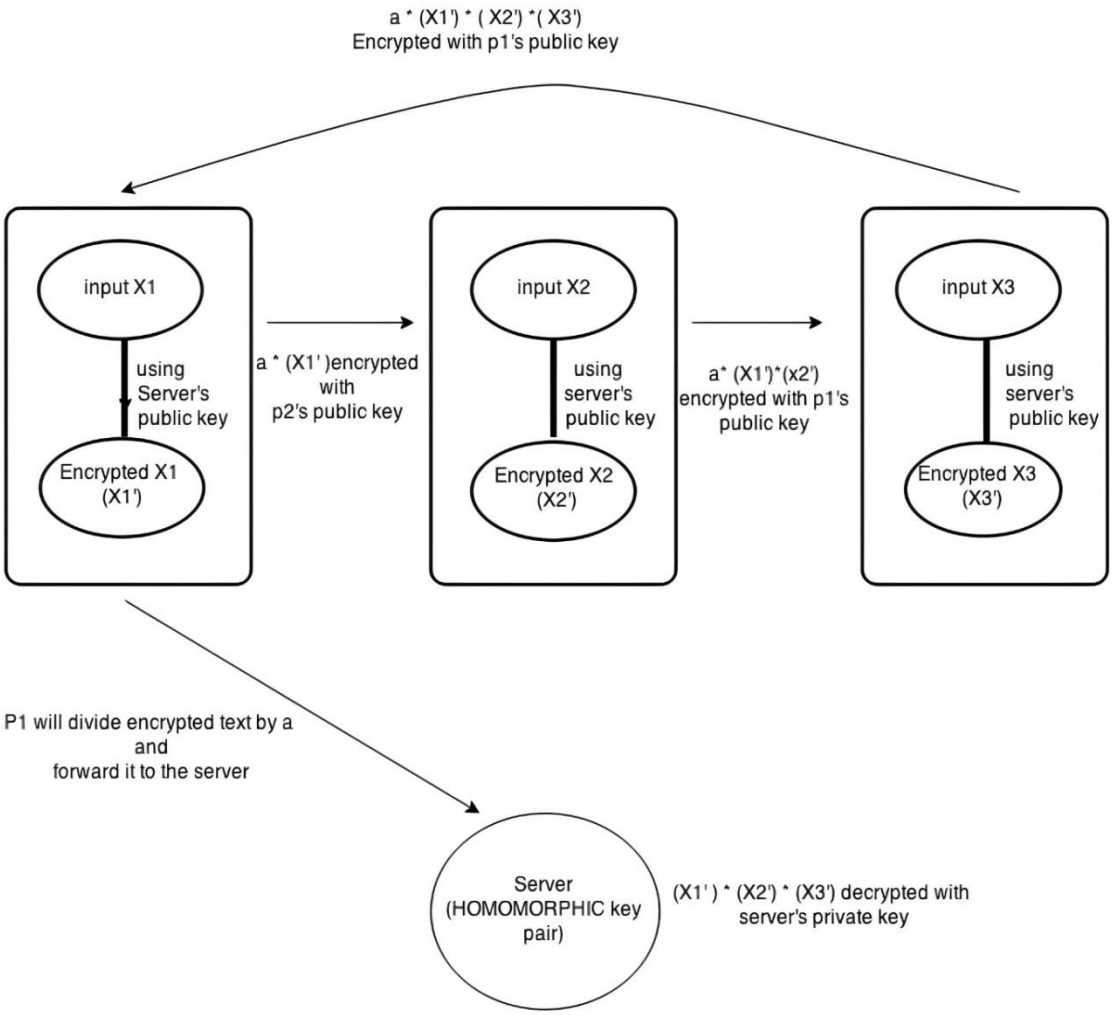

Figure 3.6: Multiparty Computation

A toy example is presented using the RSA cryptosystem (multiplicative homomorphism) to illustrate the whole process. Considering there are 3 parties that are providing their data to the server for computation. Let P be a vector containing data of all the parties that includes randomly selected 2 prime numbers, the public key and private key.

P = { 1st prime number, 2nd prime number, public key, private key}

The public key and private key selected by all 3 parties and the server using the RSA cryptosystem are:

$$P_1 = \{7,\ 19,\ 37,\ 73\} \tag{3.9}$$

$$P_2 = \{23,\ 29,\ 47,\ 367\} \tag{3.10}$$

$$P_3 = \{11,\ 31,\ 71,\ 131\} \tag{3.11}$$

$$S = \{11,\ 17,\ 23,\ 7\} \tag{3.12}$$

The function to be computed by the server is

$$F(X_1, X_2, X_3) = X_1 \times X_2 \times X_3$$

And the input of the 3 parties are X_1=2, X_2=3 and X_3=4. The parties will encrypt their input using the public key of the server. Their values after encryption are:

$$X_1\text{'}=162\ \ X_2\text{'}=181 \text{ and } X_3\text{'}=64 \tag{3.13}$$

1st party P_1 will multiply encrypted input by a random variable, say 'a' (let the value of 'a' be 1) and encrypt the whole value by P_2's public key and forward it to P_2.

$$Enc(a \times (X_1\text{'})) = 70 \tag{3.14}$$

P_2 will decrypt 70 using its private key and will get 162 back. Now P_2 will find the product of Enc (a×(X_1')) and Enc (X_2') whose value will be 29322. P_2 will encrypt this value by P_3's public key and transmit it to P_3. Then P_3 decrypt it with its private key and follow the same procedure as followed by P_2 and transmit the same to P_1. Now P_1 will decrypt it and divide it by 'a' and forward it to the server. The server will get the solution of the function by decrypting the output with its private key.

3.7.3 Secret Sharing

In secret sharing, a secret is distributed among different parties, and each party shares some part of the secret. The secret is reconstructed only when a sufficient number of shares (say k) is combined; this is termed as thresholding scheme where k shares are mandatory for secret reconstruction. Anyhow, less than k shares will not reveal the secret, and also, the individual shares are of no use. Each secret can be formulated

into a polynomial where the constant term represents the secret. The degree of polynomial is equal to one less than the number of parties involved. The constant term of the polynomial is the secret. Assume there are ' m ' parties involved in this protocol. The constant term, a_0 is the secret that is to be shared among m parties. Therefore a polynomial formulated for this secret can be written as

$$f(x) = a_0 + a_1 * x + a_2 x^2 + \ldots\ldots a_{m-1} x^{m-1} \tag{3.15}$$

here the coefficients $a_1, a_2, \ldots a_{m-1}$ are randomly chosen by the one who is sharing this secret with m parties. Each share is a tuple $(x, f(x))$. The secret cannot be reconstructed till m parties are involved. Just as a minimum of 2 points are necessary for finding the equation of a line, 3 points are required for formulating a quadratic equation, and 4 points for finding the equation of a curve. Similarly, m shares are required to reconstruct the equation of degree m -1.

3.7.4 E-Auction

E-Auction is a mechanism in which participants bid for the items, and item allocation is done based on their bidding prices. E-Auction protocol consists of an auction server, auctioneer, bidders and a bulletin board that is used to broadcast the encrypted bid value in order to ensure that no bidder repudiates his bid. First of all, bidders register themselves to the auction server so that they can participate in the bidding process. Each bidder generates a bidding vector and publishes his encrypted bid vector on the bulletin board.

Consider the case when there are n bidders bidding for an item and a set of biddable prices (say from 1 to X). Operations are based on Group Z_{13}. Every bidder selects a bid price Y_i from the predefined range 1 to X and generates a random vector of length X . Based on his bidding value, the bidder constructs his bid vector B_i , which is also of length X . Y_i values of the bid vector of a bidder are same as his random vector values, and the remaining values are 0. Then each bidder N_i splits his bidding vector into N (equal to the number of bidders involved in the process) a random

vector and sends the N^{th} random vector to N^{th} bidder. Now the bidder B_i gets his final bidding vector B_i' by adding all these random components and publishes it to the bulletin board. This way, all the bidders publish their final bid vectors. Using these vectors, a deciding vector is formed by adding all the vectors that were published by the bidders. Find out the maximum value in this deciding vector. If this value matches with any of the bidder's random vector's values, then that bidder will win the auction. This whole process can be explained by a toy example:

Let's say there are 2(N =2) bidders and the bidding range is 1 to 5(here X =5). N_1 chooses 2 and N_2 chooses 4 as their bidding prices. Operations are based on group Z_{13}. Random vectors chosen by them be (2,4,8,10,12) and (1,3,7,5,11). Their bidding vectors will be:

- B_1=(2,4,0,0,0) { 2 values are the same as its random vector's values, and the remaining values are 0}
- B_2 =(1,3,7,5,0) {4 values are same as its random vector's values remaining values are 0}

Now each of the bidders will divide his vector into n sub-components under the mod 13 operation. The 2 subcomponents of N_1 can be B_{11} = (11,9,7,5,12) and B_{12} = (4,8,6,8,1) and for N_2 sub components can be B_{21} = (2,10,4,10,4) and B_{22} = (12,6,3,8,9) Now their vectors are exchanged with other bidders. B_1 will forward it B_{12} to B_2 and B_2 will forward its B_{21} sub-vector to B_1

The final bid vectors that will be published on the board will be:

- B_1'=(11+2 mod 13, 9+10 mod 13, 7+4 mod 13, 5+10 mod 13, 12+4 mod 13) = (0,5,11,2,3)

Similarly

- B_2 '=(4+12 mod 13, 8+6 mod 13, 6+3 mod 13, 8+8 mod 13, 1+9 mod 13)=(3,1,9,3,10)

Final deciding vector BV = (3,6,7,5,0)

The maximum value of the final vector is 7. This value matches with the 3rd value of the random vector of the bidder N_2. Therefore B_2 wins the auction.

3.7.5 Homomorphic Lottery Protocol

In homomorphic lottery schemes, there is an auditor whose homomorphic key pairs are used in the entire process. A winning lottery ticket is selected by all the participating parties randomly such that the winning probability of each participant is the same. Suppose there are N number of S participants, and each participant selects a random number in the predefined range {0 to N -1} and encrypts it with the auditor's public key before publishing. These encrypted numbers are added homomorphically using any additive homomorphic cryptosystem. The sum S is obtained by adding all the numbers chosen by the participants. S Mod N is computed efficiently to get the winning ticket of the lottery. Thus the process ensures fairness because the decryption process is not kept private. Even the auditor will not be able to deceive.

3.7.6 Private Information retrieval

If the client wants to obtain i^{th} index of outsourced data without giving information about i to the server (which may be remote), one of the solution which suffices the needs of the client is sending the entire database to the client machine, but this will increase the communication cost which is not desirable because the database can be of large size. A homomorphic PIR scheme can do this with significantly less overhead and more security. Let's say the server contains a vector (database) of integer values that are in the range *(0,m).* The client formulates a vector whose i^{th} (index to be retrieved) value is 1, and the remaining values are zero. Encrypt all the values of a vector and send it to the server where homomorphic multiplication of the client's encrypted vector and vector present at the server is performed. After this additive, the HE algorithm is applied to add all the values obtained after multiplication. This will output the data present at i^{th} index of the vector but is encrypted. The server sends this encrypted data to the client for decryption.

3.8 Summary

The field of homomorphic encryption is attracting many researchers these days, and they are taking a keen interest in developing homomorphic cryptosystems that can be deployed practically. The focus is on its practical applicability and on how real-world problems can be solved easily, preserving the privacy of the client. In this chapter, a number of homomorphic schemes are implemented and analyzed. It is shown how homomorphic encryption can be helpful in some practical problems and also presents how these problems can be dealt with. Applying homomorphic encryption to traditional approaches makes them more secure and reliable.

To extend the concept further, homomorphic encryption has applications in secret sharing, cloud, and related technology such as blockchain. These areas are extensively studied in the forthcoming chapters.

CHAPTER 4

A HYBRID BASED VERIFIABLE SECRET SHARING SCHEME USING CHINESE REMAINDER THEOREM

Secret Sharing (SS) schemes were coined by Shamir [17] and Blakley [16] in 1979. Since then, it has attracted the interest of several researchers. SS has been found valuable in several applications, such as witness encryption [17], secure communication [47], and access control [48]. In SS schemes, there are two significant role-players, one is the dealer, and another is the group of shareholders (participants). The dealer splits the secret into n parts and distributes these shares among n shareholders. These shareholders, when combining their shares, can recover the secret. It is also mentioned as a thresholding scheme if the secret can be recovered by combining t out of n $(t \leq n)$ shares. However, less than t parts must not reveal any information about the secret.

There are some drawbacks in the SS schemes presented in [16], [17] which may act as a constraint for practical usage are listed below:

- Fake shares may be distributed by the malicious dealer, and in turn, secret reconstruction is not possible.
- A deceitful shareholder may submit a fake/invalid share, which leads to incorrect share reconstruction, and the true secret would only be known to the deceitful shareholder.
- Need for a mutually trusted dealer for the generation and distribution of shares.
- There is a requirement for a private channel for share distribution.

An advancement of SS schemes, known as verifiable secret sharing (VSS) schemes, came into the picture to handle the dishonesty of shareholders or dealers. The dealer may be biased in the distribution of shares or the reconstruction of the secret.

In traditional SS schemes, it was assumed that the shareholders and the dealers were honest and reliable enough. Though, in practice, the dealers do not entirely trust the players, it is logical to assume that the players do not trust the dealer as well. To make SS verifiable, some auxiliary information needs to be added that helps the

stakeholders to authenticate their respective shares. The shares are those shareholders who do not accept the shares if they find them inconsistent or invalid. With the help of VSS schemes, the shareholders can verify their shares without having access to the secret and even without revealing their shares. Other flavours of SS schemes include multiple [49], multilevel [50], weighted [51], and protected SS (PSS) schemes [53].

4.1 Background Information

4.1.1 Background of Secret Sharing Schemes

Simple SS schemes are not of much interest in practical scenarios. A threshold value plays an important role. The very first idea in this league is a scheme due to Adi Shamir in 1979 [17]. In every SS scheme, there are two phases, one is share generation, and another is secret reconstruction. Shamir's scheme is based on Lagrange's polynomial interpolation, which satisfies the basic requirements of SS schemes. Shareholders can unlock the secret if t (out of n) or more shares are known. Shamir's scheme is divided into two algorithms, namely, share generation and share reconstruction.

4.1.2 Share Generation

In this, the dealer selects a polynomial $f(x)$ (given by (1)) of degree $t-1$ whose coefficients are randomly chosen from a finite field by the dealer,

$$f(x) = a_0 + a_1 * x + a_2 x^2 + \ldots\ldots a_{t-1} x^{t-1} \tag{4.1}$$

The dealer computes a set of n shares $\{f(1), f(2), \ldots, f(n)\}$ and distributes them among the participants through private channels.

4.1.3 Secret Reconstruction

The secret reconstruction is not done until t parties are involved. For example, as a minimum, two points are required to construct the equation of a line, three points are required for formulating a quadratic equation, and similarly, t shares are combined to reconstruct the equation of degree $t-1$. The polynomial reconstruction is done using Lagrange's polynomial interpolation, in figure 4.2 i.e.,

$$f(x) = \sum_{i=1}^{t} f(i) \prod_{j=1, j\neq i}^{t} (x-j)/(i-j) \tag{4.2}$$

After Shamir Scheme, another landmark work in this direction was presented by Blakey [16] in the year 1979, where this scheme was based on Hyper-plane Geometry. It can be summarized that as the secret is a specific point in space, each share corresponds to a hyperplane, and the number of planes intersecting (if more significant than the threshold) reveals the secret.

The notion of verifiability in SS schemes was first presented by Choc *et al.* [118], where verification of received shares is done without any information about the secret. VSS schemes can be interactive and non-interactive [119], whereas non-interactive schemes are more efficient in comparison to interactive ones. Initially, interactive schemes were offered, in which the dealer and players communicated with one another to verify the shares' authenticity. This sometimes increases the overhead of the dealer as he has to communicate with N players. Later, non-interactive schemes were introduced, which reduced the dealer's overhead (communication). Max Mignotte [120] came with his seminal work in 1983 that was based on CRT and used a particular sequence of integers rather than utilizing an interpolation polynomial for secret construction. Another popular construction is due to Feldman [121], which is a verifiable and non-interactive based on Shamir's scheme. The security is based on the discrete logarithm problem (DLP), which is assumed to be computationally secure. To make the scheme unconditionally secure, Pedersen [122] used a commitment to function in his scheme.

On the other hand, if the dealer can get commitment values and break DLP, he can distribute fake shares. In 2008, Kaya *et al.* [123] proposed another VSS scheme based on CRT and proved its security. They also proposed a joint random secret sharing (JRSS) and proactive SS scheme protocol. In 2010, Harn and Lin [124] defined (n,t,n) the SS scheme based on Pedersen's schemes and presented the notion of strong VSS and t consistency. They also presented a robust (n,t,n) scheme based on Benaloh [125] scheme. Subsequently, from 2012 to 2014, Meng *et al.* [123] and Mahmoud [126], [127] proposed different VSS schemes. In the direction of PVSS, different schemes [41], [43],

[128], [129], [130], [130], [131], [132], [133], [134], [135] have been proposed from time to time with different capabilities.

4.1.4 Secret Sharing Schemes with Additional Capabilities

In this section, schemes with additional properties are discussed. As there are different scenarios, so scheme must possess some extended capability to handle the challenges. A few more extended capabilities include Dealer leakage resilient, Homomorphic SS, Cheater detection and identification, Robust, Cheating immune, and Proactive secret sharing schemes, which are discussed below:

i) Dealer Leakage Resilient Secret Sharing Schemes (DLR SS)

VSS captures only one type of dishonest behaviour of the dealer. There exist many other dishonest strategies that can be adopted by the dealer for cheating. Consider the case when the dealer tries to subliminally leak information in valid shares. Thus, this leads to the genuine behaviour of the dealer to every player but gives information about the secret to the attacker. The dealer's malicious behaviour does not get revealed. To overcome such threats, DLR-VSS was introduced. DLR-VSS holds the property of verifiability and DLR as well. One of the ways to achieve this property is to not allow the dealer to employ randomness, due to which the dealer will not be able to leak any information through valid shares. Communication between the dealer and shareholders outside the setting of the protocol is not allowed. If the dealer tries to do so, he will be discarded and assumed as the faulty dealer. To ensure security in the presence of a trusted dealer, VSS is used, whereas DLR-VSS ensures secrecy even in the presence of a faulty dealer.

ii) Cheater detection and identification.

There is always a chance of betrayal either by the dealer during distribution or by the participant during reconstruction. Cheating can be divided into two parts: first is cheating as disruption, and the other is collusion. The first one in the treachery done by the dealer when he distributes the share to the shareholder and the shareholder is not having any participation or knowledge of it. He may not want that; the shareholder reveals the correct share during reconstruction. Later can be done at the participant level; if one or more participants do not submit the correct share to the

combiner. The appropriate premise presumes the dealer and combiner as honest and considers betrayal at the participant level. Nevertheless, this is not always the case. A number of researchers propose different schemes to handle cheating at both ends. The initial solution to the problem is catered by [136]. These code-based schemes can detect betraying and can also detect the invalidity of shares [45], [137], [138], [139].

iii) Robust secret sharing

This type of SS Scheme is used in dishonest scenarios. In this case, dishonesty generally arises due to an adversary. The scheme in which secret can still be recovered, even if some shares are not correct, are called a robust SS scheme. There is many a scheme in which cheating can be identified, but, particularly in the case of robust schemes, some additional information is attached along with the shares so that the correctness of the share can be checked. Various possible models of robust schemes were discussed by Rogaway *et al.* [140] in 2007.

iv) Cheating immune secret sharing schemes

This type of scheme forestalls the cheaters (a subset of participants) from gaging an advantage in recognizing the secret information as they are submitting the fake shares. The very first comment on this type of scheme was by Tompa and Woll [141] in 1988. They established an attack on Shamir SS scheme and proved that cheaters are able to retrieve the correct secret by the false secret, which was recovered by the reconstruction. Also, the honest participant didn't get the correct secret. Moreover, this type of attack can penetrate any linear SS scheme. There are various approaches to forestall this attack. In can be referred in [118], [136], [140], [142], [143], [144]. A thorough analysis is done by Martin [145] to handle the dishonest dealer, participant, and combiner. Recent construction in 2019 in proposed by Romar *et al.* by considering the work of [146]. It is proved that Maiorana-McFarland Boolean functions may be used for better construction. They used the technique of Carlet [147] and constituted that the proposal is a generalization of [141].

An initial solution to the above problem posed in [141] was proposed by Pieprzyk *et al.* [148] in 2001. This solution was in the form of cheating prevention over Galois Field. The next solution proposed by Pieprzyk *et al.*. [149] in 2002 was cheating

immune, not cheating prevention. A more acceptable solution [146] was proposed by the same authors in 2004. They considered binary shares and boolean functions, and two versions were proposed. First is *t*-cheating immune, where an adversary who submits t incorrect shares gains no advantage, and the other is a more general construction, strictly t-cheating immune, where an adversary who submits up to t incorrect shares gains no advantages. A nice property of cheating-immune schemes is that the share size is the same as the secret size (in other schemes, either have large shares or the recovery of the secret requires more than the minimum number of shares). The main problem in the theory of cheating-immune schemes is the construction of such schemes for any access structure. Properties and constructions of such schemes are studied in [148], [149], [150], [151], [152], [153], [154].

v) Proactive secret sharing

In this type of secret sharing scheme, shares are refreshed at regular intervals, regardless of the fact whether the shares are compromised or not. To avoid such situations, use protective sharing scheme concept, in which the share can be refreshed at regular intervals of time, so the system may tolerate the fault, which may occur due to the server (or party compromised) by recovering the server and filling the server with a refreshed value. Inquisitive readers may refer [155], [156], [157], [158]. Byzantine fault tolerance, in which one party may be compromised and generally occurs in the asynchronous medium.

Another variant of VSS is an asynchronous verifiable SS (AVSS) scheme where fault tolerance in multiparty computation can be handled. Basu *et al.* [159] proposed an optimistic AVSS scheme where the payoff cost of failure possesses linearity, i.e., proportional to the number of failures. Further reading may be referred [160], [161], [162], [163], [164], [165]. A different approach that adds non-malleability to SS scheme was proposed by Goyal *et al.* [166]. With this scheme, if the shares are tempered, then either the original secret can be recovered or the recovered secret is unrelated to the original secret.

4.2 Applications of Secret Sharing Schemes

There are various applications of SS schemes ranging from traditional to contemporary [17], [47], [48], [167]. SS schemes can be used for hierarchal organizations to share a single secret. The Proposed scheme can be used to share multiple secrets in a multilevel environment with fulfilling the necessary security requirements. Other applications of SS are as follows:

i) ***Securing Cryptographic Keys*** plays an essential role in any cryptosystem. In such cases, the key is split into different parts. Each part is termed as a share of the key, and these shares are distributed to all the participants who pool their shares for key construction.

ii) ***Electronic Voting,*** also called E-Voting, uses electronic systems for casting and counting votes. To avoid plausible dishonesty, SS schemes can be adopted in the E-Voting system. Each vote can be treated as a secret, and shares of the vote are distributed among the authorities who are counting the votes. Now only t authorities can access the vote, and it cannot be manipulated by any $t-1$ authorities. SS schemes add security and reliability to the E-Voting system. Another possible application of the electronic system is *E-Auction.* In this system, participants put an offer for the items, and allocation is done based on their offered prices.

iii) ***Threshold Schemes for Multiple Servers-*** Shares are spread across multiple servers, and even $t-1$ shares do not give any information. The scheme works even if one or two servers meet any failure, and the secret can still be recovered.

iv) ***Distributed Signature*** is a mathematical way to authenticate a message. It is generally a hash code of the message, encrypted with a secret key. The sender puts his signature to authenticate the message. If there are multiple co-signers, each of them signs the message one by one according to the priority. However, this is not an efficient way because any co-signer can repudiate. The SS schemes can be adopted in such a scenario. The signing key acts as a secret that is shared among all the co-signers. Each share is given to each co-signer, and no one has complete control over the secret. Minimum t co-signers need to pool their shares for signing key construction. Thus the scheme is secure, and repudiation is not possible.

