CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY, ISLAMABAD

Student Privacy Preserving Framework Based on Blockchain Technology

by

Saba Noureen

A thesis submitted in partial fulfillment for the degree of Master of Philosophy

in the

Faculty of Computing

Department of Mathematics

2019

I wish to dedicate this thesis to my late father. He taught me to preserve and prepared me to face the challenges with faith and humility. He was a constant source of inspiration to my life. Although, He is not here to give me strength and support. I always feel his presence that used to urge me to strive to achieve my goals in life.

and
my mother

who always had confidence in me and offered me encouragement and support in all my endeavors.

CERTIFICATE OF APPROVAL

Student Privacy Preserving Framework Based on Blockchain Technology

by

Saba Noureen

MMT-171005

THESIS EXAMINING COMMITTEE

S. No.	Examiner	Name	Organization
(a)	External Examiner	Dr. Majid Khan	IST, Islamabad
(b)	Internal Examiner	Mr. Zeeshan Qaiser	CUST, Islamabad
(c)	Supervisor	Dr. Rashid Ali	CUST, Islamabad

Dr. Rashid Ali
Thesis Supervisor
April, 2019

Dr. Muhammad Sagheer
Head
Dept. of Mathematics
April, 2019

Dr. Muhammad Abdul Qadir
Dean
Faculty of Computing
April, 2019

Author's Declaration

I, **Saba Noureen** hereby state that my MPhil thesis titled "**Student Privacy Preserving Framework Based on Blockchain Technology**" is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/abroad. At any time MPhil if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my Degree.

(Saba Noureen)

Registration No: MMT-171005

Plagiarism Undertaking

I solemnly declare that research work presented in this thesis titled "***Student Privacy Preserving Framework Based on Blockchain Technology***" is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been dully acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MPhil Degree, the University reserves the right to withdraw/revoke my MPhil degree and that HEC and the University have the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized work.

(Saba Noureen)

Registration No: MMT-171005

Acknowledgements

First of all, I would like to thank **Almighty Allah** for His countless blessings in my life. He has gifted me a loving family and excellent teachers. He supports me in every path of life.

I would like to express my special thanks to my kind supervisor **Dr. Rashid Ali** for his motivation. He encourages me during my research study. His kind effort and motivation would be never forgotten. I have appreciated the guidance for my supervisor and feeling proud to be a student of such great teacher.

Secondly, I am thankful to all teachers of CUST Islamabad Dr. Muhammad Sagheer, Dr. Abdul Rehman Kashif, Dr. Shafqat Hussain, Dr. M.Afzal, Dr. Dur e Shehwar and Dr. Rashid Ali for conveying the excellent lectures.

I am grateful to my mother for her prayers, love and motivation. I would like to thank my brothers Shahzeb Sarwar, Shahzad Sarwar for their support in completing my degree program. They supported and encouraged me throughout my life. I would like to thank my all family members for their continuous support and patience during my research work.

I would like to thank my all friends Saba Majeed, Sania Mahmood, Sadia Noor, Sundus Iqbal and Mehwish Sehar for supporting me during degree programs. Especially, I would like to thanks Saba Majeed for motivating me during research work.

Finally, I am grateful to the management staff of Capital University of Science and Technology, Islamabad for providing a friendly environment for studies.

I am obliged to all people who share their knowledge with me and support me.

Abstract

Blockchain technology enables distributed, encrypted and secure logging of digital transactions. It is underlying technology of Bitcoin and other cryptocurrencies. Blockchain is expected to revolutionize computing in several areas, particularly where centralization was unnatural and privacy was important.

In this thesis, we propose Student Privacy Preserving Framework (SPPF) based on blockchain technology that preserves users privacy and increases accessibility, while keeping the ESR secure. Our design implements a university-scaled ESR framework which utilizes Ethereum blockchain, smart contracts and identity based proxy re-encryption scheme (IB-PRES) for better access control and transfer of records.

The security of (IB-PRES) is based on the decision bilinear Diffie-Hellman assumption (DBDH) in the random oracle model. Also because of (IB-PRES) our framework can resist the collision attacks and chosen plaintext attack. The goal of this thesis is to analyze how blockchain can be used for different needs of users, providers and third parties and to understand how our framework could address the privacy and security concerns in the educational industry.

Contents

List of Figures

List of Tables

Abbreviations

SPPF	Student Privacy Preserving Framework
ESR	Electronic Student Records
EHR	Electronic Health Records
IB-PRES	Revocable identity-based proxy re-encryption Scheme
DBDH	Decision Bilinear Diffie Hellman
PKI	Public key Infrastructure
PoW	Proof of Work
PoS	Proof of Stake
PBFT	Practical Byzantine Fault Tolerance
DPoS	Delegated Proof of Stake
PKC	Public Key Cryptography
PKA	Public Key Authority
PKG	Private Key Generator
COC	Consensus Smart Contract
CLC	Classification Smart Contract
SHC	Service History Contract
OC	Ownership Contract
PC	Permission Contract
RC	Re-encryption Contract
HTTPS	Hyper Transfer Protocol Secure

Symbols

Tx	Transaction
H	Hash function
M	Plaintext or Message
CT	Ciphertext
$E^{'}$	Encryption Algorithm
$D^{'}$	Decryption Algorithm
$SK^{'}$	Private Key/ Secret Key
$PK^{'}$	Public Key
ID	Recipient's Identity
$\mathcal{C}'$	Ciphertext Space
$\mathcal{M}'$	Message Space
$\mathcal{A}$	Adversary
$\mathcal{PP}$	Public parameters
$\mathbb{G}$	Group
$\mathbb{G}_a$, $\mathbb{G}_b$	Multiplicative Group
UL	User List
RL	Revocation List
$p,\ q$	Prime number
$\mathbb{Z}$	Set of integers

Chapter 1

Introduction

Currently, most of the educational establishments depend on internet accessible database. Their dependence on internet accessible databases causes risk of a hacker to the system who can change student's grades, add an unearned certificates or even can delete the entire database. In December 2016, Lynda.com the online learning platform owned by LinkedIn [53] was hit by a hack. An unauthorized external party accessed student data with 9.5 million user accounts affected [26]. Universities are collecting a huge amount of students data, which can be used for very important situations. This data is kept on private companies or universities centralized server. This imposes great risks and liabilities for student's security on behalf of the private companies who develop these systems, and the university system using it. What if the server gets shut down? What if the company must shut down? and therefore, takes all that data and useful information with it? What if these servers are hacked? What if the university system's databases are hacked?

Blockchain technology is a very effective solution to these problems. By using this technology, we can provide security to students data while using these systems. Blockchain can provide a secured database that keeps a record of each student's individual data and in these systems that data can be owned by the student or parent, with appropriate access for teachers and administrators. It can also keep a detailed history of the student's growth over the years, which can be very valuable to an educator, especially when a student is having a problem or has a possible disability. Blockchain also allow us to securely digitize other student information. We keep records of the students from the beginning of their education, including immunization letters, behavior reports, medical problems, learning deficits, referrals parent contact information, proof of living within a district, and so much more.

Now the question is What is Blockchain? and why this technology could be so powerful? A blockchain is a database which stores permanent blocks of information, such as transaction history, to be shared within a particular company. The well-known application of blockchain technology is Bitcoin [47], which is a cryptocurrency whose users without going through the central banking authority can make and receive payments. Considerably, by using blockchain technology one can send and receive only to the point bits of data to particular parties and every party got a copy of the data, shared liability keeps the data exact and secure. Blockchain uses the concept of public key cryptography. By using private key all transactions in blockchain are signed to establish the identities of different parties. In the context of storing student data in a blockchain, cryptography has to play the additional role of encrypting the data, so that only authorized users can read it. In 2017 joint research center report [5], the European Commission explains why blockchain technology is so powerful. Some significant properties of blockchain technology are:

- Every participant keeps a copy of data owned by them and updates must be validated collectively by all participants.
- The information could be anything like transactions, identities, assets, contracts or the thing which can be explained in digital form.
- Data is accessible, transparent and permanent Because of that participants can have a look on “transaction histories” collectively.
- Every amendment refers as a “new block” appended where chain ends. A protocol oversees how new entries are started, approved, recorded and distributed.

- Blockchain uses cryptography as a keeper of trust, with complex algorithms running by all participants of blockchain to approve the integrity of the whole system which removes the need of third-party intermediaries.

Data on a blockchain can be considered as a form of the public ledger for a particular community, designed to store data securely and accessible to only those users to whom it is most relevant. Our design follows storage approaches closely related to the concept proposed by Linn and Koo [40], Ekblaw *et al.* [24], Ivan [31], and Gaby *et al.* [20] which uses blockchain as an access control layer while storing patient information in an existing database of providers. We have used these concepts for electronic student records (ESR) to secure storage and transfer of data.

In this thesis, we propose a "privacy-preserving framework" for the electronic student records which we named as "Students privacy preserving framework"(SPPF). In SPPF, although student's information continues to be stored in providers databases, but similar to previously proposed schemes [20, 24, 31, 40] our scheme expand user control (where user can be student, teacher or administration) over private data in which user validate the exchange of their data with other providers and third party. Our design is based on Ethereum blockchain [59], [60] like Bitcoin, Ethereum is used as cryptocurrency but the difference is another functionality of Ethereum that is the use of "Smart contracts". The "Smart contracts" are those functions of blockchain that are written to the blockchain and then operate by every node on the block. By costing "gas" in the form of cryptocurrency to the nodes Ethereum network manages these smart contracts. The purpose of this cost is to allow only a limited number of people to run programs in the system. Furthermore, by using Ethereum network, the users can create permissioned blockchain [13] which can only be controlled by a smaller number of users.

Our design focuses on the increased interoperability with the use of permissioned blockchain technology. Our blockchain based design covers the following key points;

- Users (where users can be students, teachers, administrators with appropriate access) will have the ownership and final control of all the electronic records.
- The accessibility of documents can be securely controlled and track how data are utilized.
- Securely transfer the records and reduces the possibility of an unauthorized person to derive the users protected information.

1.1 Our Contribution

Our proposal of SPPF includes contributions for increase privacy and interoperability. It focuses on the secure interaction between different users, providers, and third party. We have the following major contributions;

1. In our design, we have used permissioned blockchain for Electronic student's records (ESR) that keeps the record of hashes of the data references while forwarding the original query link information in a secret transaction over Hypertext Transfer Protocol Secure (HTTPS) channel.
2. For secure transfer of records, we use "Revocable identity-based proxy re-encryption Scheme (IB-PRES)" [41]. The use of IB-PRES permits us to keep a record of keys and small encrypted data straight on the blockchain. It eases the process of transferring records without involving the certification authority for secret keys. The security of IB-PRES is based on the "decision bilinear Diffie-Hellman (DBDH)" assumption.

   Figure 2.1 shows the ability of our framework to deal with blockchain securely for multiple parties and their data. The purpose of using smart contracts on permissioned blockchain is to differentiate between different roles of user, provider and a third party for access control. These contracts permit to arrange different roles which can be suitable for various needs of users. Similar to Gaby *et al.* [20], we have used Qourumchain algorithm [18] for consensus purpose So by utilizing consensus algorithm instead of using proof of work increases the authentication when appending nodes to the system or deleting risky users in our framework.

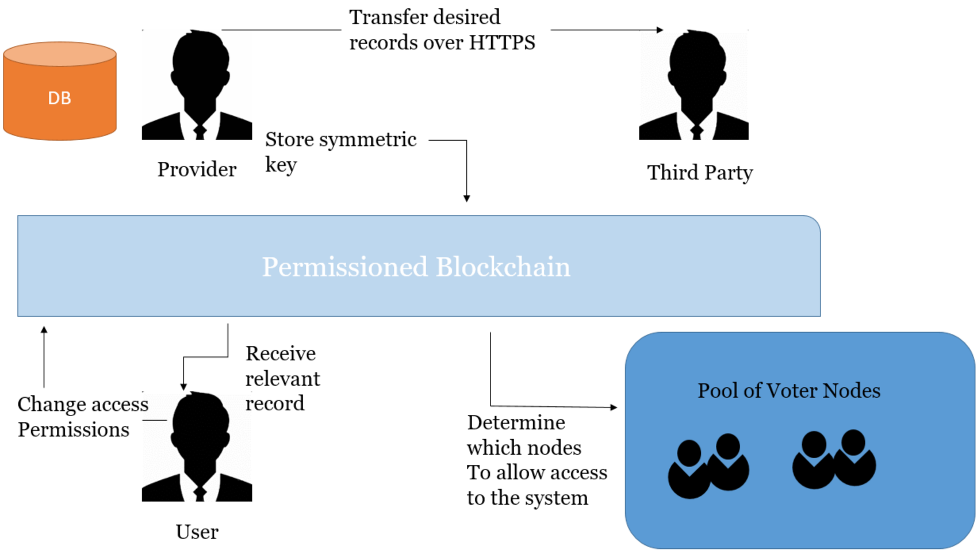
Transfer desired records over HTTPS
DB
Provider
Store symmetric key
Third Party
Permissioned Blockchain
Receive relevant record
Change access Permissions
User
Determine which nodes To allow access to the system
Pool of Voter Nodes

Figure 1.1: An illustration that how different parties deal with one another by utilizing our design.

1.2 Literature Review

Blockchain provides interesting research fields particularly from the perspective of its applications. Many articles have been published in regard to implementing the blockchain in the different industries. The main objective of these articles is to introduce that what is blockchain system which would organize for personal information and permits the users to have a wide overlook of their private data. Li *et al.* [39] presented an organized survey on the security threats to blockchain and corresponding real attacks on popular blockchain systems. In this paper, they have also analyzed the vulnerabilities exploited in these attacks. In medical field many articles are published using permissioned blockchain for Electronic health records (EHR). Ivan [31] has been discussed blockchain as a novel approach to store health data securely, implementation hurdles, and a plan for developing EHR from current technology to a blockchain solution. Ivan presents three various suggestions to avoid the "man in the middle attack" for the interchanging of medical data.

- The provider's databases might be straight attached to blockchain system.
- The already existed provider's system can deliver data to the blockchain.
- Providers can deliver the data to the patients who can add the information manually later on the blockchain.

Linn and Koo [40], Brodersen *et al.* [10], Ekblaw *et al.* [24], Alexander [4] propose different methods to establish and secure access control. To ensure privacy preservation, healthcare blockchain must have to secure EHR from different attacks on the system. For instance, a system access control can be compromised by the man in the middle attack. Linn and Koo [40] for verification of parties proposed bio-metric identity systems. Peterson *et al.* [49] suggested the encryption of public data with the "symmetric key" and secret data with "secret keys". Moreover, when a transfer of record from one party to another party occurs, the Alexander [4] gives the solution of privacy preservation problem by utilizing a "deposit box ". In this scheme, once the transfer of personal medical data is verified by a patient. The data is duplicated to the deposit box for a particular time period and the receiver received access permission to the data for that period of time. Brodersen *et al.* [10] recommends reliable sovereignties like "banks and employers" to improve the verification of an individuals identity on the blockchain which gives authentication of an individuals identity in the system. Similarly, Gaby *et al.* [20] utilizes consensus authentication process prior to registering a node and here in our design we also use the same method to authenticate the individual's identity.

Some of the articles use private transactions to improve access control. In the article presented by the Quorum [18], every node authenticates the public transaction over the blockchain whereas "private transactions" are only authenticated by the node party to the transactions and are stored off-chain. Because of this, the "public state record" is the same for every participant while "private state

record" differs between nodes. "IBM's Hyperledger" [15] and "Quorum" [18], use encryption methods for private data transactions to confirm that only authenticated participants have access to confidential information. Brown *et al.* [11] in transactions for integrity also uses "cryptographic hashes" of data. Likewise, our design use encryption and hashing of recorded data on and off the blockchain.
Privacy of student record, particularly access control is an important task in the educational institutions. Ekblaw *et al.* [24] recommend the application of "permissioned blockchain" and to encrypt data of the blockchain. Our design also uses a "permissioned blockchain" system which permits user communication with the blockchain. It gives a clear level of control over personal records to every node.
To append "new blocks" to the "permissioned blockchain" Gaby *et al.* [20] implements Qourumchain algorithm [18] in which a particular number of nodes have authority to vote for which block to append to the blockchain. The "Quorumchain algorithm" is used in a smart contract, to simplify the process of voting. Our framework uses this algorithm in the process of mining and manages blockchain.
Gaby *et al.* [20] uses the "distributed proxy re-encryption scheme" to tackle the transfer of data among nodes without revealing the symmetric keys by utilizing a proxy. But here certification authority is needed for private and public key generation. In our work, we have removed the need for certification authority by using "revocable identity-based proxy re-encryption" [41]. It reduces the possibility of "man in the middle attack".

The thesis is organized as follows:

- In **Chapter 2**, we discussed the blockchain technology and terms related to it. After this, we explained basic definitions for cryptography and issues related to its key management. Then we presented the drawbacks of the certificate authority and its solution in " identity-based encryption scheme". Lastly, we explained the identity-based proxy re-encryption scheme.

- In **Chapter 3**, we presented the review of Ancile framework for electronic health records system presented by Gaby *et al.* [20].

- In **Chapter 4**, we introduce our framework Students Privacy Preserving framework (SPPF) for electronic student records (ESR) management system and its comparative performance analysis.

Chapter 2

Preliminaries

In this chapter, we give a gentle basic knowledge related to our work.

2.1 Blockchain-A Disruptive Innovation

"Blockchain is a shared, distributed ledger that facilitates the process of recording transactions and tracking assets in a business network. An asset can be tangible a house, a car, cash, land or intangible like intellectual property, such as patents, copyrights, or branding. Virtually anything of value can be tracked and traded on a blockchain network, reducing risk and cutting costs for all involved [38]."

The Blockchain is an obviously inspired technology, That was first introduced by the person Satoshi Nakamoto [47]. But after that, it has progressed into something greater, and every single person asking the main question is, what is blockchain? Blockchain technology can be considered as an operating system, such as Microsoft Windows or MacOS. It was originally designed for the cryptocurrency Bitcoin, we can think of Bitcoin as one of the many applications of Blockchain. Blockchain gives a shared ledger for storing the bitcoin transaction. This shared ledger can also be utilized to store any transaction and trace the movement of any tangible, intangible or digital asset [38].

Blockchain technology depends on three fundamental principles. First of all, the data stored in blockchain is unable to change. So, it is recorded in a unchangeable, public transaction ledger, which can be accessed by anyone. There is always a full and undeniable storage of all transactions, since the transactions are immutable. Secondly, blockchain is implemented in decentralized system of computing nodes, because of which it is secure opposed to attacks and failure. Decentralization also means that nobody can be the owner of or runs the blockchain. Third, the metadata explaining every transaction is accessible to every node on the network, but that does not give sense that the recorded information is readable in the blockchain. The use of pseudo-anonymity and public key infrastructure (PKI) in the blockchain permits the blockchain to encrypt the data in such a way which makes it difficult to decrypt. blockchain has the ability to drastically lessen back-office information input and maintenance costs and enhance information precision and security [31].

2.1.1 Characteristics of Blockchain Network

Following are key characteristics of Blockchain network [38]:

1. **Consensus**: To ensure the validity of any transaction, all members must admit on its validation.
2. **Provenance**: Members should aware from where they came and how its ownership change by time to time.
3. **Immutability**: When a transaction is stored to the ledger, no member can change the transaction. If a transaction is appear with an error, a new transaction should be utilized to reverse the error, and then both transactions can be visible to the ledger.
4. **Finality**: From a single shared ledger one can find the ownership of an assset and can determine either the transaction is completed or not.

2.1.2 Architecture of Blockchain

The blockchain is a grouping of blocks, which like conventional public ledger consists a complete history of transaction records [3]. Figure 2.1 shows an example of a blockchain where Blockhead of the blockchain contains the value of previous block hash, and possess only one parent block that parent block is known as Genesis Block. It is important to note that ethereum blockchain [14] also stores the children of the blocks ancestors hashes. Following are the details of the internal blockchain.

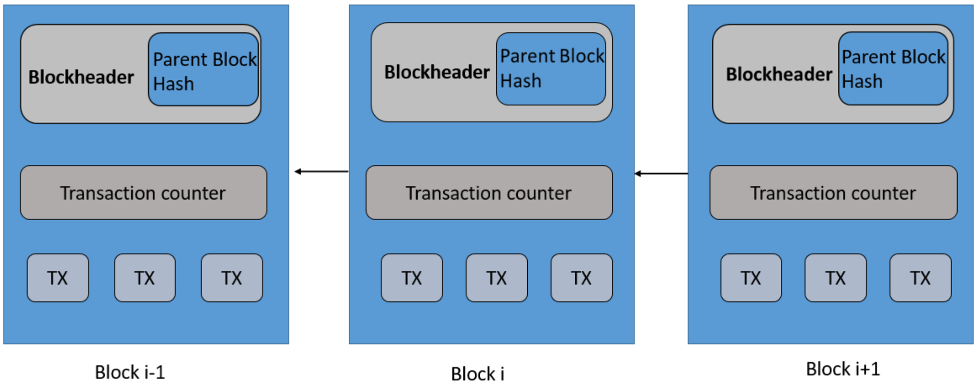

Figure 2.1: An illustration of Blockchain which consists of continuous grouping of blocks.

