PRIVACY PRESERVING ALGORITHMS FOR NEWLY EMERGENT COMPUTING ENVIRONMENTS

BY

DOYEL PAL

Bachelor of Science in Computer Science
University of Calcutta
Calcutta, West Bengal
2006

Bachelor of Technology in Information Technology
University of Calcutta
Calcutta, West Bengal
2009

Masters of Technology in Computer Science and Engineering
University of Calcutta
Calcutta, West Bengal
2011

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2015

PRIVACY PRESERVING ALGORITHMS
FOR NEWLY EMERGENT
COMPUTING ENVIRONMENTS

Dissertation Approved:

Tingting Chen

Dissertation Adviser

Johnson P Thomas

Committee Chair

Chris Crick

Weihua Sheng

LIST OF PUBLICATIONS FROM THIS WORK

- Pal, D., Chen, T., Zhong, S., & Khethavath, P. (2014). Designing an Algorithm to Preserve Privacy for Medical Record Linkage With Error-Prone Data. *JMIR medical informatics*, *2*.1(2014).
- Pal, D., Khethavath, P., Thomas, J. P., & Chen, T. (2015). Multilevel Threshold Secret Sharing in Distributed Cloud. In *Security in Computing and Communications* (pp. 13-23). Springer International Publishing.
- Pal, Doyel, et al. "Secure and Privacy Preserving Biometric Authentication Using Watermarking Technique." *Security in Computing and Communications*. Springer International Publishing, 2015. 146-156.
- Khethavath, Praveenkumar, and Doyel Pal. "Privacy Preserving Distributed Cloud Storage." *Journal of Computers*, Volume 10 (2015).

LIST OF POSTERS FROM THIS WORK

- "Privacy Preserving Sequential Pattern Mining Across Multiple Medical Sites". *American Medical Informatics Association (AMIA) Annual Symposium*, November 2015.
- "Privacy Preserving Data Matching Software". *KanSec* workshop, 2013.

ACKNOWLEDGEMENTS

Completion of my PhD has been an amazing experience in my life. It would not have been possible without the guidance and support from many people.

First and foremost I would like to express my sincere gratitude to my advisors, Dr. Johnson P. Thomas and Dr. Tingting Chen, for their valuable guidance, support and kindness. I would like to thank you both for providing with an excellent atmosphere for pursuing my research and encouraging me to successfully complete my PhD.

I would like to thank my PhD committee members, Dr. Subhash Kak, Dr.Weihua Sheng and Dr. Christopher Crick for their encouragement, valuable suggestions, and insightful thoughts. Thank you all for being supportive and guiding me throughout my doctoral studies.

I would like to thank Praveen Khethavath for being a good friend, colleague and a critique to my research work. Your feedback on my research has always been very helpful.

A special thanks to my family for being supportive. I would like to thank my mom (Monica), dad (Sandip), sister (Beau) and brothers (Bubun, Bumba) for always believing in me and encouraging me to follow my dreams. A very special thanks to my darling nieces (Angel, Divine, Sandrine and Selene) for making my life brighter and wonderful.

I would like to thank my friends Jasmine, Tejaswi, Supriya, Narmada, Tanaya and Sourabh for helping me enormously in numerous ways during my doctoral studies.

I cannot thank all of you enough for making my dream come true.

NAME: DOYEL PAL

DATE OF DEGREE: DECEMBER, 2015

TITLE OF STUDY: PRIVACY PRESERVING ALGORITHMS FOR NEWLY

EMERGENT COMPUTING ENVIRONMENTS

MAJOR FIELD: COMPUTER SCIENCE

Abstract: Privacy preserving data usage ensures appropriate usage of data without compromising sensitive information. Data privacy is a primary requirement since customers' data is an asset to any organization and it contains customers' private information. Data seclusion cannot be a solution to keep data private. Data sharing as well as keeping data private is important for different purposes, e.g., company welfare, research, business etc. A broad range of industries where data privacy is mandatory includes healthcare, aviation industry, education system, federal law enforcement, etc.

In this thesis dissertation we focus on data privacy schemes in emerging fields of computer science, namely, health informatics, data mining, distributed cloud, biometrics, and mobile payments. Linking and mining medical records across different medical service providers are important to the enhancement of health care quality. Under HIPAA regulation keeping medical records private is important. In real-world health care databases, records may well contain errors. Linking the error-prone data and preserving data privacy at the same time is very difficult. We introduce a privacy preserving Error-Tolerant Linking Algorithm to enable medical records linkage for error–prone medical records. Mining frequent sequential patterns such as, patient path, treatment pattern, etc., across multiple medical sites helps to improve health care quality and research. We propose a privacy preserving sequential pattern mining scheme across multiple medical sites. In a distributed cloud environment resources are provided by users who are geographically distributed over a large area. Since resources are provided by regular users, data privacy and security are main concerns. We propose a privacy preserving data storage mechanism among different users in a distributed cloud. Managing secret key for encryption is difficult in a distributed cloud. To protect secret key in a distributed cloud we propose a multilevel threshold secret sharing mechanism. Biometric authentication ensures user identity by means of user's biometric traits. Any individual's biometrics should be protected since biometrics are unique and can be stolen or misused by an adversary. We present a secure and privacy preserving biometric authentication scheme using watermarking technique. Mobile payments have become popular with the extensive use of mobile devices. Mobile applications for payments needs to be very secure to perform transactions and at the same time needs to be efficient. We design and develop a mobile application for secure mobile payments. To secure mobile payments we focus on user's biometric authentication as well as secure bank transaction. We propose a novel privacy preserving biometric authentication algorithm for secure mobile payments.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

With the rapid growth of data sharing, online banking, mobile banking, online shopping, data analysis etc., organizations store large amount of data about different entities (e.g., patients, customers, products, students, etc.). Data about different entities has always been a top priority for more than two decades for different purposes such as, company statistics, predictions, data usage trends, etc. The sources of these data can be any industry such as, health care, banks, shopping records, colleges, libraries, etc. Any particular organization not only stores their data privately or performs computations on them but also sometimes share their data with other organizations. Data may contain sensitive information about its entities. Hence for an organization their database poses a threat to the privacy of their entities. Although databases contain sensitive data, sharing data among different organizations is beneficial for different purposes, e.g., company welfare, research, business, etc. To overcome the threat to sensitive data organizations share their data in a privacy preserving way, i.e., without revealing sensitive information to other organizations with whom they share data. Sharing data in a privacy preserving way is a real challenge as it involves sharing data among two or more number of parties without revealing confidential information.

Preserving privacy of data is needed in a wide range of various scenarios in everyday life. Some examples in this wide range of scenarios are outlined below:

- In the health care industry patients' medical record or information may be shared among different health care providers, e.g., hospitals , pharmacies, health insurance carriers, etc. to improve patients' treatment. Under Health Insurance Portability and Accountability Act (HIPAA) regulations [1], preserving patient's privacy is important and no patient's data apart from the requested ones should be disclosed.
- In the education system, the Family Educational Rights and Privacy Act (FERPA) [2] states that any educational agencies and institutions under a program administered by U. S. Department of Education cannot disclose any information about student's grades, enrollment, billing information or behaviors without permission from a parent or eligible student. Therefore no educational institution can disclose any student's data and even if they share student information they should do that in a privacy preserving manner to meet FERPA requirements.
- To investigate a criminal case federal law enforcement agencies, such as the FBI needs information about the suspects from different sources, such as the local police, military, suspects' employer, Department of Motor Vehicle (DMV), and Internal Revenue Service (IRS). Data mining using social networking data analysis is used by criminal investigators to get a picture of the suspects such as their bank accounts, telephone numbers and location. These procedures for criminal investigations need data exchange among different institutions or organizations in a privacy preserving way so that no additional information, such as information about unrelated suspects is revealed.
- In the aviation industry, for safety reasons the Department of Homeland Security (DHS) [3] checks each and every passenger's details on each flight from/to the US. The DHS denies boarding or disembarkation to passengers based on several secret lists such as the Terror Watch List (TWL), No Fly list etc. To do that the aviation industry shares a passenger's sensitive information with DHS. To preserve privacy about innocent passengers' data the

data should be exchanged in such a way so that DHS will only receive details about only those passengers who are on their watch lists.

Modern or emergent computing environments consist of many different paradigms as show in Fig. 1. In this thesis we focus on health care, data mining, biometrics, mobile payments, and distributed cloud computing environments (Fig. 1).

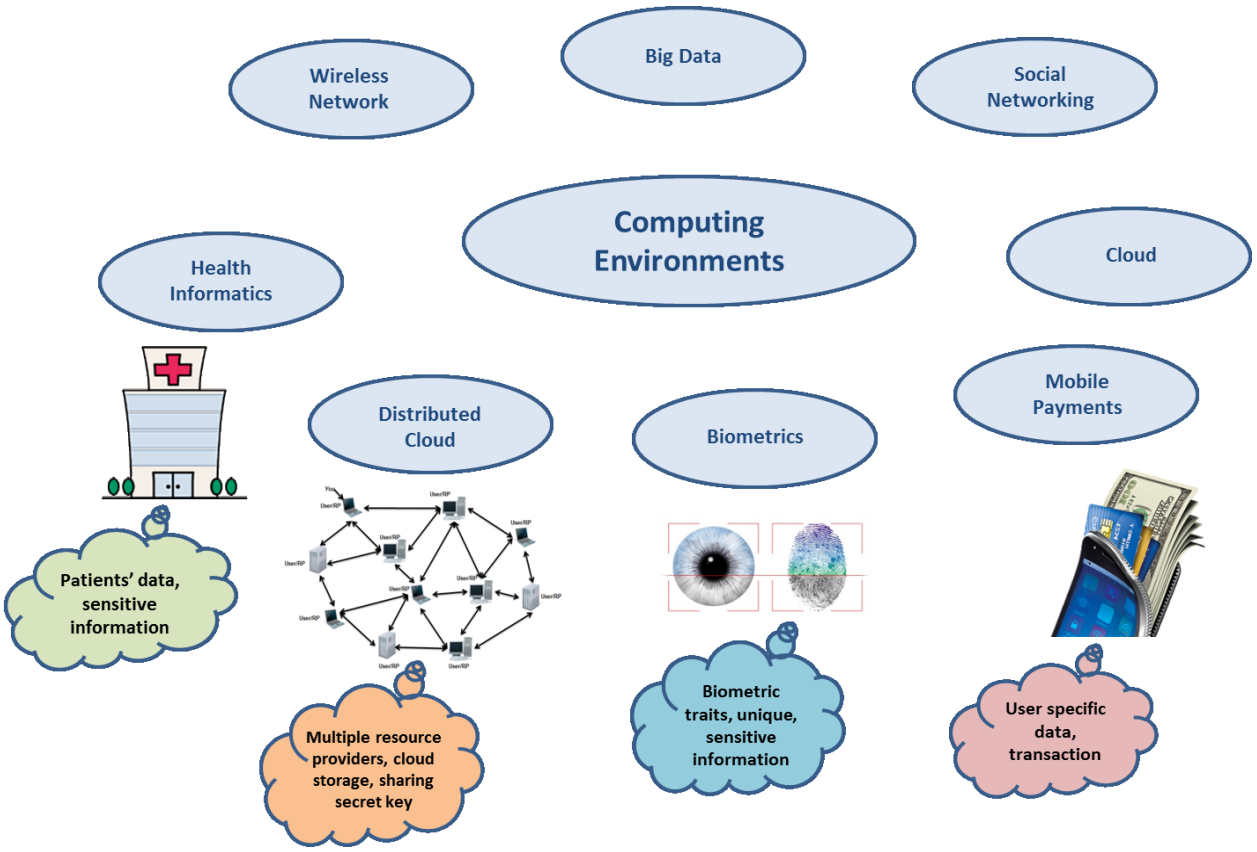

Figure 1: Overview of Computing Environments

We propose privacy preserving algorithms in these computing environments, particularly on privacy preserving techniques in health care, data mining, biometrics, distributed cloud environment and mobile payments environment as shown in Fig. 2.