A hybrid-based VSS scheme is proposed, and in the subsequent sections, the results are analyzed and compared with the existing VSS schemes.

4.3 Proposed Algorithm

The proposed scheme works for multiple secrets in a multilevel structured environment (hierarchal organization). In this, the shareholders are divided into z levels $(L_1, L_2, ..., L_Z)$ with L_1 and L_Z as the highest and lowest levels, respectively. Each i[th] Level is assumed to have N_i shareholders. For example, if $N_3 = 4$ it implies that there are 4 shareholders at level 3. There is a dealer D who wants to share k secrets $(M_0, M_1, ..., M_{k-1})$ among the shareholders and let t be the threshold of the protocol. The approach is divided into two main sections: share generation and secret reconstruction.

The essential conditions necessary for successful secret reconstruction are:

- The secret can be reconstructed if there are t or more valid shares available.
- The secret cannot be reconstructed if the number of shares is less than t.

Each shareholder keeps $k+t$ values as their shares, which are used to reconstruct k secrets. The whole algorithm is explained below:

4.4 Share Generation

Assume there are k secrets and all are from Z_p^* where p is a big prime.

Case 1: Intra-level secret sharing

- D forms a polynomial $f(x)$ of a degree $(t+k-1)$ from Z_p^*, i.e.,

$$f(x) = \sum_{i=0}^{t+k-1} a_i * x^i \bmod p \tag{4.3}$$

Where $a_0 = M_0, a_1 = M_1, ..., a_{k-1} = M_{k-1}$ and $a_k, a_{k+1}, \ldots, a_{k+t-1}$ are the private values given by the shareholders to the dealer through a private channel.

- D Selects an integer I_0. For each level, a sequence of pairwise co-prime positive integers is selected and made public. Integers at each level equal to the number of shareholders at that level, i.e., $\left(I_1^i, I_2^i, ..., I_{N_i}^i\right)$ with $\left(I_1^i < I_2^i < ..... < I_{N_i}^i\right)$, where $i = 1, 2, ..., z$ and Greatest Common Divisor (GCD) of I_0 with every other selected integer should be 1.

- D creates $t+k$ shares of the polynomial, and for each share $f(r)$, the dealer forms $f(r) + \delta_{x,N_i}^i * I_0$, where δ_{x,N_i}^i is a random value selected by the dealer for the share number x of shareholders N_i at i^{th} level with x varying from 1 to $t+k$. In N_i, N is the number of shareholders at i^{th} level. The value δ_{x,N_i}^i is different for each level and each share. $\left(f(r) + \delta_{x,N_i}^i * I_0\right)$ should lie between

$$\left(I_{N_{i-t+2}}^i * I_{N_{i-t+3}}^i * ... * I_{N_i}^i\right) < \left(f(r) + \delta_{x,N_i}^i * I_0\right) < \left(I_1^i * I_2^i ..... I_t^i\right) \tag{4.4}$$

This is the threshold range for every level, and secrets should lie in this range; otherwise, the algorithm would be inconsistent, i.e., reconstruction can be possible by combing less than t shares. The value to be shared is S_{x,N_i}^i : (S_{x,N_i}^i corresponds to share a number x of the shareholder N_i at i^{th} level with x varying from 1 to $t+k$).

$$S_{x,N_i}^i = \left(f(r) + \delta_{x,N_i}^i * I_0\right) \bmod I_{N_i}^i \tag{4.5}$$

- Before distributing S_{x,N_i}^i, D computes its hash values, and these values are made public so that everyone can access it. Shareholders accept the share if and only if its hash value matches with the previous hash value published by the dealer; otherwise, discard it. This mechanism checks the dishonesty of the dealer and makes the scheme verifiable. Thus the dealer is not able to distribute invalid shares.

- The dealer distributes shares S_{x,N_i}^i. Similarly, $t+k$ polynomial values are shared among all the shareholders at each level.

Case 2: Inter-Level Secret Sharing

For interlevel SS, D needs to select another parameter $I^i_{N_{i,j}}$ where the shareholder N_i contributes his share to j^{th} level for secret reconstruction with

$$I^j_t < I^i_{N_{i.j}} < I^j_{N_{j-t+2}} \tag{4.6}$$

Then the dealer computes $\Delta S^i_{x,N_{i,j}}$

$$\Delta S^i_{x,N_{i,j}} = f(r) + \delta^i_{x,N_i} * I_0 - S^i_{x,N_i} \tag{4.7}$$

With a share of the shareholder in interlevel sharing as $S^i_{x,N_i} + \Delta S^i_{x,N_{i,j}}$.

4.5 Secret Reconstruction

A system of equations is formed based on the distributed shares. Dealer D accepts shares only if the share is valid, which is verified using the hash value published by D before. An equation that is formed is given as:

Case 1: Intra-Level Secret Sharing

$$\delta^i_{x,N_i} * I_0 \bmod I^i_{N_i} \tag{4.8}$$

Case 2: Inter-Level Secret Sharing

$$S^i_{x,N_i} + \Delta S^i_{x,N_{i,j}} \bmod I^i_{N_i,N_j} \tag{4.9}$$

Using CRT, a unique solution for $X = f(r) + \delta_{x,i} * I_0$, $f(r)$ can be reconstructed by

$$f(r) = x \ mod \ I_0 \tag{4.10}$$

After getting all the polynomial shares by CRT, the following equation is used to reconstruct the polynomial

$$\begin{aligned} f(x) &= \sum_{i=1}^{t} f(i) \prod_{j=1, j\neq i}^{t} (x-j)/(i-j) \bmod p \\ &= a_0 + a_1 \bullet x^1 + ...a_{k+t-1} \bullet x^{k+t-1} \end{aligned} \tag{4.11}$$

Thus, the authorized set of shareholders reconstructs the k secrets.

The proposed scheme is demonstrated in Figure 4.1, which shows 3 levels with 3 shareholders at each level.

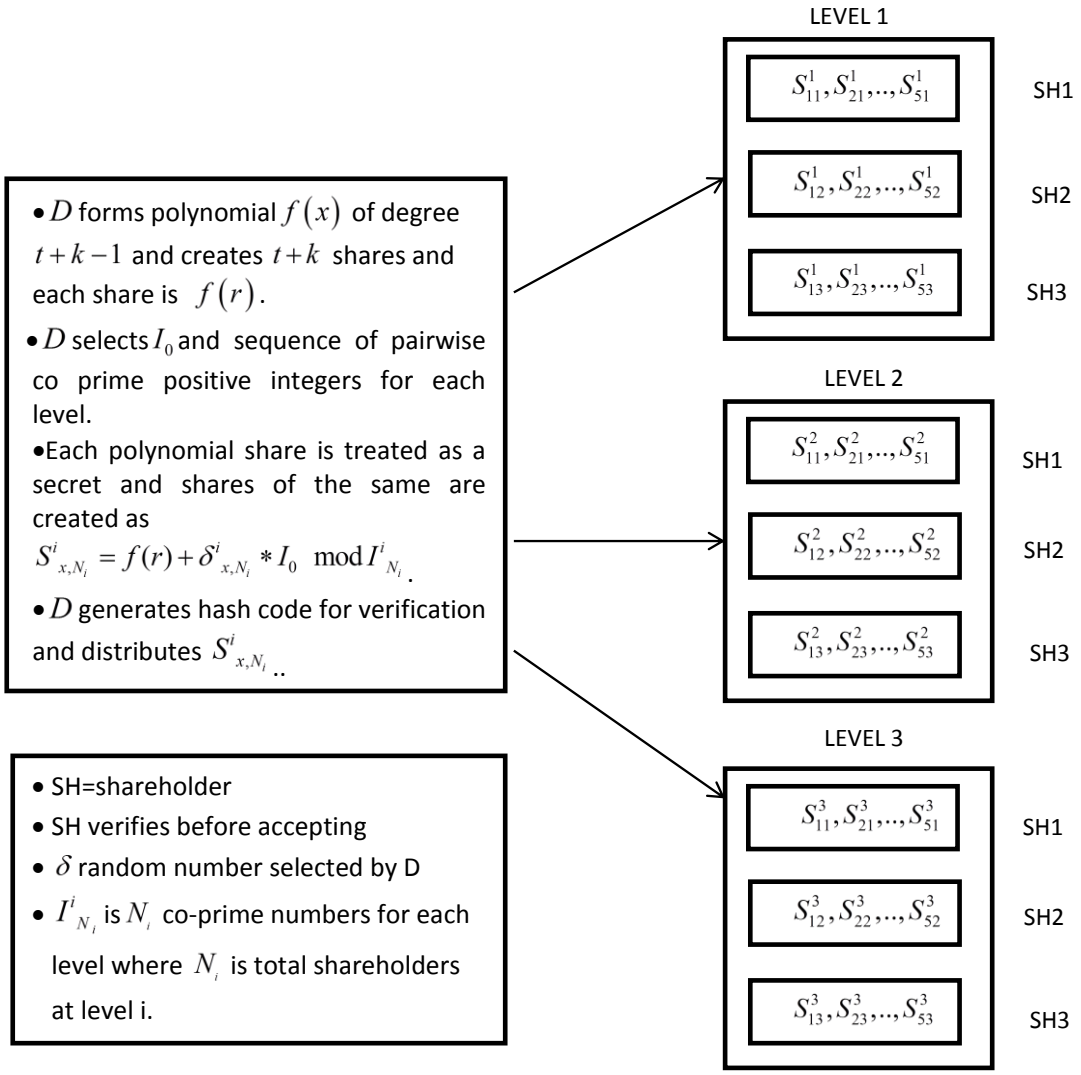

Figure 4.1: Proposed Algorithm

4.6 Implementation Results

The proposed scheme described in Section 4 is implemented in C/C + + using GMP (GNU Multiple Precision) and NTL (Number Theory Library) libraries and tested on a 3-GHz third-generation system. GMP is an open-source multi-precision library that can be used for various types of operations on signed integers, floating point numbers

and rational numbers. The richness of function, friendly interface, and freely availability make it so popular and useful. The limit of precision depends upon the machine, not on the library. The application includes cryptography and its application, security over the internet, algebraic number theory, and many more. For better insight, implementation results are presented for small numbers, and the algorithm is tested for large numbers as well. For demonstration, shareholders are divided into 3 levels $(z=3)$, namely L_1, L_2, and L_3 with 3, 4, and 7 as the number of shareholders at respective levels, i.e., $N_1=3$, $N_2=4$, $N_3=7$. The threshold (t) and the prime (p) being considered are 3 and 563, respectively. The number of secrets to be shared is 2 $(k=2)$ where $K_0=3$ and $K_1=2$ and the coefficient values provided by an authorized set of players are 2, 1, 0 (the authorized set used for secret reconstruction comprises 3 players).

(i) Dealer forms the polynomial of a degree $t+k-1$ using the values, i.e.,

$$f(x)=3+2*x+2*x^2+1*x^3+0*x^4 \tag{4.12}$$

(ii) Dealer selects $I_0=863$, and the sequence of pairwise co-prime integers selected for each level are

- For level 1: $I_1=137$, $I_2=139$, $I_3=250$ and threshold range for this level is (34750, 4760750)
- For level 2: $I_1=293$, $I_2=307$, $I_3=313$, $I_4=319$ and the threshold range for this level is (99847, 28154663)
- For level 3: $I_1=229$, $I_2=233$, $I_3=239$, $I_4=241$, $I_5=277$, $I_6=281$, $I_7=283$ and the threshold range for this level is (79523, 12752323)

(iii) Dealer creates $t+k$ shares:

$f(1)=8$, $f(2)=23$, $f(3)=54$, $f(4)=107$, $f(5)=188$ and selects $\delta_{x,i}$ as a value for each level which is shown in Table 4.1.

Table 4.1: Values Selected by Dealer

Levels	$\delta_{1,i}$	$\delta_{2,i}$	$\delta_{3,i}$	$\delta_{4,i}$	$\delta_{5,i}$
Level 1	550	558	510	620	456
Level 2	9864	8824	8999	9345	9500
Level 3	10946	10567	11001	10765	10899

Therefore, all 5 shares of a 1st shareholder are:

$$\begin{aligned} 8+550*863 \ mod \ 137 &= 90 \\ 23+558*863 \ mod \ 137 &= 22 \\ 54+510*863 \ mod \ 137 &= 3 \\ 107+620*863 \ mod \ 137 &= 45 \\ 188+456*863 \ mod \ 137 &= 115 \end{aligned} \tag{4.13}$$

Similarly, shares of other shareholders are calculated, which are shown in Table 4.2.

Table 4.2: Shares of Shareholders

Shareholders	1st share	2nd share	3rd share	4th share	5th share
1st shareholder of level 1	90	22	3	45	115
2nd shareholder of level 1	112	81	110	17	68
3rd shareholder of level 1	158	77	184	167	216
1st shareholder of level 2	111	65	226	17	255
2nd shareholder of level 2	144	0	12	259	253
3rd shareholder of level 2	292	158	35	84	279
4th shareholder of level 2	125	286	136	203	69
1st shareholder of level 3	156	106	35	1	79
2nd shareholder of level 3	120	190	99	126	48
3rd shareholder of level 3	170	60	120	133	180
4th shareholder of level 3	170	145	204	234	36
5th shareholder of level 3	152	227	19	276	213
6th shareholder of level 3	29	51	51	161	112
7th shareholder of level 3	149	235	116	261	237

Case 1: Intra Level Secret Sharing

While recovering the secret from level 2, 3 out of 4 shareholders need to contribute their shares. Say first 3 are taking part in the protocol. Following the system of equations needs to be solved using CRT:

$$\begin{aligned} X &= 111 \; mod \; 293 \\ X &= 144 \; mod \; 307 \\ X &= 292 \; mod \; 313 \end{aligned} \tag{4.14}$$

This gives $X = 8$. In the same way, equations can be formed with other shares of the shareholders, and results can be obtained accordingly.

Case 2: Inter Level Secret Sharing

Considering secret reconstruction done at level 3, a 1st shareholder of each level is contributing their shares for reconstruction. Dealer selects two values (because 2 out of 3 shares belong to other levels) between 241 and 277, which are co-prime to one another. Say the values are 253 and 263; the following system of equations is formed for secret recovery:

$$\begin{aligned} X &= 90 + 55 \; mod \; 253 \\ X &= 111 + 124 \; mod \; 263 \\ X &= 156 \; mod \; 229 \end{aligned} \tag{4.15}$$

Solving these equations using CRT, we get $X = 8$, and similarly, other shares are obtained. Further, these values are used in Lagrange's interpolation to reconstruct the polynomial

$$f(x) = 3 + 2*x + 2*x^2 + 1*x^3 + 0*x^4 \tag{4.16}$$

moreover, the secrets are recovered.

4.7 Application of the Proposed Scheme

The proposed scheme can be advantageous for sharing multiple secrets in a multilevel environment (for organizations having hierarchal structures). Considering an example of the Indian Army, suppose a Colonel has some secrets (secret keys/passwords), and

he is on leave or on some special mission for some days. Practically, it is not advisable to hand over the secrets to a single officer (superior or subordinate). So, he would make share the secrets and hand them over to officers of various ranks. He may give some shares (these shares include shares formed by splitting multiple secrets to be circulated) to higher rank officers (Lieutenant Colonel or Brigadier) and others to peers or subordinates (Major, Captain, and Lieutenant). Now, if an emergency arises for secret reconstruction at the Captain level, then the officer only at a peer or higher rank can contribute in share reconstruction. The secret is reconstructed, provided the threshold condition is satisfied. This algorithm can be used in the case when a higher rank officer (say Colonel) does not want entities or members at lower levels (Major, Captain, and Lieutenant) to recover secrets on their own without any members of lower levels. For this, he can set the threshold value more than the number of entities present at that (lower) level or another alternative is to provide more shares to entities at a higher level and less number of shares to entities at lower levels.

4.8 Security Analysis and Comparison

The Proposed work is analyzed in this section, and a comparison with some existing schemes is also performed.

4.8.1 Security Analysis

i) Traceability: The algorithm is said to be traceable when it is possible to find out whether any participant during the reconstruction phase has submitted any invalid or fake share or not.

Proof: let $f(i)$ be the original valid share and $f'(i)$ is the fake or invalid share. If any participant sends $f'(i)$ to the dealer instead of $f(i)$, then the dealer does not accept the share because

$$H(f(i)) \neq H(f'(i)) \tag{4.17}$$

Here H is a one-way hash function, and it is complicated to find 2 values that result in the same hash value. Thus, the algorithm is traceable.

ii) Robustness: The scheme is said to be robust if all the secrets can be recovered by pooling t or more shares. The use of Lagrange Interpolation has made the scheme more robust. Any t honest players can unlock the shared secret.

iii) Confidentiality: The scheme holds confidentiality if even $t-1$ players are not able to reveal the secret. Assume $t-1$ participants are available for secret recovery and the product of their moduli is X'. These $t-1$ shareholders use CRT to recover a secret. Suppose they obtained a value S', the relation between the original secret and the recovered secret is

$$S = S' + \delta * X' \tag{4.18}$$

Here, S is the original secret. Predicting the correct value of δ to reach the original secret is very difficult. Thus, even with $t-1$ shares, the scheme does not leak any information about the secret.

iv) Consistency: The algorithm holds consistency if any set of valid shares of the secret reveals the same secret. Here, in the proposed algorithm, consistency is achieved due to Lagrange's interpolation, and whether the share is valid or not is verified through one way hash function.

v) Dealer Leakage Resilient (DLR): The dealer is said to be dishonest if he subliminally leaks the information in the valid shares. This dishonest strategy allows the dealer to preserve consistency in the system and helps the attacker to unlock secret before the reconstruction phase from the leaked information. The system exhibits DLR-VSS property if the attacker does not gain information about the secret before the reconstruction phase.

Proof: The DLR-VSS property is achieved by taking the power of randomness from the dealer. The dealer does not have the capability of employing randomness in the system. By this, the dealer will no longer be able to hide information because no value is selected by his own choice.

vi) Salted Hashing: can be used in place of simple hashing. In salted hashing, a random number, referred as salt, is added to the share before using one-way hash

functions. Salted hashing ensures that no two similar secrets yield similar hash codes. However, only the dealer can verify shares submitted by shareholders. It is assumed that the dealer is honest and not distributing invalid shares. When hashes were randomized, tools like rainbow tables, lookup tables, and reverse lookup tables were not able to produce deceitful behavioir. For the pre-computation of the rainbow or lookup table, salt needs to be known in advance, and this is not possible.

Another possible method to use salted hashing and still verification is possible from both ends, i.e., shareholders can verify shares before accepting them from the dealer, and the dealer can verify shares before accepting them from shareholders before the reconstruction phase. This can be achieved by treating salt (or random number) as one of the secrets. The constant term of the polynomial will be the salt and degree of the polynomial $t+k$.

vii) Knowledge of Number of Shares: If t and k are not publicly known values, then it is desirable that the adversary must not get any information about some secrets (in the case of multi-secret schemes) just by looking at the number of shares of each shareholder. This property is achieved by distributing $t+k$ shares instead of k shares and t and k are kept a secret, so the adversary is not able to access these values. Table 4.3 shows the comparison of the schemes by security assumptions. The acronyms R, C, V, and T stand for robustness, confidentiality, consistency, and traceability, respectively. (R, C, V, T) of VSS schemes. Therefore, it is confirmed that the scheme is verifiable.

Table 4.3: Comparisons through Security Property

Scheme No.	Robustness (R)	Confidentiality (C)	Consistency (V)	Traceability (T)
[21]	Yes	Yes	No	Yes
[50]	Yes	Yes	No	Yes
[139]	Yes	Yes	Yes	Yes
[168]	Yes	Yes	No	Yes
[169]	Yes	Yes	No	Yes
[170]	Yes	Yes	No	Yes
[171]	Yes	Yes	Yes	Yes
Proposed	Yes	Yes	Yes	Yes

In Table 4.4, the suggested algorithm is compared with various techniques by communication cost over secure and insecure channels. Communication cost over secure channels is analyzed separately for both share distribution and share reconstruction. Here, n, t and C respectively denotes the number of shareholders, threshold, and any constant number. Communication cost over a secure and insecure channel is calculated using p (1024 bits), q (1024 bits), and $N = p*q$ (1184 bits). In addition to communication cost, the security assumption of different schemes is also analyzed.

Table 4.4: Comparison through Communication Cost

Scheme	Distribution Cost over Secure Channels	Reconstruction Cost over Secure Channels	Communication Cost over Insecure Channels	Security Assumption
[21]	$160n$	$160k*t$	$160*n*k + C*n*k$	Hashing
[50]	$1024\left(n_1 + n_2 + ...n_z\right)$	$1024*t$	$1024\left(n_1 + n_2 + ...n_z\right)$	Unconditionally secure
[139]	$320n$	$160t$ or $160k$	$C*n$	Hashing
[168]	-	$160t$ or $160k$	$1184n + 160n + 1024t$ or $1184n + 160n + 1024k$	DLP
[169]	$1184n$	$160t$ or $160k$	$1024n + 160n$	RSA & DLP
[170]	-	$160t$ or $160k$	$1184n + 1184n + 160n$ or $1184n + 1184n + 160k$	RSA & DLP
[171]	$320n$	$2048t$	$2048\left(n+1\right) + 2048n$	DLP
Proposed	$160*\left(t+k\right)*$ $\left(n_1 + n_2 + ...n_z\right)$	$160k*t$	$\left(160 + C\right)*$ $\left(n_1 + n_2 + ...n_z\right)$	Hashing

In [50], variables n_1, n_2, and n_3 used are some shareholders at different levels, where z is the total number of levels. This Scheme is multilevel secret sharing and uses the Chinese remainder theorem. Consider there are z levels. Here dealer publishes sequences of co-prime numbers equal to the number of shareholders for each level, which are of 1024 bits each. This adds to the communication cost of

$1024(n_1 + n_2 + ...n_z)$ bits over insecure channels. Dealer computes shares for n shareholders and distributes them, which are of 1024 bits each which make the communication cost over the secure channel to $1024*n$ bits; then t shareholders collaborate to reconstruct the secret. Thus, the communication cost for reconstruction is $1024*t$ bits.

In [168], each participant selects his secret shadow of 1184 bits and sends it to the dealer through a secure channel which makes the communication cost over the secure channel to $1184n$ bits. There are two possible cases, first is when $t > k$, in this case, the dealer forms a polynomial of degree $(t-1)$. He creates shares of the polynomial of 160 bits each and publishes them $(160*n)$. Dealer computes t values of 1024 bits that are used in the verification phase and publishes them $(1024*t)$. t Shares of 160 bits each are pooled to recover the secret, which makes the cost of reconstruction as $160*t$. The second case arises when $t < k$. This time dealer forms a polynomial of degree $(k-1)$. He creates shares of the polynomial of 160 bits each and publishes them $(160*n)$. Dealer computes t value of 1024 bits that are used in the verification phase and publishes them $(1024*t)$. k Shares of 160 bits each are pooled to recover the secret, which makes the cost of reconstruction to $160*k$ in the second scenario.

In [169], which is a multi-secret sharing scheme where each participant selects his secret shadow of 1184 bits and sends it to the dealer through a secure channel which makes the communication cost over the secure channel to $1184*n$ bits. After some computation dealer publishes a value of 1024 bits for each participant, which adds $1024*n$ bits to the communication cost over insecure channels (public channel); then, the dealer uses the public channel to distribute shares of 160 bits for each shareholder $(160*n)$. t or k shareholders submit their shares for secret reconstruction. Thus, the reconstruction cost becomes $160*t$ if secrets are less than the threshold, and it is $160*k$, if secrets are more than the threshold.