Block:

A block is made-up of block-header and block body as illustrated in Figure 2.2. Specifically, the block-header contains:

1. **Block version**:
   shows that which set of rules to follow for the block validation.
2. **Merkle tree root hash**:
   Each block in the blockchain uses Merkle tree to contain the overview of all transactions in the block. To summing up and validating the integrity of greater amount of data efficiently, blockchain utilizes Merkle tree. In this situation, it behaves like a data structure. A binary Hash tree is another

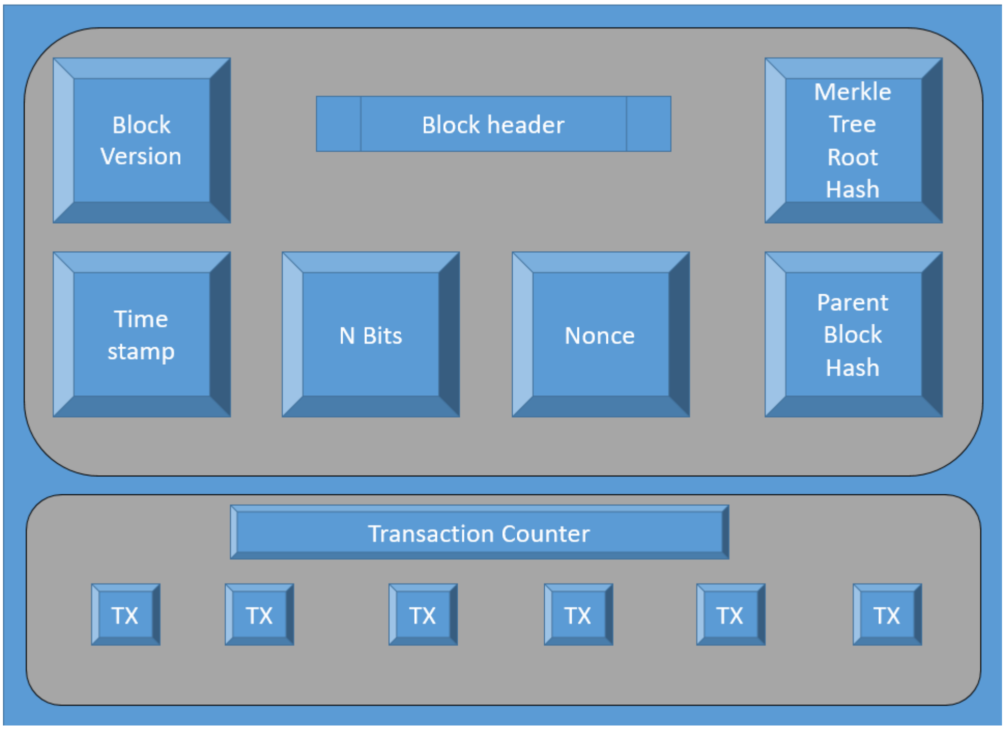

Figure 2.2: Block structure

name of the Merkle tree. Merkle trees act as binary trees which holds cryptographic hash functions. The word “tree” here is obtained from the area of computer science giving a detailed account of the branching data structure. For entire set of transactions Merkle trees build all-inclusive digital fingerprint. A “Merkle tree” is generated by repetitive hashing pairs of nodes until obtaining one hash only, this last hash is called as “Merkle root”. The double-SHA 256, the cryptographic hash algorithm is applied in the symbolic Bitcoins Merkle trees. SHA is a secure hash algorithm, which is a secure set of cryptographic hash functions, in the SHA-2 family.
When N record elements are hashed and summarized in a Merkle tree, you can examine to look if a specific element is added in the tree with at most $2^{*}log_2(N)$ number of computations, which gives a very effective way to validate either transaction is added in a block or not [48].

The “Merkle tree” is created bottom-up. In Figure 2.3, we start with four transactions; denoted asTxA, TxB, TxC, TxD. In “Merkle tree”, these transactions are not recorded, instead of this their information is hashed and the outsourced hash is recorded in every leaf node as $H_A, H_B, H_C, andH_D$.

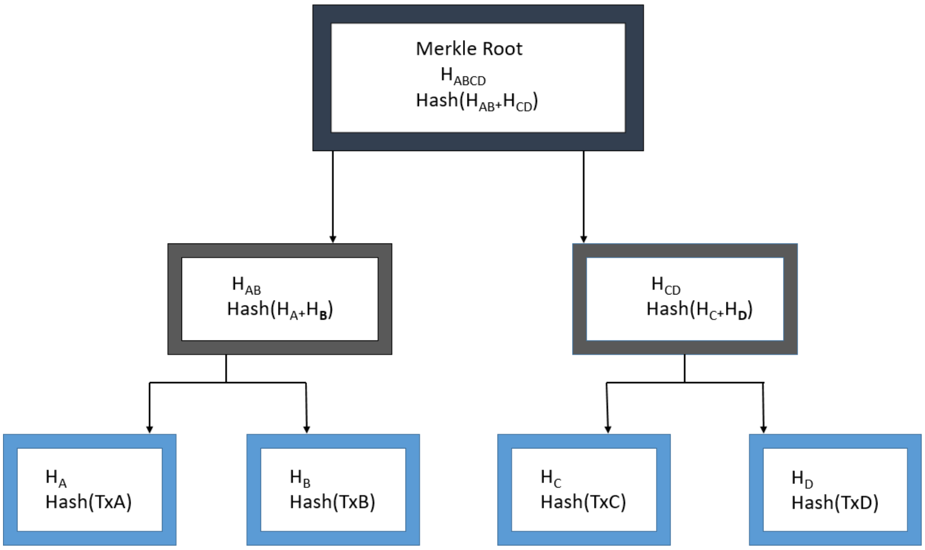

Figure 2.3: Merkle tree visual illustration

The mathematical function for the derivation of H_A can be seen as $H_A = SHA256(SHA256(TransactionA))$, where transaction A has been cryptographically hashed twice using SHA256. Consecutive pairs of nodes are then merged in a parent node, by concatenating the two hashes and hashing them together. Following the example, to construct the parent node H_{AB}, the two 32-byte hashes of the children are concatenated to create a 64-byte string. That string is then double-hashed to produce the parent nodes hash:

$$H_{AB} = SHA256(SHA256(H_A + H_B))$$

3. **Timestamp**:

   The procedure of securely storing track of the construction and moderation of document time is known as "Timestamp". It permits concerned participants to have knowledge that certain document in question, without any doubt existed as a specific time and date.

4. **nBits**:

   "The target is the threshold below which a block header hash must be in order for the block to be valid, and nBits is the encoded form of the target threshold as it appears in the block header [38]."

5. **Nonce**:

   Nonces are an integral part of many cryptosystems, such as block cipher modes, and password security. But they found use in carrying economic signals for anti-spam as well. Nonces are essentially a counter, one that is constantly incremented, usually to change the outcome of a hash function when applied to both a payload (such as a series of transactions in a Merkle tree) and the nonce. This is, in effect, how Bitcoin mining [47] works, and with any other Hashcash-style proof of work. Incrementing a nonce and finding a hash below a certain number requires a lot of computing power, and this is referred to as work. Including a valid nonce as a solution to the block, problem rewards miners and makes blocks hard to change; pseudo-finality.

   Nonces are also used to order outgoing transactions. This is mainly used in Ethereum, where user account transactions all have a nonce attached that is one more than the previously confirmed transaction.

6. **Parent Block Hash**:

   “A 256-bit hash value that points to the previous block [38].”

The transactions check and transactions are contained by the “block body”. The “block size” and measurement of each transaction decide how many numbers of transactions block can contain. To verify the authentication of cryptography asymmetric cryptography is used in the blockchain. Digital signature used in blockchain is based on asymmetric cryptography.

Digital Signature

A “digital signature” is an approach to demonstrate that data is coming from an authentic entity, not from anybody else like a hacker.

In Public key encryption framework, users create something to refer to as a key pair, by using some known algorithm. One key is Public key and another is known as a secret key. Both keys are related to each other by an algorithm. The Public key is open to everyone and user can use it as an address to accept a message just like home address and email address. The secret key is stayed confidential and is utilized to digitally sign those communications which are sent to other users. In order to give utility to the receiver to validate the communication using the sender public key "digital signature" is incorporated in the communication as shown in Figure 2.4.

In such a way, the receiver can ensure that authentic sender has delivered the relevant data. To create two keys is as same as to generate an account on the blockchain, but to create two keys user don't need to register anywhere. In the blockchain, sender digitally signed each transaction by utilizing their secret key which is executed on the blockchain. "Elliptic curve digital signature algorithm (ECDSA)" is the generally digital signature algorithm utilized in the blockchain [34].

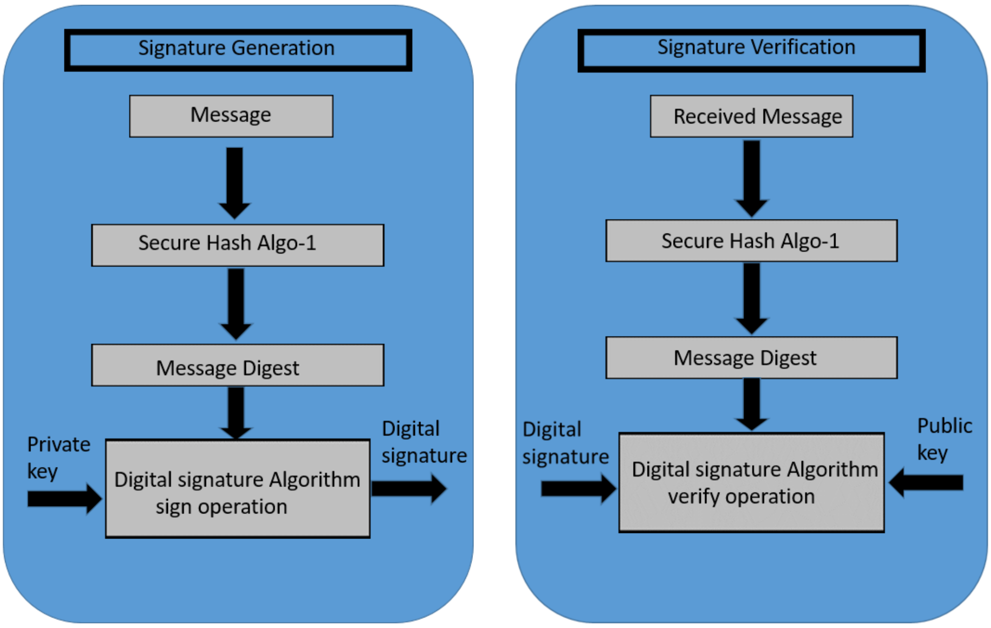

Figure 2.4: Digital signature

Smart Contracts

"A smart contract is an agreement or set of rules that govern a business transaction; its stored on the blockchain and is executed automatically as part of a transaction. Smart contracts may have many contractual clauses that could be made partially or fully self-executing, self-enforcing, or both. Their purpose is to provide security superior to traditional contract law while reducing the costs and delays associated with traditional contracts.For example, a smart contract may defne contractual conditions under which corporate bond transfer occurs or it may encapsulate the terms and conditions of travel insurance, which may be executed automatically when, for example, a flight is delayed by more than six hours [38]."

2.2 Classification of Blockchain System

Recent Blockchain system is classified roughly into three categories: Permissionless blockchain, Permissioned blockchain, and consortium blockchain [13]. In permissionless blockchain also known as Public blockchain, everyone can see all the records and could take part in the consensus process. While in consortium blockchain, the only a number of pre-selected nodes could participate in the consensus process. In case of permissioned blockchain also called as Private blockchain, only those nodes could be permitted to participate in the consensus process who came from one specific organization.

Since one organization manages the permissioned blockchain, So it is considered a centralized network. The consortium blockchain is "partially decentralized" because only some number of nodes are allowed to decide consensus and this type of blockchain is constructed by several organizations.

Hashing and digital signature are main components of the blockchain. Everyone on the blockchain admits the recent world state through hashing and digital signature. Everyone can confirm that all the transactions are coming from authentic owners. We depend on the hashing and digital signature to confirm that the blockchain has not been manipulated.

The comparison between the three types of blockchain is given in Table 2.1.

1. **Consensus Determination:** In permissioned blockchain, everyone can see all the records and could take part in the consensus process. While in consortium blockchain, only those nodes are permitted to participate in the consensus process who belongs to a specific organization.

2. **Read Permission:** In permissionless blockchain transactions are visible to everyone but in the case of permission blockchain and consortium blockchain, it depends on the current situation where we are using these types of the blockchain.

3. **Immutability:** In a permissionless blockchain, records are stored in a large number of participant, so it is almost impossible to tamper transactions.while in permissioned blockchain and consortium blockchain, only limited participants are there so their transactions could tamper.

4. **Efficiency:** Permissionless blockchain network needs a lot of time to broadcast, because of a large number of nodes transactions and blocks. Subsequently, latency is high and transactions turn out to be limited. Consortium blockchain and permissionless blockchain could be more efficient with a limited number of nodes.

5. **Centralized:** Centralization is the main difference between the three types of blockchains. Permissioned blockchain is completely decentralized while consortium blockchain is partially centralized. Since permissioned blockchain is controlled by a single organization, So it is fully centralized.

6. **Consensus Process:** In the case of the permissionless blockchain, anyone who wants to join a consensus process could join it. As compared to the passionless blockchain, Only selected number of nodes could participate in both permissioned and consortium blockchain.

   since anyone can join permissionless blockchain so numerous permissionless blockchain rises gradually. “Consortium blockchain” have many applications in the business industry. Hyperledger [32] is one of the emerging business application of “Consortium blockchain”. Ethereum has also given a mechanism to build “Consortium blockchain” [46].

Table 2.1: Comparison between public blockchain,consortium blockchain and private blockchain

Poperty	**Public blockchain**	**Consortium blockchain**	**Private blockchain**
Consensus determination	All miners	Selected set of nodes	One organization
Read permission	Public	public or restricted	public or restricted
Immutability	Nearly impossible to tamper	could be tampered	could be tampered
Efficiency	low	High	High
Centralized	No	partial	Yes
Consensus process	Permissionless	Permissioned	Permissioned

2.2.1 Consensus Algorithms:

In any distributed environment to reach a consensus is a challenge. Since blockchain is a distributed network so it is also a challenge for a blockchain. There is no central node in the blockchain that guarantee records on distributed nodes are all equivalent. We require some protocols to make sure the similarity of ledgers in various nodes.

This section will focus on how the nodes in the blockchain network agree with one another about the ledger that they hold. Also we have given general techniques to have a "consensus" in blockchain.

a: Proof based Consensus Algorithm

The first Proof based consensus algorithm was Proof of work (PoW) proposed by Nakamoto [47]. Until now, many other forms of proof-based consensus algorithms have been proposed and that all are based on PoW, Proof of stake (PoS) and their hybrid form. The main idea of proof-based consensus algorithm is that the node in the blockchain network who performs sufficient proof will have the authority to add new block to the chain and in returns get the reward on performing this proof.

b: Original Proof of work

In the blockchain network, confussion will arise if each of the node attempts to broadcast their blocks having the verified transactions. For instance, if many nodes verifies a transaction then place it to their blocks and broadcast it on the whole network. If the broadcasting work is free, then this transaction could be duplicated in different blocks which makes the ledger useless. The PoW requires every node to solve a difficult puzzle with adjusted difficulty, to get the right to append the new block to the current chain. The first node who solves the puzzle will have this right. In this way, all nodes will get the agreement about the newly added block. In particular, all the verifying nodes would need to put their transactions as well as other information like prev-Hash and Time-stamp into a block, before solving this puzzle. Then by guessing a secret value, which is the nonce field as defined earlier these nodes start solving this puzzle and put it into the block. The information contained in the block header will be combined together and put into an SHA-256 hash function [35]. If the output value of this function is below a defined threshold T, the secret value is accepted. Otherwise, the node has to make another guess of the secret value, until this node gets the answer. The difficulty of the puzzle will be adjusted after every 2016 blocks are appended so that the average speed for adding a new block in the chain is 1 block per 10 minutes. Also, the more difficult the puzzle is, the smaller the threshold T is. Figure 2.5 describes the processing for handling the guessed value. Thanks to the usage of SHA-256, guessing this value is extremely difficult, which makes every node guess many times to get the answer unless they are lucky enough. Because of the efforts paid for guessing the right value, this work is called the PoW. Also, the node joining the network using PoW can be called a miner, and the action of finding a suitable nonce is called mining.

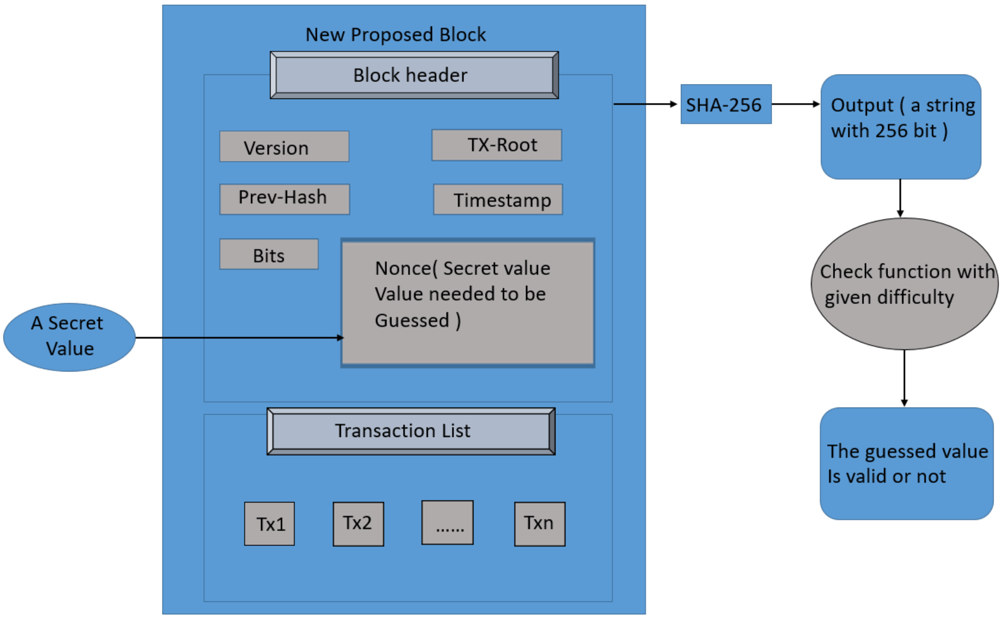
New Proposed Block
Block header
Version
TX-Root
Prev-Hash
Timestamp
Bits
Nonce(Secret value Value needed to be Guessed)
A Secret Value
Transaction List
Tx1
Tx2
......
Txn
SHA-256
Output (a string with 256 bit)
Check function with given difficulty
The guessed value Is valid or not

Figure 2.5: handling Nonce process:guessed secret value

c: Proof of Stake (PoS)

It is one of two famous consensus validation algorithm for validating blockchain transactions. In terms of cryptography, the term stake refers to the cryptocurrency which is owned by the user and promise to participate in validation. In proof of stake mechanism, network nodes (called as miners) invest cryptocurrency in the blockchain network, demonstrating their stake in the block. A miner chance of validating a block depends on its stake in the block [45].
Recently, "Max Thake" has defined "What is Proof of Stake" in his article as:
"Proof-of-Stake algorithms achieve consensus by requiring users to stake a number of their tokens so as to have a chance of being selected to validate blocks of transactions, and get rewarded for doing so [57]."
PoS is more effective than PoW. Most of the blockchain has used PoW as a consensus process at the start and now they are trying to adopt PoS gradually. For example, ethereum network is moving from Ethash (a type of PoW) [60] to Casper (a type of PoS) [8].

d: Practical Byzantine Fault Tolerance (PBFT)

PBFT algorithm is similar to tolerate Byzantine faults(By definition, Byzantine Fault Tolerance means a network can continue to function correctly even if some nodes are dishonest and attempt to propose invalid blocks, or blocks that benefit certain parties at the expense of others) [17]. Since PBFT could manage prior to 1/3 malicious byzantine duplicates, therefore Hyperledger Fabric [32] uses the PBFT as its "consensus algorithm". The entire process of PBFT have three phases: pre-prepared, prepared, commit. In every phase, if a node has been given votes from over 2/3 of all nodes, then it would allow entering the next phase. So in PBFT, every node needs to be known in the system [43].

e: Delegated Proof of Stake (DPoS)

PoS and DPoS are different from each other. PoS is “direct democratic” on the other hand DPoS is “representative democratic”. This is the main difference between both of them. Stake owner selects their representative to create and verify blocks. With remarkably lesser nodes to verify the block can be approved rapidly, which leads to rapidly approve the transactions. In the meantime, block size and block intervals could be adjusted by the representatives. Moreover, the corrupt representatives could be voted out easily. DPoS works in Bitshares as its backbone [3].

f: Ripple

In Ripple [51] consensus algorithm, a greater network uses overall trusted subnetworks within it. Nodes in Ripple have two types in the network: one type is a server which partakes in consensus process and the other type is a client for only transferring assets. Every server posses a unique node list (UNL) which is very significant to the server. The server would ask for the consensus to the nodes in UNL when they need to determine whether to include transaction into the ledger or not. If the server has given 80 percent of agreements, the transactions would then be included into the ledger. The correctness of the ledger will remain to exist if the percentage of faulty nodes in UNL do not exceed 20 percent.

g: Tendermint

“Tendermint [37] is a byzantine consensus algorithm. A new block is determined in a round. A proposer would be selected to broadcast an unconfirmed block in this round. It could be divided into three steps:

1. **Prevote Step:** Validators choose whether to broadcast a prevote for the proposed block.