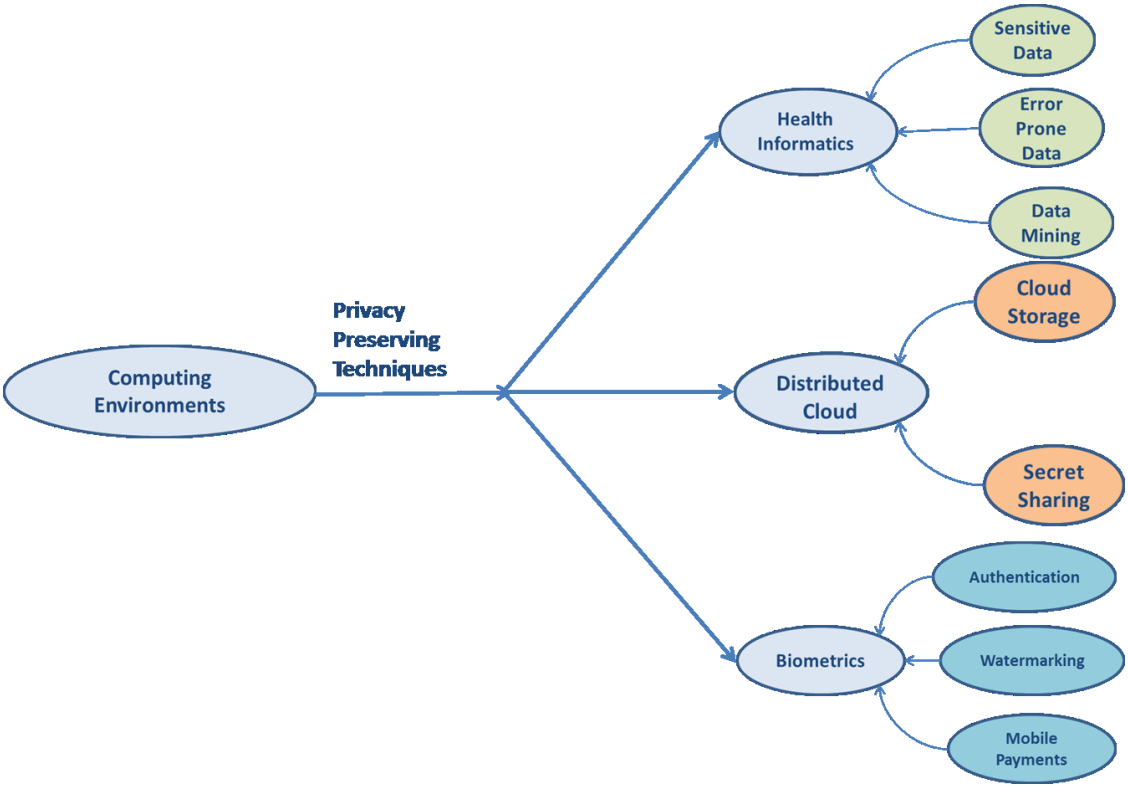

Figure 2: Focus of Computing Environments in this thesis

1.1. MOTIVATION

1.1.1. PRIVACY PRESERVATION IN HEALTH CARE

With the popularity of electronic patient records and the expanded use of medical information systems [4], many different health care providers store medical records of patients electronically. To enhance the quality of health care treatment, often it is required to collect patients' information from different healthcare providers. In many cases different health care providers hold the same patient's data. Public health surveillance often requires linking patient records of the same patient from different health care providers [4]. In [5] to identify whether a particular patient's information is held by more than one health care provider, a matching technique is used on the key attributes of patient's demographic information.

To analyze and identify frequent patterns from medical databases, data mining plays an important role. In health informatics patterns are very common as it contains patterns in symptoms of any disease, patterns in patient paths, patterns in curing methods of any particular disease, patterns in patients' daily activity etc. [61]. Sequential pattern mining from multiple medical institutions databases help to enhance healthcare quality by making predictions and detecting events based on existing patterns in the databases.

In order to monitor the quality of health care treatment provided in a region and to analyze a patient's medication interaction, it is necessary to collect correlated data from different sources such as, clinics, pharmacy, laboratory and health care providers. With the increasing need to keep, link and mine electronic patient records, it becomes very challenging to maintain security and preserve privacy. Under HIPAA regulations, preserving patient's privacy is vitally important. As medical databases contain different identifiers of a patient, e.g., patient's name, surname, date of birth, SSN, contact no., address, etc., using these identifiers for linking or sharing the patterns in patients' data violate patient privacy. Moreover due to privacy, security, and business concerns different health care providers may not be willing to reveal their health data information other than the linking or mining result to the other provider.

1.1.2. PRIVACY PRESERVATION IN DISTRIBUTED CLOUD ENVIRONMENT

Cloud computing is being increasingly used in both business and academic fields. Cloud computing refers to a way of computing over the internet where dynamically scaled shared resources (mostly virtual) are provided as a service to avoid costs of resource over provisioning. Cloud computing [9] is a combination of several concepts from virtualization [62], resource pooling, resource monitoring, dynamic provisioning, utility computing, multi-tenancy and elasticity. All cloud providers provide services defined by their service models, such as, Infrastructure as a Service, Platform as a Service, or Software as a Service. Companies like Amazon, Google, IBM, and Yahoo

are leading providers of some of the services offered by the cloud. Characteristics of cloud such as elasticity, broad network access makes the use of cloud more scalable and cost effective.

A distributed cloud is formed using existing resources and it provides storage and computation resources similar to the existing cloud to users in a distributed peer-to-peer fashion. Distributed cloud has all the characteristics of existing cloud architecture including proper management of resources, scalability and data security. The distributed cloud is more than an existing cloud with multiple data centers distributed in different locations. In the distributed cloud model shown in Fig. 3, resource providers (RP) are distinctive i.e. these distributed cloud servers are individuals with resources to offer. Users of the distributed cloud discover these resource providers and request for using resources. Nodes in the distributed cloud can be both users and resource providers (RP). The distributed cloud uses a completely decentralized mechanism to discover and allocate resources. Network constraints will become less using the distributed cloud which is internet based, because data and resources distributed would be closer and is therefore more

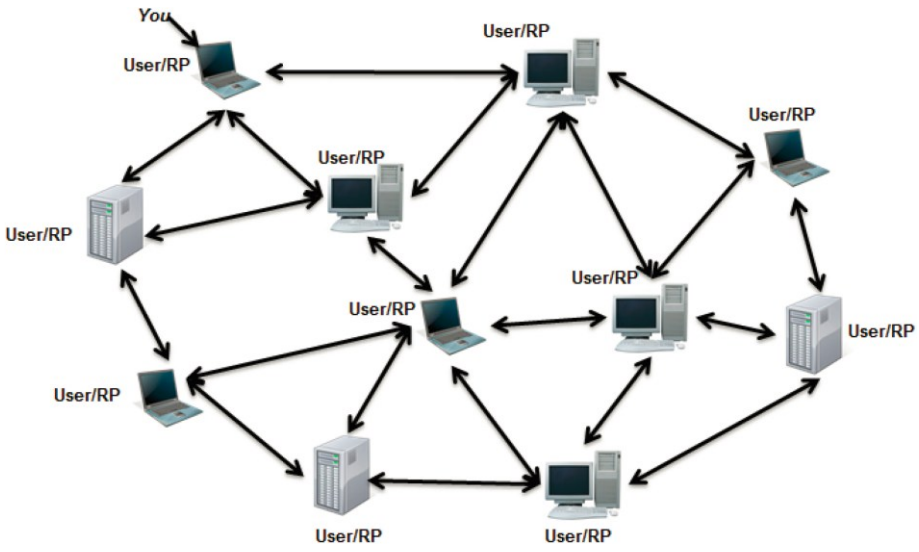

Figure 3: Distributed Cloud Model

likely to be accessed from closer location. Moreover latency would be reduced because resources would be chosen closer to the users. We intend to use multi-valued hash table mechanism for resource discovery and use a game theoretic mechanism for allocating the resources available in a distributed cloud. The distributed cloud is dynamic in nature as users come in and out of the system. In a distributed cloud users store and do computation on resources provided by other users. This requires a secure and privacy preserving mechanism to store and retrieve data. Users of the distributed cloud have control over resource selection which makes them capable of performing encryption techniques to secure data. However protecting and managing the secret keys are of concern in a distributed cloud environment.

1.1.3. PRIVACY PRESERVATION IN BIOMETRIC AUTHENTICATION

Authentication plays an important role in identifying or authenticating a user. Authentication is based on what a user knows (e.g., password, PIN) or what a user has (e.g., token, chip card). However, these can be disclosed, stolen, forgotten or lost. Biometric authentication is a secure way to authenticate any user since it is related to the uniqueness of a user's characteristics such as the physiological or behavioral characteristics of a user (e.g., finger print, iris, face image, handwriting). Though biometric traits are unique and secure, unimodal biometric authentications that uses single biometric trait have some limitations [63, 64]. It is vulnerable to spoof attacks, noise in sensed data, distinctiveness and, non-universality. One way to overcome the limitation is using more than one biometric traits of a user which uses more than one biometric trait. Although multimodal biometrics is secure for authentication purposes, it is still possible to steal or forge when attacked by any determined attacker [65]. Privacy of biometric traits of any individual should be preserved and should not be compromised under any circumstance since it represents their unique traits.

1.1.4. PRIVACY PRESERVATION IN MOBILE PAYMENTS

Mobile payments have become popular with the extensive use of user friendly hand-held mobile devices and easily available wireless networks [6]. Such payment systems are supported by many leading IT industries such as PayPal, Google, Apple etc. Mobile Payments and M-commerce [7] refer to the purchases made by mobile devices such as smart phones. Although mobile devices are widely used for different purposes including mobile payments and mobile banking etc., it has limited computational power and cannot therefore provide a highly secure environment. To enhance security and improve performance, cloud computing [8] can be used for storing large volumes of data and also provide high computational resources. Although biometrics [10] provides more security, its high complexity requires large storage and computational resources. Performing user authentication is one of the most important steps to prevent unauthorized users. There are different authentication mechanisms [11] such as password based, token based, identity based and biometric based. Biometric authentication [12] refers to identification of a user based on their characteristics or behavior. There are different biometric factors such as iris, face, fingerprint and voice recognitions that are currently being used. The cloud's advantages such as scalability and high computing power make biometric mechanisms more feasible and enhance security needed for highly secure applications such as banking. However biometric mechanisms come with privacy issues [13] as biometric information is highly sensitive. Hence we need to have appropriate secure privacy preserving mechanisms when using biometric features.

Different existing mobile payment applications use different mechanisms [14], e.g., account-based, token-based, password-based etc. Any unauthorized user with access to these mobile devices and passwords can exploit these payment mechanisms. So we use a secure and privacy preserving biometric authentication mechanism to access these mobile payment applications. Since Biometric data is highly sensitive we need to protect privacy of user biometrics. Moreover biometric authentication requires high computation power and it is time consuming therefore a light weight

biometric authentication mechanism using the cloud would be a solution to privacy preserving mobile payments authentication mechanism.

1.2. CONTRIBUTIONS

This dissertation topic proposes novel privacy preserving algorithms for newly emergent computing environments, namely privacy preserving techniques in health informatics, data mining, distributed cloud, biometric, authentication, and mobile payments environments. The major contributions are as follows:

- **Privacy Preserving Medical Record Linkage Algorithm with Error-Prone Data:** In the health care industry, often patients' medical record or information has to be shared among different health care providers, e.g., hospitals, pharmacies, health insurance carriers, etc. to improve patients' treatment. Under Health Insurance Portability and Accountability Act (HIPAA) regulations, preserving patient's privacy is important and a health care provider can only disclose a patient's data for whom the data is requested. We have proposed a newly designed privacy preserving linking algorithm named the Error-Tolerant Linking Algorithm that allows the error-prone data to be correctly matched if the distance between the two records is below a threshold. We designed and developed a comprehensive and user-friendly software system that provides privacy preserving record linkage functions for medical service providers.
- **Privacy Preserving Sequential Pattern Mining Across Multiple Medical Sites:** Sequential pattern mining is essential in medical research to identify patterns in patient path, disease treatments, analysis of DNA sequences etc. However, privacy is a concern when mining data across multiple medical sites that contain private, sensitive information. We introduce a privacy preserving sequential pattern mining scheme across multiple medical sites.

- **Multilevel Threshold Secret Sharing in Distributed Cloud:** Threshold secret sharing scheme [99] is widely used to secure different computing environments. It can be used to reduce the risks of managing secret keys. We propose a multilevel threshold secret sharing scheme to provide security of secret key in the distributed cloud.

- **Privacy Preserving Distributed Cloud Storage:** Cloud computing models store data and computing resources in virtual servers located in large data centers' using a centralized architecture. A distributed cloud uses resources provided by users who are geographically distributed over a large area. The distributed cloud is decentralized and resources are provided in a P2P fashion. Since the resources are provided by regular users, security and privacy of data storage and computing resources are main concerns. We propose a privacy preserving storage and file sharing mechanism among different users in a distributed cloud.

- **Privacy Preserving Biometric Authentication using Watermarking Technique:** Biometric authentication ensures user identity by means of users' biometric traits. Though biometrics is unique and secure, it can be still stolen or misused by any adversary. Watermarking is used for content authentication, copyright management and tamper detection. We propose a secure and privacy preserving biometric authentication scheme using a watermarking technique.

- **Privacy Preserving Biometric Authentication in Mobile Payments:** Mobile payments provide additional security than traditional credit cards. Present online authentication mechanisms do not provide adequate security for mobile payments and are vulnerable to several security challenges, e.g., confidentiality, authentication, authorization, integrity etc. There is therefore a need for enhanced and efficient secure privacy preserving mobile payment mechanisms. We present a privacy preserving biometric authentication technique in mobile payments.

1.3 DISSERTATION OUTLINE

The thesis dissertation is organized as follows:

- In chapter 2 we discuss different privacy preserving techniques. We study existing privacy preserving techniques in health informatics, database and data mining, authentication, the cloud and mobile payments environments.

- In chapter 3 we study different existing record linkage schemes in healthcare. We introduce our novel Privacy Preserving Error-Tolerant Linking algorithm. We have designed and developed a comprehensive and user friendly software with a novel privacy preserving algorithm that provides privacy preserving record linkage across multiple medical sites such that it meets HIPAA regulations. Across multiple medical service providers the novel Error-Tolerant Linking Algorithm provides privacy with error-prone data without requiring a trusted third party.