In [139], there are k secrets to be shared. There are two cases. In the first case, $t > k$, the dealer forms two polynomials of degree $(t-1)$. One is used to generate shares of multiple

secrets, and the other is used for the verification phase. Dealer computes n shares of 160 bits each, of each polynomial and sends it via a secure channel to each shareholder, which makes the communication cost over the secure channel to $(2n*160)$ bits. Further, using insecure channel, one hash code for the two shares of constant length C is published, which add $C*n$ communication cost over insecure channels. Now, in the reconstruction phase, the combiner/dealer combines t out of n shares of 160 bit each of 1st polynomial to recover the secret, which adds communication cost of $160*t$ over the secure channel. In the second case, $t < k$, here dealer forms polynomials of degree $(k-1)$. In this case, the communication cost for the reconstruction of secrets is $160*k$.

In [170], it is also a multi-secret sharing scheme where participants select their secret shadow of 1184 bits and send it to the dealer via the insecure channel, which leads to a cost of $1184*n$. Dealer publishes n values of 1184 bits $(1184*n)$, which are used for verification. Dealer computes shares of 160 bits for the shareholders, which are also published, which makes the communication cost over in secure channel to $160*n$ bits. Then, t or k shares are collaborating to reconstruct secrets, which depend on the value of k. Thus the communication cost in the reconstruction phase will be $160*t$ or $160*k$ bits over the secure channel.

In Scheme [21], multi secret sharing scheme uses hashing for verification. The dealer sends a private value of 160 bits each, to each shareholder via a secure channel which makes the communication cost over the secure channel to $160*n$ bits. He publishes k shares of 160 bits each $(160*n*k)$ and their hash codes of constant length $(C*n*k)$ for each participant. Futher, t shares of k secrets are combined for the reconstruction, which makes the cost of reconstruction $160*k*t$.

Here, in [171], each of n the shareholders sends 2 private values via a secure channel, which are of 160 bit each, to form two polynomials that make $2n*160$ cost over a secure channel. Then, dealers generate and publish (insecure channel) commitment value, each of 1024 bits for the $2n$ values, which leads to $2n*1024$ cost over an insecure channel. Now, the dealer forms two polynomials, each of degree $t-1$

. After this, the dealer computes n shares to from each polynomial and publishes them. This adds $2n*1024$ bits to communication cost over an insecure channel. In the reconstruction phase, the combiner/dealer combines t out of n shares of 1024 bits each to recover the secret, which adds communication cost of $2t*1024$ over secure channel.

In the proposed algorithm (multi-level and multiple secret SS scheme), z levels are considered with n_i shareholders for each level (i varying from 1 to z). Dealer forms a polynomial of degree $t+k$ and, for each level, publishes a sequence of co-prime numbers that are equal to the number of shareholders at that level $\left(\left(n_1+n_2+....+n_z\right)*160\,\text{bit}\right)$. Each shareholder sends a private value to the dealer through a secure channel which adds $\left(160*n\right)$ bits to the communication cost. Dealer forms $t+k$ shares of this polynomial and creates shares of a polynomial using CRT, and these shares, which are of 160 bits, are distributed via a secure channel to shareholders $\left(160*\left(t+k\right)*\left(n_1+n_2+....+n_z\right)\right)$. He also publishes the hash code $\left(C*\left(n_1+n_2+....+n_z\right)\right)$ of all the shares for verification. In the reconstruction phase, $t*k$ shares are used, which makes reconstruction cost $160*t*k$ bits.

Table 3.5 shows the comparison of our scheme with other schemes [21], [50], [139], [171], [172], concerning various parameters mentioned in the table. Some of the properties are explained below:

- The scheme is multi-use if the shares of participants are different for different secrets.
- The algorithm can resist conspiracy attack if $t-1$ corrupt shareholders cannot unlock the secret. A conspiracy-resistant scheme ensures that the reconstruction of recovered secrets does not give information about open secrets.
- SETUP (Secretly Embedded Trapdoor with Universal Protection) is a technique where an attacker breaks the security of the system, and secret information is

leaked, but other parties of the protocol are not able to detect this malicious behaviour. All VSS schemes are not SETUP resilient.

- The scheme is unconditionally secure if its security does not depend on any mathematical construct. It is said to be secure even if the adversary has unbounded computational power.

Table 4.5: Comparison through various parameters

Property	[21]	[50]	[139]	[171]	[172]	**Proposed algorithm**
Dealer Publishes the Share	Yes	No	No	Yes	No	No
Hash For Verifiability	Yes	No	Yes	No	-	Yes
Modular Exponentiation or DLP	No	No	No	Yes	No	No
Multi-Use Scheme	Yes	-	No	No	No	Yes
Can verify Dealer's honesty	No	No	Yes	Yes	No	Yes
Can verify Shareholder's honesty	Yes	No	Yes	Yes	No	Yes
Has unconditional security	No	Yes	No	No	Yes	No
Outside Adversary does not know Number of Secrets	No	-	Yes	-	No	Yes
Secret Revealing Order	Any	-	All at a time	All at a time	Any	Any
Conspiracy Attack Resistance	Yes	Yes	Yes	Yes	Yes	Yes
SETUP Attack Resistance	No	No	No	Yes	No	Yes

4.9 Summary

SS is an important sphere of Information Security and is attracting a lot of research interest these days. The primary objective is to develop efficient schemes that are secured and can be deployed practically. In this regard, this study proposes a hybrid based VSS scheme that may be utilized to communicate various secrets in a multilevel framework. The technique can reveal several secrets at different levels in a single run. The proposed algorithm holds for all requirements of VSS (Table 4.5), and the scheme is computationally efficient, too (Table 4.4). The scheme also exhibits the property of dealer

leakage resilience, which is achieved by restricting the dealer to employ randomness. Consequently, the dealer is not able to hide secret information in the share of the shareholders. So, the proposed scheme is to work in an environment where dealers and/or shareholders are not honest and also when they are not mutually trusted.

CHAPTER 5

A BLOCKCHAIN-BASED LIGHTWEIGHT SCHEME FOR CONSTRAINED IoT ENVIRONMENTS

Blockchain technique is now being used in a variety of industrial and day-to-day applications. Technology is paving the way for technological advancement, which would help the community to build secure systems. Blockchain has the ability to greatly improve data privacy and security while also improving cloud data accuracy and integrity [173].

5.1 Introduction

Blockchain was first proposed by David Chaum [174] and later modified by Haber and Storenetta [175]. Blockchain is a novel concept for storing data. In blockchain technology, blocks are used to store the data, and blocks are connected via the chain. In the blockchain, once the data of a block is chained, then its data cannot be changed until and unless legitimate permission is there. Each block has its own digital signature, which is unique, and if some change happens in a block due to an external attack, the signature will be altered [176]. The first block is known as the genesis block, with the hash value 00000000. Using the data and hash of Genesis's block, a new hash value is computed as 0234ABED4. This hash value and data of the subsequent block are used to compute the new hash value A4CE23847, as shown in Figure 5.1(a).

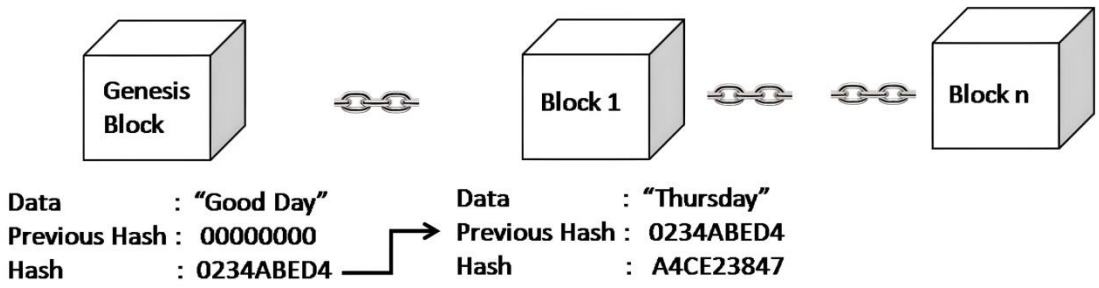

Figure 5.1(a): Schematic of Block linkage through Hash

The detailed block structure is shown in Figure 5.1(b). Block header is used to identify a particular block, mentioned hash of the parent block is received from the

previous block, i.e., mentioned parent block hash in Block *i* is received from block *i*-1. The timestamp represents the time of block generation; nonce ("number only used once,") is 32, its unique number and used in Hash calculations. Merkle root is the root value of all the transaction or data values.

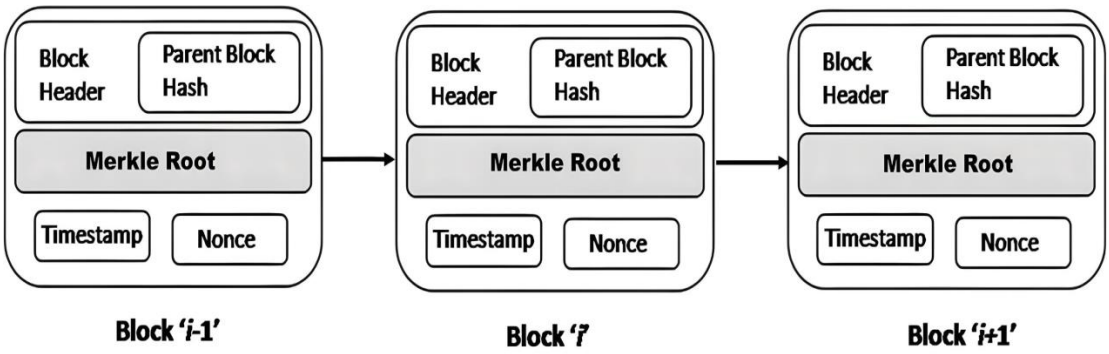

Figure 5.1(b): Detailed Block Structure for Blockchain

Each user has corresponding key pair, namely, private and public keys. To ensure non-repudiation, every transaction is signed with a private key, and it is kept secret. These transactions are signed digitally and broadcast across the whole network. Digital signature comprises two phases: the signing phase follows by the authentication phase. By taking, an instance, Alice wishes to send a message to user Bob. During the signing procedure, Alice encrypts his data with her private key and sends both the encrypted and original data to Bob. In the verification procedure, Bob checks the value by using Alice's public key. Bob could easily tell whether the information had been tampered with or not. The elliptic curve digital signature method (ECDSA) [177] is the most extensively used digital signature algorithm in the blockchain.

Every network transaction is recorded in the blockchain, which is duplicated in each node. A blockchain is a method for performing reliable computation for a large number of suspicious users. For blocks, the blockchain employs a data structure that contains the hash value of previous and subsequent blocks. The data structure's hash values are used to ensure that data in a block cannot be modified unless the post block is accepted. By allowing invisibility, only a few blockchain systems hide the user's identity and transact using the high-level programming language. The blockchain

ensures an unequalled level of security, which is both promising and reassuring. Bitcoin has become increasingly essential in online payment due to the development of blockchain technology.

5.2 Key Features of Blockchain

Blockchain technology is much more than just a backup network for currency. There are some of the key characteristics of blockchain which make it appealing and useful for users. The important features of blockchain are discussed below:

5.2.1 Immutability

After a transaction has been recorded in the ledger, it cannot be changed [178], [179]. If a transaction contains an error, it must be replaced with a new transaction to fix the problem. Both of these transactions are then available [180]. An error in a transaction happens when a transaction fails or is rejected by hackers. Transactions with extremely low fees are frequently rejected.

5.2.2 Decentralized

Decentralized network, the authority is not centrally located; instead, there are multiple authorities which is looking after it. The network is decentralized, and it is managed by multiple nodes, which may be at diverse locations. It is an excellent example of a crucial feature of blockchain technology [181]. Users can simply access the system via the internet and can save their assets therein since it does not require any regulatory authority.

Some of the advantages of the decentralized feature of blockchain –

i. Minimal Failures
ii. User Control
iii. Less Prone to Breakdown
iv. No involvement of Third-Party
v. Zero Scams
vi. Transparency

5.2.3 Enhanced Security

Encryption gives an extra layer of security to the system. Every bit of information that exists on the chain structure is secured using hashing. Every block contains its own data (transactions) and previous block hash. Any attempt to change or data tampering will lead to alternation in all hash IDs, thereby making it a secure system [182].

5.2.4 Distributed Ledgers

A public ledger will usually offer all required information about a transaction and its stakeholders because the content is publically available. Any change or alternation to such a ledger is verified using any consensus mechanism, where there is no central authority, and all the members of the blockchain agree upon a consensus to validate the transition [181]. There is a timestamp associated with every record, and also cryptographically secure, making it auditable and immutable history is recorded.

Benefits of Distributed Ledgers are as follows:

- Very fast
- No suspicious activity
- Controls the authentication Process

5.2.5 Consensus

For a transaction to be valid, all participants must agree on its legitimacy. Users can choose from a variety of consensus algorithms on the blockchain, depending on the needs of the blockchain application [182].

5.2.6 Smart Contract

Smart contracts [183] are computer programs that aid in the transmission of money or other valuables. These programs are launched automatically when a policy is met. A contract address, defined functions, and private storage are all included in each smart contract. Ethereum [184] is an open-source, decentralized platform for smart contract execution.

5.3 Types of Blockchain

Blockchain is divided into four types based on the features, it provides to cater the particular application as shown in Figure 5.2.

	Public (Permission less)	Private (Permissioned)	Hybrid	Consortium
Advantages	+ Independence + Transparency + Trust	+ Access Control + Performance	+ Access Control + Scalability + Performance	+ Access Control + Scalability + Security
Disadvantages	- Scalability - Performance - Security	- Trust - Audiability	- Transparency - Upgrading	- Transparency
Applications	• Cryptocurrency • Document validation	• Supply chain • Asset ownership	• Medical records • Real estate	• Banking • Research • Supply chain

Figure 5.2: Types of Blockchain and Their Applications

5.3.1 Permission-less blockchain

A blockchain can be classified as permissioned if control of participants or how access to fresh block contents and the creation of new blocks is not regulated by authorities. It allows any node without a valid address (a unique ID) to join and leave the network without requiring permission. Furthermore, such nodes can accept, send, and validate blocks using the same rules. The Bitcoin network is an excellent example of this, as users may conduct transactions using Bitcoin. This type of blockchain operates in a zero-trust environment, necessitating efficient consensus techniques to prevent malicious users from compromising the network. This type of blockchain is also called a Public Blockchain [185]. Examples: Bitcoin, Ethereum, Litecoin and NEO.

5.3.2 Permissioned blockchain

Permissioned blockchain demands proper authorization of the members before participating in network operations. The consensus body either be subsidiaries of a group of companies or a single private company. Permissioned blockchains are ideal for multiparty or internal business applications due to their effective management and identity-revealing requirements. Furthermore, permissioned blockchains facilitate the

deployment of more effective consensus algorithms with larger transaction capabilities due to the limited size of their consensus body. This type of blockchain is also called a Private blockchain [186]. Examples: Multichain, Hyperledger Fabric, Hyperledger Sawtooth, Corda, etc.

5.3.3 Consortium blockchain

A consortium blockchain is managed by a group of organizations rather than a single organization. This results in a higher degree of decentralization and a more secure system. Hyperledger [187], which uses a specified group of nodes to control the consensus process, is the most commonly accepted example of a consortium blockchain. Further to that, there is always a challenge to connect the various organizations to use similar technology and maintain infrastructure. Example: Marco Polo, Energy Web Foundation.

5.3.4 Hybrid blockchain

Hybrid blockchain, as the name suggests, is between the two technologies. This is managed by a private entity but can have some public part in it. This may depend upon the type of application and its associated requirement. For example, the joining of a new member/block may be managed privately, but consensus mechanisms are driven using public authorities. An implemented example of a hybrid blockchain is IBM Food Trust, which is a landmark in the industry that manages the supply chain using blockchain.

5.4 Recent Literature on Consensus Algorithms

In this section, the different types of consensus algorithms are discussed. In Figure 5.3, consensus algorithms are categorized into two different types: the first is Proof based, in which two different nodes compete with each other to do more justification with their respective transactions. The second is vote based, with the majority winning to decide upon the execution of a transaction.

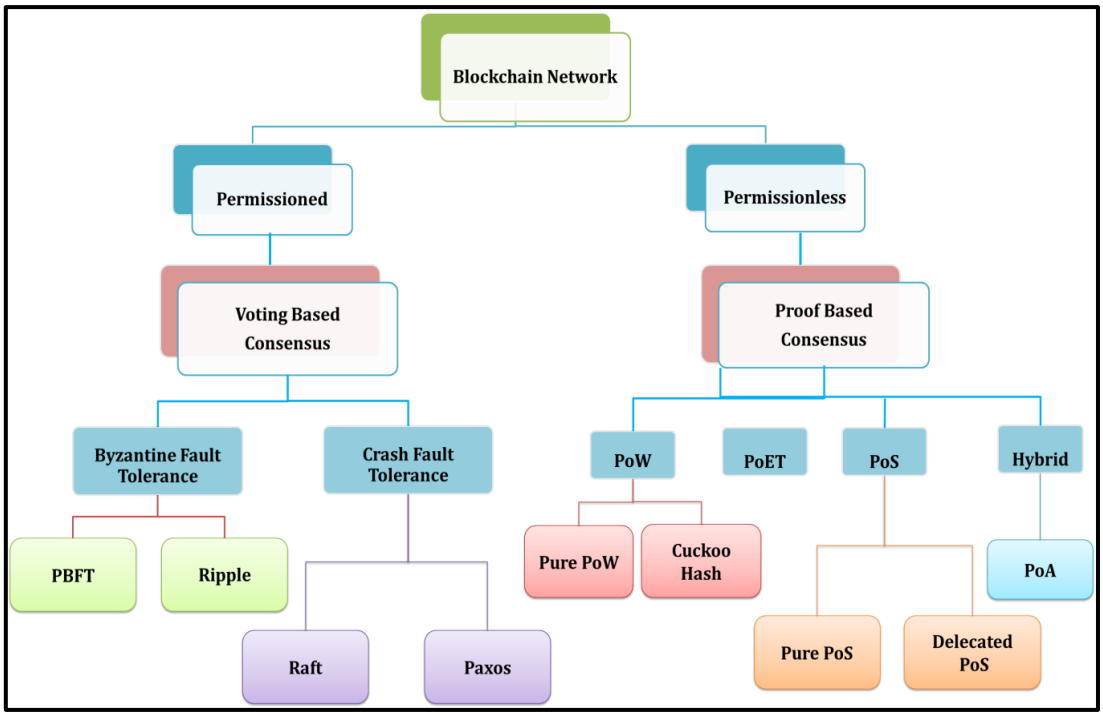

Figure 5.3: Taxonomy of Consensus Algorithm

5.4.1 Proof-based Consensus Algorithms

The miner node has the power to connect all blocks in their consensus. Some examples of proof-based algorithms are PoW, PoS, and a hybrid form of PoW and PoS.

i) Proof of Work and other Related Consensus Algorithm

It prevents the attack of fraudulent nodes over honest nodes, because of which it is a widely used method for Bitcoin consensus. Basically, in this technique, the minor node resolves any problem by connecting to an existing blockchain [188], [189], [190], [191], [192]. The nodes evaluate the hash value of the block header, which also includes nonce [193]. This difficult mathematical problem affects the average mining time of a block and the number of forks in a chain [190].

ii) Hybrid

Hybrid consensus does not get involved in fraudulent activities until the group of nodes acquires more than 50% mining power which can control the entire network [193], [194]. In such a scenario, the miner node develops a special block that works as a regular block and retains the same number of transactions. However, this method

does not work in the case of special transactions. The miner may be given the opportunity to utilize a new block based on the amount of money spent on this special transaction. Any mathematical problem which is similar to PoW, can be resolved by mining technique. The miner who answers the problem first receives back the 1% of what he spent on the unique transaction [194].

5.4.2 Voting-based Consensus Algorithms

As discussed above, the permissioned type of blockchain requires only a limited number of nodes to participate in the process of consensus. In such situations, the blockchain system employs consensus processes based on voting. The common examples of Permissioned or Private Blockchain Systems are Banks, corporations, and other institutions.

Here, a threshold value is required before a new block is attached to an existing blockchain in order to extend the chain. Byzantine Fault Tolerance (BFT) and Crash Fault Tolerance (CFT) are two types of voting-based consensus methods (CFT).

i) Byzantine Fault Tolerance (BFT)

To broadcast any message, it has to be transferred from one node to another in the form of a blockchain. The message is forwarded to all of the receiving roots' siblings. The problem with such a type of message transmission is that if any malicious node alters the message and transmits it to its siblings, the entire process gets hampered. This type of problem is termed as Byzantine problem. However, to resolve such problems, BFT is used, which takes action based on the statement of the majority of nodes [195]. PoW is used in the Byzantine fault tolerance scheme to combat corrupted nodes that pass false information. In real-world applications, it is not guaranteed that the message will be received in a specified time, which is also termed as asynchronous approach. To address such issues, practical byzantine fault tolerance, a form of BFT, has been proposed [193].

a) Practical Byzantine Fault Tolerance (PBFT)

Consensus techniques like PoW and PoS are inefficient for handling the applications of the real world. However, techniques like PBFT are based on the concept that it can handle a network where one–third of nodes are corrupted. Basically, PBFT can work even if the network contains fraudulent nodes [188], [192], [193]. There is a primary

node, and the others are treated as minor nodes. The PBFT process is segregated into three phases, namely pre-prepared, prepared, and committed [196], [197].

ii) Crash Fault Tolerance (CFT)

CFT prevents distributed systems from collapsing due to the failure of nodes. The consensus algorithms like Paxos [198] and Raft [199] are used to prevent node crashing situations. The main aim behind using Paxos and Raft is that at least [n/2+1] nodes should be honest every time among all the nodes working in a network. Here, Raft-based algorithms are discussed in detail. Raft is derived from the Paxos, and Raft is a simplified version of Paxos, which is being used by the industry for practical industry processes.

a) Raft

This algorithm is developed on the presumption that a minimum of (n/2 + 1) nodes are honest in the entire set of nodes in the network. A verifying node can play three different roles depending on the situation: follower, candidate, and leader. Initially, all nodes behave as followers and elect their leader node, followed by voting for the leader node, where they act as candidate nodes and increase their term numbers. The message requesting a vote, as well as the current term number, is broadcast. The receiver discards the received message if the received term number is less than or equal to the receiver's term number; otherwise, the receiver accepts it.

The node which receives the maximum request will be the leader node. After the selection of a leader, the stages in the algorithm are followed to reach a consensus. This process helps in identifying the node crash [194], [200], [201]. Following that, the leader node selects an index number from the log entry and verifies all transactions that occurred prior to the selected index number. The valid transactions are then added to a block, which is then attached to the existing chain. Finally, the outcome will be broadcasted to all the followers [202].

5.5 The Proposed Scheme

This section describes the system model and the problem formulation.

5.5.1 System Model

A BC-envisioned lightweight scheme is proposed for constrained IoT environments. The details are presented in Figure 5.4. In the proposed scheme, entities are considered as $E = \{E_{CH}, E_{OG}, E_{BC}, E_M\}$ to denote CH, overlay gateways (OG), local blockchain clients (BC), and miners. In the scheme, a total of k clusters $\{C_1, C_2, ...C_k\}$ exists that includes end devices like sensors, actuators, and low-powered computational servers inside any k^{th} cluster. Each cluster k selects a node N as CH that forms a group $\{H_1, H_2, ...H_k\}$. Two groups are considered as CHs, H_k and H_q, $k \neq q$, that communicate in the Proximity Sensor Nodes (PSN). H_k is selected based on computational and storage power or the node that has maximum coverage [203]. CH elects an additional node as alternate CH $H_k^{'}$, in case H_k departs from the k^{th} cluster due to network connection issues or H_k joins any other cluster H_k is assumed to favour local miner nodes L_m to process blocks and add transactions to BC. The gathered data at Elected Cluster Head (E_{CH}) is now forwarded to $E_{OG,}$ which includes a control module Q and kernel manager W. Q is responsible to communicate with E_{BC} to facilitate E_M with computational requirements for the management of local chain structures. W is responsible for setting up security parameters for lightweight signcryption of the data. The aggregated data from E_{OG} is stored in E_{BC} that is accessed by client applications through a distributed application programming interface (API) in JavaScript object notation (JSON) format to leverage lightweight exchange. The signcrypted data E_d is then added to the global chain structure via public miner local P_m (Figure 5.4).