2. **Precommit Step:** If the node has received more than 2/3 of prevotes on the proposed block, it broadcasts a precommit for that block. If the node has received over 2/3 of precommits, it enters the commit step.
3. **Commit Step:** The node validates the block and broadcasts a commit for that block. if the node has received 2/3 of the commits, it accepts the block. In Contrast to PBFT, nodes have to lock their coins to become validators. Once a validator is found to be dishonest, it would be punished."

h: Quorumchain

Quorum uses a majority voting protocol dubbed QuorumChain, where a subset of nodes within the network has authority to vote on blocks. The voting role allows a node to vote on which block should be the canonical head at a particular height. The block with the most votes will win and is considered the canonical head of the chain. Block creation is only allowed by nodes with the maker role. A node with this role can create a block, and in doing so, will sign it such that, on block import, other nodes can verify that the block was signed by one of the nodes that have permission to make blocks.
"QuorumChain" used a smart contract which manages consensus, and importantly, the consensus-upgrade process. The "smart contract" are able to tracks voter and block maker lists, both of which can be maintained through standard transactions, thereby providing further control and clarity over how and by whom the network is managed.

2.3 Cryptography

"Cryptography's aim is to construct [19] schemes or protocols that can still accomplish certain tasks even in the presence of an adversary. A basic task in cryptography is to enable users to communicate securely over an insecure channel

in a way that guarantees their transmissions, privacy, and authentication."
A cryptographic system consists of following five components;

1. **Plaintext**: The original message or data which sender converts into coded message.
2. **Encryption**: The conversion of plaintext into ciphertext is known as Encryption.
3. **Keys**: The process of encryption and decryption relies on some parameters known as keys.
4. **Decryption** : The conversion of ciphertext back into plaintext is the process of Decryption.
5. **ciphertext**: The coded or scrambled message converted by the sender from the original message.

Cryptosystem can be categories as:

1. **Symmetric key cryptosystem**
2. **Asymmetric key cryptosystem**

2.3.1 Symmetric key Cryptosystem

"Alice and Bob can share the same key K, unknown to the attacker, and use it to encrypt and decrypt their communication. The shared key is usually a uniformly distributed, random string of k bits for some parameter k. Alice can apply an encryption algorithm to the plaintext M under the key K to get a ciphertext C. This ciphertext is then sent to Bob, who applies the corresponding decryption algorithm to recover the plaintext M. This is the symmetric encryption setting, in which users share the same key K [30]."
Symmetric key cryptography has following drawbacks;

Sharing Key: For n number of members who communicate with each other, the distribution of key is an issue. If at least one member reveals the key then the whole communication will be in danger.

Authentication: Authentication is one of the issue. If two person usually referred to as Alice and Bob communicate with each other then how can Alice will prove that the message is coming from Bob.

2.3.2 Asymmetric key Cryptosystem

In 1976, Diffie-Welman [22] gives the idea of Asymmetric cryptography to resolve the problem with symmetric key cryptography. To interchange the key between two parties they used the concept of one way trapdoor function. In the public-key cryptography [54] (or asymmetric cryptography), two different keys are used for encryption and decryption. Essentially, a party is the owner of two keys one is “Public key (PK)”, and the other associated key is “secret key (SK)”. The public key is used for encryption which is open to all while the other one is used for decryption which is kept hidden.

Before the development of Public Key Cryptography (PKC), virtually all cryptosystems were based on permutation and substitution. But the public key cryptosystem used mathematical function rather than substitution and permutation. The asymmetric encryption scheme utilizes six main components as shown in 2.6. To get the ciphertext CT, the communicant who sends the message encrypts the plaintext M by using receivers public key PK' and an encryption algorithm E'. then receiver utilizes his secret key SK' (that is only known to him and decrypts the ciphertext using corresponding decryption algorithm D'.

Thus,

$$CT = E'(PK', M) \tag{2.1}$$

$$M = D'(SK', CT) \tag{2.2}$$

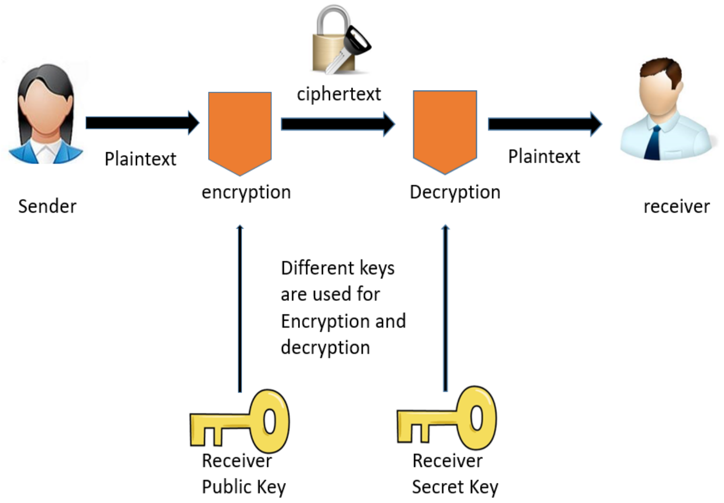

Figure 2.6: Asymmetric Cryptography

2.4 Key Management Issues

In assymetric cryptography, the main issue is the distribution of public key regarding the key management. For the distribution of public key many methods have been proposed. Some of them are as follow.

- Public Announcement
- Public Available Directory
- Public key Authority
- Public Key Certificate

2.4.1 Public Announcement

One of the major issue faced by the Asymmetric cryptography is that public key must be known to everyone. There are algorithms utilized PGP(Pretty Good Privacy) [29] in which any communicant, sends its public key to other communicants via email or make public announcement [54] as shown in Figure 2.7.

The main weakness of public announcement is forgery. Anyone could play a role of user A and send his public key to B. In this way, forger can have access to all encrypted messages.

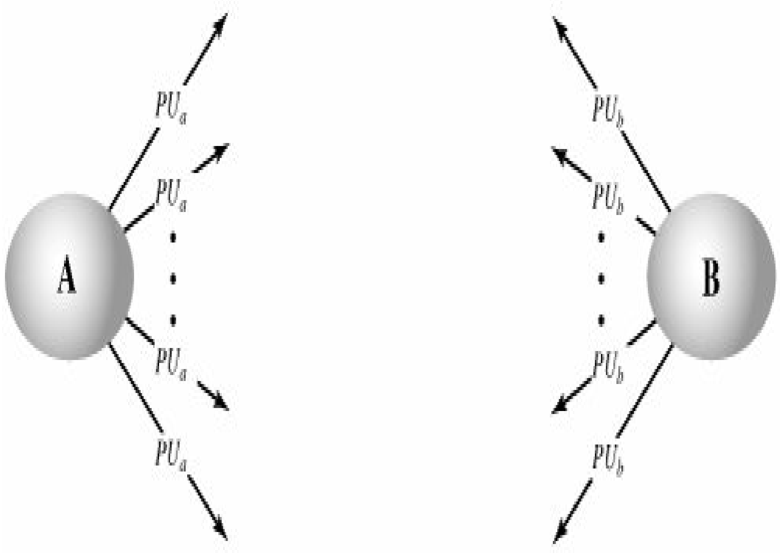

Figure 2.7: Public Announcement [54]

2.4.2 Public Available Directory

One can attain the greater security by public available directory [54]. For the maintenance and distribution of public keys, the trusted authority or system would be responsible as shown in Figure 2.8. Following are the main features of the system.

1. The authority will be responsible for the maintaining records of the "name" and "public" key of every recipient.
2. Any entity can enroll his public key with the certification authority. enrollment would be in the form of secure communication.
3. If the private key is compromised, then participant can replace his existing key with a new one at any time.
4. The participant can access the directory electronically. For this purpose, it is mandatory for participants to communicate with authority securely.

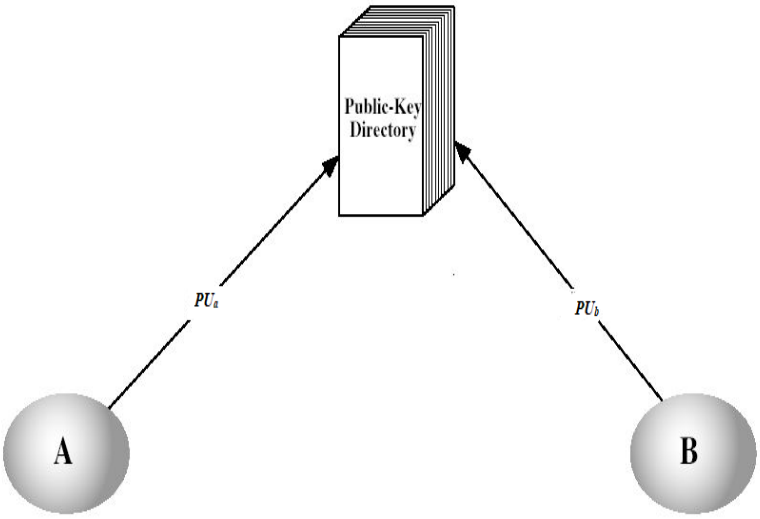

Figure 2.8: Public Available Directory [54]

2.4.3 Public Key Authority

Greater security can be achieved by tightening control over the central authority or directory. In this scheme [2], the public key authority, is employed to maintain the directory of public key of all recipients. Therefore, all participants reliably know the public key from central authority, with only authority knowing their corresponding private key. The following are steps as presented in Figure 2.9.

1. A sends time stamped request to central authority for B's recent "public key".
2. "Authority" encrypt message with his private key (PR_{auth}). The message of authority contains the B's public key PU_b, original request and time stamped as in equation 2.3. So, in this way A can verify that this is not the old message containing B's public key.

$$E(PR_{auth}, [PU_b||Request||Time]) \tag{2.3}$$

3. A keeps B's "public key" and utilizes it message encryption that contain A's identity (ID_A) and nonce (N_1) generated by A as describe in 2.4

$$E(PU_b, [ID_a||N_1]) \tag{2.4}$$

4. The same procedure is repeated by B for obtaining A's public key (PU_a) as described in (1) and (2).
5. When a message is delivered from B to A he encrypts message with A's public key (PU_a), and with random number (N_1) this can be used to verify the original message generated by A and another random number (N_2) generated by B as described in equation 2.5

$$E(PU_a, [N_1||N_2]) \tag{2.5}$$

6. A returns the random number N_2 by using B's public key PU_b to ensure that the original message is sent by B.

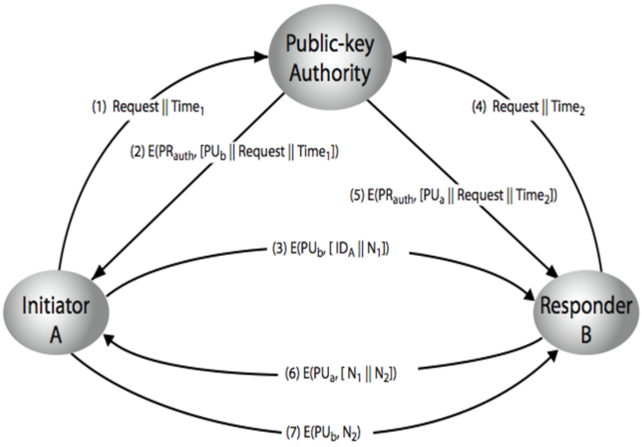
Public-key Authority
(1) Request || Time$_1$
(2) E(PR$_{auth}$, [PU$_b$ || Request || Time$_1$])
(4) Request || Time$_2$
(5) E(PR$_{auth}$, [PU$_a$ || Request || Time$_2$])
(3) E(PU$_b$, [ID$_A$ || N$_1$])
Initiator A
Responder B
(6) E(PU$_a$, [N$_1$ || N$_2$])
(7) E(PU$_b$, N$_2$)

Figure 2.9: Public Key Authority [54]

2.4.4 Public Key Certificate

Although, public key authority (PKA) is an efficient scheme, but it possesses some disadvantages. The "public key authority" could be a greater threat to a system because user has to obtain "public key" from authority to contact with other users. If some adversary had broken the "public key authority", then the whole system will be compromised. Even without breaking the "public key authority", some imprisonment is also possible by tampering the record of directory that is maintained by the public key authority. Furthermore, the use of public key authority frequently needs a large and complex system and it is really difficult to update such a system securely.

Therefore, the concept of public key certificate (PKC) had been introduced by Felder [36] to use certificate for communication without contacting the public key authority. The certificate is the signed message that contains a "public key" as well as "identity" of the owner and the entire blog is inscribed by the mediator. Generally, this mediator is "certificate authority". Note that Figure 2.10 shows the certificate scheme, in which both recipients A and B supply their public keys PU to certificate authority and requesting for certificate. Certificate authority (CA) issues certificate for both recipients by using their private keys PR. So, A may pass their certificate to B, and B reads and verifies it by using authority's public key PR_{auth} and certificate C_A.
There are some benefits of certification which are stated as under:

1. Any entity can have access to a certificate and can find out the certificate's owner "name" and "public key".
2. Any entity can validate the certificate that is created by Certificate.
3. Atmost "certificate authority" could create, modify and manage the certificate.
4. The entity could also validate each certificate's time period.

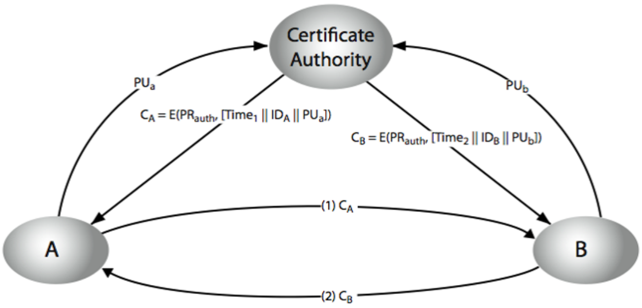
Certificate Authority
PU$_a$
PU$_b$
$C_A = E(PR_{auth}, [Time_1 || ID_A || PU_a])$
$C_B = E(PR_{auth}, [Time_2 || ID_B || PU_b])$
(1) C_A
(2) C_B
A
B

Figure 2.10: Public Key Certificate [54]

2.4.4.1 Drawbacks of Certificate Authority

Although, the public key certificate [1] is a very efficient scheme, but it has some drawbacks.

1. When user A wants to communicate with user B, both recipients need a certificate in order to communicate with each other. For offline operations, a certificate is required in order to communicate with each other. So for that purpose, large scale directory is needed for managing the certificates for offline use.
2. Certificates are large and complex structure so it is hard to update such a system securely.
3. Since the certificate keeps all public and private keys, therefore, these are large and very expensive schemes.
4. The authority does not give warning when it changes the certificate.
5. A user blindly trusts on certificate authority, if some third party generates the fraudulent certificate and gains access to someone's personal computer. So, in this way certificate authority does not give warning when any site uses the fraudulent certificate.
6. In PKI (Public Key Infrastructure) before the communication takes place the system must register its encryption and signature key to CA, then CA issues the certificate for the proof of its identity. Then this certificate is used by recipient for secure communication (Figure 2.11). Therefore, this method is also time-consuming.

2.5 Introduction to Identity Based Encryption Scheme

To solve the certificate management system, Shamir [52] introduced a new scheme called as Identity Based Encryption Scheme (IBE) in 1984. IBE is a very efficient scheme and currently active in research area of cryptography. This scheme uses an

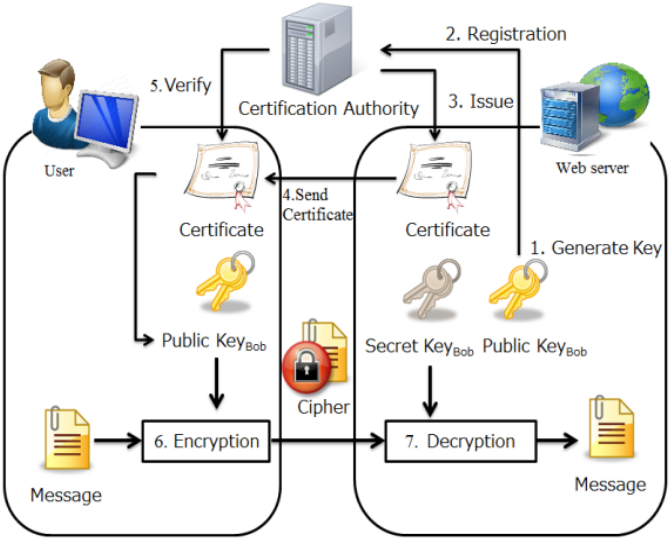

Figure 2.11: Public Key Infrastructure

arbitrary string such as a user's identity, email address or IP address and derives the public key from it. The direct derivation of public key eliminates the role of the certificate. Only private keys are generated from trusted third party also called Private Key Generator (PKG). So, in this way, large directories are not required for managing public keys of users. In IBE, the private key authority exists only, it does not need to be online, its action replaces with mathematical pairing. Note that in Figure 2.12 when Alice sends message to Bob she must contact to certificate authority for Bob's certificate CA look up Bob's certificate from certificate server and send certificate to Alice. From certificate, Alice uses the public key of Bob, PU_{Bob}, and apply the encryption by using PU_{Bob}. When Bob receives encrypted message he sends his public key to CA and receives the certificate that includes his private key. Bob decrypt the message by using his private key. Where in Figure 2.13 shows that IBE does not need certificate server for keeping the record of recipient's public key. No certificate lookup required. IBE need only the private key generator for deriving the private keys by using recipient's identity.

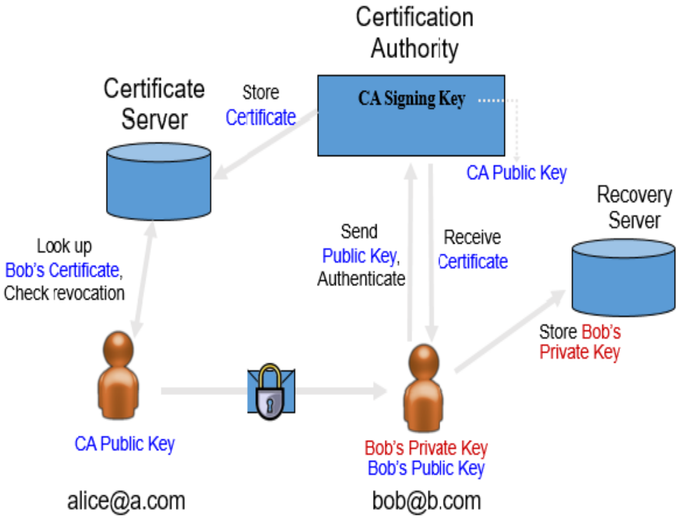

Figure 2.12: Public key Certificate

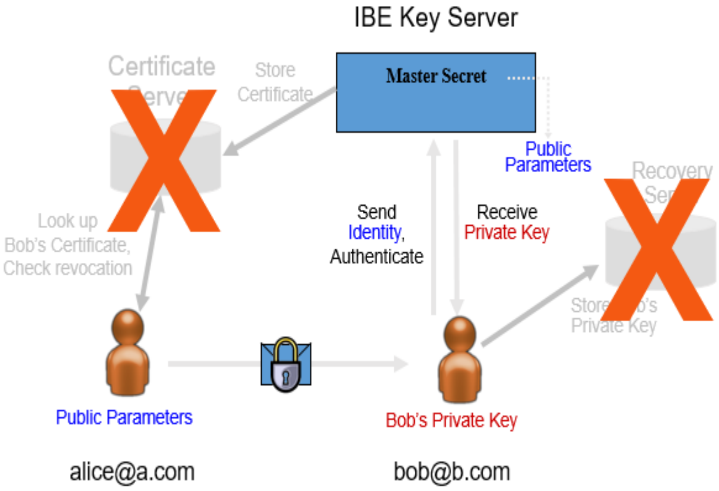

Figure 2.13: Identity Based Encryption Scheme

2.5.1 Identity Based Encryption Scheme

As discussed in previous section, identity-based encryption scheme (IBE) was first proposed by Shamir [52] in 1984. In this scheme, the pairs of users can communicate and verify each other without sharing their public and private keys, without keeping key directories and without taking the services of third parties. In IBE, the third party is used to generate the private keys in the shape of smart cards when users first connect the network.