- In chapter 4 we study existing privacy preserving data mining and sequential pattern mining techniques. Chapter 4 also presents a novel algorithm for privacy preserving sequential pattern mining that meets the HIPAA regulations. A salient feature of our algorithm is that it can be applied across multiple medical sites and publishes the common, most frequent sequential patterns from those sites without revealing patients sensitive information. To mine sequential patterns at each site we use the SLPMiner algorithm [81]. Cryptographic techniques and similarity measurement have been used to find frequent sequential patterns across multiple medical sites while protecting the privacy of patients' data.

- Chapter 5 introduces a multilevel threshold secret sharing scheme to enhance the security of secret keys in a distributed cloud environment. We create replicas of secret shares and distribute them among multiple resource providers to ensure availability.

- Chapter 6 presents a privacy preserving storage and file sharing mechanism among different users in a distributed cloud. This mechanism involves processing, storing and retrieving encrypted data in a secure and privacy preserving manner.

- In chapter 7 we propose a secure and privacy preserving biometric authentication scheme using a watermarking technique. We watermark a user's face image with a finger print and encrypt the watermarked biometric to protect its privacy from an adversary. For privacy preserving authentication we use encrypted watermarked biometrics.
- In chapter 8 we introduce a secure privacy preserving biometric authenticated mobile payment mechanism. We propose a novel algorithm named Privacy-Preserving Biometric-based Authentication Algorithm to protect user biometrics. We also design and develop a privacy preserving mobile payments application which consists of privacy preserving log in and secure transaction. For secure transaction we incorporate the RSA scheme [15] on a direct debit method [14, 16, 58].
- In chapter 9 we present conclusion from our research work and research works that we intend to do in future.

CHAPTER II

PRIVACY PRESERVATION TECHNIQUES

In this thesis dissertation we concentrate on cryptographic approaches to privacy preserving. The different types of cryptographic privacy preserving techniques are homomorphic encryption, secure multi-party computation, oblivious transfer, order- preserving encryption, zero – knowledge proofs, oblivious keyword search, searchable encryption, private information retrieval, differential privacy, frequency preserving encryption etc. Some of these techniques are discussed below:

2.1. HOMOMORPHIC ENCRYPTION

Homomorphic encryption schemes allow specific types of computations on cipher text and supports homomorphism. In this technique, arithmetic computation on the cipher text generates encrypted result and when decrypted the result matches with the result of arithmetic operations on plain texts without inherent loss because of the encryption. In group theory homomorphism is a mapping $\varphi : G \to H$, given $(G,\star)$ and $(H,\diamond)$ such that

$$\forall x, y \in G, \qquad \varphi(x \star y) = \varphi(x) \diamond \varphi(y)$$

Whereas in ring theory, homomorphism is a mapping $\varphi : R \to S$ that respects both addition and multiplication. Therefore,

$$\forall x, y \in R, \qquad \varphi(x + y) = \varphi(x) + \varphi(y) \; and \; \varphi(xy) = \varphi(x)\,\varphi(y)$$

The homomorphic encryption schemes which are limited by the preservation of a single operation such as a group homomorphism is known as partially homomorphic encryption and those which preserves both of the addition and multiplication, such as ring homomorphism is known as fully homomorphic encryption. Several cryptosystems, e.g., ElGamal, RSA, Paillier, Benaloh, etc. [15, 17, 18] support the partially homomorphism, that is, these cryptosystems preserves the structure addition or multiplication but cannot do both at the same time. For example, RSA and ElGamal [15, 17] supports the multiplicative homomorphic property, i.e., if $E(x)$denotes the ciphertext of plain text x then according to multiplicative homomorphism, $E(x_1) \cdot E(x_2) = E(x_1 \cdot x_2)$. Whereas the additive homomorphic property of Paillier cryptosystem [18] denotes that, $E(x_1) \cdot E(x_2) = E(x_1 + x_2)$. Several applications, such as, E-Voting, E-Cash, cloud computing, private information retrieval use partially homomorphic encryption. The fully homomorphic property preserves the addition, multiplication structure of ring homomorphism modulo 2. Using fully homomorphic encryption any arbitrary computation can be performed on cipher texts preserving the encryption. In 2009, Craig Gentry proposed the first fully homomorphic encryption scheme [19] using ideal lattices structure in ring theory. In this scheme the plaintext is injected with a certain level of noise during encryption. Each operation on the cipher texts results in compounding the noise and the compounding noise yields the loss of homomorphic property after certain number of operations. Although the noise production followed by noise reduction makes the scheme impractical, it was a significant breakthrough in cryptography. The fully homomorphic property can be applied on several scenarios, including private queries on search engines and cloud computing. In our thesis we exploit the partially homomorphic property of the ElGamal cryptosystem and Paillier cryptosystem to apply it to health care and mobile payments environment respectively.

2.2. SECURE MULTI-PARTY COMPUTATION

The secure multi-party computation relies on the secure two-party computation introduced by Yao [20] where two parties can compute a function f on their inputs x, y while keeping their inputs private. No party can learn anything about the other party's input and the adversary is semi-honest. This constant-round protocol privately computes the probabilistic polynomial-time functionality f on the inputs and uses a technique called garbled circuits. For example, Part A has input x and Party B has input y and the protocol works by having one of the parties, say Party A, first generates an "encrypted" or "garbled" circuit $f(x,.)$ which is then transferred to Party B who obtains a series of keys for his input using a private 1-out-of-2 Oblivious Transfer Protocol of $|y|$ instances. Nothing can be learned at this stage from the encrypted circuit but Party B can obtain the output $f(x, y)$ by "decrypting" the circuit. The decryption is partial here to ensure that nothing is learned beyond the output $f(x, y)$.

2.3. OBLIVIOUS TRANSFER

In this cryptographic approach the sender sends one of the many potentially pieces of information and the receiver comes to know only about the information. Receiver remains oblivious to what piece of information has been transferred. Suppose the sender has the information $x_1, x_2, \dots, x_n$ and the receiver wants to retrieve x_i . This protocol ensures that the sender does not learn which message is received and the receiver learns no other message. This technique is applied for exchanging secrets, private information retrieval, protocol for signing contracts and electronic cash scheme. This scheme was first introduced by Rabin [137].

2.4. ORDER PRESERVING ENCRYPTION

Conventional cryptographic techniques degrade the performance of database systems as cipher texts do not preserve the numerical orders of the corresponding plaintexts and therefore B-trees can no longer be used for any range queries [21]. Order preserving encryption techniques preserve the same order of the cipher texts as the order of the plain texts [21, 22]. The idea behind this type of

encryption technique is to apply range queries, indexing, query processing, binary search on encrypted data as exactly and efficiently as it can be done on unencrypted data. Order preserving schemes are often used in dictionary based string compression [23, 24]. Besides the conventional database application of order preserving encryption, it is also used in cloud computing for secure ranked keyword search over encrypted cloud data [25].

2.5. ZERO KNOWLEDGE PROOF

Zero knowledge proof (ZKP) [26, 27] is a public cryptosystem to make the data secure against chosen cipher text attack. In this technique one party, say Party A proves to another party, say Party B that a statement is true. In the whole procedure of proving, Party A does not convey any other information apart from the fact that the statement is true and Party B also cannot learn anything as a result. Therefore Party B gains zero knowledge. It forces any malicious user to behave according to a predetermined protocol. Provided that one way functions exist, ZKP exists for any NP set.

To preserve privacy many techniques have been developed and applied in different fields, including health informatics, database, data mining, cloud computing, authentication. [1, 28] illustrate some privacy approaches in health information exchange. One obvious approach is to link data using the identifiers in encrypted format, e.g., [29, 30, 31, 32]. An elegant approach to encrypt identifiers is using one way hash functions as in [31, 32]. To ensure security these methods are based on the irreversible transformation property of one way hash functions on identification data. These methods are vulnerable to some common cryptographic attacks. In [32] the authors propose a computerized hash encoding and anonymous record linkage procedure on medical information. To consolidate security against dictionary attack [32], two pads are used, one for the sender and the information and the other one for the recipient. Some other approaches have been proposed as well towards privacy preserving medical records linkage algorithms [33, 34]. A trusted third party has been used in [34] to make the algorithm more secure. Here each party is involved in computing the

set of bigrams for each string. Each party exchanges the power set of encrypted bigrams with the trusted third party and then string similarity is performed using Dice co-efficient. However these approaches usually have high false negative rates. Using indirect pseudonym identifiers [33], besides giving the patients control over what information is revealed about them, an architecture has been proposed to link medical records.

Privacy preserving data matching algorithms have been proposed in database and data mining. In [28], the authors propose a solution where two parties can run data mining algorithms on the union of their own confidential databases, without revealing any unnecessary information. In this particular solution the authors focused on the problem of decision tree learning as the input sizes of data mining algorithms are huge and the data mining algorithms themselves are very complex. At each party's end, this method uses a computation of the same order as computing the ID3 algorithm on its own databases. It combines the result using cryptographic tools. Some solutions of privacy preserving record linkage are based on the perturbed information. For example, [35] uses randomizing functions such as the Gaussian function or Uniform perturbations to perturb the sensitive data and builds a decision tree classifier from these perturbed data. This solution offers a reconstruction procedure to accurately estimate the original data value distribution. The cryptographic technique which relies on the Secure Multi-party Computation (SMC) Protocol [36] compute functions over private inputs. [37] proposes a more efficient protocol based on cryptographic techniques which preserves privacy of database schemas. Here the authors consider the scenario that two parties want to link their data in string format and can have different schemas. They propose a protocol which consists of a data matching protocol and a schema matching protocol. The protocol builds an embedding space and two parties embed their data strings using a variant of Sparse Map [38] ensuring the contractiveness of the embedding. A semi-honest third party collects the embedded strings from the two parties and computes the similarities. Whereas [35] concentrates on exact matching, [37] focuses on approximate matching. A hybrid method that

combines both the data perturbation and cryptographic techniques is presented in [39]. The basic idea of this method is to first classify the data into two classes as matches and mismatches, and then apply the general SMC protocol to compute the distance for the records in the matches class. The problem with this method is that general SMC protocols are costly to use in practice. In [37, 39] the proposed solutions use a trusted third party which is a major issue since a web-based trusted third party may not be a good choice to link records in a privacy preserving way.

Some recent works [40, 41, 42, 43] focus on security and privacy in biological services and in medical data. By secure encapsulation and publishing of bioinformatics software in a cloud computing environment, the authors have derived a prototype system of the biological cloud in [40]. While they worked only on biological services and focused on achieving a prototype system of the biological cloud, our solution works on different databases and concentrates on linking these different databases in a privacy preserving manner. In [41], the authors have discussed medical data sharing by preserving privacy and data utility. They have given a clear picture about the data that generally is used for data sharing purposes, different techniques for privacy preserving data sharing and different types of threats. A new algorithm has been proposed in [43] for heterogeneous health data sharing in a privacy preserving manner. The proposed algorithm considers health data containing both relational and set-valued data and accomplishes $\in$ - differential privacy. In [41] the authors have discussed different types of medical data, e.g., Demographics, Clinical Information, Text and Genomic Information and [43] works on heterogeneous medical data. In our solution, instead of different categories of medical data and heterogeneous health data we concentrate on textual and numeric data and further categorize them into error-free data and error-prone data. [42] focuses on privacy preserving interactive record linkage. The authors have given a solution by proposing a computer-based third-party record linkage platform, Secure Decoupled Linkage (SDLink). The proposed solution decouples the data, obfuscates it and shares minimum information via encryption, chaffing and recoding respectively, to ensure protection against attribute disclosure.

A new computer-based third-party record linkage platform has been proposed in [42] but our proposed solution does not need a trusted third party.

Over the past few years biometric techniques have gained attention as a reliable means of authentication. Several authentication techniques based on different biometric traits, e.g., fingerprint, face, etc. [56, 66, 67, 68, 69] have been proposed. A cryptographic privacy enhancing technology for face recognition has been proposed in [66]. A solution for two party problems [66] has been presented using the standard Eigen face recognition system, secure multiparty computation and Paillier encryption. In [56] the authors have proposed a privacy preserving protocol for fingerprint–based authentication using multiparty computation and homomorphic encryption. A privacy preserving cloud–based and cross–enterprise biometric identification with zero data disclosure possibility has been presented in [67]. Instead of using a distance based matching approach; this technique performs the identification process in an encrypted domain by creating encrypted search queries using a k–d tree structure. [68] presents a multimodal biometric authentication system using Commercial Off–the–Shelf (COTS) fingerprint and face images. A new normalization and fusion method have been introduced to accomplish match score level fusion of multimodal biometrics. In [69] the authors have proposed a robust multimodal authentication scheme integrating three biometric modalities: iris, face, and palm print with three different fusion strategies.