Figure 5.5 denotes the flow of the proposed scheme. The proposed scheme is based on the verified hash H_c. Once H_c is verified, the call to the signcryption scheme is made. Once signatures are generated, the scheme assures a low-powered energy-efficient consensus formation scheme (*PoEEC)* that aggregates the data and blocks proposals are formed.

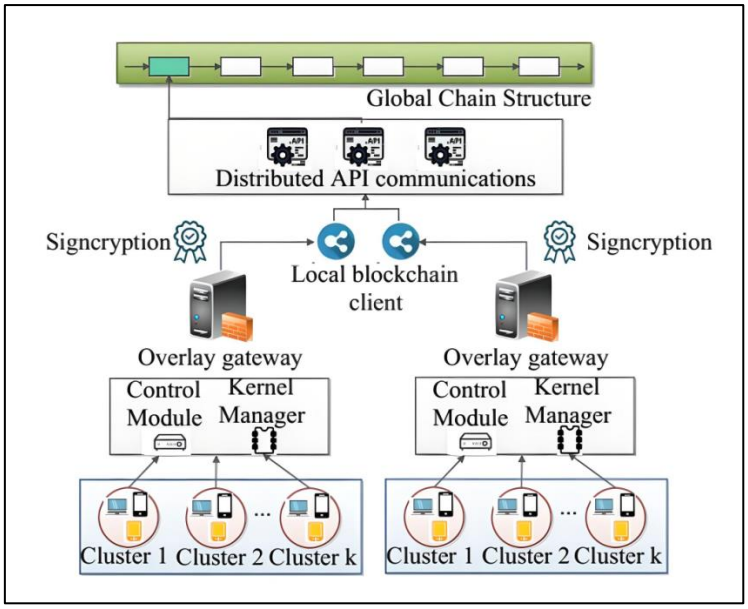

Figure 5.4: System Model

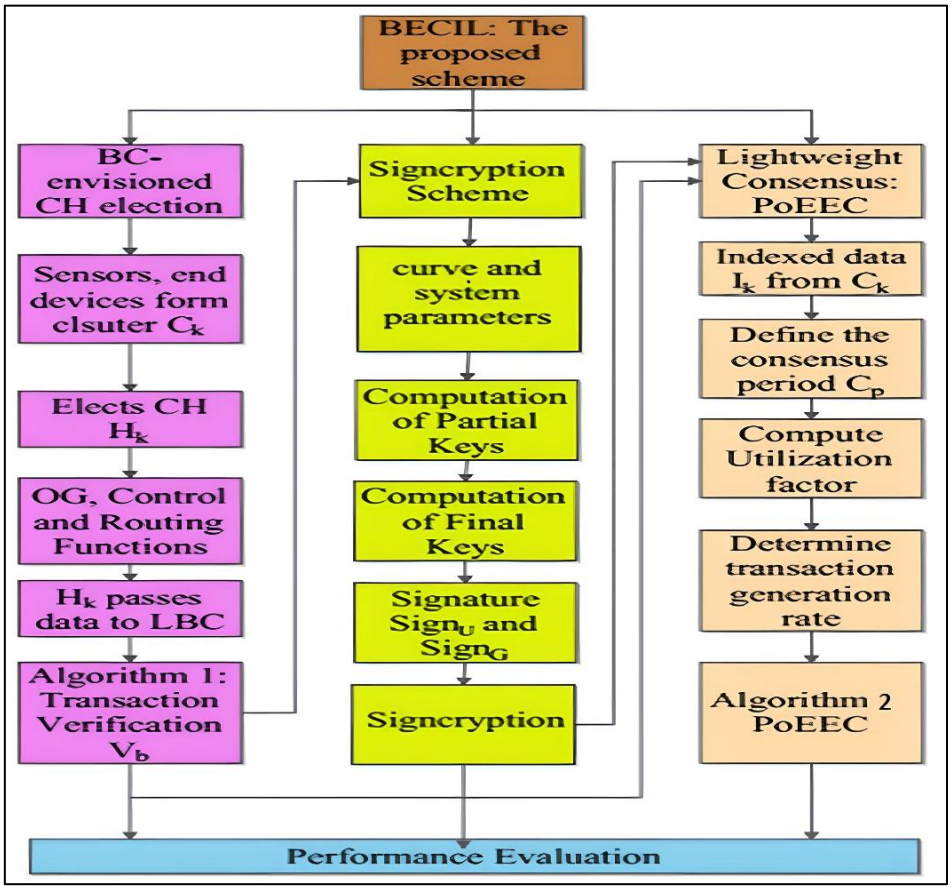

Figure 5.5: Flow of the Proposed Scheme

5.5.2 Problem Formulation

In the proposed scheme, in any k^{th} cluster C_k, a total of n MI users exists as $\{U_1, U_2, ...U_n\}$ where any n^{th} user has wallet attributes W_{Un}.

$$W_{U_n} = \{ID_{U_n}, (PU_n, PR_n), T_{U_n}, \delta, MR\}$$

where ID_{U_n} is the identity of n^{th} user in C_k, PU_n, PR_n are the public-private key pairs, T_{U_n} is the cryptocurrency in wallet W_{U_n}, and δ is the timestamp of the creation of W_{U_n}. Now, each n^{th} user U_n generates data D_n captured through m sensor nodes $S_m = \{S_1, S_2, ...S_m\}$ in C_k. For any n^{th} user, data of the sensor S_m is mapped to U_n is through a mapping function $M_1 : U_n \rightarrow S_m$ based on identifiers of U_n and S_m respectively. The mapped data is then forwarded to elected CH C_k that stores D_n based on key-value pair mapping $M_2 : C_k \rightarrow S_m$ and is defined as follows.

$$C_k = \{D_{S_1}, D_{S_2}, ..., D_{S_m}\} \tag{5.1}$$

where D_{S_m} denotes the data present at sensor node S_m. Now, any k^{th} CH C_k forwards the data to E_{OG}. At E_{OG}, Q forms multiple ledger blocks that can store $\alpha_{\max}$ transactions in a single block, B. The blocks contain the following information.

$$Q = \{B_{ID}, H_p, G_b, V_b\} \tag{5.2}$$

where B_{ID} denotes the local block identifier, H_p denotes the hash of the previous block to ensure chronology in the local chain structure. G_b denotes the generator blocks, and V_b denotes the verifier blocks. Based on appended blocks accessed via B_{ID}, G_b maintains an indexed key structure I_k that contains information as follows

$$I_k = \{B_{ID}, H_p, D_{S_m}, i\} \tag{5.3}$$

where i denotes the current timestamp. Successive entries $I = \{I_{k1}, I_{k2}, ...I_{km}\}$ are maintained to denote the generated information through G_b. Based on I_k, V_b verifies the transactions T based on hashes $H(T)$ to append to the current block C_b. The current

block hash H_c is verified to contain the block address of the previous block hash H_p on the same I_k. If the condition matches, V_b executes the signcryption scheme *S* to be added to transaction *T*, denoted as *S*(*T*). All the subsequent blocks are verified by V_b based on current hashes $H_c = \{B_1, B_2, ..., B_k\}$ to match addresses from previously linked blocks $H_P = \{B_1, B_2, ..., B_k\}$. If the match is correct, a Boolean variable *v* outputs *TRUE* to indicate successful verification by V_b, else V_b outputs *FALSE* as output. The details of the same are presented in Figure 5.6.

Algorithm 1: Transaction verification by V_b in Q

Input: Transaction set T.

Output: A Boolean variable $v = 0$, to indicate successful and failed block validations by v_b

1: **procedure** VALIDATION (T)
2: **for** $i \leftarrow 1$ to k **do**
3: **for** $j \leftarrow 1$ to n**do**
4: **for** $l \leftarrow 1$ to m **do**
5: $l_k \leftarrow \{B_{ID}, H_p, D_{S_m}, i\}$
6: $H_e \leftarrow Hash\_Append(H_p, I_k)$
7: **if** $(H(T, prev) \in H(T, curr))$ **then**
8: $V_b \leftarrow Append(T, B)$
9: **Call** SIGNCRYPTION$(G, N_G, k_v^*, L_U, ID(V_b), Y_{S_m})$
10: $v \leftarrow 1$
11: output "**Block verification SUCCESS"**
12: **else**
13: $V_b \leftarrow \text{Re } ject(T, B)$
14: BROADCAST(B)
15: $v \leftarrow 0$
16: output "**Block verification FAILED"**
17: **end if**
18: **end for**
19: **end for**
20: **end for**
21:**end procedure**

Figure 5.6: Transaction verification by V_b in Q

Subsequent to the verification process, signcryption (B. Zhang *et al.*, 2018) is performed, which has the following steps as depicted in Figure 5.6.

- In the first step, security parameter *p*, is taken as input. This outputs the master key *mk* that is kept secret. The parameter that is publically available is represented using *pm*.
- Next, is to bind the secret and public value of each user with the help of the corresponding user ID and publically available parameter *pm*.

  (y, *PubVal*) ← Bind-User-Key (pm, ID) (5.4)
- The third step involves to get the extraction of the partial private key, which is a very internal step with respect to the signcryption algorithm. This is to be sent securely.

  Par-pri ← Get-part-pri-key (*pm*, *mk*, ID, *PubVal*) (5.5)
- Via public parameters, *pm* and the partial private key *Par-pri*, and *y*, private key is generated as

  SecKey ← Get-pri-key (*pm*, *y*, *par-pri*) (5.6)
- With the help of the public parameter *pm*, the secret value *y*, *PubVal, and the* partial key is given as input to get the public key as output.

  PubKey ← Get-pub-key (*pm*, y, *Par-pri*, *PubVal*) (5.7)

Following the above steps to get the public and private keys, signcryption and unsigncryption can be performed [205].

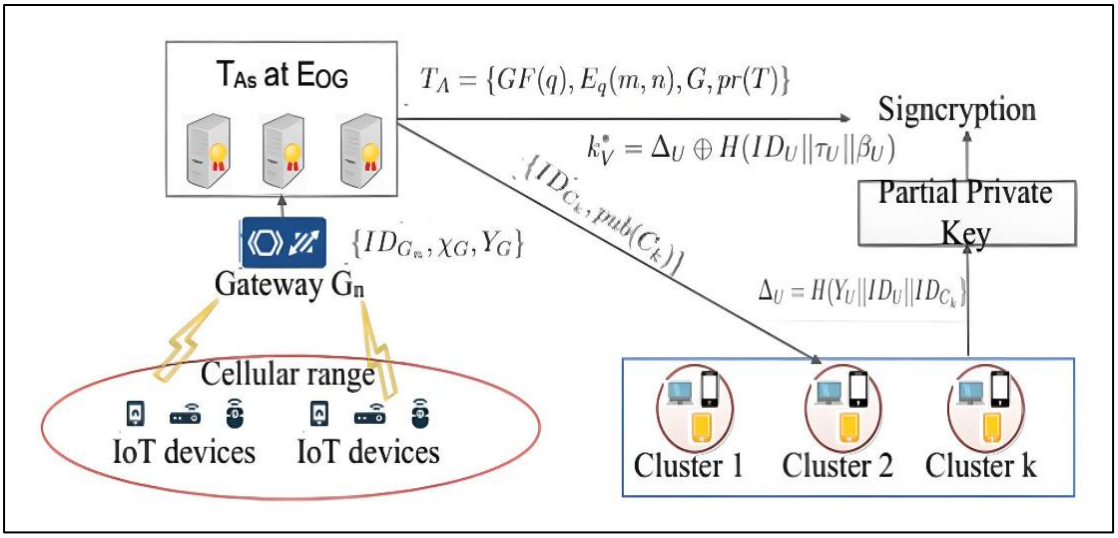

Figure 5.7: Signcryption Process in the Proposed Scheme

5.6 Proof-of-energy-efficient consensus mechanism: (*PoEEC)*

This section proposes a distributed consensus scheme *PoEEC* for encrypted data D_n signed through signcryption $Sign_U$ by U_n and $Sign_G$ at G_n, respectively. The scheme ensures block validations through V_b for *m* sensor nodes S_m installed at the user site U_n. The aggregated data D_{S_m} from C_k of *n* users $\{U_1, U_2, ...U_n\}$ are indexed as I_k. The scheme allows that forged block additions are not possible in chain *B*. To formulate the same, every transaction *T* is recorded through I_k in a trust management Table T_k in the cell range of C_k. Now, the verifier V_b checks the added transactions in T_k as follows.

$$T_k = \{sign_G, V_b, G_n\} \tag{5.8}$$

For aggregate verification, a consensus period C_p is used for transaction sets $\{T_1, T_2, ...T_p\}$. A utilization factor Φ is now computed as follows:

$$\Phi = \frac{T_n}{T_b} \tag{5.9}$$

where T_n denotes the new transactions added to C_p and T_b denotes the total number of transactions. *PoEEC* scheme is defined to keep Φ inside a window boundary $\{\Phi_{\min}, \Phi_{\max}\}$. To ensure the same, *m* sensor nodes that pass data through E_{OG} are presented to miners E_M. The miners form a miner pool T_{pool}, where new transactions are appended. Now, based on the length of T_{pool}, denoted as $|T_{pool}|$, user U_n passes data through $_{EOG}$ to add to local BC B. The rate of transaction generation is denoted as Ω, which is denoted as follows.

$$\Omega = \Phi.\delta_m.C_p / T_b.G_n \tag{5.10}$$

The utilization of Ω can be maximized by the following considerations.

- Changing Φ and frequency f of block appended
- Changing G_n

Thus, through careful observation of Φ during C_p, frequency f can be optimized to

increase the transactional throughput of the block appended. In second consideration, the placement of G_n is crucial to address the bulk transactions that are aggregated through *m* sensor nodes. The details of the proposed algorithm are presented in Figure 5.8.

Algorithm 3: Proposed PoEEC scheme for added transaction T by E_m

Input: $\Omega, \Phi, \Phi_{\max}, \Phi_{\min}$

Output: Mined transactions T added to B by E_m satisfying constraint $\Phi_{\min} < \Phi < \Phi_{\max}$.

1: **procedure** PoEEC($\Omega, \Phi, \Phi_{\max}, \Phi_{\min}$)
2: **while** True **do**
3: **for** $j \leftarrow 1$ to n**do**
4: **if** $(\Phi > \Phi_{\min})$ **then**
5: $\Phi_{av} = \Phi_{\min} + \Phi_{\max} / 2$
6: **If**$\left(T_N < \Phi_{av}\right)$ **then**
7: **If**$\left(C_p \le T_B\right)$**then**
8: Update C_p to Φ_{av}
9: Broadcast C_p to E_m
10: $T_{pool} \leftarrow T_{pool} + T_{av}$
11: Add T_{pool} to B
12: **else**
13: Update C_p to $\Phi_{\min}$
14: Broadcast C_p to E_m
15: $T_{pool} \leftarrow T_{pool} + T_{av}$
16: Add T_{pool} to B
17: **end if**
18: **else**
19: Change frequency f to map C_p
20: **end if**
21: **else**
22: Update C_p to $\Phi_{\max}$
23: **If**$\left(\left|T_{pool}\right|\right) \leftarrow T_{\max}$ **then**
24: Reset Φ to $\Phi_{\min}$
25: GOTO Line 4

26: **else**
27: Wait for T_N to be serviced in T_{pool}
28: **end if**
29: Broadcast C_p to E_m
30: Add T_{pool} to B
31: **end if**
32: **end for**
33: **end while**
34:**end procedure**

Figure 5.8: Proposed PoEEC scheme for added transaction T by E_m

5.7 Performance Evaluation of Proposed Scheme

In this section, the performance of the scheme is evaluated in terms of simulation and security evaluations.

5.7.1 Simulation Results

In this section, the simulation results of the proposed scheme are presented. Performance is evaluated against parameters chosen on the basis of node commit latency against Singh *et al.* [206].

Another observation is performed in terms of signing delay, which is compared with Zhang *et al.* [75] and Dorri *et al.* [74]. Next, the storage cost of the proposed scheme is compared against Dorri *et al.*[74]. The scheme is also compared for energy consumption in block additions against traditional consensus schemes such as PoW, PoS, and PoET. For node throughput, efficiency is measured against the baseline approach in [46].

5.7.2 Simulation setup

The setup deploys Network Simulation Toolkit for the experimentation purpose. The path libraries are set on a virtual machine (VM) running on Ubuntu Linux *v18.04* LTS with two virtual CPU cores. The internal memory is 4 GB RAM with a 30 GB external hard drive. The testing is performed on Node.js *v8.9.1* with npm *v6.7.0.*

5.8 Simulation Results

Simulation results based on the proposed algorithm are discussed in this section, which includes the benefits of local verification by V_b, the signcryption scheme, and the *PoEEC* consensus scheme. The details are presented as follows.

iii) Benefits of local BC client at E_{OG}

To measure the performance of mining efficiency, the sensor data D_{Sm} is forwarded through C_k to E_{OG}. At E_{OG}, a multi-ledger block is created that improves the mining cost, where it can store αmax transactions in a single block B.

The benefits of integrating a local verifier V_b that verifies the transaction based on the indexed set I_k, are depicted in algorithm 1. The parameter is defined as node commit and measures the impact of node commit latency. The proposed scheme is compared against Singh *et al.* [206], which uses Proof-of-Stake (PoS). In PoS, a miner difficulty F is used to finalize blocks, and that depends on the particular stake of a miner node. Figure 5.9 presents the impact of processed blocks on the commit latency (node finality) for the proposed scheme.

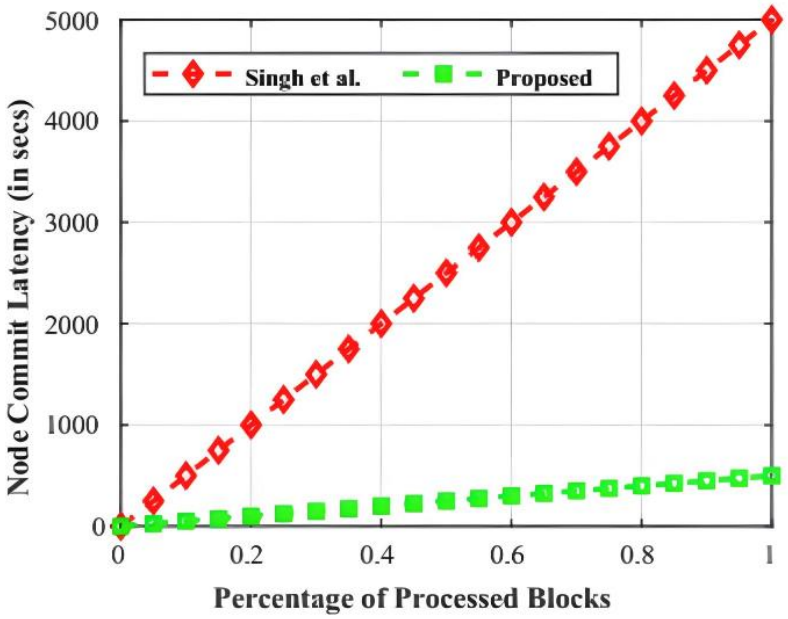

Figure 5.9: Benefits of Local BC Verification in Terms of Observed Node Commit Latency for Verified Blocks

ii) Benefits of the signcryption scheme

Next, signcryption scheme is discussed that sets up T_A and selects the public-private key pairs. Based on user U registration, two signatures $Sign_G$, and $Sign_U$ are proposed for G_n and U_n, respectively. Figure 5.10 (a) depicts the efficiency of the signcryption schemes to generate signatures $Sign_G$ and $Sign_U$, respectively. As evident from the Figure 5.10, at n = 200 transactions, the proposed scheme has a signing delay of 1397 ms, compared with Zhang *et al.* [75] with a signing delay of 1489 ms, and Dorri *et al.* [74].with a delay of 1612 ms. As signature operations are validated through secret key η_k and master key MK at E_{OG}, partial signatures, S_1 and S_2 are computed faster and are then combined with secret functions γ_1 and γ_2, respectively. Next, based on the stored blocks, the impact on the storage cost is measured. Figure 5.10 (b) presents the details. In the proposed scheme, C_k allows local miners L_m to process local transactions in blocks, which are later assigned a B_{ID}. Generator blocks G_b maintains I_k, which verifies transactions based on hashes $H(T)$. Once blocks are locally verified, only the transactional meta-information is required to be stored in the global BC. This effectively reduces the transaction size, and thus more transactions are added in each block. This improves the storage cost of the on-chain structure, which is evident in the results. For 20 blocks appended, the storage is 2.6 KB, compared to 3.7 KB for Dorri *et al.* [74]. On an average, an improvement of 23.78 % in storage cost is observed in the scheme.

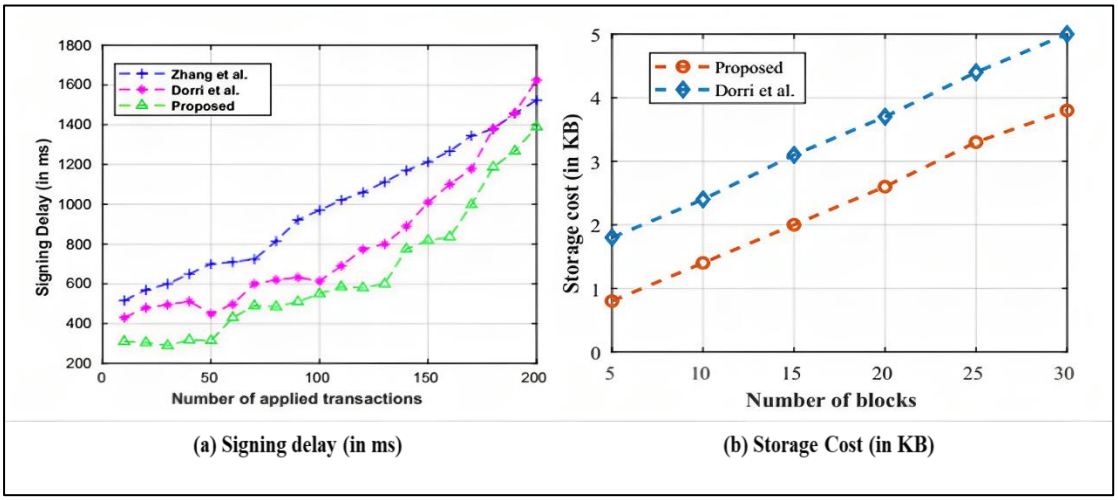

Figure 5.10: A Comparative Analysis of the Signcryption Scheme Against Traditional Signing Schemes in Terms of Signing Delay and Storage Costs

iii) *Benefits of the PoEEC scheme*

Next, the *PoEEC* scheme is presented that defines a consensus period *Cp* for transaction sets {C_1, C_2,..., *Cp*}. The scheme is compared for energy consumption against traditional consensus schemes like PoW, PoS, and PoET. For node throughput, the efficiency against the measured baseline approach in [46] is measured. As evident in Figure 5.11 (a), the proposed *PoEEC* scheme has a low consumption of 34.56 kJ at n = 80 block additions, compared to 76.12 kJ in PoET, 122.34 kJ in PoS, and 148.54 kJ in PoW respectively. The average energy consumption is 23.45 kJ for 100 transactions. As D_{Sm} is passed to E_M, the miners add only those transactions to T_{pool} that satisfy the constraints of window boundary $\{\phi_{min}, \phi_{max}\}$. Thus, only new transactions are broadcast through $C_{p,}$ which results in improved energy consumption. Figure 5.11 (b) measures the node throughput. As evident, at 450 appended transactions to T_{pool}, the proposed scheme has a utilization of 1792 kbps, compared to 1423 kbps. As ϕ is maintained in the window boundary, new transactions are added to T_{pool}, and that decides the rate Ω of transaction generation. As Ω depends on consensus period C_p, and T_b, the utilization can be tweaked based on frequency f of block appends, or secondly, by passing more data of m sensor nodes through G_n.