IBE scheme is based on public key cryptosystem but holds some extra key points. Instead of generating the random public keys by using the help of a third party, IBE scheme uses any combination of a user's name, IP address, telephone or office number etc. as a public key. IBE scheme resembles the mail services: if one user knows someone's e-mail address then he will be able to communicate with that user.

Identity based encryption scheme works as follows:

1. User A wants to communicate with B, he signs it with his secret key in his smart card. He encrypts the message by using B's identity (B's name, address etc.) and sends it to B.
2. When B receives the encrypted message, he contacts to the third party for obtaining private key (PR_b).
3. B decrypt the message by using his (PR_b) in smart card or verify the message by user's A identity

Here the third party or key generation center is the trusted party that generates the secret keys of all users. Centre knows some secret information (such as factorization of large numbers). The secret key is issued in the shape of smart cards to all users who join the mesh. The smart card contains a microprocessor, RAM, ROM that contains secret key and the program that contains the message encryption and decryption algorithm. The query is how user can secure his smart card? The user must secure his smart card by using password system or memorizing the part of the key.

The Figure 2.14 show the system of IBE. Shamir's IBE consists of four algorithms.

1. **Setup:** The setup is the component of a "private key generator (PKG)". PKG creates the "master key $\mathcal{K}$" and "public parameter $\mathcal{PP}$". Where master key is kept secret. Public parameter contains the information about "message space $\mathcal{M}'$" and "ciphertext space $\mathcal{C}'$".

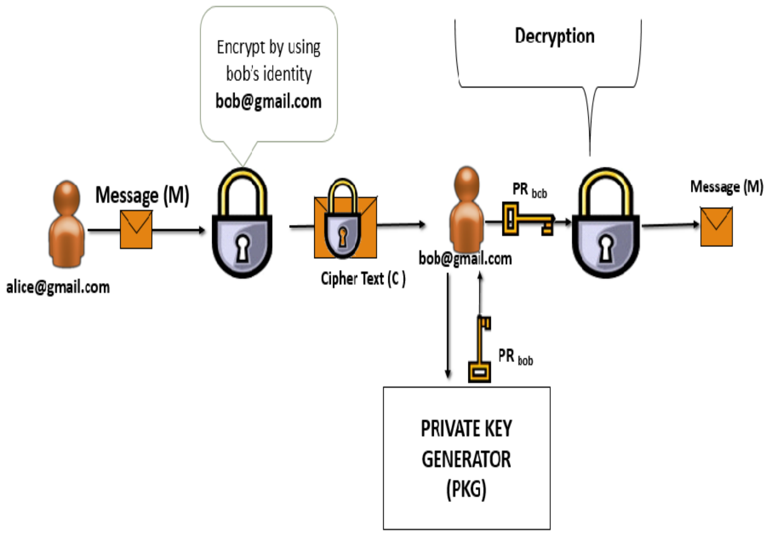

Figure 2.14: Identity Based Encryption Scheme

2. **Extract**: The PKG runs this extract algorithm, makes session keys or private key for user using his master key and user's identity (ID). This algorithm accepts the identity (ID) of user and master key $\mathcal{K}$ generates private key (S_{ID}) of corresponding identity (ID).
3. **Encrypt**: This algorithm accepts identity (ID) and message as input and produce ciphertext as output.

$$C = E(M, ID)$$

4. **Decrypt**: This algorithm takes ciphertext (C) and private key (S_{ID}) as input and returns messages.

$$M = D(C, S_{ID})$$

There have been several proposals for IBE see [21, 42, 56, 58], but none of these are fully acceptable. Some solutions take a lot of time in generating the secret key from "private key generation (PKG)". The first successful scheme was presented by Boneh and Franklin [9] in 2001. Their scheme is based on bilinear maps defined on prime order groups.

2.5.2 Revocable Identity based Proxy Re-encryption

In “Proxy re-encryption (PRE)” a semi-trusted proxy transform original ciphertext of Alice into the Ciphertext for Bob by encrypting the original ciphertext. The proxy only has the re-encryption key which is delivered by the Alice to proxy and have access to the plaintext encrypted. This has many applications in different fields, for example, confidential email, digital right management, and distributed storage are some of the applications of PRE.

“An Identity Based Proxy Re-encryption Scheme (IB-PRES)” is an extension of “ Identity-Based Encryption scheme”.In the foremost additional algorithm “re-encryption keys” are created which are then sent to the proxy. In the second algorithm, proxy use these “re-encryption keys” to re-encrypt the ciphertext and change the original ciphertext from one identity to another. Our secure “revocable identity-based proxy re-encryption scheme” uses four main entities as illustrated in Figure 2.15 “the private key generator (PKG), the proxy server (PS), the data owner (I) and the requester (R)”. We can not fully trust PKG for data security which generates secret keys for users. Since it is sincere but at the same time unexpected. Therefore, in our secure “IB-PRES” the PKG only creates user’s partial secret keys which confirms that the data of user is confidential and secure. The encryption of data by utilizing the identity as a public key is done by the data owner which then deliver it to the proxy server. The proxy server keeps the ciphertexts, re-encrypts them and then deliver these re-encrypted ciphertexts to the requester who has access permission. The owner of the data authenticates the requester, creates the “re-encryption keys” and deliver them to PS. The requester who have the “access permission” can do the decryption of “re-encrypted ciphertext” [41].

The main key points of “IB-PRES” are given as follow:

1. In IB-PRES, to confirm the confidentiality of user’s data and privacy security, the PKG only creates partial secret keys. Apart from this, the PKG does not partake in the creation of re-encryption keys.

2. The process of secret key creation and data access give the authentication. It confirms that the only validated users can have access to the data they desire and that data cannot be hacked.

3. The secret key of data owner could not be accessed by anyone, even if the allocated decryptor colludes with the proxy server.

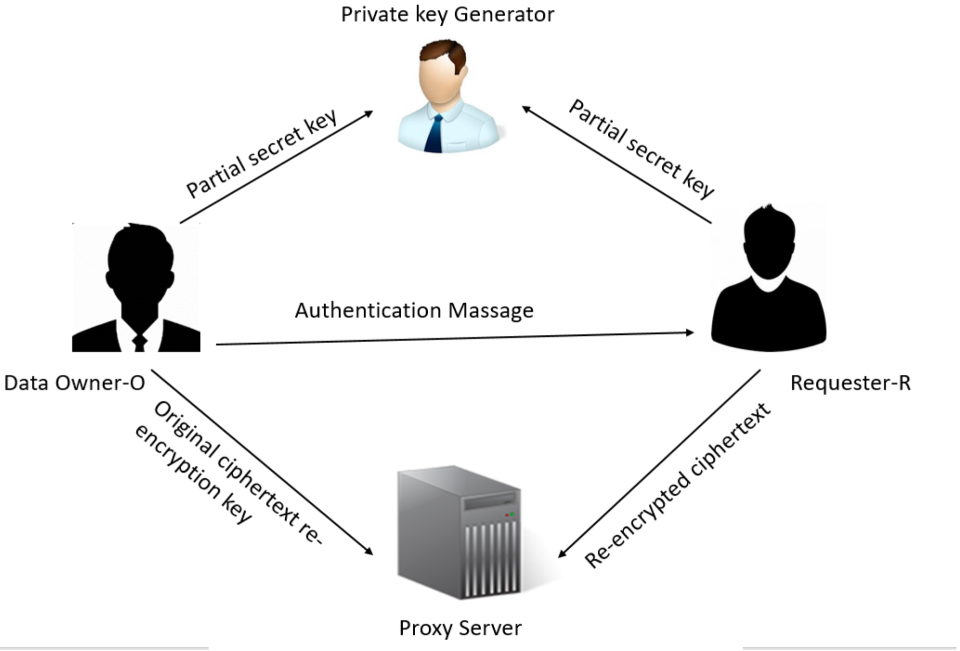

Figure 2.15: System Model

2.6 Mathematical Background

Before introducing our framework, we first recall some definition from algebra that will be used through the thesis.

Definition 2.6.1 (Groups)

The ***group*** [50] $\mathbb{G}$ denoted by $(\mathbb{G}, *)$ is the set of element under the binary operation * that satisfies the following properties:

1. **Closure:** For all $x, y \in \mathbb{G}$, $x * y \in \mathbb{G}$

2. **Associative:** For all $x, y, z \in \mathbb{G}$ satisfies $(x * y) * z = x * (y * z)$

3. **Identity:** There exist an element $i \in \mathbb{G}$ that satisfies $x * i = i * x = x \quad \forall x \in \mathbb{G}$. i is called the identity of $\mathbb{G}$.

4. **Inverse:** For each element $x \in \mathbb{G} \;\; \exists \; x' \in \mathbb{G}$ the satisfies $x * x' = x' * x = i$. Where i is the identity element of $\mathbb{G}$

Example 2.6.2 Following are the examples of groups.

1. Set of integers $\mathbb{Z}$, real number $\mathbb{R}$, rational number $\mathbb{Q}$, complex number $\mathbb{C}$ are all group under binary operation addition $+$.

2. Set of real numbers $\mathbb{R} \setminus \{0\}$, rational number $\mathbb{Q} \setminus \{0\}$ and complex number $\mathbb{C} \setminus \{0\}$ all group under binary operation multiplication ×.

3. Let $\mathbb{Z}_m = \{0, 1, 2, ...m - 1\}$ and $m > 0$ and $m \in \mathbb{Z}$ is group under addition $x * y = x + y$ where $x + y < m$. The binary operation $+$ is called addition modulo m.

4. Set of integers $\mathbb{Z}$ does not form a group under multiplication because multiplicative inverse does not exist (Inverse of 2 is $\frac{1}{2}$ but $\frac{1}{2} \notin \mathbb{Z}$)

Definition 2.6.3 (Abelian Group)

The group $\mathbb{G}$ is said to be abelian group [50] if it satisfies commutative law i.e. for all $x, y \in \mathbb{G}$ we have $x * y = y * x$.

Example 2.6.4 Following are the example of abelian groups.

1. Sets $\mathbb{Z},\ \mathbb{R},\ \mathbb{C},\ \mathbb{Q}$ are abelian group under addition.

2. Sets $\mathbb{R} \setminus \{0\}$, $\mathbb{Q} \setminus \{0\}$, $\mathbb{C} \setminus \{0\}$ form abelian groups w.r.t multiplication.

3. **General Linear Group** is defined as $GL(m) = \{A \in M(m, m) | det(A) \neq 0\}$ where $M(m, m)$ is matrix of order $m \times m$ is a group under multiplication. It is not an abelian group because matrix multiplication is not commutative.

Definition 2.6.5 (Generator)

The generator g is a group element that is capable to generate all group elements.

Definition 2.6.6 (Cyclic Group)

The finite group $\mathbb{G}$ of order n is known as cyclic if $\exists \; g \in \mathbb{G}$ which generates all elements of $\mathbb{G}$. That is,

$$\mathbb{G} = \{g, g^2, g^3, \ldots, g^n = O\}$$

Where O is the identity element of group $\mathbb{G}$ where g is the generator of $\mathbb{G}$.

2.7 Cryptanalysis

The cryptanalysis is the branch of cryptology that tries to break the ciphertext and cryptosystem. In this process, attacker tries to find the weakness in the cryptosystem and decipher the ciphertext without knowing the secret key. Any attempt to break the cryptosystem is known as attacker. There exist various cryptanalysis attacks but here we will focus on chosen plaintext attack and chosen ciphertext attack.

2.7.1 Chosen Plaintext Attack

In chosen plaintext attacks, the attacker can select the plaintext by choosing the smaller block instead of choosing the big block of text and obtain the corresponding ciphertext. The main goal of attacker is to recover the secret key or break the secret ciphertext [16].

2.7.2 Chosen Ciphertext Attack

In chosen ciphertext attack, the hacker can select ciphertext and can find out the corresponding plaintext. He would be able to decrypt the ciphertext and then regenerate the resulting plaintext from system. In this way, he can obtain the secret key [16].

Chapter 3

Ancile Framework

Recently, Dagher *et al.* [20] has proposed a framework which is named as "Ancile" for access control and interoperability of "Electronic health records (EHR) using blockchain technology". This framework is based on "Ethereum blockchain" which utilizes smart contracts (discussed in section 2.1.2) to improve the functionality of the system. Moreover, Ancile provides patients with greater control over their medical records, specifically in transferring the records from one party to another party.

In this chapter, we will explain the main overview of this framework which includes a basic explanation of "Blockchain mining, Eth-calls, and internal transactions, proxy re-encryption, types of nodes" which are being used by the Ancile. We also define the important software components and smart contracts which confirms the integrity of Electronic Health Records (EHR) Database used in this framework and discuss the architecture of Ancile framework. In the end, we will give its performance analysis.

Before the Ancile framework, many other authors have used permissioned blockchian for Electronic Health Records (EHR). Ancile framework uses the same concepts of Ekblaw *et al.* [24] and Linn and Koo [40] particularly for access control and storage. Specifically for the storage Ekblaw *et al.* [24] and Linn and Koo [40] refrain from storing entire records on the blockchain. Ekblaw *et al.* [24] stores hashed pointers to medical records and permissions, while Linn and Koo [40] stores indices to records on the blockchain. By utilizing this type of ideas, the scalability of system can be increased.

3.1 Contributions of Ancile Framework

In this section, we will have a detail look at the basic contributions of Ancile framework. To increase privacy and interoperability Ancile includes several contributions. Before this proposed framework many other authors had proposed blockchain based EHR systems. Unlike previously proposed frameworks, the Ancile blockchain stores hashes of the data references while sending the actual query link information in a private transaction over HTTPS. JP Morgan's Quorum [18] had used private transactions for privacy but in his scheme, he had not used "proxy re-encryption", that is implemented by Ancile to securely transfer EHR. Moreover, one of the main benefits of using "proxy re-encryption" is that one can keep small encrypted data and keys straight on the blockchain, which make it easy to transfer the records, for instance, doctor's instructions to drugstore or any other third party. It also allows patients to remove access permission if they want because user doesn't require to store keys locally.

Secondly, the center of attention of Ancile framework is the patient ownership rights. In consequence, this framework suggests that data will only be known to patients without any exchange of currency. As in this framework, there is no money required for mining, therefore this design simply using the system for mining. They consider that "providers and governments" have already the motivation to protect the patient's medical data. Moreover, to handle the different responsibilities of "patients, providers and third party" on the blockchain for access control, Ancile uses functionality of smart contract. By using these smart contracts Ancile makes the stratification of roles easy which can be suitable for different requirements of users. For instance, in a case in which a patient is a miner while allowing the access control for parents or guardians, patient keeps the data ownership.

Finally, Ancile uses consensus algorithm for permssioned blockchain structure rather than proof of work. Because of this, validation increases when appending nodes to the system or eliminating users that are harmful to the system.

3.2 Technical Difficulties of the Blockchain

An EHR management system gets many benefits from "Blockchain technology". But on the other hand, there also exists some built-in restrictions on this technology. When 51% of miner nodes collide with each other then the rewriting of chain construction occur which is the basic restriction of blockchain [59, 60]. Therefore, the benefits of the decentralized systems can be achievable, if at least 50% miner nodes guarantee blockchain immutability.

Secondly, "permissioned blockchain" lessen the potential of a hacker to have accessibility of Protected Health Information (PHI), but it could not hide the transactions record. This permits nodes to manage inauspicious network analysis. A hacker may be able to find out the frequency of particular node with which it meets "a doctor or the providers or third parties" with which a particular node make relationship.

Lastly, Since blockchain act as a "distributed systems" therefore for their operations it needs high storage cost[59, 60]. As a consequence, a greater amount of data cannot be recorded on the blockchain effectively. Hence, during the use of blockchain for the purpose of access control and data integrity, the data itself should be recorded elsewhere and this data could be at risk totally separate from the blockchain.

3.3 Important Definition

Before presenting the Ancile framework, we first discuss some necessary terminology and definitions.

3.3.1 Eth Calls and Internal Transactions

Every valid transaction executed is stored on the blockchain [12, 13]. Due to this, blockchains can suffer from scalability issues. Valid transactions sent to smart contracts in the Ethereum blockchain are considered state changeable calls and consume gas. To reduce gas consumption and the number of transactions on the blockchain, the Ethereum blockchain allows eth-calls to be utilized in addition to transactions. Eth-calls allow nodes to send messages to other nodes or smart contracts to retrieve its current state without storing the message on the blockchain. Therefore, eth-calls are similar to simulations of transactions. By executing eth-calls to send notifications/messages or to retrieve current states, the size of the blockchain can be greatly reduced.

Same as the eth-calls, "internal transactions" are not recorded on the blockchain. After being started off by the transactions smart contract interrelate with each other, then internal transactions happen. By decreasing the amount of stored information on the blockchain, "internal transactions" also improve the scalability of systems which are based on blockchain

3.3.2 Proxy Re-encryption

"Proxy re-encryption solves the issue of transferring encrypted records between nodes without sharing symmetric keys by using a proxy. The proxy is responsible for reconstructing an encrypted message in such a way that another user could use their private key to decrypt the document, even if it was not originally encrypted with their associated public key [6, 7, 23, 33]. This system allows secure sharing between parties without fully decrypting the document during transfer process, as shown in Figure 3.1.

In this paper, we utilize a distributed proxy re-encryption scheme with blinding, where multi-parties (proxies) partake in the re-encryption process [61]. To do this, a message is encrypted with a master public key, and the associated private key is then distributed in pieces to the proxies. In doing so, the proxies can re-encrypt the message while unable decrypt the full message. To further prevent proxy nodes from accessing the message, a blinding re-encryption scheme, like [61], will use homomorphic multiplication to create an encrypted blind value from random numbers chosen by each proxy. The message is then homomorphically multiplied by the blind value, thus creating a plaintext message that is obscured unless the blind value can be determined [61]. Thus, distributing the private key and blinding the message, the message can only be decrypted by the intended receiver or after every proxy agrees to collude [20]."

$$\begin{tikzcd}
{[m]_A} \arrow[r, "{* \ [p]_A = X_{n\in N}[p_n]_A}", "\text{Blind}"'] & {[m_p]_A} \arrow[dl, "\text{Decrypt}"'] \\
m_p \arrow[r, "\text{Re-Encrypt}", "{* \ [p]_B^{-1} = {}^1/_{X_{n\in N}}[p_n]_B}"'] & {[m]_B}
\end{tikzcd}$$

Figure 3.1: The procedure of "blinded re-encryption" [61]

3.3.3 Types of Nodes

The blockchain deals with three various types of nodes "Full nodes, light nodes and archieve nodes" which helps to maintain the scalability of the system [28]. As the number of nodes rises in the blockchain network, the scalability of the system can be compromised because in the blockchain only those users can participate as "miners or full nodes" who have greater depository and computing power.

Full nodes deal with each "transaction" and keep each "block" in the blockchain while the light nodes only keep "block header" which includes the "previous block hash, the hash of the Merkle Root and the nonce". As light nodes stores block header, because of this without using the large part of memory of light nodes of the blockchain can validate the certain transaction which has not been altered. Light nodes can also access particular data which they want.

Same as the "full nodes", archive nodes keeps each "transaction and block" on the blockchain network [12, 13]. Moreover, archive nodes have a record of receipts of transactions and the whole state tree [55]. Archive nodes use this information to help network to retrieve required data [12, 13]. The different functions of mentioned three kinds of nodes provide greater scalability of the "Ethereum Blockchain" because of which large organizations and users separately can use blockchain with their available resources and for their respective purposes.

3.4 Proposed Framework: Ancile

3.4.1 Overview

The Ancile framework utilizes six different types of smart contracts for different functions. These smart contracts are "Consensus, Classification, Service History, Ownership, Permissions, and Re-encryption". These "smart contracts" allows patients to get advantage from enhanced effectiveness of these contracts by decreasing the requirement for patients to interact with each one of them. This will have an effect on the efficiency of patient experience which improves and also decreases privacy threats. Ancile utilizes the contracts to create other contracts, to generate a greater level of partitioning So that the location of patient's personal data can be given directly to the patient only.