Security is an important concern for cloud architecture as cloud storage is vulnerable to security threats. Protecting data storage and key management are two of the most important concerns. [70-74] concentrate on cloud storage security and secure key management schemes in different cloud environments. Cloudstash [70] concentrates on providing the security of cloud storage. It considers the file as secret and applies secret sharing scheme directly on the multiple shares of secrets. This scheme hashes and signs each share of file and then distributes these signed multi shares into multi clouds. To overcome the limitations of a single domain cloud, a new second layer in the dependable

cloud computing stack named Intercloud has been introduced in [71, 113]. This scheme performs symmetric encryption on the data and splits the key into shares using a secret sharing scheme. It attaches the key shares as metadata to the pieces of data and distributes them to the clouds. The N cloud scheme [72] proposes an improved cloud storage scheme in terms of availability, performance and confidentiality in the cloud. It splits the file in many chunks and uploads the encrypted file chunks in geographically separated cloud storages in parallel. It also replicates the chunks in a non-overlapping manner into many cloud storages. To reconstruct the file it downloads the chunks, decrypts them and reassembles them back to a file. This technique manages the encryption keys at the client side and does not save them in the cloud. To overcome the limitation of a single cloud, the DepSky system [73] builds a cloud of clouds named Intercloud, on top of a set of storage clouds by using a combination of diverse commercial clouds. It uses the combination of Byzantine quorum system protocols, cryptographic secret sharing, erasure codes and diversity of several clouds to improve the availability and confidentiality provided by the commercial storage cloud. The CloudSeal scheme [74] provides an end to end content confidentiality protection mechanism for large scale content storage. It uses integrated symmetric encryption, threshold secret sharing, and a proxy based re-encryption scheme to protect the content and to manage user access. It also supports forward and backward security. Though many privacy preserving mechanisms have been proposed for the cloud environment, privacy preserving techniques in a distributed cloud has not been studied much.

Mobile payments have gained popularity and we need to have an efficient, secure and privacy preserving payment protocol. There are many schemes for mobile payments which try to solve one of the issues such as efficiency, security or privacy with mobile payments. A secure account-based payment protocol [44] proposed for wireless networks, makes use of symmetric key properties which reduces computations. Their protocol increases transaction security compared to a SET [45] based approach. Use of a highly secure, multi-factor authentication scheme can be expensive in

mobile devices. In [46], the author has proposed a multi-factor security protocol for wireless payments for J2MEE based clients and J2EE based servers. This paper utilizes a TIC code and SMS to authenticate the transactions and assumes the mobile device which has the TIC codes is secure. The only security for applications provided is username and password. The rest of the payments transactions is based on TIC codes and SMS. S. Karnuskos et al [47], presented a secure mobile payment service as a part of the SEMPOS project which can handle security and privacy of users. The SEMPSO project makes use of mobile network operators and data centers to be part of the payment mechanism. In [48] the author utilizes a NFC based solution for managing payments. Since there are many parties involved for transactions, a cloud based approach is used. Secure elements which are attack resistant microcontroller are downloaded to the mobile device when a user wants to make payments and after user authorization, the transaction is executed and the secure elements are deleted from the device for security purposes. This architecture was envisioned to replace debit and credit cards in the future. The existing methods for mobile payments are device specific as in [46, 48]. In [46, 48] the payment technology can be applied only for devices with J2MEE and NFC enabled devices. Existing payment techniques such as NFC based solutions or google wallet are similar to traditional credit cards and are provided for ease of customer and are still prone to thefts and various attacks.

This review shows that existing privacy preserving medical record linkage algorithms do not allow errors in data and uses a trusted third party. In the real world data errors in medical record is common. In our work we address these drawbacks by proposing a novel error tolerant record linkage algorithm which works on error–free and error–prone data and does not need a trusted third party. SLPMiner algorithm is one of the most efficient algorithms for sequential pattern mining algorithm [78] which is based on the PrefixSpan algorithm. Though many privacy preserving sequential pattern mining algorithms have been proposed, no research has been done on privacy preserving property of the SLPMiner algorithm. In our thesis dissertation we introduce the privacy

preserving property of the SLPMiner algorithm that can be applied across multiple medical sites. Key management and data protection are two important concerns in the cloud and distributed cloud environment. From our review it is evident that cloud storage and key management are vulnerable to security threats in a distributed cloud environment and has not been studied much. We propose novel privacy preserving schemes to protect data and the secret key in a distributed cloud environment. Biometric authentication is a reliable technique to verify a genuine user. However biometrics is not secure when it is attacked by any determined adversary. In this thesis dissertation we propose a privacy preserving biometric authentication technique using the watermarking technique. We introduce a secure mobile payments scheme which uses our novel privacy preserving biometric authentication algorithm.

CHAPTER III

DESIGNING AN ALGORITHM TO PRESERVE PRIVACY FOR MEDICAL RECORD LINKAGE WITH ERROR-PRONE DATA

Linking medical records across different medical service providers is important to the enhancement of health care quality and public health surveillance. In records linkage, protecting patients privacy is a primary requirement. In real-world health care databases, records may contain errors due to various reasons such as typos. Linking the error-prone data and preserving data privacy at the same time are very difficult. Existing privacy preserving solutions for this problem are only restricted to textual data.

3.1. LITERATURE REVIEW

Existing works [52] do not meet all the requirements of medical record linkage all the time in practice. For instance in earlier work [29] on data record linkage, identifiers were transmitted in encrypted format. This does not allow any kind of error in identifiers, but this may happen frequently in real-cases. Spelling mistakes and typographical errors are very common in databases. Some researches [41, 42, 49, 50, 51] have focused on error-prone data and on missing data. In [42] the authors have proposed a solution to build cross-site records and link data for a particular patient as she moves between participating sites. They consider the hypothesis that most variation in names occur after the first two letters. The date of birth is considered to be one of the most reliable attributes. The composite identifier is generated based on real identifiers in such a way that the possibility of identifying a common patient is maximized. This composite identifier is the hashed string of the first two letters of the patient's first name and last name plus

their date of birth. Considering this composite identifier they have shown that it has a higher sensitivity rate compared to other identifiers, e.g., SSN and identifier based on patient's full name and date of birth.

Most of the existing algorithms for error- prone data are concentrated on textual data. They are very useful for linking records for any customer identifying information. Some approaches towards error-prone data in privacy preserving record linkage have been proposed. One of these proposed solutions uses Bloom Filters [49]. This solution applies Bloom Filters with Hash–based Message Authentication Code (HMACS) on q-grams of identifiers and allows errors in identifiers. Compared to other privacy preserving record linkage methods with encrypted string type identifiers these methods have lower false negative rates. However, the existing proposed solutions of this category are designed for textual data. On the other hand privacy preserving record linkage for error-prone numeric data is also very important. For instance, medical records usually contain ample numerical attributes, such as the patient's blood pressure, height, weight, and other test results. In different medical databases, medical data may be stored at different precisions. Then even two very close numbers (e.g., 392.1 and 392.11) may cause a totally negative linking result. The consequence of high false negative results may be very harmful, especially when querying a patient's records for emergency treatments.

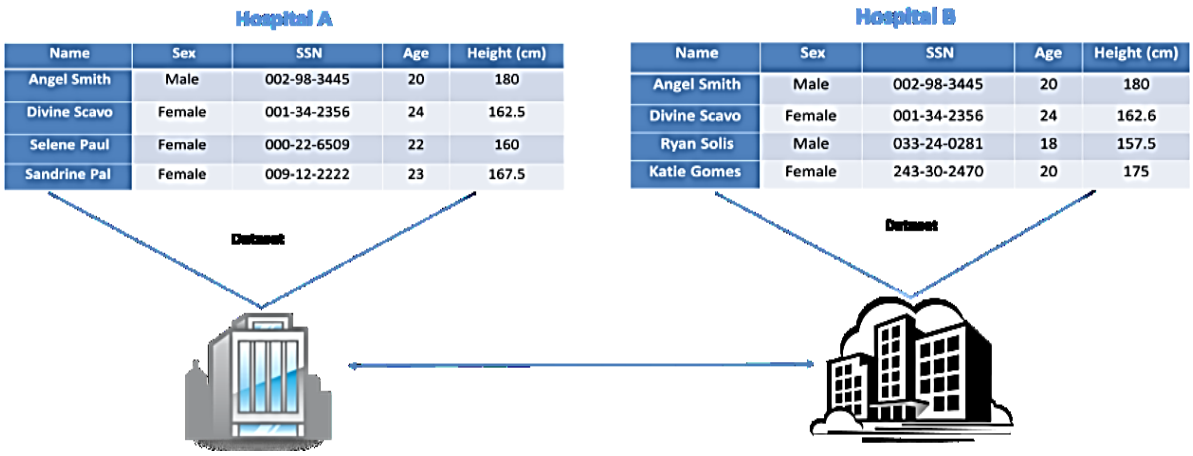

Figure 4: Privacy Preserving Record Linkage Problem

In this work we aim to address the privacy preserving record linkage problem with the presence of error-prone data. In Fig. 4, we illustrate an example of a real world case where privacy preserving record linkage for error-prone data is needed. In this figure, each of the two hospitals have their own database of patients' information. They want to find out the common patients they share (e.g., *Angel Smith* and *Divine Scavo*) in order to share information about the patients. However, due to HIPAA requirements, they cannot exchange data in clear texts. Moreover, we notice that for patient *Divine Scavo*, all attributes are the same at both hospitals except the height (one is 162.5 cm and the other is 162.6 cm). If traditional cryptographic schemes are used, the record belonging to the same patient will be labeled as a mismatch, leading to an error result. In order to avoid the mismatch for the records belonging to the same patient, we need new software to enable privacy preserving record linkage for error-prone data. Such work has not been reported in the literature.

3.2. DESIGN CONSIDERATION

Assume the privacy preserving record linkage takes place between two medical organizations. Each organization holds information about its entities, e.g., patients/customers. Along with the different entities both of these organizations have some common entities too. It is very difficult to get the data of only these common entities from the entire dataset of two organizations and preserving privacy at the same time.

For example, suppose privacy preserving record linkage takes place between two hospitals, *Hospital A* and *Hospital B*. Each of these hospitals maintains detail information about patients, such as, patient's name, date of birth, address, SSN, sex, etc. as shown in Fig. 5. *Hospital A* has four patients, *Angel Smith, Divine Scavo, Selene Paul* and *Sandrine Pal* and *Hospital B* has four patients, *Angel Smith, Divine Scavo, Ryan Solis* and *Katie Gomes*. All the information of patient *Angel Smith*

in *Hospital A* matches with the patient *Angel Smith* in *Hospital B*. Some of the information, i.e., name, address, SSN, age and sex of patient *Divine Scavo* in *Hospital A* matches with patient *Divine Scavo* in *Hospital B*. Assume that *Hospital A* takes the initiative of record linkage and *Hospital B* is the participating organization as it participates in the record linkage. (We use the terms initiating organization and participating organization in this chapter.) During this entire procedure it is implicit that *Hospital B* agrees to share its patients database with *Hospital A* without revealing any sensitive information about the patients. *Hospital A* will get *Angel Smith* as matched data, *Divine Scavo* as partially matched data and *Selene Paul* and *Sandrine Pal* as mismatched data. Matched data means that the data *Hospital A* and *Hospital B match*, mismatched data means data that belongs to *Hospital A* and *Hospital B do not match* and partially matched data means data for an entity of which some of the attributes match at both hospital ends.

Hospital A

Name	Address	Sex	SSN	Age	Body Mass Index	Fasting Blood Sugar
Angel Smith	74 S University Place	Male	002-98-3445	20	24.8	70
Divine Scavo	214 S. West St.	Female	001-34-2254	24	23.3	85
Selene Paul	222 N. Duck St.	Female	000-22-6509	22	22.5	70
Sandrine Pal	200 S. Duncan St.	Female	009-12-2222	23	23.0	72

Hospital B

Name	Address	Sex	SSN	Age	Body Mass Index	Fasting Blood Sugar
Angel Smith	74 S University Place	Male	002-98-3445	20	24.8	70
Divine Scavo	214 S. West St.	Female	001-34-2254	24	23.4	86
Ryan Solis	242 Jade Clover Lane	Male	033-24-0281	18	21.5	71
Katie Gomes	201 Amethyst Lane	Female	243-30-2470	20	24.0	75

Figure 5: Data from two hospitals

Errors in database data are common. Therefore privacy preserving record linkage for error-prone data is necessary. For error-prone numerical data we can formulate the problem as follows. We assume that for any client, the common part of her records stored in both entities has n attributes. The goal of the linkage is to find out the records held by party *B* that are within a small distance (or very close) to the record hold by party *A*. Formally, the problem for error-prone data (i.e., privacy preserving error-tolerant linkage) is defined as follows. Given two databases D_A $(a_1, a_2, \ldots\ldots, a_m)$

and D_B $(b_1, b_2, \ldots\ldots, b_m)$ with the same attributes where a_i and b_i is the record in database D_A and D_B respectively. The error-tolerant linkage function takes a tuple < a, D_A, τ > as input, where a is any record in D_A and τ is the distance threshold and outputs a vector of boolean numbers, $(r_1, r_2, \cdots, r_m)$, where $\forall i$ s.t., $1 \leq i \leq m$,

$$r_i = \begin{cases} 1 & Dist(a_i, b_i) \leq \tau \\ 0 & otherwise \end{cases} \quad (1)$$

in which $Dist(i, j)$ is the distance function defined for input records between database records i and j. The Euclidean distance function is used in our work. Privacy preserving error-tolerant linkage guarantees that computing the error-tolerant linkage function is secure in the semi-honest model [27, 28], without a trusted third party.

By being secure in the semi-honest model, the two parties (or any other adversary) cannot efficiently obtain more information than the input and the output of the algorithm. In particular, for our error-tolerant linking algorithm, the two parties will know only the output $(r_1, r_2, \cdots, r_m)$ and no information about the values of records (either linked or not-linked) will be revealed.