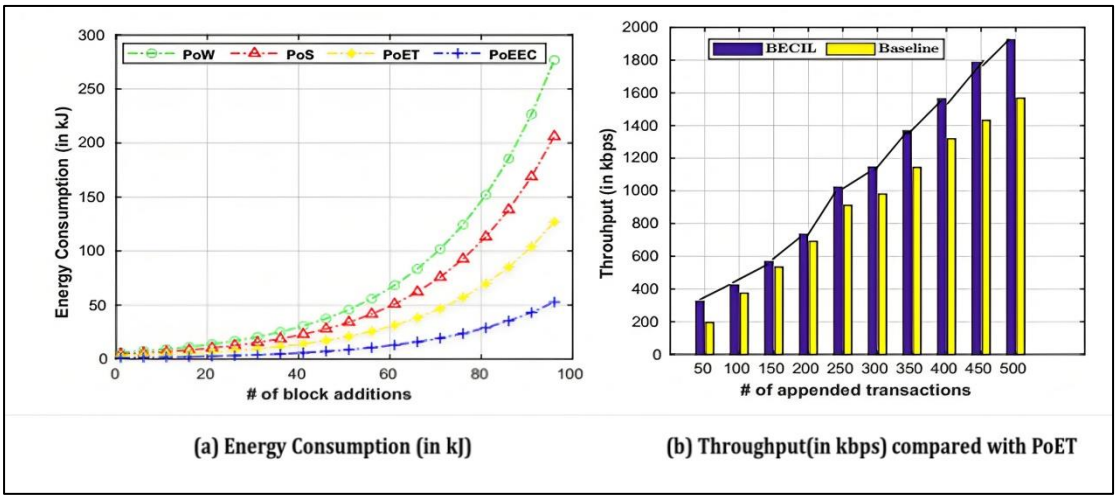

Figure 5.11: Comparative Results for PoEEC Consensus

5.9 Summary

This chapter proposes a scheme *that* integrates BC and lightweight encryption and a signing scheme for constrained IoT environments. The proposed scheme addresses the gaps in earlier studies to address the security trade-off and computational mining through a dual mechanism of secure signcryption scheme and low-powered consensus. In the first phase, the secure transfer of sensor data is proposed signing mechanism for gateway and user nodes. To exploit the same, a local CH in each cell allows aggregated data to GN. GN adds the data to a local blockchain and is based on transaction verification. In the second phase, the collected data is mined by miners through an energy-efficient *EEPoC* scheme that increases the scalability of mined transactions in BC at low computational and storage overhead.

CHAPTER 6

DESIGNING AN OPTIMAL ECC ALGORITHM WITH AN EFFECTIVE ACCESS CONTROL MECHANISM FOR BIG DATA STORAGE

In recent years, one of the most discussed and well-known technological terms is big data. It is defined as the collection of huge data sets, and its primary characteristics are volume, variety, and velocity [207]. Nowadays, the usage of big data is inevitable in various fields such as healthcare, education, natural resources, social networking, etc. Some of the benefits of big data towards organizations are considered are: improvement of operations, adaptation to the new environment, and optimization of available resources [208], [209]. In general, the cloud is believed as a promising platform for storing and processing big data.

Moreover, cloud computing provides an immense resource pool and enables unlimited virtual storage space by connecting network resources [210]. The most desirable properties of the cloud are scalability, rapid elasticity, fault tolerance, resource pooling, and pay-per-use [211]. In addition, the essential part of cloud service is to store the data. The users are primarily concerned with the confidentiality of data, its integrity, security issues, and privacy of open networks, including cloud computing [212]. Yet some users and companies hesitate to use storage services of cloud networks due to security and privacy concerns [213]. In cloud computing, some issues are considered as the most significant challenges, which can degrade the trustworthiness of the users. First, users are mainly scared about the risk of leakage of data and unauthorized access to stored data on the cloud. Second, some attacks and intrusions toward data are the biggest challenges for cloud service providers (CSP). Next, cloud users have trust issues with the cloud service provider for managing their data in terms of data storage, backup, migration, deletion, updating, and access in the cloud. Further, cloud users always remain in fear that the computation of data could disclose their privacy or related entities to unauthorized parties [214]. To combat with the problem of big data security and privacy, cloud computing employs a variety of

security strategies. It includes encryption, authentication, access control, firewalls, data leak prevention systems (DLPSs) and intrusion detection systems (IDSs) [215].

Cryptography is a widely used technique that helps in hiding sensitive or confidential data from unauthenticated users. In order to maintain data confidentiality and security, highly trusted outsourced encrypted data has been used in the cloud, which makes it difficult to perform any auditing in data management and greatly reduces the risk of privacy leakage [214]. Some of the techniques which are widely used for cryptography include Data Encryption Standard (DES) algorithm, Advanced Encryption Standard (AES) algorithm, RSA algorithm, Diffie-Hellman Key Exchange etc. Both DES and AES algorithms are symmetric-key algorithms, and they use the same key for both encryption and decryption. On the contrary, RSA and Diffie-Hellman Key Exchange are the most common asymmetric-key algorithms. This method belongs to the class of public key schemes, which use different keys for encryption and decryption in the cloud [216]. Apart from these security suggestions, the security of the IT infrastructure [217] is also critical [209]. Monitoring, evaluating, and learning from data usage is also seen as an important part of developing security services for data storage infrastructure [218].

6.1 Literature Review

Bruno Guazzelli Batista *et al.* [219] have proposed a strategy for deciding the QoS-driven methodologies in the cloud environments based on the different security mechanisms. Based on this, certain performance evaluation has been carried out that resulted in additional computational overhead. To overcome these additional overheads, computational resources are allocated on the basis of requirements. By estimating the results, it is quite evident that it is possible to manage the performance, even if there is some overhead due to security parameters.

Laurence T. Yang *et al.* [220] proposed an algorithm that manages resource allocation over the cloud. It works on the theory of general number field sieve (GNFS). They focused on solving the large and sparse linear systems over GF (2), which are

assumed to be the most critical steps of the GNFS algorithm. The crucial step which was introduced is the novel parallel block Wiedemann algorithm. It includes partitioning strips and cyclic, which are used to parallelize the algorithm, thus improving the efficiency of the system. By doing this way, GNFS outperforms the existing techniques.

Muthu Ramachandran [221] has given concise strategies, procedures, and the best preliminary design for requirement engineering in terms of managing critical resources and service requirements in cloud infrastructure. They give certain insight regarding requirements engineering, i.e., the requirement is gathered and provided the services accordingly. They also provided some roadmaps on software security as a service. Additionally, the Integrated-Secure SDLC model (IS-SDLC) is also discussed, which can explain the large cloud system Amazon EC2 service.

Power consumption in rendering cloud services is always a matter of concern. Dinh-Mao Bui *et al.* [222] recommend an energy-efficient answer for arranging the resources in cloud computing. Most concerns are related to energy-efficient solutions for managing the cloud environment's resources. The proposed methodology was based on the Gaussian procedure regression strategy for foreseeing resource usage. They have applied a convex optimization system for computing a proper amount of physical servers for each controlling window. They needed to guarantee that a low number of servers were required to render the nature of services that are matching with the goal. In their final step, the servers are turned off and kept idle for the defined time. The method was also tested using the model data taken from the Google servers. The approach was observed to reduce energy consumption significantly but without any major access control mechanisms.

Hu Xiong *et al.* [223] have proposed a scheme that is identity-based and offers end-to-end security between the analytical system and data collectors. The secure big data interactions between the data collectors and analytical data systems were performed using an identity-based (ID-based) signcryption technique with quick revocation and the ability to outsource unsigncryption. All the major security requirements, such as

confidentiality, authentication, integrity and non-repudiation, were achieved by the scheme. Unsigncryption, which is also a major step in the proposed work, is outsourced to untrusted cloud servers, which may hamper security concerns.

Jian Shen *et al.* [224] have proposed a third-party-helped scheme that uses cloud computing as a technology that is searchable and verifies data. For a better understanding, the data storage is done using cube data storage. Further, a user-differentiated system is used. Based on the framework model and information structure, the plan enables the clients to survey the trustworthiness of their transferred or downloaded information and look through the content that is available online with encrypted keywords.

Gang Chen *et al.* [225] have demonstrated the development of various tools such as MapReduce. These tools are progressively developed to cater to practical industry needs, such as the variety of data being added as content. They used the information security framework Netease, for instance, to show the development. The aim was to develop a system that is robust and works in real-time scenarios for Net ease too. It can also be used for tasks such as user behaviour mining, spam detection, and game log analysis.

With the consideration of privacy-preserving arrangement, Kan Yang *et al.* [226] suggested a practical and fine-grained (big data) access control mechanism. In particular, they conceal the entire attribute instead of the key values in the access management. A novel Attribute Bloom filter is designed, which would help filter out the user according to the access policy and locate the exact point of access policy alteration. This helps in correct decryption according to the access policy.

Yang *et al.* [227] have proposed a novel algorithm for a healthcare system that stores big data. The core is to improve the patient's security, for example, healthcare data, acknowledge access control mechanism for personal data and bolster smart deduplication, and supplements the storage space in big data storage system. A safe deduplication strategy helps with restorative records with indistinguishable data, which might be scrambled with different access arrangements. Furthermore, that

method was utilized to spare the storage overhead in the big data storage system. The principal objective of those strategies was to check records after the deduplication. It can still be accessible with approved data administrators at different levels, with varied access control methods. That smart healthcare big data storage system was formally demonstrated.

Fugkeaw *et al.* [228] have presented a method for policy updates, which is very secure and can be used for lightweight applications. Their proposed access control mechanism demonstrates secure policy refreshes in a big data setup. The significant point of their technique was to reduce the re-encryption cost, which was a successful policy updating strategy. Their trial result demonstrates that their approach presents an effective policy updating plan with less computation cost. Amid the few occasions of policy updating, a large number of requests need to be catered to.

Wei Zhou *et al.* [229] proposed a model to improve the storage requirement of data in a hybrid kind of environment. A hybrid environment deals with the storage of heterogeneous data and also the real-time processing of data. The major concern is to handle the asymmetric characteristics of data, which occurs due to the varied source and different source devices. This study provides a model which addresses storage performance imbalance when data is distributed through various devices to quantitatively weight and then distribute the data to the storage device with the best equivalent data access features. The implemented Preference-Aware HDFS (PAHDFS) demonstrates outstanding performance, efficiency, and scalability.

Kavitha *et al.* [230] proposed a secure healthcare private data cloud that fools the attackers by providing fake and concealing the original data. Security of the cloud is achieved by sharing keys. i.e., using the bilinear map that is very secure, and users can communicate securely by generating session keys. The decoy technique is used for the implementation process, which provides additional security. Further, the frequent advancement in the area of Big data, access policy management, and the IoT has widened the horizon of challenges for securing data which may be of any form and

also can be decrypted and accessed by the authorized user only [231], [232], [233], [234], [235], [236].

6.2 Problem Formulation

The proliferation of Big data leads to specific problems such as data hugeness, data security, and many more. But in today's era, there is an urgent need to handle the issue because of gaining multimedia usage and related application. Data generated by various internet devices is so massive that it must be stored on the cloud, but that leads to privacy and security concerns over the cloud. The limitations of existing big data access control schemes are as follows.

The limitations of existing big data access control schemes are listed below:

- Existing big data processing methods face many scalability and efficiency issues mainly due to the increased volume of data. The increased volume of data occupies more space for storing the data in the cloud, and hence they are difficult to process.
- Traditional encryption schemes have many flexibility issues, and hence they cannot provide complete security to the data.
- Since the volume of data is more, it is impossible to identify the sensitive part of the information.
- Although ABE-based approaches allow the data owner to predefine their eligible users for data access, users had to suffer greatly because of the difficulty of modifying access policies and in the encrypted text [211].
- To avoid the risk of privacy leakage from access policy, there are measures that can hide the attributes. When these attributes are hidden, unauthorized as well as authorized users would not be able to recognize them and hence cause a problem in decryption.

These are the few gaps in previous studies that drive us to conduct this research on big data access control techniques.

6.3 Proposed Method

Big data is often identified by 3Vs: the volume of data created, the variety of data coming from different sources, and the speed and time (velocity) associated with the data. Such massive amounts of data might come from a variety of sources, including corporate sales records, scientific experiment results, and real-time sensors. There is a possibility of many security issues in the big data stored in the cloud. The existing access control technique is weak and hence leads to different attacks like collusion attacks, denial of service attacks, and insider attacks on the data stored in the cloud. Hence, , a method is proposed to get rid of all these issues and preserve the data from attacks. Here, the input is, KDD data set is due to easy availability. Initially, the input data is fed into the MapReduce framework, where it is clustered into groups by means of a clustering algorithm. For clustering, the suggested technique utilizes a fuzzy c means algorithm (FCM). Further, clustered data of every mapper is fed as input to the reducer. Next, the reducer is utilized to reduce the number of clusters produced by the mapper without any data loss. Finally, the reduced output is sent to the data owner for further processing. The proposed method offers an optimal cryptographic solution for a secure and verifiable access control scheme, which can also be applicable in big data storage. The main stages of the proposed method are Setup and encryption stage, Data storage construction stage, Data reconstruction stage, and Access policy update stage. The proposed method utilizes optimal elliptic curve cryptography (OECC) for security. Here, the key parameter is selected using the modified grasshopper optimization algorithm (MGOA). Finally, the secured data is stored in the cloud along with the access policy of the data owner. The overall proposed system model is shown in Figure 6.1.

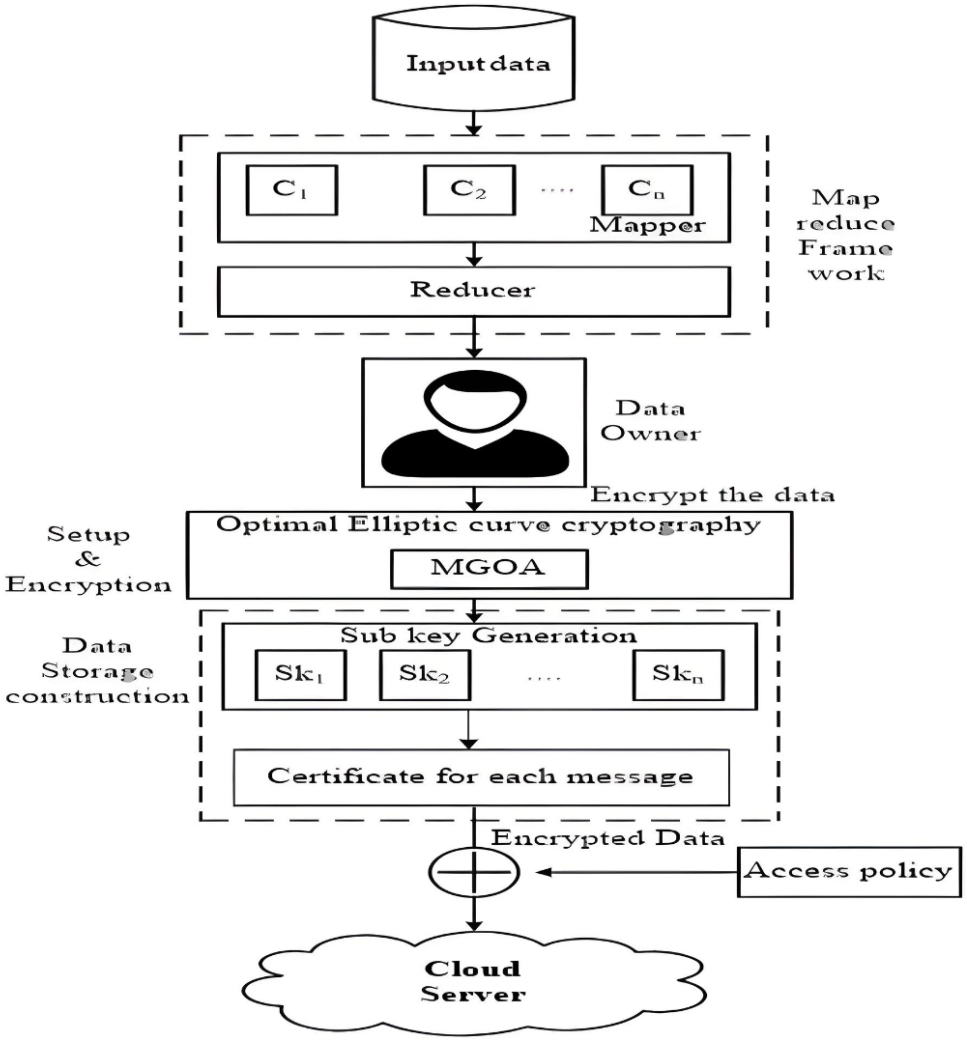

Figure 6.1: The Proposed System Model

6.3.1 MapReduce Framework

Mapreduce is an architecture for analyzing and processing large amounts of data on a network of servers. It is a parallel and distributed large-scale data processing paradigm that has received a lot of attention and is frequently used in big data applications. MapReduce is powerful, flexible, and cost-effective and has salient applications in cloud computing. The MapReduce framework consists of the mapper step and reducer step. The step-by-step flow of the MapReduce framework is depicted in Figure 6.2,

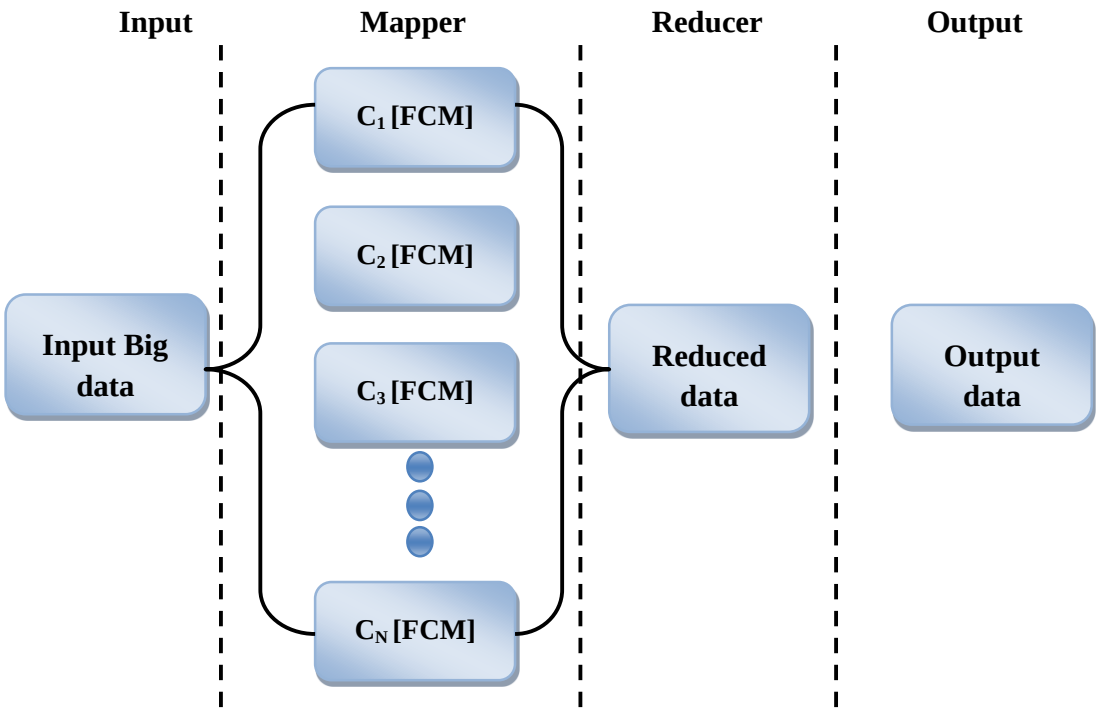

Figure 6.2: The Proposed MapReduce Framework

Figure 6.2 illustrates the general arrangement of the proposed MapReduce structure; there are four segments in MapReduce, i.e., Input (big data), Mapper, Reducer, and Output (collect all the reduced output). MapReduce framework is usually explored to process big files.

- ***Input Segment:*** Here, the Knowledge Discovery and Data Mining (KDD) dataset is considered as the input for the experimentation.
- ***Map Segment:*** In this segment, the proposed method groups the available data using fuzzy c means clustering algorithm into M partitions. The basic idea behind clustering is to divide data/object based on the highest similarity between objects in the same cluster and the minimum similarity between objects in different clusters. All mappers run in parallel to attain better efficiency.
- ***Reduce Segment:*** In this segment, the data from each of the mapper is condensed and reduced. The MapReduce framework will not allow duplicate data, and the resultant data will contain only consistent data values.

- ***Output segment:*** Finally, reduced data is collected from the MapReduce framework, and then the reduced output is sent to the data owner.

On the data owner side, the main stage of the proposed method is the Setup and encryption stage, Data storage construction stage, Data reconstruction stage, and Access policy update stage. The detailed process of every step is illustrated in subsequent sections.

6.3.2 Setup and encryption stage

In the setup and encryption stage, the data owner initializes the system parameters to generate the public and private keys for encryption and decryption, respectively. The proposed method utilizes the optimal elliptic curve cryptography (OECC) method for encrypting the data. Here the key parameter is optimally selected by means of a modified grasshopper optimization algorithm (MGOA). The procedure of optimal elliptic curve cryptography (OECC) is described in the following section.

i) Optimal Elliptic Curve Cryptography (OECC)

The suggested technique uses the MGOA to select key values that are optimal. The ECC method generates private and public keys; these keys are responsible for data confidentiality. The elliptic curve's general equation is given below:

$$y^2 = x^3 + ax + b \tag{6.1}$$

To create both a public key and a private key, the proposed method needs a key generation part. To encrypt the message, the owner uses receiver's public key, and finally, the receiver will decrypt the encrypted data using its private key. The key generation part is clarified underneath.

ii) Key Generation by ECC

In the key generation process, the operation [237] is explained over two fields: prime and binary fields. The appropriate field is chosen with a finite large number of points for cryptographic operations. The prime field operations select a prime integer, resulting in many fundamental points on the elliptic curve.

$$PU_{key} = a * P_C \tag{6.2}$$

where,

a- Random value

P_C- Generator Point of the curve

PU_{key} - Public key

Here, the proposed method optimally selects the "a" values based on the optimization technique. For optimization, the MGOA is used to determine the random value of a.

6.3.3 Modified Grasshopper Optimization Algorithm (MGOA)

In the suggested work, the MGOA is used to select the random values. The Grasshopper Optimization algorithm (GOA) [237] is a nature-inspired swarm-based algorithm that mathematically models and simulates the behaviour of grasshoppers in nature. The grasshopper swarm has one distinguishing feature. It is a swarm-based nature-inspired algorithm that mimics and mathematically models the behaviour of a grasshopper swarm in nature. The grasshopper swarm possesses a unique feature: the swarming behaviour is found in both the nymph and adulthood grasshoppers. The major difference between the nymph and the adult grasshopper is the distance they can travel. When they become adults from nymphs, they form a swarm in the air and cover a considerable distance. In the proposed work, GOA is modified by means of oppositional-based learning strategy (OBL), which increases the searching ability. In the proposed method, a grasshopper position represents the random values which are basically prime numbers. The objective function is defined as the key breaking time. The step-by-step procedure of MGOA is described below. The following steps are used to select the optimal key value. The mathematical model presented that is used to replicate the MGOA swarming behaviour is as follows:

Step 1: Solution representation

To optimize the key value, MGOA initially creates an arbitrary population of solutions. Solution creation is an important step of an optimization algorithm that

helps to identify the optimal solution quickly. In MGOA, possible key values are selected randomly, and that is the position of grasshoppers. Each swarm member is represented as a grasshopper in a D-dimensional search space.