These six unique contracts help "Ancile" to maintain "cryptographic hashes" of recorded data and "query link", which approve the EHR database integrity. Smart contracts also manage access control by allowing patients to have a look and to manage who can have access to their personal information. Additionally, patients have the authority to allow other nodes to transfer record. This is because of the usage of "identity-checking", which confirms who is allowed to approach data and "proxy re-encryption", which make it possible to avoid re-encryption of record for each transfer. Moreover, Ancile confirms that three components needed to approach an EHR, "the encrypted record, the query link, and the symmetric key" which are located at different places, by conveying the query link for the data securely off the blockchain.

In the following sections, we will explain the different software modules used by

Ancile, the particular function of every “smart contract” and the architecture of the proposed framework.

3.4.2 Software Components

Ancile contains following main software components:

- Database Manager
- Cipher Manager
- Ethereum-Go Client

A. Database Manager

Ancile create “query link” to the recorded data in the existing system in such a way it incorporates EHR Databases. The function of Database Manager is explained in [20] as: “The Database Manager is used to navigate existing EHR Databases and for generating the link that maps to a record. Moreover, the Database Manager will also create hashes of both the record and the query link to place on the blockchain.”

Someone can alter or remove the records directly from the database because of using an existing database which may destabilize the security of blockchain. That is why use of hashes approve the data integrity. In case if any node did not get back the data, the usage of cryptographic hashes enables these nodes to confirm what particular record has been lost.

B. Cipher Manager

Ancile uses Cipher Manager for all cryptographic tasks. The Cipher Manager manages all encryption and decrypting of documents. Firstly, in Ancile it handles the “Symmetric key encryption” on greater files.

The reason of using symmetric key encryption is its efficiency and ability to remove the need of re-encrypt the files later. Secondly, Cipher Manager utilizes the "Public key encryption" to secure data while distribution and to mention who can have accessibility to protected health information (PHI). Thirdly, Cipher Manager handles the "proxy re-encryption", proxy nodes use Cipher Manager to "re-encrypt" recorded symmetric keys on the blockchain, when they need to give access of stored record to a third party. Finally, it is also responsible for the decryption of every encrypted information recorded from Ancile.

C. Ethereum-Go Client

"The Ethereum-Go client [28] sometimes called as Geth, is the main Ethereum CLI client written in Go programming language. Geth is an access point to Ethereum networks, including the public, test, and private Ethereum networks. Ancile is designed to function on a permissioned Ethereum blockchain; thus the Geth client would be used by permitted nodes to access the private blockchain. Having the Geth client would signify to Ancile that a particular system is a node. Additionally, Geth [28] may be accessed using JSON RPC endpoints on the internet.
These nodes may be full or light nodes, allowing for versatile roles of patients, providers, and third parties. Users may access their node's information with Geth on the client side using a wallet [25]. The functionality of wallets may vary depending on the type of node. By using wallets, users may access their node's information over HTTPS [25]. As a result, the Geth client allows Ancile to have a user-friendly interface that can be accessed on the web and adapted as needed [20]."

3.4.3 Smart Contracts

Following are the detail of each contract used by Ancile: "Concensus contract (COC), Classification Contract (CLC), Service History Contract (SHC), Ownership Contract (OC), Permissions Contract (PC) and Re-encryption Contract

(RC)".

3.4.3.1 Consensus Contract

The "Consensus Contract" is a universal contract. The "blockchain mining, user registration and some overwrite procedures" are maintained by this contract. As shown in Figure 3.2, the COC keeps the Ethereum addresses of that node which have voting permission. The COC operates by utilizing the Qourumchain [27] consensus algorithm (which is also explained in section 2.2.1) for mining purpose. The COC for "user registration" is utilized to authenticate nodes who require a greater level of categorization while appended to the system. In the consensus process, the pre-existing nodes are continuously ensuring that upcoming nodes will not harm the system. At the beginning stage of Ancile, the COC would be unoccupied. Hence, initial nodes would need to be added by the temporary administrator node. For example, long-approved providers and third parties. The removal of the temporary administrator is allowed after having enough full nodes for the consensus process. Then the process of consensus would be implemented. Another use of COC is for overwriting nodes which are considered to be dangerous to the network. For instance, when an insurance company becomes bankrupt, the related node's permissions are them required to be canceled. For this, a voting node put forward a request, and to remove node from system the remaining node must need to approach a majority. To eliminating them from COC, and deleting their data from different PC's and OC's, this would then require to overwrite their categorization as terminated.

3.4.3.2 Classification Contract

The "Classification Contract (CLC)" classify the different roles of nodes in the system as patients, providers, or third parties. Ancile utilizes only one CLC for the entire blockchain. As shown in Figure 3.2, the CLC keeps two information fields, one is Ethereum addresses of all nodes and another one is their associated

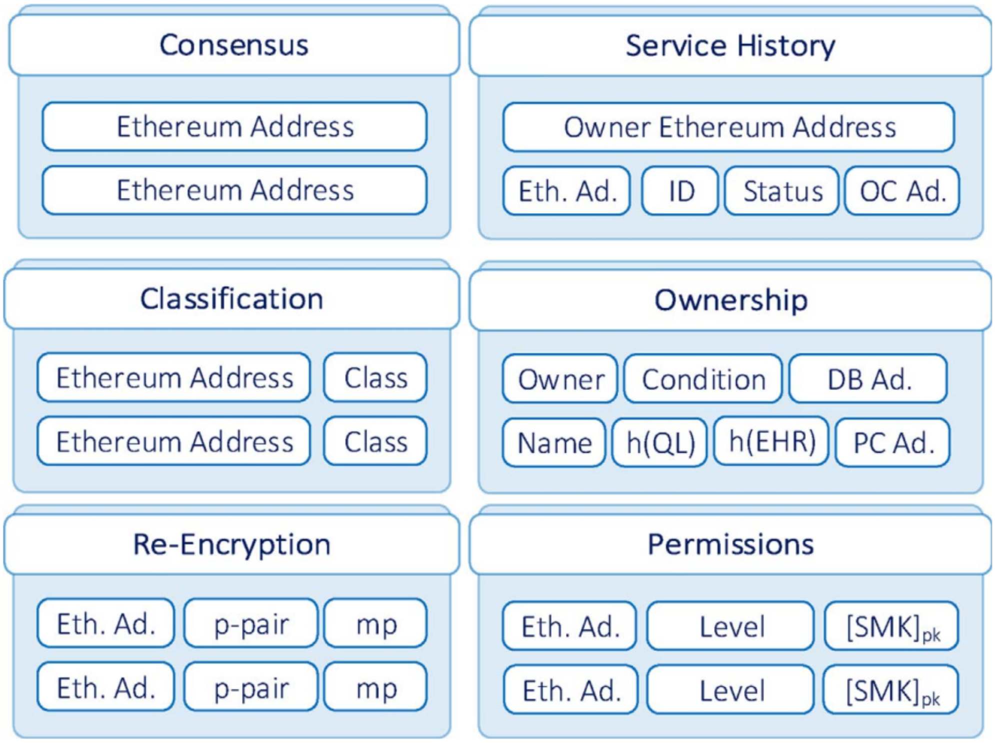

Figure 3.2: This figure illustrates the memory fields which would be maintained by every smart contract for tracking. Query link are represented by QL, while symmetric and public keys are represented by SMK and PK respectively. Hashed information are illustrated by a statement started by "h" and enclosed in parenthesis. Brackets demonstrates "encryption" utilizing the key mentioned in subscript. [20]

classifications. By using this information, the CLC can prevent double registration by confirming already registered nodes in the system.

Additionally, to find out the node classification during the node registration process the COC is used, by avoiding repetition of "Consensus process" COC may be utilized to approve the identity of a node. Hence, the utilization of CLC decreases the difficulty of "access control" in later contracts by providing a single reference point which is approved by the process of consensus.

3.4.3.3 Service History Contract

The "Service History Contract" is responsible for maintaining the histories of relationship between nodes. During the registration process, a new SHC is generated for each node. As shown in Figure 3.2, the SHC keeps the "Ethereum address of the patient, the Ethereum addresses of all relevant nodes, their related ID's, relationship status and applicable ownership contract (OC) address". An active or inactive relationship may signify by the status field.

The SHC of nodes gives them complete detail of their previous and recent medical related associations while interacting with Ancile. Moreover, SHC use ID's to provide facility to "providers" to recognize patients, similarly to "patients" to identify providers, utilizing existing ID's. Another responsibility of SHC is to querying the patients for permission confirmation. When a "provider" wish to connect a patient, the patient's SHC will ask for permission before upgrading. This allows patients to be aware of their every relationship.

3.4.3.4 Ownership Contract

The Ownership Contract has control of tracking the records which are stored by the providers for the patients. When a new connection is required to be established between two nodes, an OC is generated. As shown in Figure 3.2, the OC have different fields with different purposes. The owner field can recognize the OC, that indicates ownership of listed data by the patient. "Condition and Date fields" may come after the Owner field. They signify the special conditions for the owner if there are any.

For instance, "a parent or guardian" can be the keeper of a kid's information until they reach to maturity. Then the time at which the transfer of ownership should occur to the OC would be shown by the Date field. The shared ownership of the data can also be signified by the condition field, the example of this case is a provider-provider relationship. Moreover, to determine the provider's EHR Database, the OC has the data required for the patient node. The OC then enter "each patient record with a file name, a hash of the file's query link, a has of the record itself, and an address to a Permissions Contract (PC)".

To maintain the data integrity the two hash function fields are necessary. By recording the data in EHR databases of the provider permits users to decrease the blockchain's storage needs and also enables users to utilize existing system; Although as a consequence, the provider will have ability to create changes to a data without signifying the alteration on the blockchain. Since the hash function may be utilized to approve that there have been no alterations occur off of the blockchain. Additionally, to confirm that the "query link, key, and record" are located in various places, Ancile keeps the "hash of the query link" instead of query link itself. This hash ensures that the link has not been changed during the process of transfer because query link itself is delivered over HTTPS.

3.4.3.5 Permissions Contract

Every record has a specific permissions Contract (PC) and an OC create it when upcoming new data is appended to the network. In Figure 3.2, it can be seen that PC is constructed to keep "the Ethereum addresses of all nodes that may interact with a record, a level of access and a symmetric key encrypted with the public key of each node. The patient, the provider of the origin, and the Re-encryption Contract (RC) will be automatically written to the PC with Owner, Read and Blind level access respectively [20] ". There may exist a exceptional case, for instance "psychotherapy notes", for which provider may signify in the transactions that the data must be kept secret from the patient. Then the "Owner level access" would be given to the provider and "Blind access" to the patient. The various level of access is given as:

- **Read:** When for the first time data is appended to the system or by using proxy re-encryption, a node would have a "Symmetric key" created for them.
- **Transfer:** By using Read access a node can add other nodes. Special conditions are applied when a node is given this level of access. Such as, a provider may have authorization to provide only "Read access" to another provider.
- **Owner:** This type of node have complete permission of the PC. These type of nodes can "add other nodes of any level of access, remove nodes from the PC, and alter the levels of access for any existing nodes."

- **Blind:** In this type a node have only PC's addresses. This access level is utilized by those patients who would be only enable to view that who can see their data but may not be permitted to see the data themselves. It is also utilized to give control to the RC that it can recover "the symmetric key" encrypted for the proxy nodes.

3.4.3.6 Re-encryption Contract

The "proxy re-encryption" is managed by the Re-encryption contract RC. In Ancile framework, "a master public key with a shared secret key" is given to a set group of proxy nodes. Each time, when new proxy nodes set is generated a RC will also be established. The effectiveness of re-encryption schemes depends on the greater number of proxy nodes each set while confirming that the possibility of proxy collusion is low. Every proxy in Ancile selects a "blind value p", encrypt it and decrypt portions of blinded message on their own system. Each proxy will send their contribution to the RC when they need to combine their values.
As shown in Figure 3.2, the RC keeps "the addresses of proxy nodes, pairs of encrypted p values, and the plaintext blinded message." This is because of RC which use "homomorphic encryption" to create those values in Ancile. At the moment, the limit of smart contract hold up is 256 bits [44]. The length of "symmetric keys" that would be sent for re-encryption are usually 128 bits. After encryption, the size of symmetric key would become greater. Therefore, the "symmetric keys" size should be set smaller in order to use homomorphic multiplication until progress in Ethereum permits smart contracts to keep larger values.

3.5 Framework Architecture

The architecture of Ancile is illustrated in the following diagram 3.3 which assessing in various cases how the framework would be utilized. Four different forms of actions are used by the Ancile framework as shown in Figure 3.3. First of all, the action demonstrated in solid blue is a standard blockchain transaction. These transactions are written to the blockchain and mined using the Quorumchain consensus algorithm. Secondly, the internal transactions are illustrated in dashed blue lines.

Thirdly, the action represented in orange is an eth-call, that is utilized when information requires to be sent to a smart contract but does not need to be written to the blockchain. This permits the increase in efficiency and privacy of the system when using the functions required to operate Ancile. The last type of action, illustrated in gray, is a non-blockchain action. This demonstrates the transmitted data over HTTPS or anything taking place internally to a node.

Non-blockchain actions may also illustrate the private transactions. To transmit sensitive data a private transaction uses an external method while locating the hash of the information on the blockchain. As a consequence, the legitimacy of the transaction can only be validated by those who can re-create the hash. The data integrity offered by the blockchain is preserved by using the private transactions and also increases the privacy.

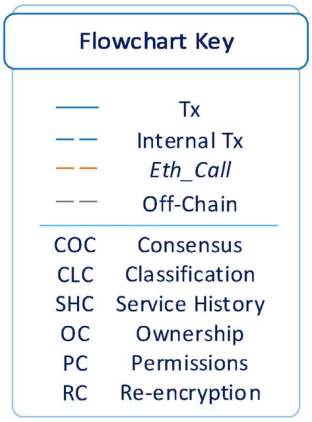

Figure 3.3: This diagram clarify the different forms of action used in Ancile and abbreviations in the framework architecture [20].

3.5.1 Adding a Node

The procedure of appending an upcoming new node is shown in Figure 3.4, that illustrates added patient to the blockchain, but it must be noted that for any classification of the node the same procedure would occur. Before this process, it must be considered that upcoming users have generated “wallets” and be given “an

Ethereum address". Another thing which must also be noted is that third parties and providers must have an identifier which is publicly known. This identifier is distinct to their organization. The provider number allocated by the Federal government is an illustration of this identifier for Medicare purpose. Moreover, since in existing provider systems patients usually already possess a numerical ID's. So the publicly open ID's possessed by the patient must be considered as the existing value. These values must be stored off of the blockchain for security issues.

Adding a node process starts when the node which has the authority of voting authenticate that public ID is suitable for the required classification. Since lowest level of permissions is needed for adding a patient on the blockchain, and since only patient's numerical ID is delivered to the voter nodes, therefore with little validation patient will be added to the system. On the other hand, the validation procedure for other parties like third parties and providers should be more substantial. It is the responsibility of voter nodes to authenticate the classification request is valid, by confirming the validation of third party or provider nodes. This process could need ensuring the existence of a non-registered provider matching the given ID.

This process of validation makes it possible that the request for adding new nodes is received from the already registered nodes, which results in low probability of unauthorized actors to overtake the system. After authentication, new node will be added to the system after generating a new account on CLC and creating related SHC. Other users or patients will then get the information of their accounts from the node who is requested to add in the system, this is similar to the case when patients visit to a new provider first time how they generate online account or fill out forms. Following are the main steps of adding a node:

1. The provider sends the address of the new node and the requested type to the CLC.

2. The CLC forwards the requested node's address and requested type to the COC.

3. The COC polls a subset of the voter nodes for type validation.

4. Each vote is returned to the COC.

5. The results are compiled.

6. The results of the poll are returned to the CLC.

7. The CLC confirms authorization.

8. A request is sent to the new node for approval to be added to the network.

9. The patient response is returned.

10. If accepted, the CLC updates its local memory with the new node's Ethereum address and type.

11. The CLC creates a new SHC for the new node.

12. The SHC address is sent to the patient.

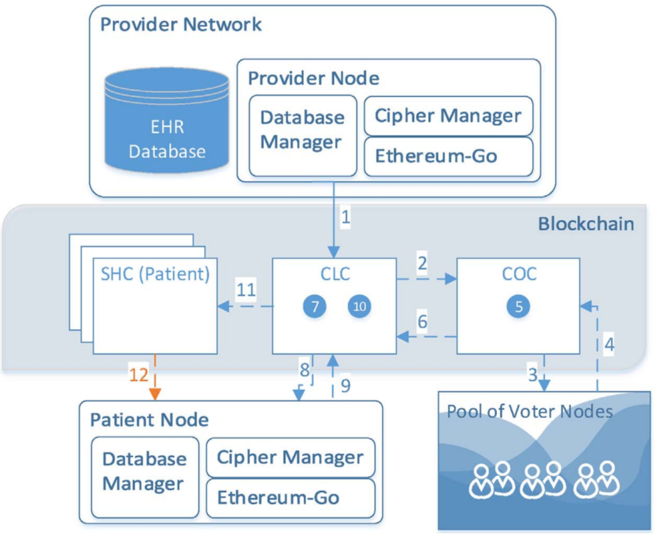

Figure 3.4: Adding a patient node is illustrated in this figure to the permissioned blockchain. It is assumed that before this process upcomig nodes have generated their relevant wallet and an Ethereum address [20].

3.5.2 Registering a Patient

Patient registration is an illustration of making an association among two distinct nodes. When a new patient wants to connect to a provider, this procedure needs to complete every time. As shown in Figure 3.5, registration starts by ensuring that the patient is enrolled node in the system. If the patient is not a node in the system, the "Adding a node" process would be completed first. After ensuring that the appropriate information is sent by the provider in a transaction to their SHC.

This procedure proceeds by asking the patient node approve the association. Because of this request patient say about the providers with which they connect. The process stop, if the patient says no to the relationship and it is informed to the provider. Although, if the offered relationship has accepted by the patient, to demonstrate the union a new OC is generated. Then the owner field would then be filled by the OC automatically with the Ethereum address of patient and provider's network information of database. At this step, Any exceptional to the owner may also be applied. In the end, for future reference, the service history contracts of provider and patient are upgraded with the OC address.

This process would be same for any association. The utilization of SHC and process of registration permits users to make connections with number of other nodes. For instance, this process is put into service when an association between an insurance company and healthcare provider need to register. In Ancile a relationship is established when two entities are required to share protected data. Following are steps for registering a patient:

1. The Databases Manager of provider node sends the patient's Ethereum address to the CLC to verify that patient is a registered node in the blockchain.
2. The CLC returns boolean value to the provider node.
3. The provider node sends the patient's Ethereum address, an ID and a status of active to its SHC.
4. The SHC confirms the patient is a patient.
5. The provider's SHC requests to create a new relationship with the patient.
6. The patient accepts or rejects the request from the SHC.

7. If accepted, the SHC of the provider is updated with the patient's information.

8. The provider's SHC creates a new OC for the new relationship.

9. The OC sends its address to both SHCs to update their databases.

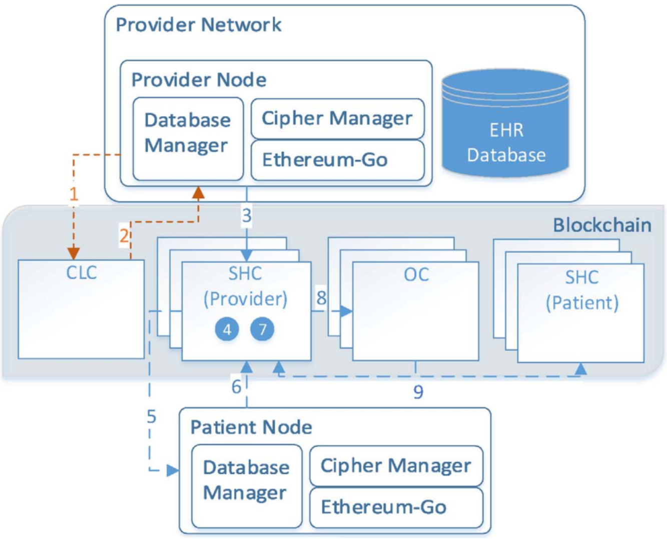

Figure 3.5: The registration process is illustrated in this figure which is demonstrating a new patient-provider association. Before this process, the patient's ID and Ethereum address must be known to the provider node; Therefoe, this process would be ended at the provider's office by the patient interrelating with the provider network [20].

3.5.3 Changing Access Permissions

Sometimes in various cases other party or provider may have increased control over their data given by the patient. The procedure of giving access control of transfer or ownership of record to the provider is represented in Figure 3.6. To give access control it must be considered that ownership of the record is owned by the patient.