3.3. SCHEMES

The main goal of our approach is that if the participating organization sends the entire dataset as encrypted format to the initiating organization then it is not possible for any other third party as well as the initiating organization to know the data values or data fields of the participating organization if the key-pair is unknown. In our proposed solution, to send data confidentially to the other party we considered three cryptographic schemes, i.e., SHA-1, SHA-2 and Error-Tolerant Linking Algorithm. Before discussing the schemes in detail we categorize the data into two different data categories: error-free data and the error-prone data. SHA-1, and SHA-2 are basic cryptographic schemes for privacy preserving error-free data linkage and the Error-Tolerant Linking Algorithm is for error-prone data.

The workflow of our scheme is as follows. To encrypt the data, the initiating organization chooses a dataset name and cryptographic scheme and sends both of them to the participating organization. If the participating organization holds the same dataset it starts the privacy preserving data linkage process by encrypting the data using the cryptographic scheme send to it by the initiating org. It sends data in cipher text format to the initiating organization. Meanwhile the initiating organization encrypts its own dataset by using the cryptographic scheme. After receiving data the initiating organization applies the privacy preserving matching scheme to obtain the results, i.e., to determine if the data sets match, mismatch and or partially match. The workflow of the solution is shown in Fig. 6.

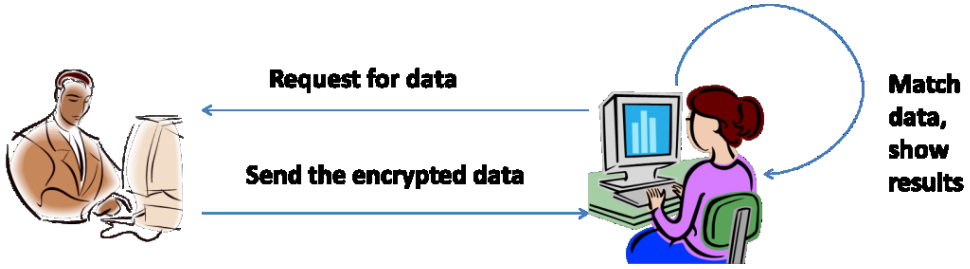

Figure 6: Overall flow of the solution

We will discuss about the Error-Tolerant Linking Algorithm in detail and two existing cryptographic schemes briefly in the following two subsections respectively.

3.3.1. SCHEME FOR ERROR-PRONE DATA

We propose a new solution for the error-prone data, named an Error-Tolerant Linking Algorithm. The Error-Tolerant Linking Algorithm uses the ElGamal[4] scheme as the basic building block. In this subsection, we review the ElGamal scheme first and then will describe the Error–Tolerant Linking Algorithm in detail. ElGamal is a public key encryption scheme. Let G be a cyclic group of prime order p with generator g. A value $x \in Z_p$ is randomly chosen as the private key. The corresponding public key is (p, g, h), where h $= g^x$. To encrypt the message m, a value r $\in Z^p$

is randomly chosen. Then the cipher text is $E(m) = (C_1, C_2) = (g^r, m.h^r)$. (We use $E(m)$ to denote the ciphertext of m encrypted by ElGamal scheme.) The decryption is as follows.

$$\frac{C_2}{{C_1}^x} = \frac{m.h^r}{g^{x.r}} = \frac{m.h^r}{h^r} = m \tag{2}$$

The Difficulty of computing discrete logarithms over finite fields forms the basis for security in ElGamal. To decrypt a cipher text any adversary would have to get the one time random integer. Determining this random integer is infeasible as it requires computing of discrete logarithms.

Error-Tolerant Linking Algorithm exploits the homomorphic property of the ElGamal scheme. That is, for two messages m_1 and m_2 it satisfies the following property:

$$E(m_1.m_2) = E(m_1).E(m_2) \tag{3}$$

In addition to linking the data from two different organizations, the Error-Tolerant Linking Algorithm preserves privacy as well. We assume that the attributes of records are preprocessed and converted to integers beforehand. For numerical attributes, this preprocessing is straightforward by normalizing the original values to integers within a certain range. For attributes consisting of strings, we can use a preprocessing method to convert the strings into integers so that the integers can still keep the distance between the records. Our algorithm is applied afterwards to complete the records linkage. This algorithm allows the input record with minor deviations less than a small threshold τ. The threshold value is to calculate the distance between the identifiers of two records. In this algorithm, neither entity can learn the records of the other's patients.

Algorithm 3.1 shows the details of our privacy-preserving Error-Tolerant Linking Algorithm. First, party *A* generates a pair of keys for ElGamal scheme and sends the public key to party *B*. For each attribute $a[j]$ in the record, party *A* computes $g^{a[j]}$ and $g^{((a[j])^2)}$, and sends the cipher texts of these terms to party *B*. For each record b_i held by *B*, *B* computes $g^{((b_i[j])^2)}$ for each attribute $b_i[j]$, and encrypts them using the public key received from *A*. Then party *B* computes C as shown in

Line 11 in Algorithm 1. After receiving the product from *B,* party *A* decrypts it using the private key and obtains a decrypted value $D(C)$. If $D(C)$equals any number in $(g^0, g^1, g^2, \ldots\ldots, g^{\tau})$, it means that $\sum_{k=1}^{n}(a[k] - b_i[k])^2 \leq \tau$, and thus we say it is a linking case. Otherwise, we say record b_i does not link a. The Error-Tolerant Linking Algorithm is correct. We discuss the correctness analysis of the Error-Tolerant Linking Algorithm in the sub-subsection named Correctness Analysis of Analysis subsection.

3.3.1.1. ALGORITHM

INPUT: Party A holds a record$(a[1], a[2], \ldots\ldots, a[n])$. Party B holds a database of m records, and each record in B is of this form: b_i : $(b_i[1], b_i[2], \ldots\ldots, b_i[m])$ where $1 \leq i \leq m$; the distance threshold is τ.

OUTPUT: Linking result for each record held by B: $[r_1, r_2, \cdots, r_m]$, where $r_i = 1$ or $r_i = 0$, $1 \leq i \leq m$.

1. Party *A* generates a pair of public key (p, g, h) and private key x for ElGamal scheme and sends (p, g, h) to *B*.
2. for j= 1 to n do
3. *A* computes $g^{a[j]}$ and encrypts $g^{a[j]}$, obtaining $E(g^{a[j]})$.
4. end for
5. *A* computes $E(g^{(a[1])^2})$, $E(g^{(a[2])^2})$, ……., $E(g^{(a[n])^2})$ and sends them together with $E(g^{a[1]})$, $E(g^{a[2]})$, …., $E(g^{a[n]})$ to *B*.
6. for i = 1 to n do
7. $r_i = 0$
8. for j = 1 to m do
9. *B* computes $g^{((b_i[j])^2)}$ and encrypts $g^{((b_i[j])^2)}$, obtaining E ($g^{((b_i[j])^2)}$).
10. end for
11. *B* computes $\prod_{j=1}^{n} E(g^{a[j]^2})\, E\left(g^{(b_i[j]^2)}\right) E(g^{a[j]})^{-2b_i[j]}$ and sends it to *A*.
12. *A* decrypts C using private key x, and obtains $D(C) = g^{\sum_{j=1}^{n}(a[j]-b_i[j])^2}$.
13. for k = 1 to τ
14. if $D(C) = g^k$ then

```
15.              Output result r_i = 1 .
16.              break
17.          end if
18.      end for
19. end for
```

Algorithm 3.1: Error- Tolerant Linking Algorithm

3.3.2. SCHEMES FOR ERROR-FREE DATA

For the regular error-free data we considered two existing basic cryptographic schemes: SHA-1 and SHA-2. SHA-1[141] and SHA-2[141] comes under the hash algorithm family. In this subsection we will review Secure Hash Algorithm (SHA) briefly. SHA is based on the design principle of MD4 [142]. Both of these algorithms are iterative and one way hash functions. SHA-1 and SHA-2 consist of two major steps: preprocessing and hash computation. In the preprocessing step, every input message is padded and then split into fixed size message blocks and the working variables are initialized to be used in hash computation. The hash computation consists of an 80–step compression function which iteratively generates hash values h_i, i.e., the i^{th} hash value. The 80–step compression function is applied to each of the message blocks. Generally two types of inputs are considered here: chaining input and message. If the message is m and chaining input is h_i then the compression function is $g(m, h_i)$ at the i^{th} stage. The chaining input at the $(i+1)^{th}$ stage, $h_{(i+1)}$ is calculated by $h_i + g(m, h_i)$. The value of the compression function at the last stage is the hash value of the message. SHA-1 and SHA-2 differ in terms of the message size, block size, word size, and message digest size as given in Table 1.

In SHA-1 five working variables: *a, b, c, d, e* are used. The message is represented by 16 32-bit words, denoted by M_i. The message is then expanded to 80 32-bit words W_i. The expanded message W_i can be defined as follows:

$$W_i = \begin{cases} M_i & for\ 0 \le i \le 15 \\ W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} & for\ 16 \le i \le 79 \end{cases} \quad (4)$$

Here $\oplus$ denotes addition modulo 2^{32}.

Secure Hash Algorithm Name	Message Size (bits)	Block Size (bits)	Word Size(bits)	Message Digest Size (bits)
SHA-1	$< 2^{64}$	512	32	160
SHA-2	$< 2^{128}$	1024	64	512

Table 1: SHA-1 and SHA-2 properties

Next working variables are initialized and computes the 80–step compression function and intermediate hash values. If there are n message blocks, i.e., $M_1, M_2, \ldots\ldots, M_n$ then the entire procedure is repeated for *n* number of times. The resulting 160–bit message digest of the message, M is,

$$H_0^{(N)} \parallel H_1^{(N)} \parallel H_2^{(N)} \parallel H_3^{(N)} \parallel H_4^{(N)} \quad (5)$$

Here, $H_j^{(i)}$ means the j^{th} word of the i^{th} hash value. The procedure of SHA-2 is similar to SHA-1. It first pads the message and divides it into 64-bit message blocks. The number of working variables here are eight, i.e., *a, b, c, d, e, f, g*, and *h*. After initializing the working variables and computing the 80–step compression function and intermediate hash values it generates a 512–bit message digest of the message M. The final message digest of M is,

$$H_0^{(N)} \parallel H_1^{(N)} \parallel H_2^{(N)} \parallel H_3^{(N)} \parallel H_4^{(N)} \parallel H_5^{(N)} \parallel H_6^{(N)} \parallel H_7^{(N)} \quad (6)$$

SHA-1 and SHA-2 are considered here since it is easy to compute the hash value of any given message and they are the one-way hash functions, i.e., they have One – way, Second preimage resistant and Collision resistant properties. SHA-1 and SHA-2 produces 160-bit and 512-bit hash

values respectively for any given message. Therefore for any given message there are 2^{160} and 2^{512} possible hash values. It is very difficult to identify the actual message from this vast range of hash values. Here in our system after applying SHA-1 and SHA-2 on the data we get the message digest/encrypted data from both of the parties and apply our data matching technique on those encrypted data.

3.4. SYSTEM ANALYSIS

We analyze our schemes especially the Error-Tolerant Linking Algorithm in terms of correctness, privacy and complexity.

3.4.1. CORRECTNESS ANALYSIS

For the proof of correctness, if the two parties follow Algorithm 1, they will jointly compute the correct Euclidean distance without each party knowing the record from the other party. The homomorphic property of ElGamal scheme helps to prove line 12 in Algorithm 3.1.

$D(C)$

$= D(\prod_{j=1}^{n} E(g^{a[j]^2})\, E(g^{(b_i[j]^2)}) E(g^{a[j]})^{-2b_i[j]})$

$= D(E(\prod_{j=1}^{n} g^{(a[j])^2}\ g^{(b_i[j])^2})\ \prod_{j=1}^{n} E(g^{a[j]})^{-2b_i[j]}))$

$= D(E(\prod_{j=1}^{n} g^{(a[j])^2}\ g^{(b_i[j])^2})\ E(\prod_{j=1}^{n}(g^{-2a[j]b_i[j]}))$

$= D(E(\prod_{j=1}^{n} g^{(a[j])^2}\ g^{(b_i[j])^2}\ \prod_{j=1}^{n}(g^{-2a[j]b_i[j]}))$

$= D(E(\prod_{j=1}^{n} g^{(a[j])^2}\ g^{(b_i[j])^2} g^{-2a[j]b_i[j]}))$

$= D(E(g^{\sum_{j=1}^{n}(a[j]-b_i[j])^2}))$

$= g^{\sum_{j=1}^{n}(a[j]-b_i[j])^2}$

If $(C) = g^k$, where $k \leq \tau$ means that $\sum_{k=1}^{n}(a[k] - b_i[k])^2 \leq \tau$. Then we can say that record a is within the distance of τ, from record b_i, and the result of error-tolerant linking is positive.

3.4.2. PRIVACY ANALYSIS

In this subsection, we explain why the Error-Tolerant Linking Algorithm is secure (i.e., privacy preserving) in the semi-honest model. Being secure in the semi-honest model means neither of the two parties can learn more than the output of the algorithm from the information received during the algorithm. In Algorithm 3.1, the only message received by *B* is the cipher texts $E(g^{((a[1])^2)})$,$E(g^{((a[2])^2)})$, ….., $E(g^{((a[n])^2)})$, and $E(g^{a[1]})$, $E(g^{a[2]})$, …., $E(g^{a[n]})$. Since ElGamal scheme is semantically secure under the decisional Diffie-Hellman assumption [25], party *B* cannot learn anything about $g^{((a[1])^2)}$,$g^{((a[2])^2)}$, $\cdots g^{((a[n])^2)}$, $g^{a[1]}$, $g^{a[2]}$, $\cdots g^{a[n]}$ but the cipher texts. For party *A*, the message it receives from party *B* is C. From the semantic security of ElGamal scheme, party *A* cannot learn the clear texts from party *B* but the $D(C)$. Here we note that $D(C)$ and $\sum_{k=1}^{n}(a[k] - b_i[k])^2$ can be derived from the output of the algorithm by trying different numbers in a small range of τ. Therefore, we say that *A* knowing $D(C)$ and $\sum_{k=1}^{n}(a[k] - b_i[k])^2$ does not violate the security requirement and party *A* can send these values to party *B* if needed.