Step 2: Opposite solution generation

After the initial solution generation, the method creates an opposite solution using (6.3). Here, every solution P_{ij} has a unique opposite P'_{ij} solution. The opposite solution $OP(P'_{11}, P'_{12}, ..., P'_{1n})$ is calculated based on the equation.

$$P'_{ij} = L_{W_i} + U_{P_i} - P_{ij} \tag{6.3}$$

where, L_{W_i} represents the lower bound coefficient, U_{P_i} represents the upper bound coefficient, P_{ij} represents the old solution. Then, the method combines the initial and opposite solution for further processing.

Step 3: Fitness calculation

In this step, the fitness of the initial solution (X_{ij}) is evaluated. Fitness is checked for optimality among the possible solutions. The essential prerequisite for designing a fitness function is key breaking time·

Step 4: Updation using grasshopper optimization

After this, the grasshopper optimization algorithm is used to update the solution. The updated equation is as follows:

$$P_i = S_i + G_i + W_i \tag{6.4}$$

where P_i represents the position of the i^{th} grasshopper, S_i is the social interaction, G_i is the gravity force on the i^{th} grasshopper, and W_i shows the wind advection.

$$S_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} s\left(d_{ij}\right)\hat{d}_{ij} \tag{6.5}$$

$$d_{ij} = \left|P_j - P_i\right| \tag{6.6}$$

$$\hat{d}_{ij} = \frac{P_j - P_i}{d_{ij}} \tag{6.7}$$

where d_{ij} is the distance calculated within the i^{th} and the j^{th} grasshopper. s represents the social force, and the force of gravitation (G_i) is computed as:

$$G_i = -g\hat{e}_g \tag{6.8}$$

where g represents the gravitational constant and $\hat{e}_g$ is a unity vector, which is directed towards the earth's centre, the wind force, i.e., W_i is presented using (6.9).

$$W_i = u\hat{e}_w \tag{6.9}$$

where u is the constant drift and $\hat{e}_w$ is a unity vector in the wind direction. Substituting values S, G, and W in (6.4) results in:

$$P_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} s(|P_j - P_i|)\frac{P_j - P_i}{d_{ij}} - g\hat{e}_g + u\hat{e}_w \tag{6.10}$$

Where N is the number of grasshoppers. Using (6.10), the proposed method can update the solution.

Using the above equation, grasshoppers quickly reach the comfort zone, and the swarm does not converge to a specified point. In order to overcome this issue, a modified version of this equation is described below,

$$P_i = C\left(\sum_{\substack{j=1 \\ j \neq i}}^{N} C\frac{U_P - L_W}{2} s(|P_j - P_i|)\frac{P_j - P_i}{d_{ij}}\right) + T \tag{6.11}$$

where T represents the target or best solution. C is a decreasing coefficient to shrink the comfort zone, repulsion zone, and attraction zone. The coefficient C reduces the comfort zone depending upon the number of iterations and is calculated as follows,

$$C = C_{\max} - t\frac{C_{\max} - C_{\min}}{t_{\max}} \tag{6.12}$$

where the highest value is $C_{\max}$, and the lowest is $C_{\min}$. Here, t denotes the current iteration and $t_{\max}$ denotes the maximum iteration.

Step 5: Termination criteria

The algorithm comes to a halt as the number of iterations has reached to the maximum, and the solution having the best fitness value is selected as the optimal value

Based on the obtained optimal prime value, the public pu_{ky} and private keys pv_{ky} are defined as:

$$pu_{ky} = a * P_C \tag{6.13}$$

$$pv_{ky} = a \tag{6.14}$$

The process of encryption and decryption is illustrated below:

6.3.4 Encryption and Decryption

In the optimal ECC algorithm, the encryption process encrypts the input message, and the output is represented as two ciphertexts as, T_1 and T_2.

$$T_1 = k * P_C \tag{6.15}$$

$$T_2 = Data + k * pu_{ky} \tag{6.16}$$

where, k symbolizes the random value 1 to *(n-1)*. P_C is the generator point on the elliptic curve. pu_{ky} is the public key. These encrypted message T_1 and T_2, is sent to the receiver. After receiving the ciphertext, the receiver decrypts the data using the following equations,

$$Data = T_2 - pv_{ky} * T_1 \tag{6.17}$$

Based on the above process, the input data is encrypted on the data owner side. After encrypting the input data, the resultant output is given to the data storage construction stage. The data storage construction stage is explained below.

6.3.5 Data Storage Construction Stage

In the data storage construction stage (figure 6.3), the data owner generates the subkey for each legitimate user who needs to use the data. Then, a certificate for each message in a set of messages is generated, and the encrypted data is stored on the cloud server along with the access policy. The process of subkey generation and certificate generation is described below:

i) ***Subkey generation***

Here, each legitimate user selects a random number and encrypts the random number with the data owner's public key, and generates the ciphertext. The user then sends the user *id* (Id_i), a hash function *(H)*, and ciphertext to the data owner. After receiving the ciphertext, the data owner checks the two conditions: the first condition is to compare the new hash value with the old hash value for the corresponding user. The second condition is to compare the random value of the corresponding user with the other user value. If two conditions are verified, data owner generates the subkey for the corresponding user. Then, the data owner securely broadcasts the Id_i and subkey to all the users.

ii) ***Certificate generation***

To validate user's legitimacy for accessing the data and to prevent cheating behaviours of the user, the proposed work generates certificate for each message. Finally, the encrypted data, certificate message, and the access policy are sent to the cloud server.

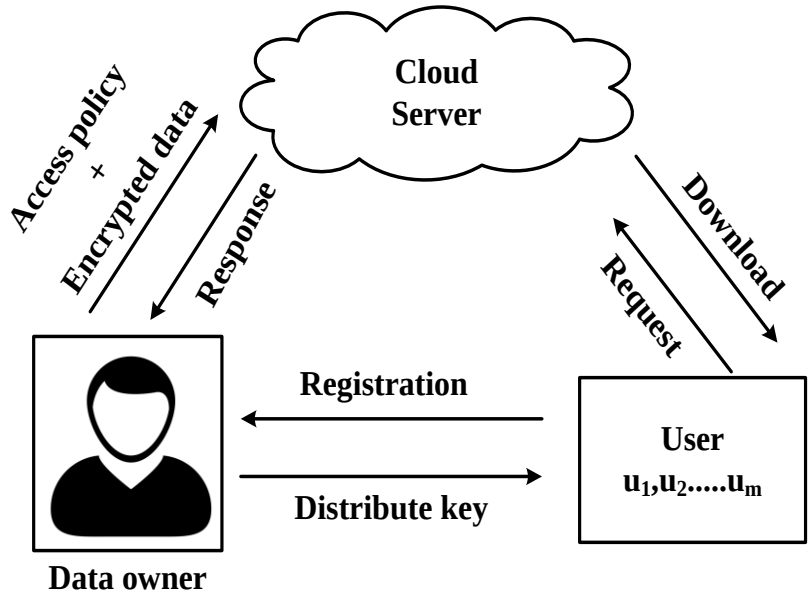

Figure 6.3: The Data Storage Construction Model

6.3.6 Data Reconstruction Stage

In the data reconstruction stage (figure 6.3), if any user needs to get the data from the cloud server, first, the user downloads the data and asks for help from other legitimate user to decrypt the data. Then, user sends the request to the data owner to obtain the message certificate. After receiving the request, the data owner encrypts certificate using the user's secret number. Finally, the ciphertext is sent to the user. Then, the user uses the subkey to compute the exchange certificate and sends it to the other legitimate user. After that, the other user verifies the certificate. The successful verification confirms the corresponding user as a valid user of the message. Once the verification is completed, the other user sends the Id_i and the random secret number to the corresponding user. After receiving the Id_i and secret number from all the users, the corresponding user verifies the Id_i. If the secret number does not pass the verification, the subkey value cannot be used by the corresponding user. When the users obtain authorization from the legal user, the corresponding user recovers the original data.

6.3.7 Policy Update Stage

The figure 6.3 demostrate the final stage is the access policy update stage, where the cloud server can update the ciphertext when the data owner specifies a new access policy, and this update can be validated by the data owner; in the meantime, each user can easily update their secret number and sub-key. The performance of the proposed method is analyzed, and the effectiveness of the proposed method is compared with other methods in the result and discussion section.

6.4 Experimental Result and Analysis

In this section, the experimental results obtained for the proposed method using MapReduce Framework in the cloud environment with the KDD dataset are discussed. The experimentation is conducted using CloudSim. The experimentation is done using Java (JDK 1.8) on a PC with a 2 GHz dual-core and 4GB of main memory on the Windows operating system.

i) Observations on varying the data size

This section presents the experimental analysis of the proposed scheme is given below in this section. Here Table 6.1 shows the performance measure of the recommended method. It shows the execution time and memory by varying the data size. For the information measurement of 10,000, the execution time found by the procedure is 113,485 ms. Similarly, other observations are taken by varying data sizes of 20000, 30000, and 40000, respectively.

Table 6.1: Experimental analysis of time by varying data size

Data size	Execution Time (ms)
10000	113485
20000	156984
30000	214785
40000	286545

ii) Observations by varying the number of mappers

Here, Figure 6.4 represents the execution time of the proposed method by varying the number of mappers. For the number of mappers to be 5, the execution time is 156,984 ms. Similarly, observations are taken for 10, 15, and 20 mappers, respectively.

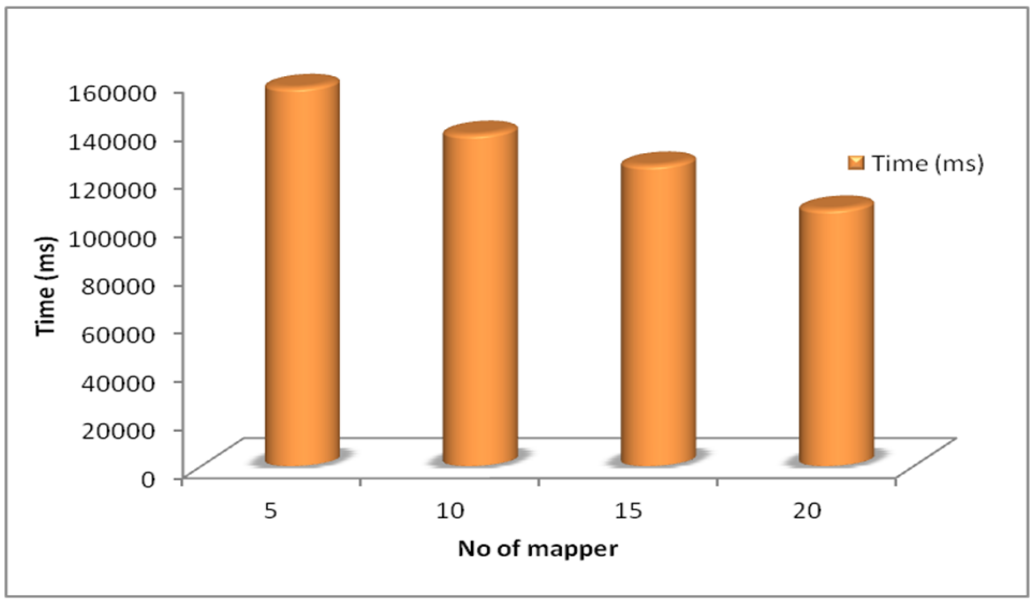

Figure 6.4: Performance Analysis of Execution Time

iii) Observations by Varying the Cluster Size

Table 6.2 represents the results obtained by varying the execution time against cluster size. For the cluster size of 4, the execution time obtained by the proposed method is 106545ms. Similarly, observations are taken for 6, 8, and 10 clusters, respectively.

Table 6.2: Experimental analysis of execution time with differentiating cluster size

Cluster size	Execution Time (ms)
4	106545
6	124785
8	137573
10	156984

iv) Comparison Analysis of recommended method based on data size

Table 6.3 compares the performance of the Proposed work with [237], [238], and [239] against the parameter time. The time under the mentioned techniques for various data sizes is depicted in the table.

Table 6.3: Comparison of the proposed method and existing methods with varying data size

Data size	Time (ms) **Proposed**	Time (ms) [237]	Time (ms) [238]	Time (ms) [239]
10000	113485	136537	158648	175754
20000	156984	185436	194723	226648
30000	214785	248765	259648	278538
40000	286545	359473	378386	407285

When the data size is 10000, the time under proposed is 113485 ms, under [237] is 136537 ms, under [238] is 158648 ms, and the time under [239] is 175754 ms. The proposed method has a lower time than all the other compared methods in all the data sizes considered for experimentation. The time consumption has to be low for a method to be efficient. Therefore, it can be confirmed that the proposed scheme is more efficient in comparison to existing methods.

The graphical representation of this performance analysis based on time consumption is presented in Figure 6.5.

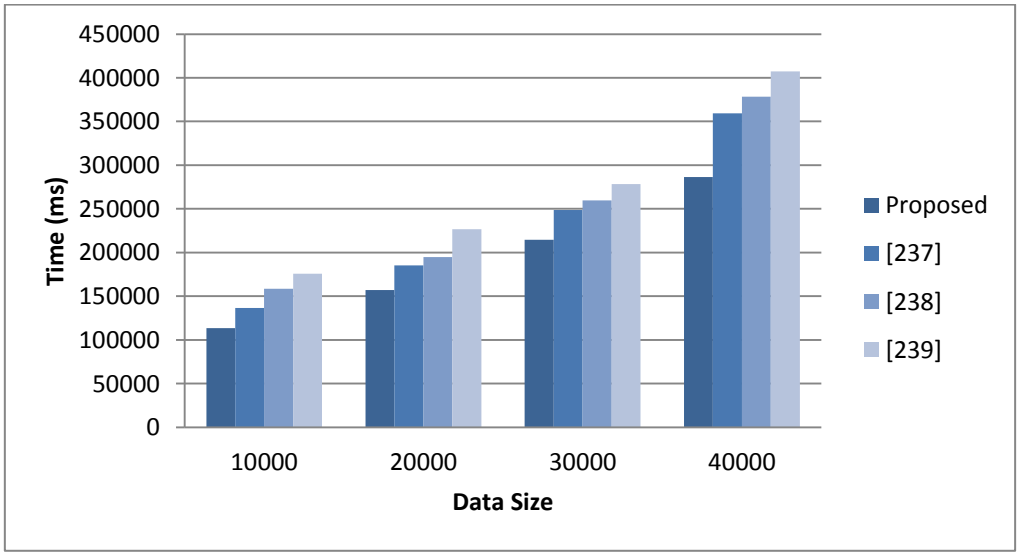

Figure 6.5: Performance Comparison of the Proposed Method with Varying Data Size

v) Comparison Analysis of a recommended method based on Mapper

The time consumption of the proposed method is compared with various techniques [237], [238], and [239] for a varying number of mappers, as shown in Table 6.4. When the number of the mappers is 5, the time under proposed is 156984ms, under [237] is 185436 ms, with [238] is 194723 ms, and the time with [239] is 226648 ms. Similar observations are taken for varying mappers, 10, 15, and 20, respectively.

Table 6.4: Comparison of the proposed and existing methods with varying numbers of mappers

Number of mappers	Time (ms) Proposed	Time (ms) [237]	Time (ms) [238]	Time (ms) [239]
5	156984	185436	194723	226648
10	137573	174636	184747	205815
15	124785	148765	169648	198538
20	106545	139473	158386	177285

The graphical representation of this performance analysis based on time for the variable number of mappers is demonstrated in Figure 6.6.

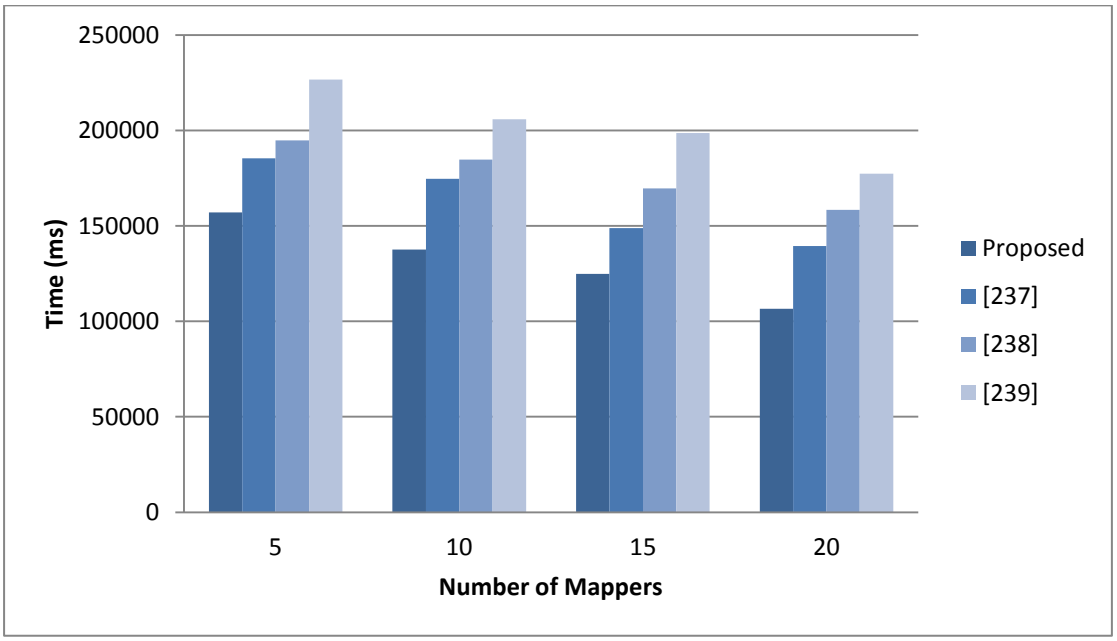

Figure 6.6: Performance Comparison of Proposed and Existing Methods with Varying Numbers of Mappers

vi) Comparison Analysis of a recommended method based on cluster size

Table 6.5 shows the performance comparison of the proposed work against [237], [238], and [239] with varying cluster sizes. The time under the mentioned techniques for various cluster sizes is depicted in the below table.

Table 6.5: Comparison of the proposed method and existing methods with time-based varying cluster sizes

Cluster size	**Time (ms) Proposed**	**Time (ms)** [237]	**Time (ms)** [238]	**Time (ms)** [239]
4	106545	139473	158386	177285
6	124785	148765	169648	198538
8	137573	174636	184747	205815
10	156984	185436	194723	226648

When the cluster size is 10, the time with the proposed scheme is 156984 ms, and with [237] is 185436 ms, under [238] is 194723 ms, and with [239] is 226648 ms. The proposed method takes lesser time than all the other compared methods for cluster size taken into account. The time consumption has to be low for a method to call it an efficient method. Thus, from Table 6.5, it is observed that the proposed method has better performance as compared to existing methods. The graphical representation of this performance analysis based on time for different cluster size is presented in Figure 6.7

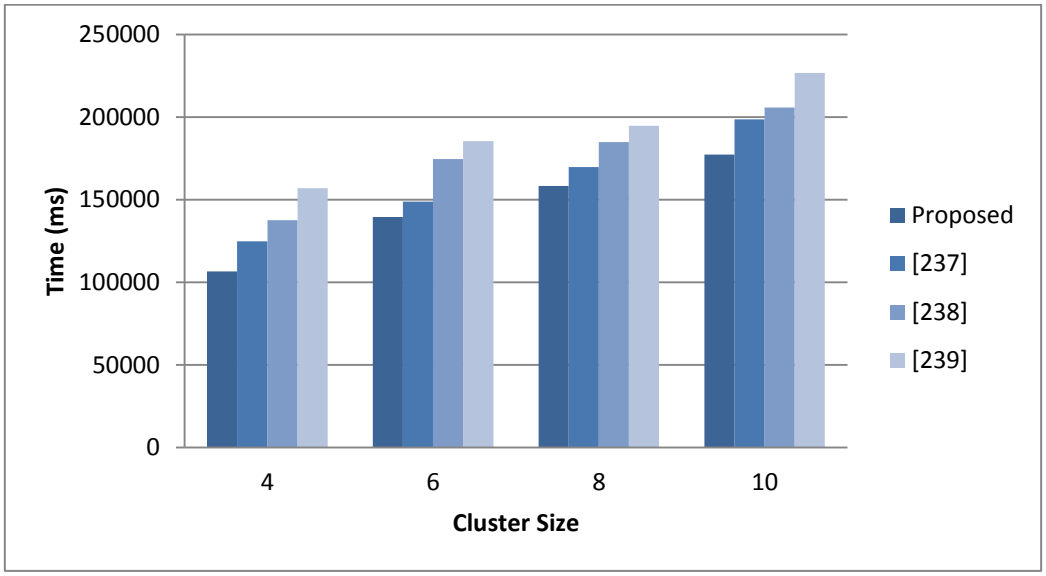

Figure 6.7: Performance Comparison of Proposed and Existing Methods with Time-Based on Cluster Size

vii) Comparison of the proposed method against VSSFA

The proposed method is also compared with the [232] (VSSFA), which possesses secure storage of big data by "variation step size firefly algorithm". The comparative results are tabulated in Table 6.6.

Table 6.6: Comparison of the Proposed Method with VSSFA

Data size (MB)	**Proposed Time (ms)**	[232] **Time (ms)**
10000	113485	254684
20000	156984	291678
30000	214785	351246
40000	286545	427643

From the above table, it is evident that the suggested method takes less time for 10000 data size in comparison to [232]. The proposed method takes 113485ms for 10000 data, but VSSFA takes 254684 ms, which is higher when compared to the proposed method. Similar observations are also taken for 20000, 30000, and 40000 data sizes.

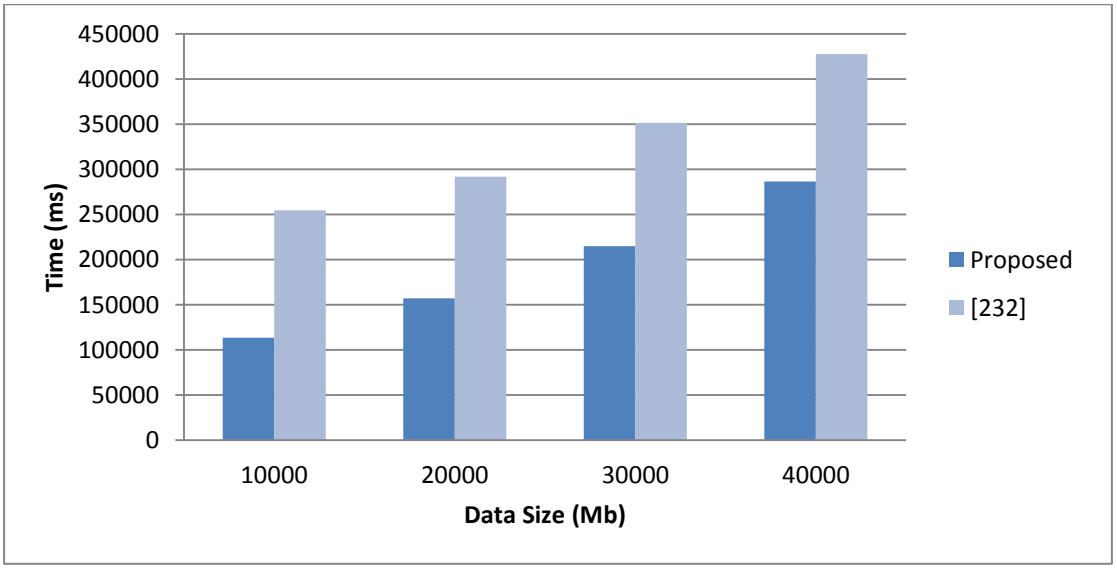

Figure 6.8: Comparison of the Proposed Method with VSSFA by Varying Data Size

Table 6.7: Comparison of the proposed method and VSSFA by varying number of mappers

Mapper size	Proposed Time (ms)	[232] Time (ms)
5	156984	284456
10	137573	245673
15	124785	223453
20	106545	218652

The proposed method takes minimum time as compared [232] for the different number of mappers. For 5 mappers, the proposed method takes 156984 ms, while [232] takes 284456 ms. Similar observations are there for 10, 15 and 20 mappers, respectively.