By locating the PC for the data and asking for the permission changes, the procedure of changing access permissions starts. The PC would then ensure that permission change can be done before proceeding a request to the patient. If the Ownership of the record did not possess by the patient, the one who has the Ownership will be sent the request. Then the local database will be updated by the PC and the provider will get either worthwhile or refusal notification. The changes in PC would be examined by the provider node, but in time-sensitive case, the utilization of notification activate the process. The repetition of this process would occur for any party who may ask for permission changes.

On the other hand, there exists a case in which a patient without a formal request may want to give enhanced permissions. For this, first of all, the patient should knowlege of Ethereum address of that nodes who will get access permission from the patient. After this, PC will get a transaction sent from patient specifying these changes. On the blockchain, nodes who have owner level access can allow more permissions to other nodes. Because of access permission, patients have the freedom to specify a life partner or other authorized representative who have rights to access their data.

1. The provider node sends the patient's ID to the provider's SHC.
2. The provider's SHC returns the address of the associated OC.
3. The provider node sends the applicable filename to the OC.
4. The address of the file's PC is returned to the provider node.
5. The provider node sends the requested access permission to the PC.
6. The PC reviews the current level of access of the provider node.
7. If the requested level of access is not the current level, the PC requests a change in the level of access from file owner.
8. The patient accepts or rejects the request.
9. If the patient approves, the PC updates the permissions for the applicable file.
10. Once the permissions have been updated, a notification is sent to the provider indicating the process was completed successfully

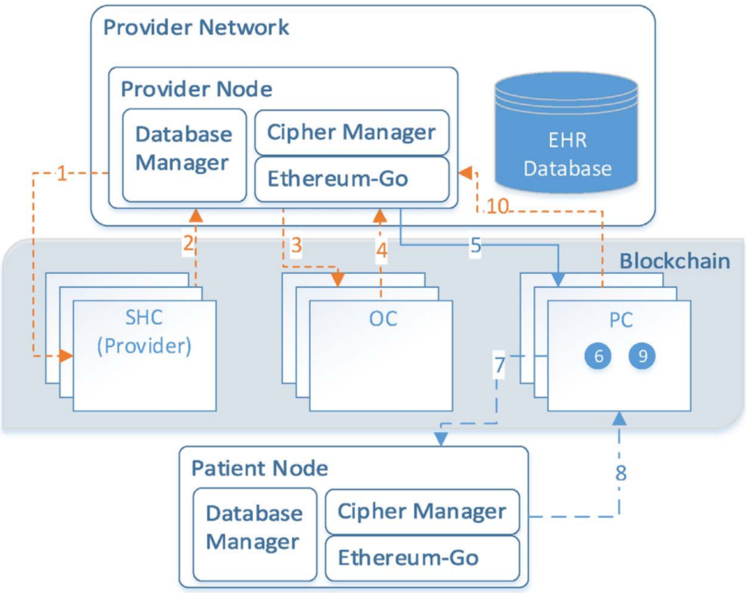

Figure 3.6: The process of giving more permission. It must be noted that the ownership access is owned by the patient for the requested record [20].

3.5.4 Adding a Record

In this process, it must be considered that the relationship between provider and patient have already set and they both have shared OC. Adding a record starts with the "internal encryption" within provider node. When a provider node generated an updated record, this updated data will then be sent to the Database Manager and created a query link to the EHR Database. The Database Manager must then start automatically creating hashes to the data and query link. Before keeping the record in EHR Database, the Database Manager make a contact with the Cipher Manager for the encryption of these hashes and query link. Succeeding the encryption, the OC address is located by the provider node for the patient and provider node also place hash of data, hash of the query link, and "encrypted symmetric key" to the blockchain. Following the placement of relevant data to the blockchain, new PC for data will be generated by the OC which will automatically add permission fields for the RC, patient, and provider of origin. The query link of data is then received by the patient who can have accessibility to the data when he wants. Figure 3.7 represents the procedure of addition of a record to the Database of EHR and how Ancile is used to confirm integrity of data and access control.

Moreover, small records may also be stored by the Ancile. This performance may be effective for fast data transfer for example instructions letter. Although, data would be required to be small in size, for the reason for mining high cost and keeping records. The similar process of encryption would need to be finished to keep the record of small data, but this data would be placed to the blockchain instead of placing the hash of the data. This would simplify the retrieval process because uploading the data on the blockchain remove the requirement of query link to get access to the data. Unfortunately, when a small data is uploaded to the blockchain, all encrypted gadget required to get access to the data is ready for use to the whole system. For precaution, Data uploaded to the blockchain must not have details like home address or social security numbers. Following are the main steps of adding a new record:

1. The provider's Database Manager generates a query link to a free location in memory, hashes the link and the record, then sends the link and record to the Cipher Manager.
2. The Cipher Manager generates a symmetric key and encrypts the symmetric key with the public keys of the provider, patient, and proxy set.
3. The Database Manager stores the record in the EHR Database.
4. The provider node sends the patient's ID to their SHC.
5. The address of the associated OC is returned.
6. The provider node sends the record name, query link hash, record hash, and encrypted symmetric keys to the PC.
7. The new PC auto-creates the provider, patient, and RC permissions.
8. The PC sends its address to the OC.
9. The record information is added to the OC's local memory.
10. The encrypted query link is sent to the patient over HTTPS.
11. The patient node stores the query link in its Cipher Manager.

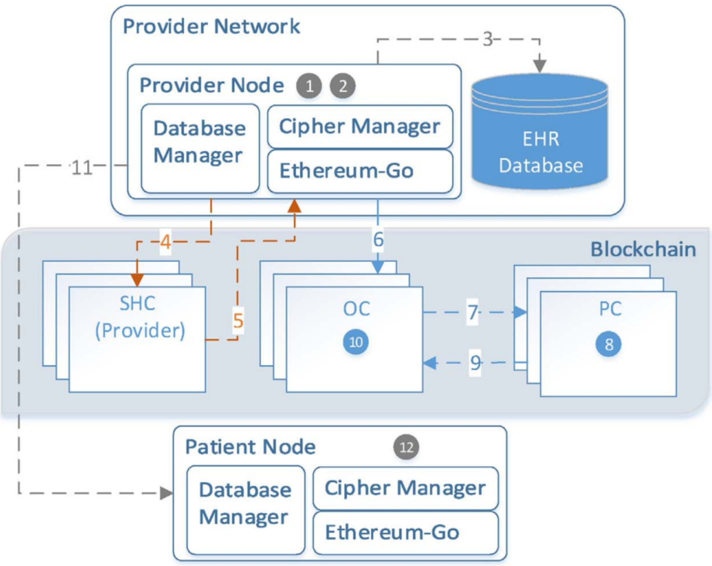

Figure 3.7: The process of addition of a new record. Before the begning of this process the registeration of rel;ationship should be completed first [20].

3.5.5 Retrieving a Record

In the retrieval of a record no transactions are needed, that is why it is a nontaxing process. As shown in Figure 3.8, the procedure starts when the patient uploads the provider's OC who keeps a record of the data. The request for the data is then proceeded by the patient. If the permission to gain access to the data is possessed by the patient, the "encrypted symmetric" is get back. When the key is decrypted by patient, they can decrypt the "query link" they must have kept the record in their Cipher Manager, in EHR Database of provider gain access of data and decrypt the data.

The very small effort of the client would be needed by this process but may to retrieve might require time and also require three tools to decrypt. Although, since the patient makes connections with their nodes by using online wallets, this enables them to gain access to their records through internet connectivity on any machine. The interoperability of EHR improves by making possible this retrieval of record importantly on any computer, or even cell phones. The procedure demonstrated in Figure 3.8 does not show the procedure of how to retrieve small data which are placed to the blockchain, but that procedure must be even straightforward. The only remarkable distinction is that in the place of the symmetric key the record itself would be returned and the patient would no more be required to reach the provider's EHR database. The steps for retrieving a record are as follows:

1. A patient node sends the provider's ID to patient's SHC.
2. The SHC returns the applicable OC address.
3. The patient node sends the filename of the requested record and Ethereum address of the patient to the OC.
4. The OC checks with the PC to confirm that Ethereum address has permission.
5. If the patient has permission, their symmetric key is sent to the OC.
6. The OC sends the encrypted symmetric key and database access information to the patient.
7. The Cipher Manager decrypts the symmetric key using the private key of the patient, then decrypts the query link with the symmetric key.
8. The Database Manager follows the related query link and retrieves the encrypted document from the provider's EHR Database.
9. The Cipher Manager decrypts the record with the symmetric key.

3.5.6 Transfer a Record

For any EHR management system, it is necessary to transfer the records smoothly. Ancile utilizes "proxy re-encryption" to cover the requirement of accessibility while also managing the security. Figure 3.9 shows the procedure of transfer of record between two different providers. It must be noticed that the process of transferring a record would be completed by "retrieving a record, decrypting and sending to another party". Therefore, Ancile only confirms that who is allowed to share

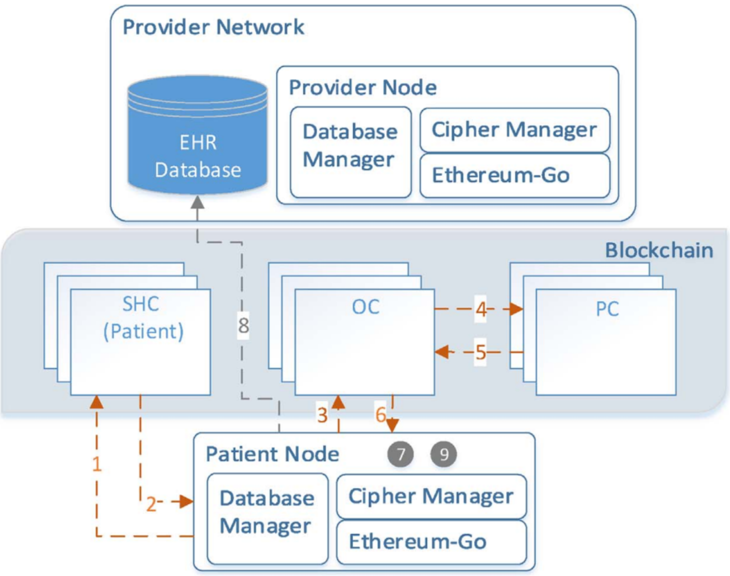

Figure 3.8: The patient's retrieval of data procedure. It should be noted that the patient must possess Read access for this data and hence, has a "Symmetric key" encrypted for their use. [20]

record and with whom data may be shared, but it can not confirms the data movement is tracked. In such a way, Ancile can be utilized as an indisputable ledger in which off-chain actions should require to be taken.

The procedure starts with a provider A who specify the necessary PC. For this Provider, A should have Owner or Transfer level access, or else PC terminate the procedure. If only transfer level access is possessed by the Provider A, then the PC and CLC meet up to ensure that Provider B is the same node with whom Provider A want to transfer a record. When this confirmation is done, the Provider B will get a permissions field in the PC, and the "proxy re-encryption" procedure starts. Figure 3.10 demonstrates the process of re-encryption. Steps 9,10,11 in Figure 3.9 corresponds to steps 1,11, and 12, respectively in Figure3.10.

To get the advantages from both "blinded distributed re-encryption" as well as blockchain technology, the process of re-encryption between RC and proxy nodes contains multiple transactions. To choose any RC the PC may be prearranged in such a way it can make a pseudo-random choice. Additionally, the threshold might be created by the RC so that some collection of proxies in the group required to participate in re-encryption and would then give back the updated "encrypted symmetric key" to the PC. It must be considered that the public key of Provider B is known to the selected proxies for the process of re-encryption. Once Provider B gets the encrypted query link through HTTPS and PC address from the blockchain, the process of transfer of record would then ends. After this, the key from the PC might be retrieved by the Provider B and they may also decrypt the data. Following steps demonstrates the process of transferring a record between two different Providers 3.9.

1. Provider A sends the patient's ID to its SHC.
2. The SHC returns the applicable OC address.
3. Provider A sends the filename to the OC.
4. The OC returns the address of the applicable PC.
5. Provider A node sends the Ethereum address and request level of Provider B to the PC.
6. The PC sends a transaction to the CLC to verify Provider B is an authorized provider on the system.
7. The CLC returns the verification to the PC.
8. The PC updates its database to give Read access to Provider B.
9. The PC sends the Ethereum address of Provider B and the master encrypted symmetric key to proxy nodes. The symmetric key is then re-encrypted.
10. The re-encrypted symmetric key is sent to the PC.
11. The PC adds the re-encrypted symmetric key to its database.
12. The PC sends the PC address to Provider B.
13. Over HTTPs, Provider A sends the encrypted query link to Provider B. Provider B may then decrypt the link and retrieves the record.

The main steps of proxy re-encryption process are the following shown in Figure 3.10 :

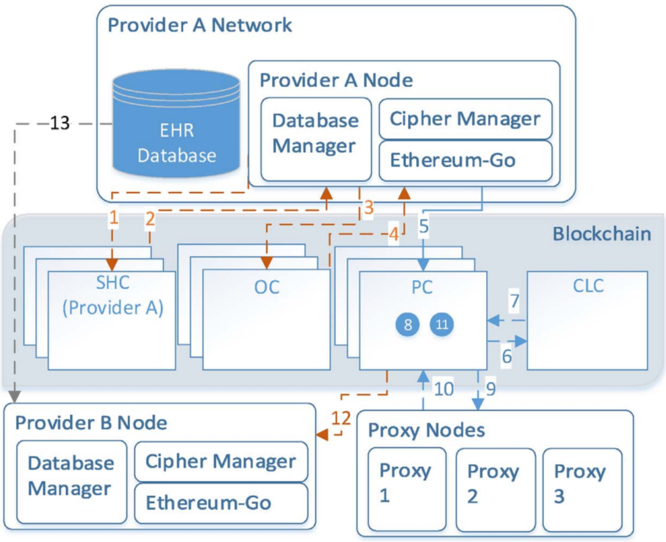

Figure 3.9: The procedure of transferring data between two providers. To complete this procedure, Transfer level access must be possessed by Provider A [20].

1. The PC sends the Ethereum address of re-encryption recipient and the master encrypted symmetric key to proxy nodes to the RC.

2. The RC sends the public key of the recipient to the proxy nodes.

3. The proxy nodes each generate a random large value p and encrypts it with the master key and the public key of the recipient.

4. The encrypted p-value pairs are sent to the RC.

5. The RC uses homomorphic multiplication to create a master key and the recipient public key.

6. The RC then uses homomorphic multiplication to combine the associated encrypted symmetric keys and p-values.

7. The RC sends the encrypted p-key value to the proxy nodes.

8. The proxy nodes decrypt the p-key value to get the blinded message.

9. The blinded message is sent to the RC.

10. The RC uses homomorphic multiplication to calculate the recipient's new symmetric key.

11. The RC sends the Key to the PC.

12. The PC adds the re-encrypted symmetric key to its database.

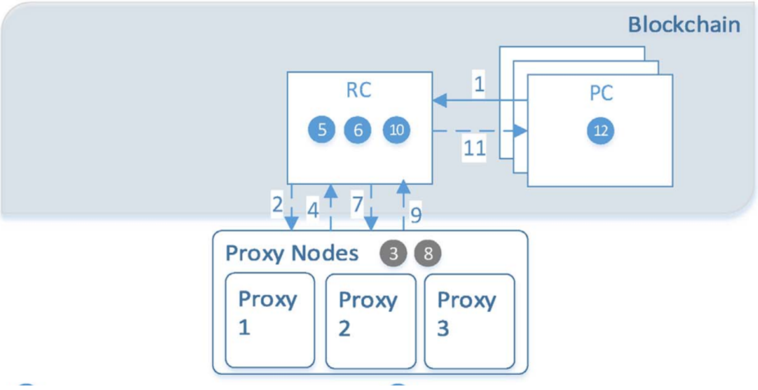

Figure 3.10: The method for symmetric key re-encrypting. The participating proxies may be selected by the PC psuedo-ranomly, on the other hand, the proxy group and hence the RC, should have been already created [20].

3.6 Comparative Performance Analysis

In this section, as given in [20] we highlight a comparative performance analysis of Ancile and MedRec [24] by comparing the approximated costs of both frameworks. The Author had classified actions happening either within blockchain or off of the blockchain actions. For a unit of measure, they had used gas costs while discussing the on-blockchain actions. In Ethereum blockchain [59, 60] the computational cost is measured in the unit of “gas costs” such that as gas costs increase the computational time also increases.

They had used adding a record process for the estimation of performance differences because MedRec [24] only gives a full procedure of appending a record, it never gives any particular computational details. In MedRec, following steps are included to append a data for a patient:

1. An appeal is forwarded by a provider to their manager of EHR, a management system which manages updates of local database, delivering notifications, and gives a link to have a look on medical data, to append a data.

2. To recover patient's address and to get back summary contract from registrar contract an appeal is forwarded to the blockchain.

3. An appeal is delivered to the blockchain to display an updated relationship contract of patient-provider and make a connection of it with summary contract.

4. The appeals to the blockchain are authenticated by the Miners and when they effectively authenticate them they get a reward defined in relationship contract of patient-provider.

5. On the patient node the summary contract is brought up to date and a notification is delivered to the patient about this update by utilizing manager of EHR.

6. After this patient accepts or rejects the alterations.

7. Based on the reaction of patient a suitable update is delivered to the relationship contract status of patient-provider in the service contract.

8. A signed query request is delivered by the patient to the database of provider and it would look over the permissions to examine what details can be delivered to the patient node with the query.

9. The keeper of database of patient node brings updates to the local database of patient node with details collected from patient node.

Off-blockchain actions are represented in steps 1, 5, 6, 8, and 9, while actions happening within the blockchain are demonstrated in steps 2, 3, 4, and 7. The actions which are happening outside the blockchain are applying hash function to the links, requesting the query or update of local databases and delivering the notification. The performance figure may be very small because of the implementation of off-blockchain modules. The recovering and keeping the record of one data in the smart contract is involved in on-blockchain actions, As the "gas cost" is based on the measure of the size of the data value which will be on lower end of the gas limit compared to doing multiple different actions in one transaction. In Ancile the steps to adding patient data is given in Figure 3.7. The off-blockchain actions are demonstrated in Steps 1, 2, 3, 11, and 12, on the other hand, actions occurring within the blockchain are represented in steps 4, 5, 6, 7, 8, 9, and 10. The creation of query link, its hashing, its encryption, and database recovering or storage are included in off-blockchain actions. Based on the application of modules of off-blockchain the cost of performance can be little but in comparison with MedRed, it will be a bit greater because Ancile involves the encryption of keys utilized to decrypt the EMR links. The retrieval and storage of records in smart contracts, forwarding internal transactions to connect various contracts by utilizing other contracts are the on-blockchain actions. As Ancile includes more steps, particularly actions happening within the blockchain, in contrast to MedREc. Therefore, Ancile will possess a greater performance cost, But Ancile provides more privacy by permitting providers to keep record of small medical data and links to greater medical data by using symmetric key encryption on keys to decrypt data. So, it can be considered best than the MedRec.

Chapter 4

Students Privacy Preserving Framework

In this chapter, we will introduce our proposed scheme "Students privacy preserving framework" (SPPF) for electronic student records (ESR), and give the details of Revocable identity-based proxy re-encryption (IB-PRES) used in this framework. We will also discuss the main overview of our framework and architecture of this framework. At the end of this chapter, we will provide the comparative performance analysis of SPPF by comparing it with other frameworks.

4.1 Complexity Assumption

The proxy re-encryption method which we have used in our work its computation problems are defined in this section.

Definition 4.1.1 (Bilinear Mapping)

Consider two multiplicative groups $\mathbb{G}_a$ and $\mathbb{G}_b$ whose prime order is q and generator of group $\mathbb{G}_a$ is s. A bilinear map $u : \mathbb{G}_a \times \mathbb{G}_a \rightarrow \mathbb{G}_b$ is a map which satisfies the following properties between the groups $\mathbb{G}_a$ and $\mathbb{G}_b$.

1. **Bilinearity** $u\ (s_1^{\alpha},s_2^{\beta}) = u\ (s_1, s_2)^{\alpha\beta}$ for $s_1,\ s_2 \in \mathbb{G}_a$ and $\alpha,\ \beta \in \mathbb{Z}_q^*$ are two arbitrary numbers.

2. **Non-degenerate** $u(s_1,\ s_2) \neq 1$ where $1 \in \mathbb{G}_b$ which is the identity element of the group $\mathbb{G}_a$.

3. **Computable** There exist an efficient algorithm which evaluate $u\ (s_1, s_2)$ for all $s_1,\ s_2 \in \mathbb{G}_a$

we call $(q,\mathbb{G}_a,\mathbb{G}_b,u,s)$ a bilinear group [41] .

Example 4.1.2 The following are the examples of bilinear mapping.