3.4.3. COMPLEXITY ANALYSIS

We analyze the computational cost of our algorithm on party *A* and party *B* respectively. The computational cost of party *A* includes computing $2n$ exponentiations, $2n$ encryptions, and 1 decryption. Suppose that the each exponentiation takes time T_e. Then the total computation cost of party *A* is $2n(T_e + T_E) + T_D$, where T_E is the time to perform one ElGamal encryption and T_D is the time to perform one decryption. Values of g^k can be computed beforehand and saved in a table for reference at line 14, Algorithm 3.1. Party B needs to compute $2nm$ exponentiations, nm

encryptions, $2nm$ divisions/ multiplications. So the computation cost for party *B* is $nm(2T_e + T_E + 2T_m)$, where T_m is the time for a multiplication and the definitions of T_e, T_E are as above. When there are *k* records in party *A* to be linked with the records held by party *B,* the total computation time at party *A* is $nm(2T_e + T_E) + 2nmk \times T_m$. Note that in the computation cost, $nm(2T_e + T_E)$ is not multiplied by k, because as long as party *A* does not change his public key, the cipher texts of party *B*'s record do not change and thus they only need to be computed once.

3.5. RESULTS

3.5.1. SYSTEM IMPLEMENTATION

This subsection explains the system implementation of record linkage for healthcare data. We implement our system using the Eclipse Integrated Development Environment (IDE). We have used the programming language Java. The entire system is divided into three modules: Connection Management Module, Data Matching Module, and Matching Record Management Module. Among these three modules the main module is Data Matching Module. The solution of the privacy preserving record linkage, i.e., Data Matching Module works for both of the error-free data and error-prone numerical data. The Matching Record Management Module shows the result/records from the recent past data matching attempts and the Connection Management Module takes care of creating the connection with the collaborator. We will discuss these three modules in more detail in the following subsections. The snapshot of selecting a function/module in our system is given in Fig. 7.

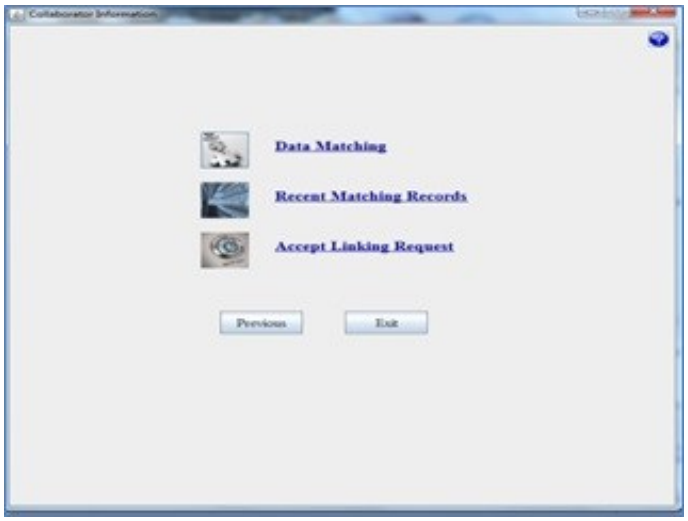

Figure 7: Snapshot of selecting a module

3.5.1.1. CONNECTION MANAGEMENT MODULE

Each party/organization keeps a list of available and reachable collaborators. To create a connection with another party/collaborator, each party needs to select that particular collaborator from the collaborator list. The snapshot of how a user selects a collaborator in our system is given in Fig. 8.

Figure 8: Snapshot of selecting a collaborator

To initialize the connection each and every organization keeps some initial information about the other collaborators. This information contains the IP address, port number, public-key private-key pair etc. Party A first selects the party B from the available participating collaborator list. Party A uses the corresponding IP address and port number of the party B for creating the connection. We follow the client–server architecture to implement our system. The communication between two parties is realized by socket API. Party B (server) creates a socket to listen requests from party A (client). Party B can handle more than one client at a time. If there is more than one client, party B creates separate sockets using multi-threading for each of the requesting clients.

3.5.1.2. DATA MATCHING MODULE

The Data Matching Module is the main module of our system. We consider two parties: party A and party B. Party A initiates the matching procedure and party B takes part in this matching procedure. The entire work flow of the system including party A and party B is illustrated in Fig. 9. As shown in Fig. 9, once a connection is created between party A and party B, data transfer between the two parties and matching can take place. Party A first selects the record set for matching data. When party A selects the dataset name then the corresponding attributes list becomes available. Party A selects the attributes names and sends the dataset name along with the attributes to party B. Party B searches the requested record set in its set of record sets. If party B has the record set, it sends an acknowledgement to the party A. In response to this acknowledgement, party A sends the user selected cryptographic scheme name to party *B* and encrypts its own selected record set. Party *B* encrypts the requested dataset with the requested cryptographic scheme. A snapshot of how a user selects a cryptographic scheme is given in Fig. 10. For encryption purposes, the Java Cryptography Library is used. To encrypt data we considered three cryptographic schemes: SHA-1, SHA-2, and Error-Tolerant Linking Algorithms. The first two schemes SHA-1, SHA-2 do not require any key pair whereas the Error-Tolerant Linking Algorithm needs key pair for encryption. The key pair will be known to the organization and its

collaborator beforehand. The first two schemes, SHA-1 and SHA-2 work in the same way. After encrypting the record set, party *B* sends the encrypted data to party *A*. After receiving the encrypted data from party *B*, party *A* applies the data matching technique on these two encrypted record sets.

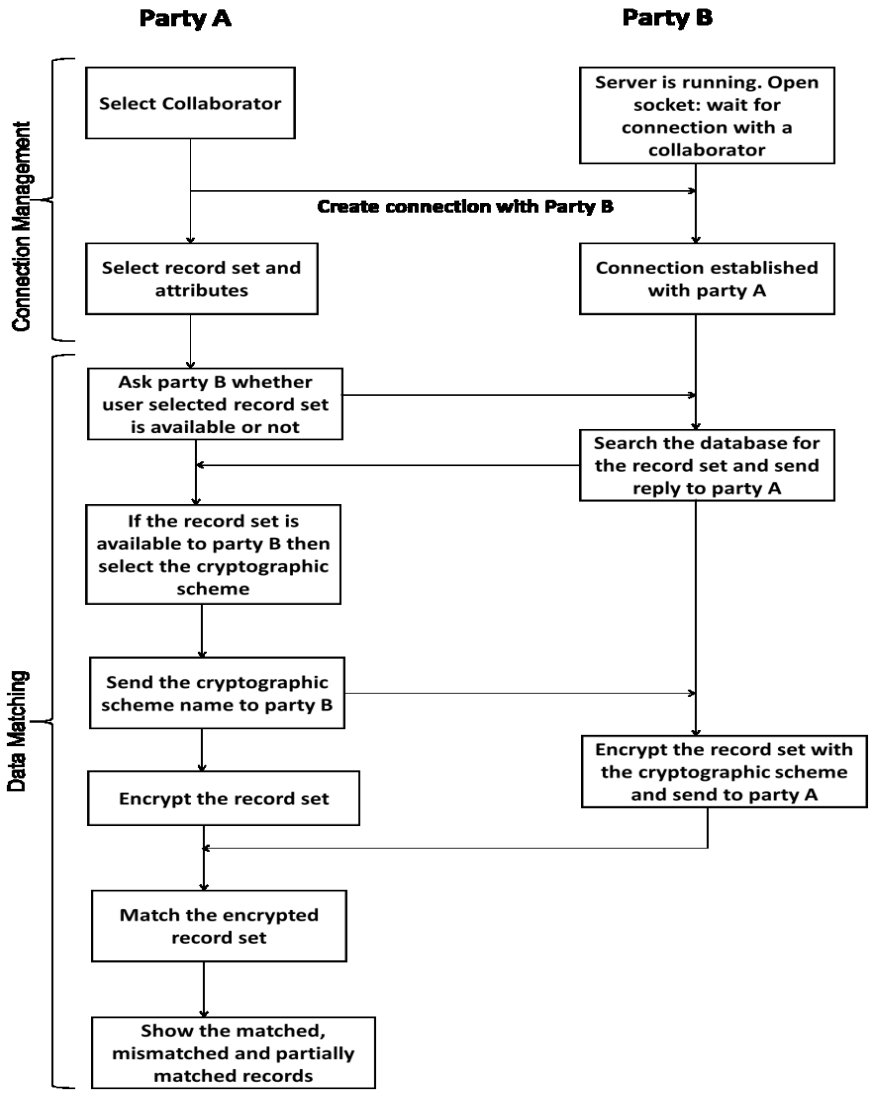

Figure 9: The work flow of the system

The Error-Tolerant Linking Algorithm works a little differently than the other two cryptographic schemes, once the dataset name and attributes has been selected. Suppose party A has one medical

record to be linked with the records held by party B. The flow can be easily extended to the case where A has multiple medical records to be linked, A send the encrypted messages generated by his data record to be linked to party B. Then B handles the encrypted messages as described in previous sections (i.e., encrypting his own data record and multiplying their inverse with A's message) and sends the multiplied encrypted message back to A. A decrypts the message and outputs the linking result. Party B moves to the next record and repeats the linking procedure. Party A does not need to encrypt his record again but only needs to decrypt the messages sent from B and output the linking result. This repeating process carries on until party B has gone through all his records. Then party A and party B close the connection with each other and the privacy preserving linkage is completed.

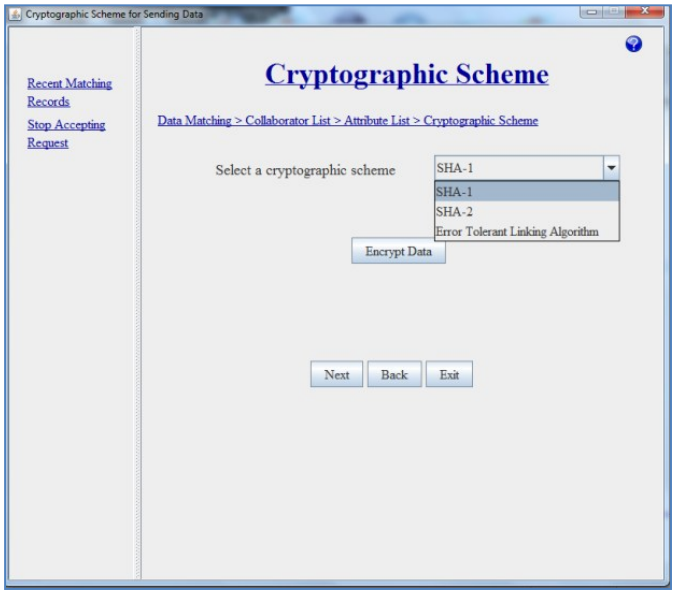

Figure 10: Snapshot of selecting an encryption scheme

The flow of the Error Tolerant Linking Algorithm after selecting the dataset name and attributes is depicted in Fig. 11. Once the entire data matching procedure is completed, the client/initiating organization closes the connection with the server/participating organization

automatically. As a result of the entire data matching procedure, party A gets the matched, mismatched and partially matched result with party B. Fig. 12 shows a snapshot of matching result of our system

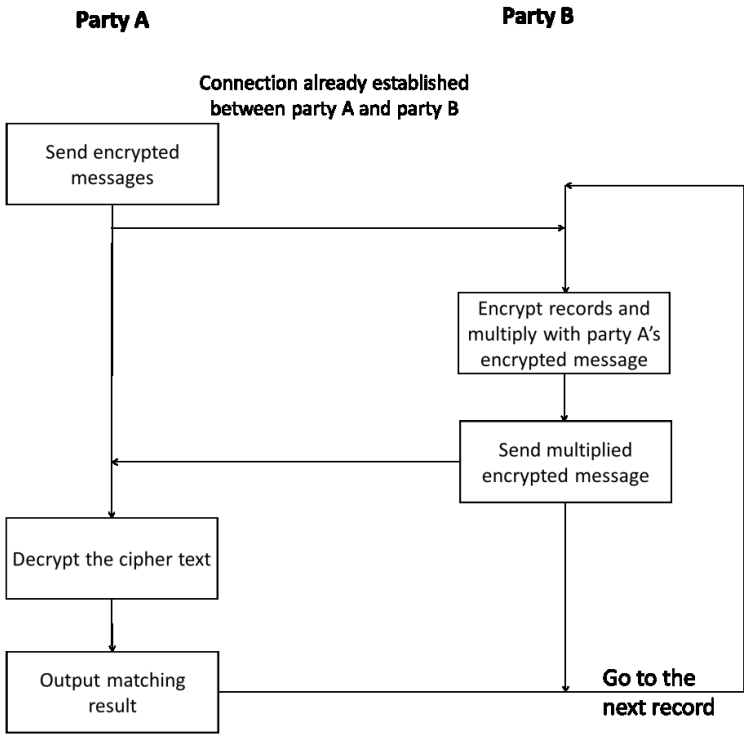

Figure 11: Flow of Error Tolerant Linking Algorithm between two parties when they are already connected and the dataset name, attributes are known to party B

3.5.1.3. MATCHING RECORD MANAGEMENT MODULE

The Matching Record Management Module shows a brief description of the matching result from the recent past data matching attempts. It shows the date-time of when the matching took place, name of the participating organization/collaborator, name of the record set, the number of matched, mismatched and partially matched data for each and every data matching attempt. A snapshot of Matching Record Management Module of our system is given in Fig. 13.