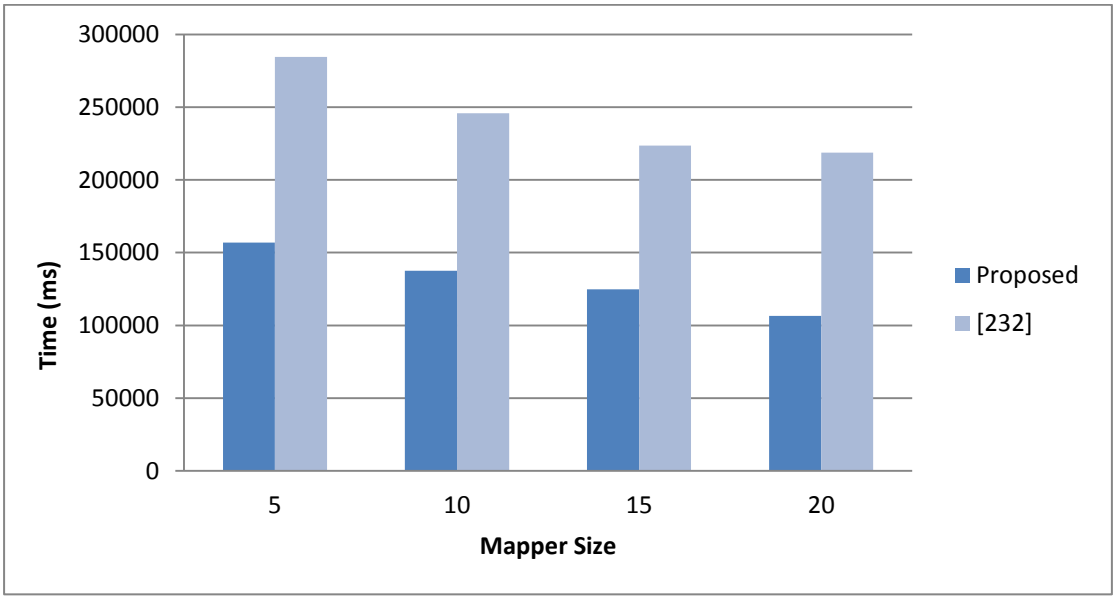

Figure 6.9: Comparison of the Proposed Method and VSSFA by varying mapper

6.5 Summary

Secure storage of big data and an effective access control mechanism is proposed in this chapter. Here the implementation is done by CloudSim with the Hadoop MapReduce framework. The performance of the recommended method is assessed by

means of memory, encryption time, decryption time, and execution time. From the experimental results, it is clear that the suggested method attains the minimum time as compared to the existing methods. This can be attributed to the fact that the proposed method utilizes the optimal ECC with the help of MGOA algorithm for secure storage.

CHAPTER 7

CONCLUSION AND FUTURE WORK

With the emergence of new trends and technologies in day-to-day applications, new challenges of securing information have arisen in the areas such as Big data, Cloud, IoT, and so forth. With the rampant use of the internet today, data is on the air; it is not only textual data but a mixture of all possible data forms, including text, audio, video, and many more. The heterogeneity and volume of data aggravate the problem of achieving confidentiality using traditional encryption schemes. Hence, newer applications require newer security solutions. Having said that, it is important to note that one of the prime aspects of achieving desired privacy is to have a secure mechanism for secret sharing. This shared secret can be used for deriving keys to provide secrecy for encrypting data in any of the above applications. Existing schemes do not offer multi-secret sharing at multiple levels; this gap has been overcome in our proposed verifiable scheme. This proposed scheme proved to be more secure and reasonably efficient when compared with existing techniques.

Along with the processing of data, its secure storage management on the cloud is also a challenge for today's world. In this light, homomorphic encryption act as a saviour, as it facilitates us to perform computation on encrypted data without decrypting it. After that, computed data can be decrypted with the use of Homomorphic Encryption. Implementation results are presented to demonstrate computation of a certain depth.

Further, parallel and distributed architectures definitely come out as a solution to handle huge data on the cloud. The term Big Data is catching the eye of researchers because of the new challenges posed in its management and from the point of security. Catering to this research needs an integrated solution combining existing methods for clustering (FCM), optimized encryption scheme (OECC), and access control mechanism. The proposed integration is implemented with the Hadoop MapReduce framework. It shows promising results with respect to computational performance with varying parameters like data size, the number of clusters, and mappers.

Lastly, a blockchain-based solution is proposed to handle the special needs of communication and security in resource-constrained IoT environments in automating manufacturing industries, healthcare, etc. Blockchain is a distributed ledger that offers inherent advantages for storing IoT sensed information securely in an immutable, trusted manner. Further, an energy-efficient consensus algorithm is proposed, which helps overcome the resource constraints, and the signcryption algorithm handles the required security aspects. The proposed scheme is proved to be efficient in terms of mining latency and throughput when compared to existing schemes.

As a part of future work, some extensions may be done if some time and resources are available. Homomorphic encryption, though it is being evolved at an exponential rate in the last ten years, still a practical scheme having high security and reasonable efficiency for resource constraint environment is needed. Homomorphism is a growing field, and there is much more to explore in it. The community always demands a universal homomorphic scheme, which may be used in all real-time applications. One may focus on designing or modifying existing homomorphic cryptosystems and extend the usability and practicality of homomorphic Encryption Schemes. Few possible directions in SS to find a method in which each level or layer can have different thresholds instead of a global threshold. Another is to use a suitable mathematical construct by which one can distribute one master share instead of multiple shares for multiple secrets. In view of lightweight schemes, parametric reduction techniques may be employed to collect sensor data that balances the load of schemes with desired computational efficiency. Lastly, there is always a need to develop dynamic and group-based access control mechanisms. Secret sharing schemes may be clubbed with attribute-based access control mechanisms to achieve the same.

REFERENCES

[1] S. Andrew, "Tanenbaum computer networks," *Comput. Networks, Englewood Cliffs*, pp. 141–148, 1996.

[2] B. Singh, *Network security and management*. PHI Learning Pvt. Ltd., 2011.

[3] M. E. Whitman and H. J. Mattord, *Principles of Information Security*. 2018.

[4] J. R. Vacca, *Computer and Information Security Handbook*. 2009.

[5] U. H. Rao and U. Nayak, *The InfoSec handbook: An introduction to information security*. Springer, 2014.

[6] L. Dong and K. Chen, "Introduction of Cryptographic Protocols," in *Cryptographic Protocol*, Springer, 2012, pp. 1–12.

[7] A. Abraham and M. Paprzycki, "Significance of steganography on data security," in *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, 2004, vol. 2, pp. 347–351.

[8] C. E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.

[9] C. Chris, "Review of History of Cryptography and Cryptanalysis by John Dooley," *Cryptologia*, vol. 43, no. 6, pp. 536–538, 2019.

[10] M. Faheem, S. Jamel, A. Hassan, Z. A., N. Shafinaz, and M. Mat, "A Survey on the Cryptographic Encryption Algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 11, pp. 333–44, 2017.

[11] W. Diffie, W. Diffie, and M. E. Hellman, "New Directions in Cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[12] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–6, 1978.

[13] T. Elgamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–72, 1985.

[14] L. Mahabadi, "'Introduction to Modern Cryptography' by Jonathan Katz and Yehuda Lindell Chapman \& Hall CRC, 2008," 2011.

[15] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–99, 1984.

[16] G. R. Blakley, "Safeguarding cryptographic keys," in *1979 International Workshop on Managing Requirements Knowledge, MARK 1979*, 1979.

[17] A. Shamir, "How to Share a Secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[18] G. Yong-Jun, F. Xiao-Hong, and H. Fan, "A new multi-secret sharing scheme with multi-policy," in *The 9th International Conference on Advanced Communication Technology*, 2007, vol. 3, pp. 1515–1517.

[19] P. Liaojun, L. Huixian, and W. Yumin, "An efficient and secure multi-secret sharing scheme with general access structures," *Wuhan Univ. J. Nat. Sci.*, vol. 11, no. 6, pp. 1649–1652, 2006.

[20] Y. Wei, P. Zhong, and G. Xiong, "A multi-stage secret sharing scheme with general access structures," in *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, 2008, pp. 1–4.

[21] A. Das and A. Adhikari, "An efficient multi-use multi-secret sharing scheme based on hash function," *Appl. Math. Lett.*, vol. 23, no. 9, pp. 993–6, 2010.

[22] M. Rambabu, S. Gupta, and R. S. Singh, "Data mining in cloud computing: survey," in *Innovations in Computational Intelligence and Computer Vision*, Springer, 2021, pp. 48–56.

[23] C. Gentry, "A Fully Homomorphic Encryption Scheme," *Dissertation*, no. September, p. 169, 2009.

[24] T. Cisco and A. Internet, "Cisco: 2020 CISO Benchmark Report," *Comput. Fraud Secur.*, vol. 2020, no. 3, pp. 4–4, 2020.

[25] B. J. Gantz, D. Reinsel, and B. D. Shadows, "Big Data , Bigger Digital Shadow s , and Biggest Grow th in the Far East Executive Summary: A Universe of Opportunities and Challenges," *Idc*, pp. 1–16, 2012.

[26] K.-Y. Lin, L.-H. Xu, and J.-H. Wu, "A fast fuzzy C-means clustering for color image segmentation," *J. Image Graph.*, vol. 9, no. 2, pp. 159–163, 2004.

[27] R. Xu, S. Member, and D. W. Ii, "Scholars' Mine," 2005.

[28] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. 2009.

[29] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "T h d f s," *hadoop Distrib. file Syst. In2010 IEEE 26th Symp. mass storage Syst. Technol.*, pp. 1–10, 2013.

[30] S. N. Khezr and N. J. Navimipour, "MapReduce and its application in optimization algorithms: A comprehensive study," *Majlesi J. Multimed. Process.*, vol. 4, no. 3, 2015.

[31] J. Wang, D. Yuan, and M. Jiang, "Parallel K-PSO based on MapReduce," *Int. Conf. Commun. Technol. Proceedings, ICCT*, pp. 1203–1208, 2012.

[32] C.-Y. Lin, Y.-M. Pai, K.-H. Tsai, C. H.-P. Wen, and L.-C. Wang, "Parallelizing Modified Cuckoo Search on MapReduce Architecture," *J. Electron. Sci. Technol.*, vol. 11, no. 2, pp. 115–123, 2013.

[33] Y. Lu, C. Liu, I. Kevin, K. Wang, H. Huang, and X. Xu, "Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robot. Comput. Integr. Manuf.*, vol. 61, p. 101837, 2020.

[34] A. Diro, H. Reda, N. Chilamkurti, A. Mahmood, N. Zaman, and Y. Nam, "Lightweight authenticated-encryption scheme for Internet of Things based on publish-subscribe communication," *IEEE Access*, vol. 8, pp. 60539–60551, 2020.

[35] F. C. de Oliveira, J. J. P. C. Rodrigues, R. A. L. Rabêlo, and S. Mumtaz, "Performance delay comparison in random access procedure for NB-IoT, LoRa, and SigFox IoT protocols," in *2019 IEEE 1st Sustainable Cities Latin America Conference (SCLA)*, 2019, pp. 1–6.

[36] C. Gomez, A. Minaburo, L. Toutain, D. Barthel, and J. C. Zuniga, "IPv6 over LPWANs: Connecting low power wide area networks to the Internet (of Things)," *IEEE Wirel. Commun.*, vol. 27, no. 1, pp. 206–213, 2020.

[37] S. Zeadally, A. K. Das, and N. Sklavos, "Cryptographic technologies and protocol standards for Internet of Things," *Internet of Things*, vol. 14, p. 100075, 2021.

[38] S. A. Jesudurai and A. Senthilkumar, "An improved energy efficient cluster head selection protocol using the double cluster heads and data fusion methods for IoT applications," *Cogn. Syst. Res.*, vol. 57, pp. 101–106, 2019.

[39] U. Bodkhe, D. Mehta, S. Tanwar, P. Bhattacharya, P. K. Singh, and W.-C. Hong, "A survey on decentralized consensus mechanisms for cyber physical systems," *IEEE Access*, vol. 8, pp. 54371–54401, 2020.

[40] Y. N. Liu, Q. Zhong, M. Xie, and Z. B. Chen, "A novel multiple-level secret image sharing scheme," *Multimed. Tools Appl.*, vol. 77, no. 5, pp. 6017–31, 2018.

[41] M. Hadian Dehkordi and R. Ghasemi, "A lightweight public verifiable multi secret sharing scheme using short integer solution," *Wirel. Pers. Commun.*, vol. 91, no. 3, pp. 1459–1469, 2016.

[42] S. Mashhadi, "Secure publicly verifiable and proactive secret sharing schemes with general access structure," *Inf. Sci. (Ny).*, vol. 378, pp. 99–108, 2017.

[43] A. Behnad and T. Eghlidos, "A new, publicly verifiable, secret sharing scheme," *Sci. Iran.*, vol. 15, no. 2, pp. 246–251, 2008.

[44] R. Li, Y. Xiao, C. Zhang, T. Song, and C. Hu, "Cryptographic algorithms for privacy-preserving online applications," *Math. Found. Comput.*, vol. 1, no. 4, pp. 311–330, 2018.

[45] B. Rajabi and Z. Eslami, "A verifiable threshold secret sharing scheme based on lattices," *Inf. Sci. (Ny).*, vol. 501, pp. 655–661, 2019.

[46] P. Bhattacharya, S. Tanwar, R. Shah, and A. Ladha, "Mobile edge computing-enabled blockchain framework—a survey," in *Proceedings of ICRIC 2019*, Springer, 2020, pp. 797–809.

[47] U. Martínez-Peñas, "Communication efficient and strongly secure secret sharing schemes based on algebraic geometry codes," pp. 1–23, 2016.

[48] K. Peng, "Threshold distributed access control with public verification: a practical application of PVSS," *Int. J. Inf. Secur.*, vol. 11, 2012.

[49] V. P. Binu and A. Sreekumar, "Threshold Multi Secret Sharing Using Elliptic Curve and Pairing," *arXiv Prepr. arXiv1603.09524*, vol. 9, no. 4, 2016.

[50] L. Harn and M. Fuyou, "Multilevel threshold secret sharing based on the Chinese Remainder Theorem," *Inf. Process. Lett.*, vol. 114, no. 9, pp. 504–509, 2014.

[51] S. Iftene and I. Boureanu, "Weighted Threshold Secret Sharing Based on the Chinese Remainder Theorem Threshold Secret Sharing Schemes," pp. 1–9, 2014.

[52] L. Harn, C. F. Hsu, Z. Xia, and J. Zhou, "How to share secret efficiently over networks," *Secur. Commun. Netw.*, 2017.

[53] A. N. Amroudi, A. Zaghain, and M. Sajadieh, "A verifiable (k, n, m)-threshold multi-secret sharing scheme based on NTRU cryptosystem," *Wirel. Pers. Commun.*, vol. 96, no. 1, pp. 1393–1405, 2017.

[54] M. Li, J. Yu, and R. Hao, "A cellular automata based verifiable multi-secret sharing scheme without a trusted dealer," *Chin. J. Electron.*, vol. 26, no. 2, pp. 313–318, 2017.

[55] A. N. Tentu, A. Basit, K. Bhavani, and V. C. Venkaiah, "Multi-secret sharing scheme for level-ordered access structures," in *International Conference on Number-Theoretic Methods in Cryptology*, 2017, pp. 267–278.

[56] C. Hu, X. Liao, and X. Cheng, "Verifiable multi-secret sharing based on LFSR sequences," *Theor. Comput. Sci.*, vol. 445, no. 52, p. 62, 2012.

[57] B. Duari and D. Giri, "An ideal and perfect (t, n) multi-secret sharing scheme based on finite geometry," *Adv. Intell. Syst. Comput.*, vol. 699, pp. 85–94, 2019.

[58] T. Zhang, X. Ke, and Y. Liu, "(t,n) multi-secret sharing scheme extended from Harn-Hsu's scheme," *Eurasip J. Wirel. Commun. Netw.*, no. 1, pp. 0–3, 2018.

[59] L. Harn and C. F. Hsu, "(t, n) Multi-Secret Sharing Scheme Based on Bivariate Polynomial," *Wirel. Pers. Commun.*, pp. 193–195, 2017.

[60] V. P. Binu, D. G. Nair, and A. Sreekumar, "Secret Sharing Homomorphism and Secure E-voting," *arXiv Prepr. arXiv1602.05372*, 2016.

[61] L. Harn and M. Fuyou, "Weighted secret sharing based on the chinese remainder theorem," *Int. J. Netw. Secur.*, vol. 16, no. 6, pp. 420–426, 2014.

[62] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms. Foundations of secure computation." pp. 11, 169--180, 1978.

[63] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 169–178.

[64] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Annual international conference on the theory and applications of cryptographic techniques*, 2010, pp. 24–43.

[65] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Annual cryptology conference*, 2011, pp. 505–524.

[66] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, 2012, pp. 1219–1234.

[67] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–35, 2018.

[68] J. H. Silverman, J. Pipher, and J. Hoffstein, *An introduction to mathematical cryptography*. Springer, 2008.

[69] C. Moore, N. Hanley, J. McAllister, M. O'Neill, E. O'Sullivan, and X. Cao, "Targeting FPGA DSP slices for a large integer multiplier for integer based FHE," in *International Conference on Financial Cryptography and Data Security*, 2013, pp. 226–237.

[70] J. H. Cheon *et al.*, "Batch fully homomorphic encryption over the integers," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2013, pp. 315–335.

[71] L. Xiao, O. Bastani, and I.-L. Yen, "An efficient homomorphic encryption protocol for multi-user systems," *Cryptol. ePrint Arch.*, 2012.

[72] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *IMA International Conference on Cryptography and Coding*, 2013, pp. 45–64.

[73] S. N. Mohanty *et al.*, "An efficient Lightweight integrated Blockchain (ELIB) model for IoT security and privacy," *Futur. Gener. Comput. Syst.*, vol. 102, pp. 1027–1037, 2020.

[74] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "LSB: A Lightweight Scalable Blockchain for IoT security and anonymity," *J. Parallel Distrib. Comput.*, vol. 134, pp. 180–197, 2019.

[75] W. Zhang, Z. Wu, G. Han, Y. Feng, and L. Shu, "Ldc: A lightweight dada consensus algorithm based on the blockchain for the industrial internet of things for smart city applications," *Futur. Gener. Comput. Syst.*, vol. 108, pp. 574–582, 2020.

[76] W. Ali, I. U. Din, A. Almogren, M. Guizani, and M. Zuair, "A lightweight privacy-aware IoT-based metering scheme for smart industrial ecosystems," *IEEE Trans. Ind. Informatics*, vol. 17, no. 9, pp. 6134–6143, 2020.

[77] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial IoT: Blockchain system with credit-based consensus mechanism," *IEEE Trans. Ind. Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.

[78] S. El Kafhali, I. El Mir, and M. Hanini, "Security Threats, Defense Mechanisms, Challenges, and Future Directions in Cloud Computing," *Arch. Comput. Methods Eng.*, vol. 29, no. 1, pp. 223–246, 2022.

[79] M. Jangjou and M. K. Sohrabi, "A Comprehensive Survey on Security Challenges in Different Network Layers in Cloud Computing," *Arch. Comput. Methods Eng.*, 2022.

[80] A. Razaque, N. Shaldanbayeva, B. Alotaibi, M. Alotaibi, A. Murat, and A. Alotaibi, "Big Data Handling Approach for Unauthorized Cloud Computing Access," *Electronics*, vol. 11, no. 1, 2022.

[81] A. Devi B *et al.*, "Fully Homomorphic Encryption Scheme for Securing Data Computation in Cloud Computing," *SSRN*, 2022.

[82] S. Mittal, S. Sharma, and K. R. Ramkumar, "A Matrix-Based Homomorphic Encryption for Preserving Privacy in Clouds," *{ECS} Trans.*, vol. 107, no. 1, pp. 5441–5448, Apr. 2022.

[83] D. Singh, V. Thada, and J. Singh, "Attribute Based Access Control In Infrastructure As A Service Case Study," in *International Conference on Engineering, Construction, Renewable Energy, and Advanced Materials*, 2022.

[84] S. Belguith, N. Kaaniche, and M. Hammoudeh, "Analysis of attribute-based cryptographic techniques and their application to protect cloud services," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 3, p. e3667, 2022.

[85] S. Khan *et al.*, "An Efficient and Secure Revocation-Enabled Attribute-Based Access Control for eHealth in Smart Society," *Sensors*, vol. 22, no. 1, p. 336, 2022.

[86] P. Schmidt, "Fully homomorphic encryption: Overview and cryptanalysis," *Dortmund Tech. Univ. Dortmund*, 2011.

[87] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 2018.

[88] H. C. A. Van Tilborg and S. Jajodia, *Encyclopedia of cryptography and security*. Springer Science \& Business Media, 2014.

[89] S. Goldwasser and S. Micali, "Probabilistic encryption \& how to play mental poker keeping secret all partial information," in *Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali*, 2019, pp. 173–201.

[90] J. Benaloh, "Verifiable secret-ballot elections [Ph. D. Thesis]," *Yale Univ.*, 1987.

[91] S. Uchiyama and E. Fujisaki, "EPOC: Efficient Probabilistic Public-Key Encryption," *Submiss. to IEEE P1363a*, 1998.

[92] D. Naccache and J. Stern, "A new public key cryptosystem based on higher residues," in *Proceedings of the 5th ACM Conference on Computer and Communications Security*, 1998, pp. 59–66.

[93] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International conference on the theory and applications of cryptographic techniques*, 1999, pp. 223–238.

[94] I. Damgård and M. Jurik, "A generalisation, a simplification and some applications of Paillier's probabilistic public-key system," in *International workshop on public key cryptography*, 2001, pp. 119–136.

[95] T. Sander, A. Young, and M. Yung, "Non-interactive cryptocomputing for nc/sup 1," in *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, 1999, pp. 554–566.

[96] Y. Ishai and A. Paskin, "Evaluating branching programs on encrypted data," in *Theory of Cryptography Conference*, 2007, pp. 575–594.

[97] N. P. Smart and F. Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes," in *International Workshop on Public Key Cryptography*, 2010, pp. 420–443.

[98] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic evaluation of the AES circuit," in *Annual Cryptology Conference*, 2012, pp. 850–867.

[99] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, "Fully homomorphic encryption over the integers with shorter public keys," in *Annual Cryptology Conference*, 2011, pp. 487–504.

[100] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme," in *Annual international conference on the theory and applications of cryptographic techniques*, 2011, pp. 129–148.

[101] G. Chunsheng, "Attack on fully homomorphic encryption over the integers," *arXiv Prepr. arXiv1202.3321*, 2012.

[102] Z. Brakerski and V. Vaikuntanathan, "Efficient Fully Homomorphic Encryption from (Standard) LWE," in *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, 2011, pp. 97–106.

[103] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?," in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, 2011, pp. 113–124.

[104] A. Khedr, G. Gulak, and V. Vaikuntanathan, "SHIELD: scalable homomorphic implementation of encrypted data-classifiers," *IEEE Trans. Comput.*, vol. 65, no. 9, pp. 2848–2858, 2015.

[105] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International conference on machine learning*, 2016, pp. 201–210.

[106] H. Takabi, E. Hesamifard, and M. Ghasemi, "Privacy preserving multi-party machine learning with homomorphic encryption," in *29th Annual Conference on Neural Information Processing Systems (NIPS)*, 2016.

[107] H. Chabanne, A. De Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," *Cryptol. ePrint Arch.*, 2017.

[108] J. Zhang, Y. Yang, Y. Chen, J. Chen, and Q. Zhang, "A general framework to design secure cloud storage protocol using homomorphic encryption scheme," *Comput. Networks*, vol. 129, pp. 37–50, 2017.

[109] E. Hesamifard, H. Takabi, and M. Ghasemi, "Cryptodl: Deep neural networks over encrypted data," *arXiv Prepr. arXiv1711.05189*, 2017.

[110] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-Preserving Deep Learning via Additively Homomorphic Encryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1333–1345, 2018.