1. Matrix multiplication is bilinear mapping which is defined as $\phi : M_{n\times m} \times M_{m\times n} \rightarrow M_{n\times n}$, where $\phi(A, B) = AB$.
2. The dot product between on vector space $\mathbb{R}^n$ is also bilinear defined as $\psi(u, v) = u_1.v_1 + u_2.v_2 \ldots u_n.v_n$. It is bilinear mapping in the sense because it is linear transformation in each of its variable.

Definition 4.1.3 (Decision Bilinear Diffie-Hellman (DBDH) Assumption)

Consider a bilinear groups $(q, \mathbb{G}_a, \mathbb{G}_b, u, s)$,the advantage of an probabilistic algorithm $\mathbb{A}$ in finding the solution of the DBDH problem in the bilinear group is given as

$$Adv_{\mathbb{A}}^{DBDH} = |P[\mathbb{A}(s, s^a, s^b, s^{ab}) \rightarrow 1] - P[\mathbb{A}(s, s^a, s^b, s^z) \rightarrow 1]|$$

where a,b,z are taken from the uniform distribution on $\mathbb{Z}_q^*$ and the probablity is drawn over the choice of a,b,z and $\mathbb{A}$ is coin flip [41].

4.2 Construction of IB-PRES

In [41] Author uses the idea of Revocable identity-based proxy re-encryption ($IB-PRES$). In this section, we will give the construction of $IB - PRES$. In this scheme, the PKG generates the partial secret keys for users instead of generating full secret keys. The description of this scheme is as follows.

- **Setup(K^*):** Assume two multiplicative groups $\mathbb{G}_a$ and $\mathbb{G}_b$ whose prime order is q. The security parameter k^* is taken by the PKG as input, and outputs a bilinear group $(q, \mathbb{G}_a, \mathbb{G}_b, u)$ with prime order q,
  where $u : \mathbb{G}_a \times \mathbb{G}_a \rightarrow \mathbb{G}_b$. is a bilinear mapping and choose a cryptographic hash function $H : \mathbb{G}_a \rightarrow \{0, 1\}^*$. Suppose the generators of $\mathbb{G}_a$ are s, x. Then, the PKG puts $s_1 = s^\alpha, s_2 = s^\beta, \gamma = x^\alpha$, where $\alpha, \beta \in \mathbb{Z}^*{}_p$, and begins a user list $UL = \phi$ and a revocation list $RL = \phi$. Lastly, the system parameters params $= (q, \mathbb{G}_a, \mathbb{G}_b, u, s, s_1, s_2, x, H)$ are published by PKG that stores the master private key $MPK = (\alpha, \beta, \gamma)$ secretly.

- **KeyGen(params, I):** The public parameters "params" and an identity I taken as input by the PKG, and returns a partial private key PK_I for the user having the identity I. The PKG selects arbitrary $\ell_I \in \mathbb{Z}^*{}_q$, and calculates

  $$PK_{I,1} = x^\alpha (H(I \oplus s_2))^{\ell_I}, PK_{I,2} = s^{\ell_I}$$

  The user I have a partial private key is $PK_I = (PK_{I,1}, PK_{I,2})$. The PKG forwards (PK_I, ℓ_I) to the user I via a secure way for example email. The user I can validate the partial private key by

  $$u(PK_{I,1}, s) = u(s_1, x).u(H(I \oplus s_2), PK_{I,2})$$

  The user I selects $p \in \mathbb{Z}^*{}_q$ and calculates the private key $PK_I^{'} = (PK_{I,1}^{'}, PK_{I,2}^{'})$.

  $$PK_{I,1}^{'} = x^\alpha (H(I \oplus s_2 \oplus p)^{\ell_I}, PK_{I,2}^{'} = PK_{I,2}$$

- **IBEncrypt(params, I, m):** The message m, identity I and the public parameters params are taken as input by the data owner I and returns the ciphertext CT, which is forwarded to the proxy server PS. The data owner I selects $\theta \in \mathbb{Z}^*{}_q$ and calculates the original ciphertext $CT = (C_1, C_2, C_3)$.

$$C_1 = m.u(s, s_1)^{\theta}, C_2 = s^{\theta}, C_3 = (H(I \oplus s_2 \oplus p))^{\theta}$$

- **Query($R, SK_R^{'}, CT$):** The requester R ask for the data produced by the owner I. The identity R, private key $PK_R^{'}$ and the ciphertext CT are taken as inputs by the requester R, and returns an authentication data φ, which is forwarded to the data owner I. The requester R calculates $E = s_2^l$ and $Q = EPK_{R,1}^{'}$ and forwards an authentication information $\varphi = H(R \oplus s_2 \oplus p^{'}), R, C_2, Q, E$ to the data owner I.

- **Permit(params, $R, \varphi, PK_{R,2}$):** The data owner guarantees the requester by verifying the authentication data φ. If the data owner authenticates that the requester is legal, then the process continue to execute the re-encryption key. Otherwise, return $\perp$. Firstly, the data owner I ask the PKG for the partial private key $PK_{R,2}$ of the requester R. The PKG look for the identity of the requester in the revocation list RL. If the PKG finds the requester as a revoked user, the PKG give reply the data owner $\perp$. Otherwise, respond with $PK_{R,2}$ of the requester. After getting $PK_{R,2}$, the data owner I checks

$$u(Q, s) = u(s_1, s).u(H(R \oplus s_2 \oplus p^{'}), PK_{R,2}).u(E, s)$$

- **ReKeyGen(φ, R):** The authentication data φ and the identity R of the requester are taken as inputs by the data owner, and returns the re-encryption key $RK_{I \to R}$, which is forwarded to the proxy server PS. The data owner I calculates the re-encryption key as

$$RK_{I \to R} = (\frac{H(R \oplus s_2 \oplus p^{'})}{H(I \oplus s_2 \oplus p)})^{\theta}$$

- **ReEncrypt($CT, RK_{I \to R}$):** The original ciphertext CT and the re-encryption key $RK_{I \to R}$ are taken as inputs by the proxy server PS, and returns the re-encrypted ciphertext $CT\prime$ which is forwarded to the requester R. The proxy server PS calculates the re-encrypted ciphertext as

$$C_1^{'} = C_1, C_2^{'} = C_2, C_3^{'} = RK_{I \to R}.C_3$$

The re-encrypted ciphertext $CT^{'} = (C_1^{'}, C_2^{'}, C_3^{'})$ are sent by the proxy server to the requester R.

- **IBDecrypt:** With respect to the following two cases the decryptor give reply as follows:

  1. **Case1. IBDec**$(CT, PK_I^{'})$**:** The original ciphertext CT and his/her private key $PK_I^{'}O$ are taken as inputs by the data owner I, and returns the message m. The data owner I decrypts the original ciphertext as
  $$m = C_1.\frac{u(PK_{I,2}^{'}, C_3)}{u(PK_{I,1}^{'}, C_2)}.$$
  2. **Case2. IBDec(**$CT^{'}, PK_R^{'}$)**:** The re-encrypted ciphertext $CT^{'}$ and his/her private key $PK_R^{'}$ are taken as inputs by the requester R, and returns the message m. The re-encrypted ciphertext are decrypted by the requester R as
  $$m = C_1^{'}.\frac{u(PK_{R,2}^{'}, C_3^{'})}{u(PK_{R,1}^{'}, C_2^{'})}.$$

- **Revoke(**id, RL**):**The revocation list is updated by the PKG by
$$RL \leftarrow RL \cup id,$$
where the identity of the user is id which need to be revoked, and outsourced the updated revocation list.

Theorem 4.2.1 The IB-PRES is correct.

Proof. The verification of our IB-PRES can be examined by the following equations with respect to the two cases of decryption.

- **Correctness for case 1.** here from the construction of IB-PRES we have

$$\begin{aligned} &C_1 = m.u(s, s_1)^{\theta}, \\ &PK_{I,2}^{'} = PK_{I,2} = s^{\ell_I}, \\ &C_3 = (H(I \oplus s_2 \oplus p))^{\theta}, \\ &PK_{I,1}^{'} = x_{\alpha}(H(I \oplus s_2 \oplus p))^{\ell_I}, \end{aligned}$$

since x and s be the generator of $\mathbb{G}_a$ and $s_1 = s^{\alpha}$,

so we can take $x^{\alpha} = s_1$

$$PK^{'}_{I,1} = s_1(H(I \oplus s_2 \oplus p))^{\ell_I}, \quad C_2 = s^{\theta},$$

$$C^{'}_1 . \frac{u(PK^{'}_{I,2}, C_3)}{u(PK^{'}_{I,1}, C_2)} = m.u(s, s_1)^{\theta} . \frac{u(s^{\ell_I}, (H(I \oplus s_2 \oplus p))^{\theta})}{u(s_1(H(I \oplus s_2 \oplus p)^{\ell_I}, s^{\theta})}.$$

Since by symmetric property of bilinearity

$$u(s^a, s^b) = u(s, s)^{ab},$$

$$= u(s, s)^{ba},$$
$$= u(s^b, s^a),$$

where $a, b \in \mathbb{Z^*}_p$ and prime numbers are commutative

$$\begin{aligned}
C^{'}_1 . \frac{u(PK^{'}_{I,2}, C_3)}{u(PK^{'}_{I,1}, C_2)} &= m.u(s^{\theta}, s_1) . \frac{u(s^{\theta}, (H(I \oplus s_2 \oplus p))^{\ell_I})}{u(s_1(H(I \oplus s_2 \oplus p))^{\ell_I}, s^{\theta})}, \\
&= m.u\left(\overbrace{s.s.s...s}^{\theta}, s_1\right) \frac{u\left(\overbrace{s.s.s...s}^{\theta}, \overbrace{H(I \oplus s_2 \oplus p)...H(I \oplus s_2 \oplus p)}^{\ell_I times}\right)}{u(s_1(H(I \oplus s_2 \oplus p))^{\ell_I}, s^{\theta})}, \\
&= m. \frac{u(s^{\theta}, s_1(H(I \oplus s_2 \oplus p))^{\ell_I})}{u(s_1(H(I \oplus s_2 \oplus p)^{\ell_I}, s^{\theta})}, \\
&= m. \frac{u(s_1(H(I \oplus s_2 \oplus p))^{\ell_I}, s^{\theta})}{u(s_1(H(I \oplus s_2 \oplus p)^{\ell_I}, s^{\theta})}, \\
&= m.
\end{aligned}$$

- **Correction for case 2.** To check validity of case 2 one can do the following calculations:

$$\begin{aligned}
&C_1^{'} = C_1 = m.u(s, s_1)^{\theta},\\
&C_3^{'} = RK_{I\to R}.C_3,\\
&C_3 = (H(I \oplus s_2 \oplus p))^{\theta},\\
&RK_{I\to R} = (\frac{H(R \oplus s_2 \oplus p^{'})}{H(I \oplus s_2 \oplus p)})^{\theta},\\
&C_3^{'} = (\frac{H(R \oplus s_2 \oplus p^{'})}{H(I \oplus s_2 \oplus p)})^{\theta}.(H(I \oplus s_2 \oplus p))^{\theta},\\
&C_3^{'} = (H(R \oplus s_2 \oplus p^{'}))^{\theta},\\
&C_2^{'} = C_2 = s^{\theta},\\
&PK_{R,2}^{'} = s^{\ell_R},\\
&PK_{R,1}^{'} = s_1(H(R \oplus s_2 \oplus p^{'}))^{\ell_R},\\
&C_1^{'}.\frac{u(PK_{R,2}^{'}, C_3^{'})}{u(PK_{R,1}^{'}, C_2^{'})},\\
&= m.u(s, s_1)^{\theta}.\frac{u(s^{\ell_R}, (H(R \oplus s_2 \oplus p^{'}))^{\theta})}{u(s_1(H(R \oplus s_2 \oplus p'))^{\ell_R}, s^{\theta})},
\end{aligned}$$

Since by symmetric property of bilinear mapping

$$\begin{aligned}
u(s^a, s^b) &= u(s, s)^{ab},\\
&= u(s, s)^{ba},\\
&= u(s^b, s^a),
\end{aligned}$$

where $a, b \in \mathbb{Z}^*{}_p$ and prime numbers are commutative.

$$\begin{aligned}
&C_1^{'}.\frac{u(PK_{R,2}^{'}, C_3^{'})}{u(PK_{R,1}^{'}, C_2^{'})},\\
&= m.u(s, s_1)^{\theta}.\frac{u(s^{\theta}, (H(R \oplus s_2 \oplus p^{'}))^{\ell_R})}{u(s_1(H(R \oplus s_2 \oplus p'))^{\ell_R}, s^{\theta})},\\
&= m.u\left(\overbrace{s.s.s..s}^{\theta}, s_1\right)\frac{u\left(\overbrace{s.s.s..s}^{\theta}, \overbrace{H(R \oplus s_2 \oplus p^{'})...H(R \oplus s_2 \oplus p^{'})}^{\ell_R times}\right)}{u(s_1(H(R \oplus s_2 \oplus p'))^{\ell_R}, s^{\theta})},\\
&= m.\frac{u(s^{\theta}, s_1(H(R \oplus s_2 \oplus p^{'}))^{\ell_R})}{u(s_1(H(R \oplus s_2 \oplus p')^{\ell_R}, s^{\theta})},\\
&= m.\frac{u(s_1(H(R \oplus s_2 \oplus p^{'}))^{\ell_R}, s^{\theta})}{u(s_1(H(R \oplus s_2 \oplus p')^{\ell_R}, s^{\theta})},\\
&= m.
\end{aligned}$$

□

4.3 Security Model

In this section, we introduce a security model of IB-PRE on the basis of which we will represent how IB-PRE is secure against chosen plaintext attacks (CPA) as compared to other proxy re-encryption schemes.

Before defining the security model, we first confirms that the following condition to be satisfied:

For given a challenge ciphertext CT^* for identity I^*, the adversary without having the knowledge of private key PK'_{I^*}, the private key PK'_R and the proxy re-encryption key $RK_{I^* \to R}$ can make the following queries. Suppose that I^* be the target identity with which the adversary want to be challenged to the challenger.

Game CPA

Setup: The challenger $\mathbb{C}$ to create the public parameters *params*, the master private key MPK runs the Setup(K^*) and forwards *params* to adversary $\mathbb{A}$.

Phase 1. $\mathbb{A}$ can make the following queries:

1. **Private key Query.** $\mathbb{A}$ takes the identity I as input, and $\mathbb{C}$ outputs the PK'_I.
2. **Proxy Re-encryption Key Query.** $\mathbb{A}$ takes the identity (I, R) as a input, and $\mathbb{C}$ outputs $RK_{I \to R}$.

Challenge. When $\mathbb{A}$ wants to end phase 1, it submits I^* and messages (m_0, m_1) of equal lengths. $\mathbb{C}$ flips a fair coin with $0, 1$ and obtain $\gamma \in 0, 1$. It computes a challenge ciphertext CT^* for the message m_γ under the identity I^* and forwards CT^* to $\mathbb{A}$.

Phase 2. $\mathbb{A}$ can make adaptively the following additional queries:

1. **Private Key Query.** $\mathbb{A}$ takes the identity I as input, where $I \neq I^*$, and $\mathbb{C}$ responds as in phase 1.

2. **Proxy Re-encryption Key Query.** $\mathbb{A}$ takes the identity (φ, R), where $I \neq I^*$ and $R \neq I^*$, and $\mathbb{C}$ responds as in phase 1.

Guess. $\mathbb{A}$ outputs a guess $\gamma^{'}$ on γ.

Definition 4.3.1 (IND-PrID-CPA)
In Game CPA, $\mathbb{A}$ wins the game if $\gamma^{'} = \gamma$. An IB-PRE scheme is said to be indistinguishable against adaptively chosen an identity and chosen plaintext attacks (IND-PrID-CPA) if there is not any polynomial time algorithm with a non-negligible advantage in winning Game CPA.

4.4 Proposed Framework: SPPF

4.4.1 Overview

In SPPF for ESR, we are using six different kinds of smart contracts which are also being introduced in Ancile framework for EHR system [20] as mentioned in previous chapter 3. These smart contracts are "Consensus, Classification, Service history, Ownership, Permission and Re-encryption contracts". Ancile used these smart contracts for EHR system but here we are using them for ESR system. We have replaced the entity patient used in Ancile with user entity where the user can be a student, teacher, or Admin with appropriate access. The difference between Ancile and our framework smart contracts comes in "Re-encryption contract".
Our Re-encryption contract is different from Ancile's Re-encryption contract because Ancile has used a "distributed proxy re-encryption scheme with blinding" [61] in which multiple entities can participate in the process of re-encryption. On the other hand, SPPF is using "Revocable identity-based proxy re-encryption" [41] that assures the increase in security of the transfer of data. The purpose of using above mentioned six contracts is that we want users to get advantage from

enhanced feasibility and at the same time reducing requirement to interrelate with each contract. This enhances user's efficiency and minimizes confidentiality risks. These contracts enable users to have directly only their respective information.

These smart contracts maintain the "cryptographic hashes of recorded data and query links", which confirms the integrity of ESR management system in SPPF. The usage of "smart contracts" in blockchain enable users to have a look and control over their private information that who can access their private information. Additionally, by using "identity checking" and "proxy re-encryption" help to avoid re-encryption of records for every transfer. Furthermore, like Ancile, SPPF deliver the "query links" for the data securely outside the blockchain. By using the concept of Ancile, SPPF also places three tools needed for the accessibility of ESR in different locations. These three tools are "the encrypted record, the query link, and the symmetric key".

SPPF uses similar software components which are used by Ancile. These three software components are "Database Manager, Cipher Manager, and Ethereum-Go Client". The functionality of all these software components is explained in the previous chapter in section 3.4.2. We will here only explain the "Identity-based Re-encryption contract".

4.4.1.1 Identity Based Re-encryption Contract

The "Re-encryption contract RC" is responsible for "Revocable identity-based proxy re-encryption" for the transfer of records. In our framework, the partial secret keys are created by the "Private Key Generator" (PKG) for the users with identity I by taking "public parameters $params$ and an identity I" as input. The data owner I uses $params$ and his/her "identity I" and encrypt the "message m" to give "original ciphertext CT" as an output that is delivered to the "proxy server PS". The requester R asks for the data encrypted by the owner I and sent the "authentication information φ" to the data owner I. The data owner validates the requester by validating the "authentication informationφ".

If the legitimacy of the requester is approved by the data owner I then data owner will take the "authentication information φ and identity R of the requester" as inputs and returns the “re-encryption key ”which is delivered to the proxy server PS. By using the “re-encryption key” the original ciphertext CT is re-encrypted by the proxy server PS. Which gives “re-encrypted ciphertext CT'” as an output, this is then received by the requester R. The requester returns the message “m'” by decrypting the “re-encrypted ciphertext CT'”.

This process closes itself by updating the revocation list “RL” which is being updated by the PKG and returns the “updated revocation list” [41]. Every time, when a new proxy server is initiated, an RC should be established. RC keeps the records of “proxy node’s address, the re-encrypted keys, re-encrypted ciphertexts CT', decrypted messages $m^{'}$ and revocation list RL”.

4.5 Framework Architecture

The architecture of SPPF is consists of on-blockchain and off-blockchain actions. These actions are illustrated in Figure 3.3 in the previous chapter. The Standard transactions and Internal transactions are on-blockchain actions. The Standard transactions are written on the blockchain. The Qourumchain Consensus algorithm [18] is used to mine these transactions.

When the records are needed to be sent to the smart contracts then the eth-call are used, but they are not written to the blockchain. This permits more efficiency and privacy of the system while deploying the functions required to operate SPPF. The off-blockchain actions include records being transferred through HTTPS or anything which occurs within a node. The hash of the sensitive data is placed on the blockchain by the non blockchain transactions while transferring sensitive data externally. The reason behind placing the hash of the data within the blockchain is to authenticate that either the transactions are legal or not. This helps us in preserving the data integrity provided by the blockchain while enhancing the privacy.

Our scheme SPPF will follow the Ancile [20] for adding a node, Registering a user, for changing access permissions, Adding or retrieving a record. The change will occur in the transfer of records.

4.5.1 Adding a Node

For adding a node in the system we will follow the same steps as given in Ancile framework [20] described in section 3.5.1 and shown in Figure 3.4. In SPPF, students are added to the system as a node in the place of patients which are discussed in Ancile framework. In our design the existing provider system will give the numerical ID's for the students. These ID's will then be used to validate the new student node in the same way in which Ancile is validating the new patient node and add it to the system mentioned in section 3.5.1.

4.5.2 Registering a Student

The process of registering a student for our design will be the same as explained in the section 3.5.2. Before starting this process, it should be confirmed that the student is a registered node in the existing database. After the confirmation of students node status, our framework will follow the steps illustrated in Figure 3.5. This process will allow students to connect with teachers and admin or with any other third party. Registering a student will be needed when SPPF wish to share secure data between two parties.