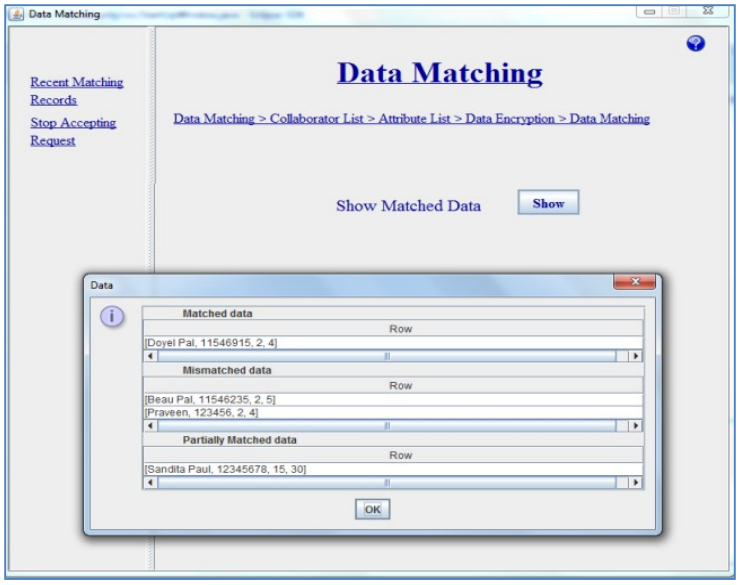

Figure 12: Snapshot of showing matching result in our system

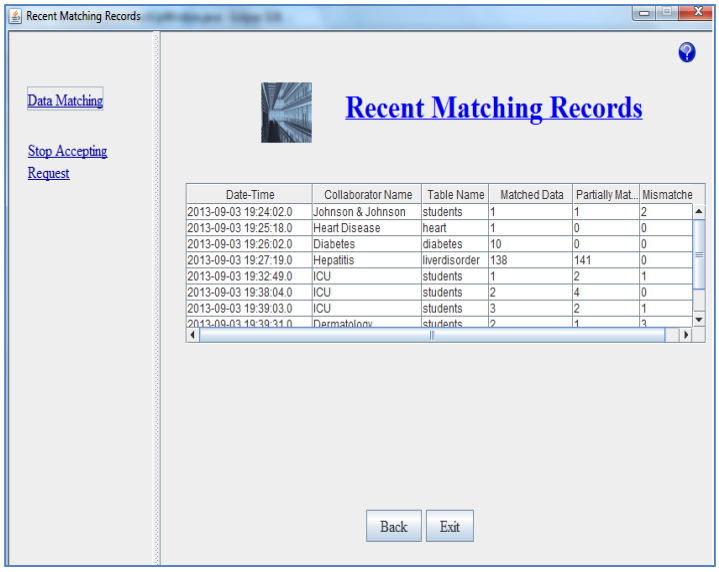

Figure 13: Matching Record Management Module

3.5.2. EXPERIMENTAL SETUP

We ran our system on two computers with 3.33 GHz Intel Core i5 processor, 4 GB RAM, 64 bit operating system. Both for party A and party B, we have constructed window applications using

Java. The applications on the different computers are connected by the Internet. The communication between party A and party B is realized by using socket API. Before running the system each client needs to know the IP address and port number of the server. If a party/server changes its IP address then it should inform the other parties/clients. We consider that each party maintains an IP address and port list of other parties.

We use two real-world medical datasets, the Pima Indians Diabetes Data Set and the Heart Disease Data Set [59] to implement our system. To handle these real-world datasets MySQL is used. Java Database Connectivity (JDBC) connects the application front end to the database end. JDBC is used to access data directly from the database and to display them to the user.

3.5.3. EXPERIMENTAL RESULTS

We test the scalability of our system in terms of time efficiency. For each cryptographic scheme in this system we vary the number of records and the number of attributes for each record, and then measure the computation time of our system. To test the efficiency of our system, we consider two real-world datasets, the Pima Indians Diabetes Data Set and the Heart Disease Data Set [24]. In the Pima Indians Diabetes Data Set we use at most eight attributes: (1) Number of times pregnant, (2) Plasma Glucose Concentration; a 2 hours in an oral glucose tolerance test, (3) Diastolic blood pressure (mm Hg), (4) Triceps skin fold thickness (mm), (5) 2-hours serum insulin (mm U/ ml), (6) Body mass index (weight in kg/(height in m)^2), (7) Diabetes pedigree function, (8) Age (years). Similarly for the Heart Disease Data Set, we use eight attributes for each record: (1) age of the patient, (2) sex, (3) chest pain type, (4) resting blood pressure, (5) serum cholesterol in mg/dl, (6) fasting blood sugar, (7) resting electrocardiographic results (8) maximum heart rate achieved. For each encryption scheme except the Error-Tolerant Linking Algorithm we vary only the number of attributes to four, six and eight and use 100 patients' records. For the Error-Tolerant Linking Algorithm we vary the number of attributes as well as the number of patients' records.

For SHA-1 and SHA-2 we use 100 patients' records from both the Pima Indians Diabetes Data Set and the Heart Disease Data Set. For each record we vary the number of attributes to four, six and eight. We show the computation times of our system using SHA-1 and SHA-2 in Fig. 14 and Fig. 15 respectively. For both of these above mentioned existing algorithms the computation time increases as we increase the number of attributes. The computation time grows almost linearly as we increase the number of attributes. Moreover, in every case the computation time does not go beyond 0.1 second.

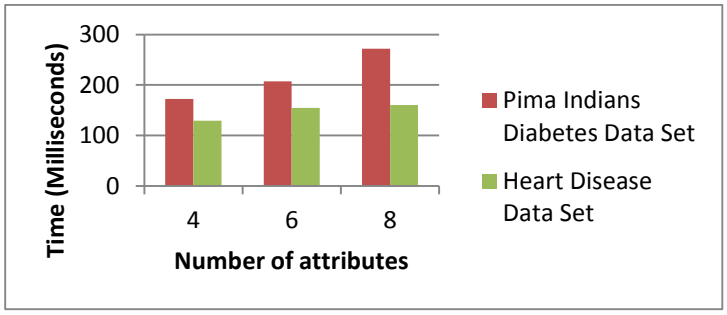

Figure 14: Computation time for SHA -1 for 100 records with varying number of attributes of four, six and eight.

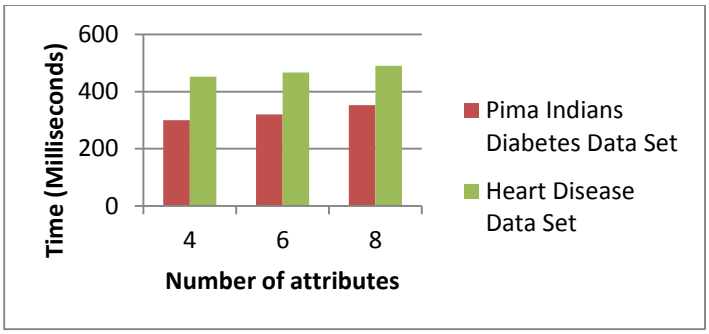

Figure 15: Computation time for SHA -2 for 100 records with varying number of attributes of four, six and eight.

Figures 16 and 17 depict the computation time for the Error-Tolerant Linking Algorithm. We implement the Error-Tolerant Linking Algorithm with both constant and varying number of attributes. The values of all the attributes are preprocessed and converted to integers. Fig. 16 shows the computation time of our system when party A conducts the privacy preserving linking on Pima Indians Diabetes Data Set. Party B holds a variable number of records of 100, 200 and 300 while keeping the number of attributes constant at four. The computation time increases linearly as the size of data to be linked grows. Fig. 17 presents the computation time for Heart Disease Data Set with varying number of records from 100 to 300 and varying number of attributes from 4 to 8. For this data set too, the computation time increases as the number of attributes and number of records grow. In both cases, when the number of records or number of attributes increases the computation time increases almost linearly. In addition to that, for this algorithm too the computation times never go beyond 0.1 second.

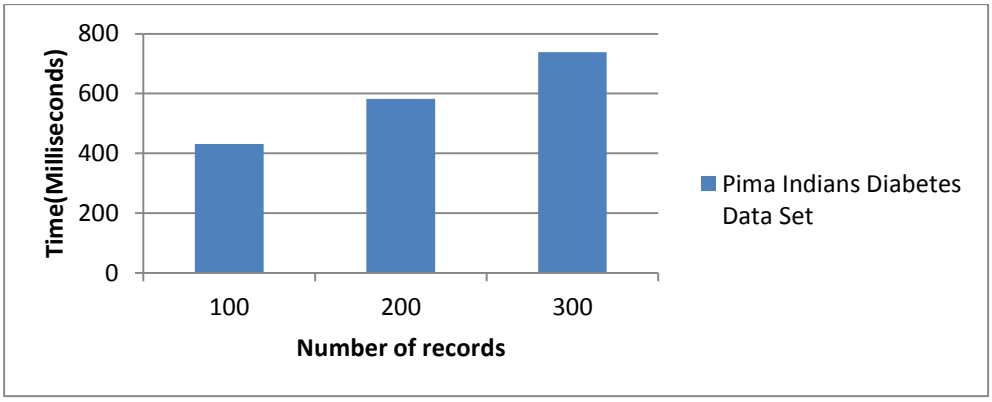

Figure 16: Computation time for Error –Tolerant Linking Algorithm with varying patients' records from Pima Indians Diabetes Data Set where each record has 4 attributes

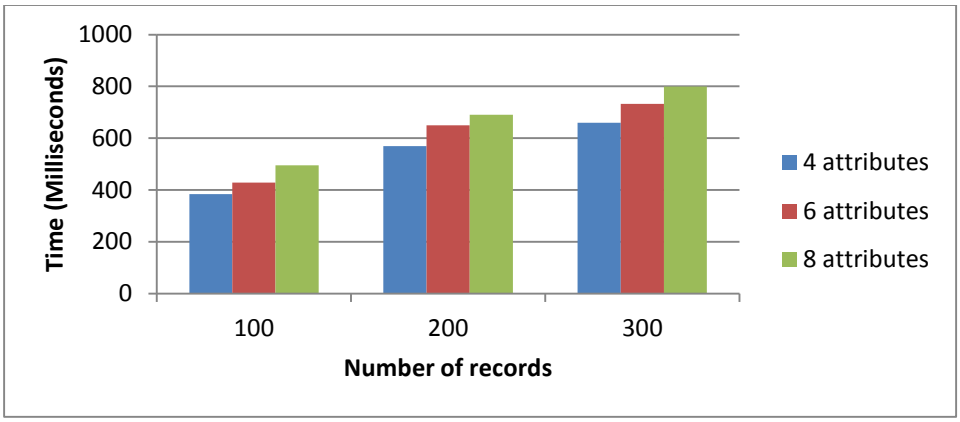

Figure 17: Computation time for Error-Tolerant Linking Algorithm with varying patients' records and varying attributes from Heart Disease Data Set

3.6. CONCLUSIONS

In this chapter we propose a solution for privacy preserving record linkage for error-free data as well as for error-prone data. For error-free data, we offer two existing cryptographic schemes: (1) SHA-1, and (2) SHA-2. A new algorithm is proposed for error-prone numeric data. We implemented our system fully and tested it on two real-world data sets. We have shown that our system is secure, correct, and efficient and does not require a trusted third party. The experimental results demonstrate the efficiency of our system. The one limitation of our proposed system is that for error-prone data our system is limited to only numeric data. Considering this fact, we designed our software in such a way that any existing solution for error-prone textual data can be easily integrated into our system. This makes our software flexible and open to integrate any existing record linkage scheme for error-prone textual data.

CHAPTER IV

PRIVACY PRESERVING SEQUENTIAL PATTERN MINING ACROSS MULTIPLE MEDICAL SITES

4.1. LITERATURE REVIEW

Data mining investigates data for unknown, hidden, predictive patterns and trends from large volumes of data. Over the past decade data mining research has led to the development of various algorithms such as clustering, classification, association rule mining etc. One of the important sequence data mining problems, sequence Pattern Mining discovers frequent sequential patterns in large sequential databases. Sequential frequent pattern mining methods are applicable in a large number of applications and domains, such as analysis of DNA sequences, patterns in patient path, disease treatments, analysis of customer purchase patterns, web access patterns, discovering trends in marketing and sales data etc.