[111] A. Al Badawi *et al.*, "Towards the AlexNet Moment for Homomorphic Encryption: HCNN, theFirst Homomorphic CNN on Encrypted Data with GPUs," *arXiv Prepr. arXiv1811.00778*, 2018.

[112] S. Wagh, D. Gupta, and N. Chandran, "SecureNN: 3-Party Secure Computation for Neural Network Training.," *Proc. Priv. Enhancing Technol.*, vol. 2019, no. 3, pp. 26–49, 2019.

[113] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE," in *International Conference on the Theory and Application of Cryptology and Information Security*, 2017, pp. 377–408.

[114] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "TFHE: fast fully homomorphic encryption over the torus," *J. Cryptol.*, vol. 33, no. 1, pp. 34–91, 2020.

[115] B. Li and D. Micciancio, "On the security of homomorphic encryption on approximate numbers," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2021, pp. 648–677.

[116] R. Hamza *et al.*, "Towards Secure Big Data Analysis via Fully Homomorphic Encryption Algorithms," *Entropy*, vol. 24, no. 4, p. 519, 2022.

[117] Y. G. Ramaiah and G. V. Kumari, "Efficient public key homomorphic encryption over integer plaintexts," in *2012 International Conference on Information Security and Intelligent Control*, 2012, pp. 123–128.

[118] B. Choc, S. Goldwasser, S. Micali, and B. Awerbuch, "VERIFIABLE SECRET SHARING AND ACHIEVING SIMULTANEITY IN THE PRESENCE OF FAULTS.," in *Annual Symposium on Foundations of Computer Science (Proceedings)*, 1985, pp. 383–395.

[119] G. Bai, I. Damgård, C. Orlandi, and Y. Xia, "Non-interactive verifiable secret sharing for monotone circuits," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9646, pp. 225–244, 2016.

[120] 2003 Mignotte & Strasbourg, "Maurice Mignotte, Strasbourg," in *Cryptography: Proceedings of the Workshop on Cryptography, Burg Feuerstein, Germany, March 29-April 2, 1982*, 2003, vol. 149, p. 371.

[121] P. Feldman, "A Practical Scheme for Non-interactive Verifiable Secret Sharing Paul Feldman Massachusetts Institute of Technology," *Network*, pp. 427–437, 1987.

[122] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Lecture Notes in Computer Science*, 1992, pp. 129–140.

[123] K. Kaya and A. A. Selçuk, "A verifiable secret sharing scheme based on the chinese remainder theorem," in *Lecture Notes in Computer Science*, 2008, pp. 414–425.

[124] L. Harn and C. Lin, "Strong (n, t, n) verifiable secret sharing scheme," *Inf. Sci. (Ny).*, vol. 180, no. 16, pp. 3059–3064, 2010.

[125] J. C. Benaloh, "Secret sharing homomorphisms: Keeping shares of a secret secret (Extended Abstract)," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1987, pp. 251–260, doi: 10.1007/3-540-47721-7_19.

[126] Q. Al Mahmoud, "Polynomial differential-based strong (n, t, n)-verifiable secret sharing," *IET Inf. Secur.*, vol. 7, no. 4, pp. 312–317, 2013.

[127] Q. Al Mahmoud, "A Novel Verifiable Secret Sharing with Detection and Identification of Cheaters' Group," *Int. J. Math. Sci. Comput.*, vol. 2, no. 2, pp. 1–13, 2016.

[128] S. Mashhadi, "Secure publicly verifiable and proactive secret sharing schemes with general access structure," *Inf. Sci. (Ny).*, vol. 378, pp. 99–108, 2017.

[129] M. Stadler, "Publicly verifiable secret sharing," in *Lecture Notes in Computer Science*, 1996, pp. 190–199.

[130] E. Fujisaki and T. Okamoto, "A practical and provably secure scheme for publicly verifiable secret sharing and its applications," in *Lecture Notes in Computer Science*, 1998, pp. 32–46.

[131] A. Ruiz and J. L. Villar, "Publicly verifiable secret sharing from Paillier's cryptosystem," *Lect. Notes Informatics (LNI), Proc. - Ser. Gesellschaft fur Inform.*, vol. P-74, pp. 98–108, 2005.

[132] S. Heidarvand and J. L. Villar, "Public verifiability from pairings in secret sharing schemes," in *Lecture Notes in Computer Science*, 2008, pp. 294–308.

[133] M. P. Jhanwar, “A practical (non-interactive) publicly verifiable secret sharing scheme,” in *Lecture Notes in Computer Science*, 2011.

[134] T. Y. Wu and Y. M. Tseng, “A pairing-based publicly verifiable secret sharing scheme,” *J. Syst. Sci. Complex.*, vol. 24, no. 1, pp. 186–94, 2011.

[135] A. Ben Shil, K. Blibech, R. Robbana, and W. Neji, “A New PVSS Scheme with a Simple Encryption Function,” *Electron. Proc. Theor. Comput. Sci.*, vol. 122, pp. 11–22, 2013.

[136] R. J. McEliece and D. V. Sarwate, “On Sharing Secrets and Reed-Solomon Codes,” *Commun. ACM*, 1981.

[137] P. Feldman, “Practical Scheme For Non-Interactive Verifiable Secret Sharing,” in *Annual Symposium on Foundations of Computer Science (Proceedings)*, 1987.

[138] Y. Liu, C. Yang, Y. Wang, L. Zhu, and W. Ji, “Cheating identifiable secret sharing scheme using symmetric bivariate polynomial,” *Inf. Sci. (Ny).*, 2018.

[139] J. Shao, “Efficient verifiable multi-secret sharing scheme based on hash function,” *Inf. Sci. (Ny).*, vol. 278, pp. 104–109, 2014.

[140] P. Rogaway and M. Bellare, “Robust computational secret sharing and a unified account of classical secret-sharing goals,” in *Proceedings of the ACM Conference on Computer and Communications Security*, 2007.

[141] M. Tompa and H. Woll, “How to share a secret with cheaters,” in *Lecture Notes in Computer Science*, 1987.

[142] S. Cabello, C. Padró, and G. Sáez, “Secret sharing schemes with detection of cheaters for a general access structure,” *Des. Codes Cryptogr.*, vol. 25, no. 2, pp. 175–188, 2002.

[143] K. Kurosawa, S. Obana, and W. Ogata, “t-cheater identifiable (k, n) threshold secret sharing schemes,” in *Lecture Notes in Computer Science*, 1995.

[144] T. Rabin and M. Ben-Or, “Verifiable secret sharing and multiparty protocols with honest majority,” 1989.

[145] K. Martin, "Challenging the adversary model in secret sharing schemes," *Coding Cryptogr. II, Proc. R. ...*, 2008.

[146] J. Pieprzyk and X. M. Zhang, "On cheating immune secret sharing," *Discret. Math. Theor. Comput. Sci.*, vol. 6, no. 2, pp. 255–64, 2004.

[147] C. Carlet, "On the propagation criterion of degree l and order k," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1998.

[148] J. Pieprzyk and X. M. Zhang, "Cheating prevention in secret sharing over GF(pt)," in *Lecture Notes in Computer Science*, 2001.

[149] J. Pieprzyk and X. M. Zhang, "Constructions of cheating immune secret sharing," in *Lecture Notes in Computer Science*, 2002.

[150] A. Braeken, S. Nikova, and V. Nikov, "On Cheating Immune Secret Sharing.," *IACR Cryptol. ePrint Arch.*, p. 200, 2004.

[151] P. D'Arco, W. Kishimoto, and D. R. Stinson, "Properties and constraints of cheating-immune secret sharing schemes," in *Discrete Applied Mathematics*, 2006, pp. 154(2):219–33.

[152] R. dela Cruz and H. Wang, "Cheating-immune secret sharing schemes from codes and cumulative arrays," *Cryptogr. Commun.*, pp. 67–83, 2013.

[153] M. Wen Ping and M. H. Lee, "New methods to construct cheating immune functions," *Lect. Notes Comput. Sci.*, pp. 79–86, 2004.

[154] W. P. Ma and F. T. Zhang, "New methods to construct cheating immune multisecret sharing scheme," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2005.

[155] J. Brendel and D. Demirel, "Efficient proactive secret sharing," *2016 14th Annu. Conf. Privacy, Secur. Trust. PST 2016*, pp. 543–550, 2016.

[156] F. Zhang *et al.*, "ChURP: Dynamic-committee proactive secret sharing," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2019, pp. 2369–2386.

[157] L. Zhou, F. B. Schneider, and R. Van Renesse, "APSS: Proactive secret sharing in asynchronous systems," *ACM Trans. Inf. Syst. Secur.*, vol. 8, no. 3, pp. 259–86, 2005.

[158] A. Low *et al.*, "Proactive Secret Sharing in Dynamic Environments," 2019.

[159] S. Basu, A. Tomescu, M. Reiter, and D. Malkhi, "Asynchronous verifiable secret-sharing protocols on a good day," *arXiv Prepr. arXiv1807.03720*, 2018.

[160] M. Backes, A. Datta, and A. Kate, "Asynchronous computational VSS with reduced communication complexity," in *Lecture Notes in Computer Science*, 2013.

[161] S. Basu, A. Tomescu, I. Abraham, D. Malkhi, M. K. Reiter, and E. G. Sirer, "sAVSS: Scalable Asynchronous Verifiable Secret Sharing in BFT Protocols," *arXiv Prepr. arXiv1807.03720*, pp. 88–97, 2018.

[162] C. Cachin, K. Kursawe, and A. Lysyanskaya, "Asynchronous verifiable secret sharing and proactive cryptosystems," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2002, pp. 88–97, doi: 10.1145/586123.586124.

[163] A. Patra, A. Choudhury, and C. Pandu Rangan, "Efficient Asynchronous Verifiable Secret Sharing and Multiparty Computation," *J. Cryptol.*, vol. 49, p. 109, 2013.

[164] A. Srinivasan and P. N. Vasudevan, "Leakage Resilient Secret Sharing and Applications," in *Lecture Notes in Computer Science*, 2019.

[165] S. Basu, D. Malkhi, A. Tomescu, M. K. Reiter, I. Abraham, and E. G. Sirer, "Efficient verifiable secret sharing with share recovery in BFT protocols," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2019.

[166] V. Goyal and A. Kumar, "Non-malleable secret sharing for general access structures," in *Lecture Notes in Computer Science*, 2018.

[167] A. Beimel, "Secret-sharing schemes: a survey," in *International conference on coding and cryptology*, 2011, pp. 11–46.

[168] J. Shao and Z. Cao, "A new efficient (t, n) verifiable multi-secret sharing (VMSS) based on YCH scheme," *Appl. Math. Comput.*, vol. 168, no. 1, pp. 135–140, 2005.

[169] M. H. Dehkordi and S. Mashhadi, "An efficient threshold verifiable multi-secret sharing," *Comput. Stand. Interfaces*, vol. 30, no. 3, pp. 187–190, 2008.

[170] J. Zhao, J. Zhang, and R. Zhao, "A practical verifiable multi-secret sharing scheme," *Comput. Stand. Interfaces*, vol. 29, no. 1, pp. 138–41, 2007.

[171] R. F. Olimid, "Dealer-Leakage Resilient Verifiable Secret Sharing," *IACR Cryptol. ePrint Arch.*, pp. 1–10, 2014.

[172] L. Harn, "Secure secret reconstruction and multi- secret sharing schemes with unconditional security," *Secur. Commun. networks*, vol. 7, no. 3, pp. 567–573, 2014.

[173] J. Bermejo Higuera, J. R. Bermejo Higuer, J. A. Sicilia Montalvo, and R. González Crespo, "Introduction to Cryptography in Blockchain," in *Implementing and Leveraging Blockchain Programming*, Springer, 2022, pp. 1–34.

[174] D. L. Chaum, *Computer Systems established, maintained and trusted by mutually suspicious groups*. Electronics Research Laboratory, University of California, 1979.

[175] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," in *Conference on the Theory and Application of Cryptography*, 1990, pp. 437–455.

[176] J. Xue, C. Xu, and Y. Zhang, "Private blockchain-based secure access control for smart home systems," *KSII Trans. Internet Inf. Syst.*, vol. 12, no. 12, pp. 6057–6078, 2018.

[177] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Secur.*, vol. 1, no. 1, pp. 36–63, 2001.

[178] F. Hofmann, S. Wurster, E. Ron, and M. Böhmecke-Schwafert, "The immutability concept of blockchains and benefits of early standardization," in *2017 ITU Kaleidoscope: Challenges for a Data-Driven Society (ITU K)*, 2017, pp. 1–8.

[179] E. Landerreche and M. Stevens, "On immutability of blockchains," in *Proceedings of 1st ERCIM Blockchain Workshop 2018*, 2018.

[180] T. Aste, P. Tasca, and T. Di Matteo, "Blockchain technologies: The foreseeable impact on society and industry," *Computer (Long. Beach. Calif).*, vol. 50, no. 9, pp. 18–28, 2017.

[181] A. S. Rajasekaran, M. Azees, and F. Al-Turjman, "A comprehensive survey on blockchain technology," *Sustain. Energy Technol. Assessments*, vol. 52, p. 102039, 2022.

[182] R. Huo, S. Zeng, Z. Wang, J. Shang, and W. Chen, "A Comprehensive Survey on Blockchain in Industrial Internet of Things: Motivations, Research Progresses, and Future Challenges," *IEEE Commun. Surv. \& Tutorials*, 2022.

[183] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *2016 IEEE symposium on security and privacy (SP)*, 2016, pp. 839–858.

[184] V. Buterin and others, "A next-generation smart contract and decentralized application platform," *white Pap.*, vol. 3, no. 37, pp. 1–2, 2014.

[185] T. Neudecker and H. Hartenstein, "Network layer aspects of permissionless blockchains," *IEEE Commun. Surv. \& Tutorials*, vol. 21, no. 1, pp. 838–857, 2018.

[186] C. Qiu, H. Yao, F. R. Yu, C. Jiang, and S. Guo, "A service-oriented permissioned blockchain for the Internet of Things," *IEEE Trans. Serv. Comput.*, vol. 13, no. 2, pp. 203–215, 2019.

[187] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Commun. Surv. \& Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.

[188] S. Kaur, S. Chaturvedi, A. Sharma, and J. Kar, "A research survey on applications of consensus protocols in blockchain," *Secur. Commun. Networks*, 2021.

[189] S. M. H. Bamakan, A. Motavali, and A. B. Bondarti, "A survey of blockchain consensus algorithms performance evaluation criteria," *Expert Syst. Appl.*, vol. 154, pp. 113–385, 2020.

[190] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 3–16.

[191] M. S. Ferdous, M. J. M. Chowdhury, M. A. Hoque, and A. Colman, "Blockchain consensus algorithms: A survey," *arXiv Prepr. arXiv2001.07091*, 2020.

[192] Q. Wang, J. Huang, S. Wang, Y. Chen, P. Zhang, and L. He, "A comparative study of blockchain consensus algorithms," in *Journal of Physics: Conference Series*, 2020, vol. 1437, no. 1, p. 12007.

[193] S. Velliangiri and P. Karthikeyan, "Blockchain technology: challenges and security issues in consensus algorithm," in *2020 International Conference on Computer Communication and Informatics (ICCCI)*, 2020, pp. 1–8.

[194] G.-T. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain," *J. Inf. Process. Syst.*, vol. 14, no. 1, pp. 101–128, 2018.

[195] W. Tong, X. Dong, and J. Zheng, "Trust-pbft: A peertrust-based practical byzantine consensus algorithm," in *2019 International Conference on Networking and Network Applications (NaNA)*, 2019, pp. 344–349.

[196] W. Cai, W. Jiang, K. Xie, Y. Zhu, Y. Liu, and T. Shen, "Dynamic reputation--based consensus mechanism: Real-time transactions for energy blockchain," *Int. J. Distrib. Sens. Networks*, vol. 16, no. 3, 2020.

[197] X. Zheng and W. Feng, "Research on practical Byzantine fault tolerant consensus algorithm based on blockchain," in *Journal of Physics: Conference Series*, 2021, vol. 1802, no. 3, p. 32022.

[198] L. Lamport, "Paxos made simple," *ACM SIGACT News (Distributed Comput. Column) 32, 4 (Whole Number 121, December 2001)*, pp. 51–58, 2001.

[199] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX Annual Technical Conference (Usenix ATC 14)*, 2014, pp. 305–319.

[200] S. J. Alsunaidi and F. A. Alhaidari, "A survey of consensus algorithms for blockchain technology," in *2019 International Conference on Computer and Information Sciences (ICCIS)*, 2019, pp. 1–6.

[201] A. Jain and D. S. Jat, "A Review on Consensus Protocol of Blockchain Technology," in *Intelligent Sustainable Systems*, Springer, 2022, pp. 813–829.

[202] S. Pahlajani, A. Kshirsagar, and V. Pachghare, "Survey on private blockchain consensus algorithms," in *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*, 2019, pp. 1–6.

[203] A. Kousaridas, S. Falangitis, P. Magdalinos, N. Alonistioti, and M. Dillinger, "SYSTAS: Density-based algorithm for clusters discovery in wireless networks," in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2015, pp. 2126–2131.

[204] B. Zhang, Z. Jia, and C. Zhao, "An efficient certificateless generalized signcryption scheme," *Secur. Commun. Networks*, 2018.

[205] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K.-K. R. Choo, and Y. Park, "Certificateless-signcryption-based three-factor user access control scheme for IoT environment," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3184–3197, 2020.

[206] A. K. Singh and K. S. Vaisla, "A lightweight signcryption scheme based on elliptic curve cryptography," in *Proc. 1st Int. Conf. Adv. Comput. Commun. Eng.(ICACCE)*, 2014, vol. 1, pp. 7–10.

[207] C. Perera, R. Ranjan, L. Wang, S. U. Khan, and A. Y. Zomaya, "Big data privacy in the internet of things era," *IT Prof.*, vol. 17, 2015.

[208] N. Kshetri, "Big data ' s impact on privacy , security and consumer welfare," *Telecomm. Policy*, vol. 38, pp. 1–12, 2014.

[209] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. Ullah Khan, "The rise of 'big data' on cloud computing: Review and open research issues," *Information Systems*. p. 47(98),115, 2015.

[210] Z. Yan, S. Member, W. Ding, X. Yu, H. Zhu, and R. H. Deng, "Deduplication on Encrypted Big Data in Cloud," vol. 2, no. 2, pp. 138–150, 2016.

[211] M. D. Assunção, R. N. Calheiros, S. Bianchi, M. A. S. Netto, and R. Buyya, "Big Data computing and clouds: Trends and future directions," *J. Parallel Distrib. Comput.*, vol. 79, 2015.

[212] Z. Wang, C. Cao, N. Yang, and V. Chang, "ABE with improved auxiliary input for big data security," *J. Comput. Syst. Sci.*, vol. 89, 2017.

[213] P. Li, S. Guo, T. Miyazaki, M. Xie, J. Hu, and W. Zhuang, "Privacy-preserving access to big data in the cloud," *IEEE Cloud Comput*, vol. 5, 2016, doi: 10.1109/MCC.2016.107.

[214] Z. Yan, R. H. Deng, and V. Varadharajan, "Cryptography and data security in cloud computing," *Inf Syst*, vol. 387, 2017.

[215] Z. Tan *et al.*, "Enhancing big data security with collaborative intrusion detection," *IEEE Cloud Comput.*, vol. 3, 2014.

[216] R. Nigoti, M. Jhuria, and S. Singh, "A survey of cryptographic algorithms for cloud computing," 2013.

[217] S. Singh, Y. S. Jeong, and J. H. Park, "A survey on cloud computing security: Issues, threats, and solutions," *J. Netw. Comput. Appl.*, vol. 75, 2016.

[218] J. Kindervag, S. Balaouras, B. Hill, and K. Mak, "Control and protect sensitive information in the era of big data." Academic Press, 2012.

[219] B. G. Batista, C. H. Ferreira, D. C. Segura, D. M. Leite Filho, and M. L. Peixoto, "A QoS-driven approach for cloud computing addressing attributes of performance and security," *Futur Gener Comput Syst*, vol. 68, 2017.

[220] L. T. Yang, G. Huang, J. Feng, and L. Xu, "Parallel GNFS algorithm integrated with parallel block Wiedemann algorithm for RSA security in cloud computing," *Inf Sci*, vol. 387, 2017.

[221] M. Ramachandran, "Software security requirements management as an emerging cloud computing service," *Int J Inf Manag.*, vol. 4, 2016.

[222] D. M. Bui, Y. Yoon, E. N. Huh, S. Jun, and S. Lee, "Energy efficiency for cloud computing system based on predictive optimization," *J Parallel Distrib Comput*, vol. 102, 2017.

[223] H. Xiong, K. K. Choo, and A. V Vasilakos, "Revocable identity-based access control for big data with verifiable outsourced computing," *IEEE Trans Big Data*, vol. 6, 2017.

[224] J. Shen, C. Wang, A. Wang, S. Ji, and Y. Zhang, "A searchable and verifiable data protection scheme for scholarly big data," *IEEE Emerg Top Com*, vol. 14, 2018.

[225] G. Chen, S. Wu, and Y. Wang, "The evolvement of big data systems: from the perspective of an information security application," *Big Data Res.*, vol. 2, 2015.

[226] K. Yang, Q. Han, H. Li, K. Zheng, Z. Su, and X. Shen, "An efficient and fine-grained big data access control scheme with privacy-preserving policy," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 563–571, 2016.

[227] Y. Yang, X. Zheng, W. Guo, X. Liu, and V. Chang, "Privacy-preserving smart IoT-based healthcare big data storage and self-adaptive access control system," *Inf. Sci. (Ny).*, vol. 479, pp. 567–592, 2019.

[228] S. Fugkeaw and H. Sato, "Scalable and secure access control policy update for outsourced big data," *Futur Gener Comput Syst*, vol. 79, 2018.

[229] W. Zhou, D. Feng, Z. Tan, and Y. Zheng, "Improving big data storage performance in hybrid environment," *J Comput Sci*, vol. 26, 2018.

[230] E. Shanmugapriya and R. Kavitha, "Medical big data analysis: preserving security and privacy with hybrid cloud technology," *Soft Comput.*, vol. 23, no. 8, pp. 2585–2596, 2019.

[231] K. S. Saraswathy and S. S. Sujatha, "Internet of Things Big Data Security in Cloud via Stream Cipher and Clustering Model," *Wirel. Pers. Commun.*, vol. 123, no. 4, pp. 3483–3496, 2022.

[232] V. S. Thiyagarajan and A. Ayyasamy, "Privacy-preserving over big data through VSSFA and MapReduce framework in cloud environment," *Wirel Pers Commun*, vol. 4, 2017.

[233] G. Viswanath and P. V. Krishna, "Hybrid encryption framework for securing big data storage in multi-cloud environment," *Evol. Intell.*, vol. 14, no. 2, pp. 691–698, 2021.

[234] Z. Peng, F. Liang, and L. Mu, “Big Data-Based Access Control System in Educational Information Security Assurance,” *Wirel. Commun. Mob. Comput.*, 2022.

[235] K. Yang, X. Jia, K. Ren, R. Xie, and L. Huang, “Enabling efficient access control with dynamic policy updating for big data in the cloud,” in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 2013–2021.

[236] H. E. Alhazmi, F. E. Eassa, and S. M. Sandokji, “Towards Big Data Security Framework by Leveraging Fragmentation and Blockchain Technology,” *IEEE Access*, vol. 10, pp. 10768–10782, 2022.

[237] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris, and I. Aljarah, “Grasshopper optimization algorithm for multi-objective optimization problems,” *Appl. Intell.*, vol. 48, no. 4, pp. 805–820, 2018.

[238] H. John, “Holland. Genetic algorithms,” *Sci. Am.*, vol. 267, no. 1, pp. 44–50, 1992.

[239] N. Koblitz, “Elliptic curve cryptosystems,” *Math. Comput.*, vol. 48, no. 177, pp. 203–209, 1987.