4.5.3 Changing Access Permissions

In many situations, students may wish to give extra access control to other parties for which they need to complete "changing access permissions" process. The ownership of data possessed by students should be validated first. The main steps of this process in our scheme are same as given in the section 3.5.3 and demonstrated in Figure 3.6. Changing access permissions enables students to give rights to their guardians to access their information.

4.5.4 Adding a Record

For appending a new record the SPPF will follow the same process of adding a record as mentioned in section 3.5.4 and its main steps are illustrated in the Figure 3.7. Here in our design we will use the Electronic student records (ESR) database instead of Electronic health records (EHR) database. The record generated by the provider node can be of any node either student, teacher or admin and it should also be assumed that user and provider have have already connected to each other and have shared OC.

4.5.5 Retrieving a Record

Students in our design can retrieve a record by following the main steps mentioned in section 3.5.5 and demonstrated in Figure 3.8. The retrieval of the record can be done by using any computer or smart phone which increase the interoperability of ESR.

4.5.6 Transfer a Record

It is necessary for any Electronic Student Record (ESR) management system to ensure the privacy and confidentiality during the transferring of records. We use "Revocable identity-based proxy re-encryption (IB-PRES)" to maintain the access control and privacy. Figure 4.1 shows the process of transferring a record between two providers (Provider 1 and Provider 2). Our framework only verifies that who is allowed to share records and with whom records can be shared.

The process starts by Provider 1 who locates the relevant PC. Provider 1 should have transfer or owner level access to transfer the record. Otherwise, PC will stop the process. If Provider 1 has "transfer level access", the PC and CLC intercommunicate with each other to verify the Provider 2 with whom Provider 1 can transfer record.

After this verification, Provider 2 will have a field of permissions in the PC and the IB-PRES process starts as shown in figure 4.1. The step 9 to 13 illustrates the process of IB-PRES.

By using IB-PRES there is no need of sharing a "complete secret key" which increases security. The RC will be in charge of maintaining the IB-PRES and outsourced the partial secret key generated by PKG to PC. The ciphertext is re-encrypted by proxies by using "re-encrypted keys and identities" for the Provider 2. The process of record transfer ended, when from the blockchain Provider 2 receives the "encrypted query link" through HTTPS and "PC address" from the blockchain. Provider 2 may then decrypt the record. Following are the main steps to transfer the records as shown in Figure 4.1:

1. Provider 1 forward the user's ID to its SHC.
2. The SHC returns the applicable OC address.
3. Provider 1 forwards the filename to the OC.
4. The OC returns the address of the applicable PC.
5. The Provider 1 Node then forward the Ethereum address and request level of Provider 2 to the PC.
6. The PC sends a transaction to the CLC to verify Provider 2 is an authorized provider on the system.
7. The CLC returns the verification to the PC.
8. The PC updates its database to give Read access to Provider 2.
9. The PC sends the Partial secret key generated by PKG to the Provider 1 then the Provider 1 generates the re-encryption key.
10. The re-encrypted key and original ciphertext is sent to the PC.
11. The PC adds the re-encrypted key to its database.
12. The PC sends Ethereum address of Provider 2 and the re-encrypted key to the proxy server.

13. The re-encrypted Ciphertext is sent to the PC.

14. The PC forwards the PC address to Provider 2.

15. Over HTTPS, Provider 1 sends the encrypted query link to the Provider 2. Provider 2 may then decrypt the link and retrieve the record.

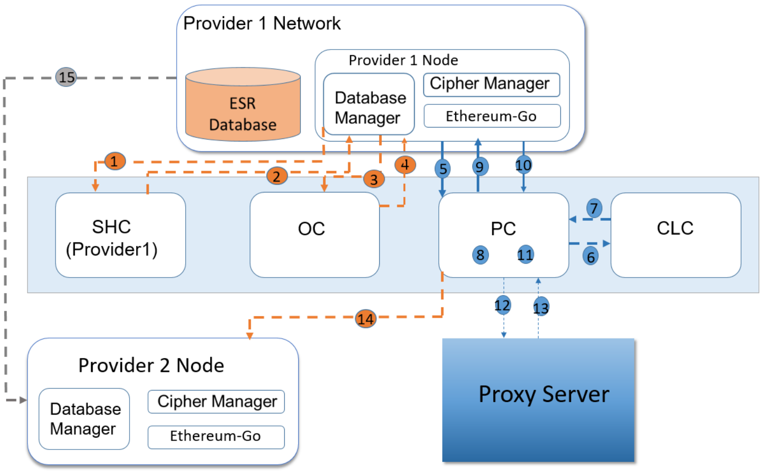

Figure 4.1: The procedure of transfer of record between two providers

Following are the main steps of re-encrypting the data as shown in Figure 4.2 :

1. The PKG generate the partial secret keys and place them on the RC.

2. RC will send this partial secret key to PC.

3. The PC forwards the Ethereum address of re-encryption recipient and the partial secret keys generated by PKG to the owner of data.

4. The data is encrypted by the owner of data who uses his/her identity and give original ciphertext CT.

5. The requester queries the data to the data owner and sent the authentication information φ.

6. After authentication data owner will generate the re-encryption key by taking “authentication information φ and identity R” of the requester as input.

7. The data owner sends “re-encryption key and original ciphertext CT'” to RC.

8. RC sends this re-encryption key and original ciphertext to the proxies.

9. The proxy server re-encrypt the original ciphertext CT by using re-encryption key and output the "re-encrypted ciphertext CT'".

10. This "re-encrypted ciphertext CT'" sent to the RC.

11. Then RC sends it to PC.

12. PC send this "re-encrypted ciphertext CT' and PC address" to the requester.

13. Requester may then decrypt the re-encrypted ciphertext CT' by using his/her secret key.

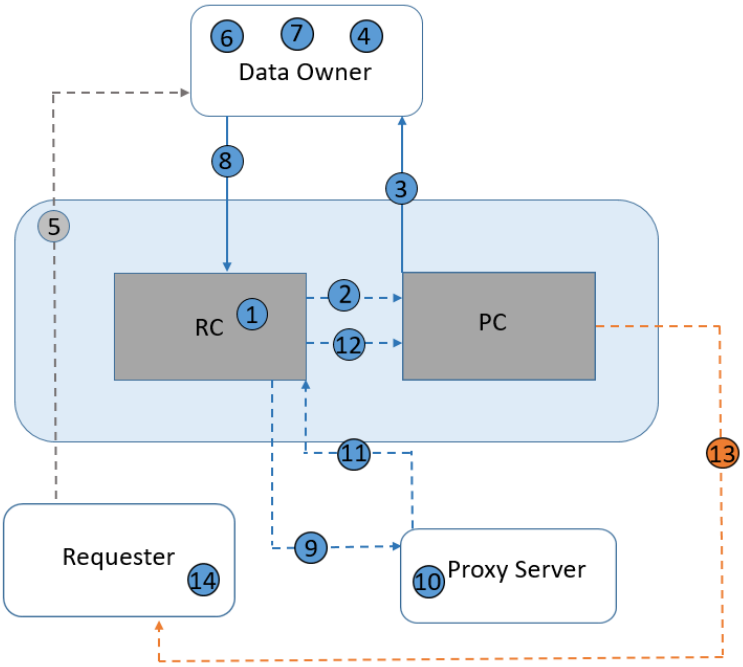

Figure 4.2: The process of re-encrypting the data.

4.6 Comparative Performance Analysis

This section present a "performance analysis" of our scheme compared to Ancile [20] and SPPF on the basis of the transfer of records and re-encryption process. The steps of Ancile to transfer data and process of re-encryption are detailed in

Figure 3.9 and 3.10. In the transfer of records almost all steps are on-blockchain, only step 13 is off-blockchain and in the process of re-encryption all steps except 3, 8 are on-blockchain. The off-blockchain action in transferring a record involves query links and retrieving of record. In the re-encryption process of Ancile, only re-encryption of data involves, and in off-blockchain actions proxies re-encrypt the data.

On the other hand, in our framework SPPF re-encryption process involves a quite few different types of off-blockchain actions which include "authentication of the requester, retrieving data, sending internal transactions to link different contracts and decryption of records". Since SPPF has additional steps, particularly actions happening within blockchain in the re-encryption process and on-blockchain actions in the transfer of records, compared to Ancile. Our scheme is secure against IND-PrID-CPA under DBDH assumption (from theorem given in [41]). Moreover, because of less computation required for our scheme, it is least in communication cost. Furthermore, Our design provides more authentication and user revocation than Ancile. We can conclude that SPPF will have least communication cost as well as permits additional features than Ancile.

4.7 SPPF in the context of Education Sector

Our research specifically focuses on generating a design for nationwide ESR management that fulfills the needs for privacy and security rules.

First of all, ESR management needs a framework which only permits a valid entity to have access to electronic students critical data. SPPF meets this requirement of ESR management system, by utilizing dual "identity-checking" first in PC and the CLC, the second is during the re-encryption process to ensure that critical data is received by the valid users. Furthermore, SPPF can control who may participate in the permissioned blockchain, by utilizing *COC* to validate the entity before the registrations. The concept of authentication of providers, third parties included users in this framework was not proposed by any other framework in the education industry.

In addition to that, SPPF use “Ethereum blockchain protocols and structure” which permits the use of “eth-calls and transactions” to manage storage of each important actions during managing privacy. SPPF confirm the data integrity through the usage of “hashing the query link and ESR itself”.

Lastly, ESR needs security when delivered to other entities other than the student or provider of origin. In SPPF identity-based proxy re-encryption make it possible to transfer the ESR without being decrypted. Generating the partial secret keys, using the proxies and query link enhances the privacy of the system. Furthermore, the use of IB-PRES maintain verification of the secret, the identity of requester and also support the user revocation which other framework does not offer. which helps to increase the interoperability.

4.8 Conclusion

The adoption of ESR management system rises in the recent years. But the transfer of electronic student records securely over the network remains difficult to achieve. Blockchain can provide us with the solution to this issue by giving a single, secure decentralized ledger of student’s data for all users. By implementing a blockchain based ESR system one can allow users to have better control over their critical information and its accessibility.

In this thesis, we get the motivation from the research paper of “Ancile:privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology” [20] which was designed for the security of EHR. Here in our thesis, we have modified the Ancile framework by using it for ESR management system and replacing the entities. We have used Ethereum based blockchain technology in electronic student records management system with smart contracts. This design focuses on the ownership and access control of the student over their personal data. We used “Revocable identity-based proxy re-encryption” [41] which demonstrates our prioritization of security and access control in which the PKG does not generate full secret keys for users. Therefore, the PKG can not decrypt the ciphertext without knowing the secret keys of users.

Although, the implementations of blockchain technology for education are in initial stages but still our design offers significant privacy and data integrity by using different tools like smart contracts and IB-PRES. We look forward to continuing research in the use of blockchain by implementing our proposed design if anyone interested.

Bibliography

[1] R. M. Abobeah, M. M. Ezz, & H. M. Harb, “Public-key cryptography techniques evaluation”. *International Journal of Computer Networks and Applications*, 2(2):14, 2015.

[2] G. Agarwal & S. Singh, “A comparison between public key authority and certification authority for distribution of public key”. *International Journal of Computer Science and Information Technologies*, 1(5):332–336, 2010.

[3] I. Ahmed & M. A. Shilpi, “Blockchain technology a literature survey”. 7: 1490–1493, 2018.

[4] Alexander, “Electronic health records implementation with blockchain, bpm, ecm, and platform”. *http://improving-bpm-systems.blogspot.ch/2016/07/electronic-healthrecords-ehr.html.*, 1:3, 2016.

[5] A. F. C. Alexander Grech, “Blockchain in education”. *JRC Science Hub*, 1: 1–131, 2017.

[6] G. Ateniese, K. Fu, M. Green, & S. Hohenberger, “Improved proxy re-encryption schemes with applications to secure distributed storage”. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):1–30, 2006.

[7] M. Blaze & M. Strauss, “Atomic proxy cryptography”. In *Proc. EuroCrypt’97*, volume 1, page 11. Citeseer, 1998.

[8] E. Blog, “Introducing casper the friendly ghost””. 1:8, 2015.

[9] D. Boneh & M. Franklin, "Identity-based encryption from the weil pairing". In *Advances in CryptologyCRYPTO 2001*, volume 1, pages 213–229. Springer, 2001.

[10] C. Brodersen, B. Kalis, C. Leong, E. Mitchell, E. Pupo, A. Truscott, & L. Accenture, "Blockchain: Securing a new health interoperability experience". *ed: Accenture LLP*, pages 1–10, 2016.

[11] R. G. Brown, J. Carlyle, I. Grigg, & M. Hearn, "Corda: An introduction". *R3 CEV, August*, 1:15, 2016.

[12] V. Buterin, "State tree prunning. ethereum blog". 1:1121–1128, 2015.

[13] V. Buterin, "On public and private blockchains. ethereum blog". 2:1143–1152, 2015.

[14] V. Buterin et al., "A next-generation smart contract and decentralized application platform". *white paper*, 3:37, 2014.

[15] C. Cachin, "Architecture of the hyperledger blockchain fabric". In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, volume 310, page 4, 2016.

[16] R. Canetti, S. Halevi, & J. Katz, "Chosen-ciphertext security from identity-based encryption". In *International Conference on the Theory and Applications of Cryptographic Techniques*, volume 1, pages 207–222. Springer, 2004.

[17] M. Castro, B. Liskov, et al., "Practical byzantine fault tolerance". In *OSDI*, volume 99, pages 173–186, 1999.

[18] J. Chase. "Quorum whitepaper", 2016.

[19] J.-S. Coron, "What is cryptography?". *IEEE security & privacy*, 4(1):70–73, 2006.

[20] G. G. Dagher, J. Mohler, M. Milojkovic, & P. B. Marella, "Ancile: Privacy-preserving framework for access control and interoperability of electronic

health records using blockchain technology". *Sustainable Cities and Society*, 39:283–297, 2018.

[21] Y. Desmedt & J.-J. Quisquater, "Public-key systems based on the difficulty of tampering (Is there a difference between DES and RSA?)". In *Advances in CryptologyCRYPTO86*, volume 1, pages 111–117. Springer, 1987.

[22] W. Diffie & M. Hellman, "New directions in cryptography". *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[23] Y. Dodis, "Proxy cryptography revisited". In *Proc. 10th Annual Network and Distributed System Security Symposium-NDSS'03*, volume 1, pages 1–20, 2003.

[24] A. Ekblaw, A. Azaria, J. D. Halamka, & A. Lippman, "A case study for blockchain in healthcare:medrec prototype for electronic health records and medical research data". In *Proceedings of IEEE open & big data conference*, volume 13, page 13, 2016.

[25] Ethereum, "Step-by-step guide: Getting started with ethereum mist wallet". *https://medium.com/@attores/step-by-step-guide-getting-started-with-ethereum-mist-wallet-772a3cc99af4*, 1:11, 2016.

[26] D. J. Garbade, "What blockchain technology can do for online education". *HACKERNOON*, 1:6, 2018.

[27] Github, "Qourumchain consensus". *https://github.com/jpmorganchase/quorum/wiki/QuorumChain-Consensus.*, 1:7, 2017.

[28] Github, "Go ethereum". *https://github.com/ethereum/go-ethereum/blob/master/README.md.*, 1:2, 2017.

[29] C. Heinrich, "Pretty good privacy (PGP)". In *Encyclopedia of Cryptography and Security*, volume 1, pages 955–958. Springer, 2011.

[30] M. S. Iqbal, S. Singh, & A. Jaiswal, "Symmetric key cryptography: Technological developments in the field". *International Journal of Computer Applications*, 117(15):23–26, 2015.

[31] D. Ivan, “Moving toward a blockchain-based method for the secure storage of patient records”. In *ONC/NIST Use of Blockchain for Healthcare and Research Workshop. Gaithersburg, Maryland, United States: ONC/NIST*, pages 1–11, 2016.

[32] O. Jacobovitz, “Blockchain for identity management”. *The Lynne and William Frankel Center for Computer Science Department of Computer Science. Ben-Gurion University, Beer Sheva Google Scholar*, 1:9, 2016.

[33] M. Jakobsson, “On quorum controlled asymmetric proxy re-encryption”. In *International Workshop on Public Key Cryptography*, volume 1, pages 112–121. Springer, 1999.

[34] D. Johnson, A. Menezes, & S. Vanstone, “The elliptic curve digital signature algorithm (ecdsa)”. *International journal of information security*, 1(1):36–63, 2001.

[35] S. King, “Primecoin: Cryptocurrency with prime number proof-of-work”. *July 7th*, 1:6, 2013.

[36] L. M. Kohnfelder, “Towards a practical public-key cryptosystem”. PhD thesis, Massachusetts Institute of Technology, 1978.

[37] J. Kwon, “Tendermint: Consensus without mining”. *Draft v. 0.6, fall*, 1:11, 2014.

[38] T. Laurence, “Blockchain for dummies”, page 41. John Wiley & Sons, 2017.

[39] X. Li, P. Jiang, T. Chen, X. Luo, & Q. Wen, “A survey on the security of blockchain systems”. *Future Generation Computer Systems*, 151:1–13, 2017.

[40] L. A. Linn & M. B. Koo, “Blockchain for health data and its potential use in health it and health care related research”. In *ONC/NIST Use of Blockchain for Healthcare and Research Workshop. Gaithersburg, Maryland, United States: ONC/NIST*, pages 1–10, 2016.

[41] W. Luo & W. Ma, “A secure revocable identity-based proxy re-encryption scheme for cloud storage”. In *International Conference on Cloud Computing and Security*, volume 1, pages 519–530. Springer, 2018.

[42] U. Maurer & Y. Yacobi, “Non-interactive public-key cryptography”. In *Advances in CryptologyEUROCRYPT91*, volume 1, pages 498–507. Springer, 1991.

[43] D. Mazieres, “The stellar consensus protocol: A federated model for internet-level consensus”. *Stellar Development Foundation*, page 32, 2015.

[44] P. McCorry, S. F. Shahandashti, & F. Hao, “A smart contract for boardroom voting with maximum voter privacy”. In *International Conference on Financial Cryptography and Data Security*, volume 1, pages 357–375. Springer, 2017.

[45] J. MICHAEL, A. COHN, & J. R. BUTCHER, “Blockchain technology”. *The Journal*, 1:7, 2018.

[46] T. Miyamae, T. Honda, M. Tamura, & M. Kawaba, “Performance improvement of the consortium blockchain for financial business applications”. *Journal of Digital Banking*, 2(4):369–378, 2018.

[47] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system.(may 2009)”. *URL: http://www. bitcoin. org/bitcoin. pdf*, 1:1–9, 2009.

[48] A. Narayanan, J. Bonneau, E. Felten, A. Miller, & S. Goldfeder, “Bitcoin and cryptocurrency technologies: a comprehensive introduction”. Princeton University Press, 2016.

[49] K. Peterson, R. Deeduvanu, P. Kanjamala, & K. Boles, “A blockchain-based approach to health information exchange networks”. In *Proc. NIST Workshop Blockchain Healthcare*, volume 1, pages 1–10, 2016.

[50] J. J. Rotman, “A first course in abstract algebra”. Pearson College Division, 2000.

[51] D. Schwartz, N. Youngs, A. Britto, et al., “The ripple protocol consensus algorithm”. *Ripple Labs Inc White Paper*, 5:8, 2014.

[52] A. Shamir et al., “Identity-based cryptosystems and signature schemes”. In *Crypto*, volume 84, pages 47–53. Springer, 1984.

[53] M. M. Skeels & J. Grudin, “When social networks cross boundaries: a case study of workplace use of facebook and linkedin”. In *Proceedings of the ACM 2009 international conference on Supporting group work*, pages 95–104. ACM, 2009.

[54] W. Stallings, “Cryptography and network security: principles and practices”, volume 3. Pearson Education India, 2006.

[55] P. Szilagyi, “Eth/63 fast synchroniation algorithm”. *https://github.com/ethereum/ go-ethereum/pull/1889.*, 89:2, 2015.

[56] H. Tanaka, “A realization scheme for the identity-based cryptosystem”. In *Advances in CryptologyCRYPTO87*, volume 1, pages 340–349. Springer, 2006.

[57] M. Thake, “What is proof of stake”. *Medium*, 1:3, 2018.

[58] S. Tsujii & T. Itoh, “An ID-based cryptosystem based on the discrete logarithm problem”. *IEEE Journal on Selected Areas in Communications*, 7(4): 467–473, 1989.

[59] G. Wood. “Ethereum: A secure decentralized transaction ledger”, 2014.

[60] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger”. *Ethereum project yellow paper*, 151:1–32, 2014.

[61] L. Zhou, M. A. Marsh, F. B. Schneider, & A. Redz, “Distributed blinding for distributed elgamal re-encryption”. In *Distributed Computing Systems, 2005. ICDCS 2005. Proceedings. 25th IEEE International Conference on*, volume 1, pages 824–824. IEEE, 2005.