For the last ten years various algorithms (e.g., [75-79]) have been developed to find sequential patterns in very large sequential databases. These large varieties of algorithms encompass mining frequent sequential patterns, closed sequential patterns, compressing sequential patterns, maximal sequential patterns, sequential generator patterns etc. from sequence database. The basic feature of most sequential pattern mining algorithms [75] is to find all frequent subsequences from a given set of sequences where each sequence consists of a list of elements and each element consists of a set of items. A user-specified minimum support threshold is also used as a threshold value to find the frequent subsequences. The frequent subsequences are those whose occurrence frequency in the set of sequences is no less than minimum support threshold. [75, 76, 77, 80]

algorithms generate constant–support–based frequent patterns which generate an exponentially large number of short patterns. Finding all frequent patterns which support length–decreasing support constraint is desirable as long patterns tend to be interesting with small support and short patterns tend to be interesting with high support. SLPMiner [81] is such an algorithm that finds all frequent sequential patterns that satisfy the length–decreasing support constraint.

On the other hand, preserving privacy is a real concern in many data mining problems. Data mining involves both extracting patterns and trends from large databases and sharing of information [82]. Databases contain sensitive information which leads to mining of sensitive data by applying data mining algorithms [83, 84]. As a result, the privacy of sensitive data may be compromised when private information is disclosed publicly or shared. Privacy preserving data mining involves the development of data mining algorithms that modify private data in such a way so that even after mining, private data remains always private. One of the different techniques that can be used to modify private data is using a cryptography based technique. Several privacy preserving sequential pattern mining algorithms [84-87] have been developed to mine sequence data. In [84] the authors proposed PriPSep algorithm which is a novel secure extension of the S-step process of the SPAM [78] algorithm. Compared to SPADE, SPAM uses a space inefficient approach. To mine sequential patterns, PrefixSpan [77] and SPADE [79] are the two fastest algorithms [78]. The database projection based approach of the SLPMiner [81] algorithm is based on the sequential version of [89] which shares the same overall structure of the PrefixSpan algorithm. The pruning methods proposed in the SLPMiner algorithm and the length decreasing support constraint make this algorithm more efficient and enable its high performance.

Finding frequent patterns and trends by exploring large databases is the main task of data mining. Sequential pattern mining is one of the important sequence data mining problems with numerous applications. Sequential pattern mining discovers the frequent subsequences as patterns from sequence databases. Numerous algorithms [77, 78, 79, 80, 81, 92, 93] have been developed to find

such patterns efficiently. The sequence pattern mining problem was first introduced by Agrawal and Srikant [75]. To discover generalized sequential patterns, the GSP algorithm [76] was proposed. In [77] the authors proposed an efficient pattern–grown method named PrefixSpan which grow sequential patterns in each projected sequence database. The PrefixSpan approach uses the pseudo projection technique to reduce the number of physical projected databases. An efficient algorithm SPAM [78] finds all frequent sequences in a list of transactions using a depth–first search strategy combined with a vertical bitmap representation. Here the depth–first search strategy integrates the depth–first traversal of the search space with effective pruning mechanisms. A fast sequential pattern mining algorithm, SPADE decomposes the original problem into smaller sub–problems using combinatorial properties. The sub–problems can be solved independently in main memory using efficient lattice search techniques and simple join operations. An efficient sequential pattern mining algorithm named LAPIN has been proposed in [91] which is based on the last position induction of the items. The SLPMiner algorithm [81] finds all sequential patterns that satisfy a length–decreasing support constraint. To reduce the search space this algorithm combines an efficient database–projection–based approach with three database pruning methods.

Considering the presence of private sensitive information in the sequence databases, privacy is a concern in sequential pattern mining. Several algorithms [83-86] have been proposed for privacy preserving sequential pattern mining. A privacy preserving sequential pattern mining algorithm named PRIPSEP [10] has been proposed for secure multi–party computation. [84] introduces a privacy preserving sequential pattern mining algorithm based on secure multi–party computation and homomorphic encryption scheme. It discovers the sequential patterns collaboratively and privately on the union of the private data sets held by the multiple parties. [85] presents a privacy preserving index based sequential pattern mining algorithm SSAP which uses an equivalent form of a sequential pattern to reduce the cryptographic operations, such as encryption and decryption. A privacy preserving collaborative sequential pattern mining algorithm has been introduced in [86].

[94] presents the way to mine frequent pattern with differential privacy. However, there has not been any research on providing a privacy preserving property for the SLPMiner algorithm.

4.2. DESIGN CONSIDERATION

We focus on identifying the most frequent sequential patterns on sequence databases over multiple parties in a privacy preserving manner. The motivation behind this work is to extract the common, most frequent sequential patterns from multiple sites in a privacy preserving manner. For sequential pattern mining we use the SLPMiner algorithm because of its high performance and efficiency. Our proposed scheme can be applied to the healthcare industries to identify the patterns in patient path, patient health, disease, medication etc. as it involves multiple parties and needs to be done in a privacy preserving manner.

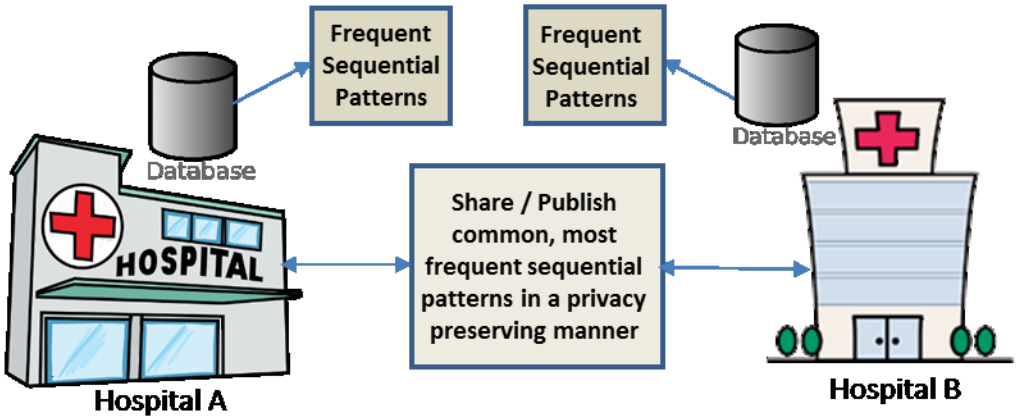

Figure 18: Overview of the scenario

As a real life scenario, to analyze effective treatment methods of different diabetic patients (grouped by ages) it is essential to collect correlated patients' data from different medical sources. According to the Health Insurance Portability and Accountability Act (HIPAA) [1], sharing any patient data violates privacy and should not be disclosed. Therefore we need a privacy preserving mechanism to analyze patients' records from different medical sources. The result of the analysis will show the

different age groups and effective treatment methods of diabetic patients. For this study, say 2 hospitals are involved. Fig. 18 shows the overall view of the scenario. Out of the collective data patterns of different patients from these two hospitals, the task is to get the most frequent patterns of patients age groups, treatment methods. The pattern can be represented as $< age\ group, treatment\ method >$. Fig. 19 shows the databases D_A, D_B from $Hospital\ A$ and $Hospital\ B$ respectively.

Name	Age Group	Method	Cured?
Tina	20 – 30	< Diet, Exercise>	Yes
Beau	30 – 40	<Medicine, Diet, Exercise>	Yes
Diane	50 – 60	<Medicine, Diet>	No
Sophie	20 – 30	<Medicine>	No
Sandy	30 – 40	<Medicine, Diet, Exercise>	Yes

Database D_A

Name	Age Group	Method	Cured?
Josh	20 – 30	< Diet, Exercise>	Yes
Melanie	20 – 30	< Diet, Exercise>	Yes
Bob	30 – 40	<Medicine, Diet, Exercise>	Yes
Eva	30 – 40	<Medicine, Diet, Exercise>	Yes
Chris	40 – 50	<Medicine, Diet>	No

Database D_B

Figure 19: Patient Databases from two hospitals

From database D_A the most frequent pattern is <30–40, {Medicine, Diet, Exercise}>. From database D_B the most frequent patterns are <20–30, {Diet, Exercise}> and <30–40, {Medicine, Diet, Exercise}>. Considering the threshold value 2 the most frequent pattern derived from these two databases is <30–40, {Medicine, Diet, Exercise}>. We identify the most frequent sequential patterns from a site using the SLPMiner algorithm.

4.3. BACKGROUND

4.3.1. BASIC NOTATIONS

In this work we use the notations that are used in the SLPMiner algorithm. Sequential pattern mining was first introduced by Srikant et al. [75]. We use the basic sequence model defined in [75]. A sequence s denoted by $< t_1, t_2, .., t_n >$ is an ordered list of itemsets s_j. An itemset $I = \{i_1, i_2, \ldots., i_n\}$ is defined as a non-empty ordered set of items i_j. A sequential database D is a set of

sequences. $|D|$ denotes the number of sequences in a sequential database D. The length of sequence s is the number of items in s and is denoted by $|t|$. Similarly the length of the itemset is denoted by $|i|$. A sequence $s = < t_1, t_2, .., t_n >$ is called a sub–sequence of sequence $s' = < t'_1, t'_2, \ldots., t'_m >$ $(l \leq m)$ if there exist l integers $i_1, i_2, \ldots, i_l$ such that $1 \leq i_1 < i_2 < \cdots < i_l \leq m$ and $t_j \subseteq t'_{ij}$ $(j = 1, 2, \ldots, l)$. If s is a sub – sequence of s', then we write $s \subseteq s'$ and say sequence s' supports s. The support of a sequence $\delta_D(s)$, is defined as $|D_s| / |D|$, where $D_s = \{ s_i | s \subseteq s_i \wedge s_i \in D \}$.

4.3.2. SLPMINER ALGORITHM

In this section we discuss the SLPMiner algorithm briefly. The SLPMiner algorithm finds all the frequent sequential patterns that satisfy a given length–decreasing support constraint. The length–decreasing support constraint is defined as below:

Given a sequential database D and a function $f(l)$ *that satisfies* $1 \geq f(l) \geq f(l+1) \geq 0$ *for any positive integer l, a sequence s is frequent if and only if* $\delta_D(s) \geq f(|s|)$.

The SLPMiner algorithm uses an efficient database–projection–based approach for generating a frequent pattern. It also use three pruning methods based on the smallest valid extension (SVE) property to prune either entire sequences, items within certain sequences or entire projected databases. The SVE property helps to prune large portions of the database efficiently that are irrelevant for finding frequent item sets that satisfy the length–decreasing support constraint. To find frequent sequential patterns the SLPMiner algorithm uses the database–projection–based approach [77, 88, 89] which grows a Prefix tree in depth–first order. The prefix tree contains all the frequent sequential patterns with a given threshold support value. The root of the prefix tree is a null sequence and it expands to create children nodes by growing the pattern in two different ways, namely, item set extension and sequence extension. At each node instead of the entire input database, the projected database is created which is much smaller in size compared to the input

database. To make the SLPMiner algorithm efficient the authors introduced three pruning methods, Sequence Pruning, Item Pruning, and Structure–based Pruning, which reduce the size of the projected database at each node of the prefix tree.

4.3.3. SIMILARITY MEASUREMENT

One of the fundamental tasks of the data mining problem involves the comparison of data and determining the similarity of data. A wide variety of real world applications, such as, information retrieval, mirror pages, document categorization, plagiarism, etc. rely on similarity computation. To measure the similarity between data the similarity measurement in terms of distance between the data is computed. Euclidean, Cosine, Jaccard, Pearson Correlation [139] are some examples of popular similarity measures. We use Jaccard Similarity to measure the similarity between frequent sequential patterns. The similarity between two finite sample sets is measured by the Jaccard Similarity Coefficient. The Jaccard Similarity Coefficient can be defined as below,

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}, 0 \leq J(S_1, S_2) \leq 1 \quad (7)$$

According to the definition of the Jaccard Similarity Coefficient if two sets, S_1, S_2 are the same, then the Jaccard Similarity Coefficient between S_1 and S_2 is 1.

4.4. PROBLEM STATEMENT

Privacy is a concern in data mining as the database contains sensitive private information. Therefore finding most frequent sequential patterns in a privacy preserving way is an important task in sequential pattern mining. The goal is to find the most frequent patterns held by two or more parties. Formally the problem can be defined as follows: – given n parties $A, B, .., I, .., N$, where each party i has a database D_I. For n parties there are therefore $D_A, D_B, .., D_I, ..., D_N$ databases. The SLPMiner algorithm finds frequent sequential patterns at each site i, e.g. $P_a, P_b, ..., P_i, ..., P_k$. The

similarity measurement function checks the tuple $< P_i, \sigma, \ \tau >$ where $P_i,$ is the frequent sequential pattern at different sites, e.g., P_i^A at site A and P_i^B at site B. σ is the count of Jaccard similarity coefficient (=1) for a particular pattern P_i (Equation 8) and τ is some threshold value.

$$\sigma(P_i) = \sum_{a,b \in [1,n]} \left(J\left(P_i^A, P_i^B\right) = 1 \right) \tag{8}$$

$\sigma \geq \tau$ (Equation 9) determines P_i as the most frequent sequential pattern among all the participating sites. Privacy preserving similarity measurement reveals only the most frequent sequential patterns among all the participating sites. A frequent pattern $P_r \ where \ P_r \neq P_i$ at site S_r whose $\sigma(P_r) < \tau$ should not be disclosed to any other site since it is not a most frequent sequential pattern among all the sites.

$$\sigma(P_i) \geq \tau \tag{9}$$

4.5. PROPOSED SOLUTION

To find the most frequent sequential patterns between at least two parties in a privacy preserving way we propose an algorithm (Algorithm 4.1) which uses the SLPMiner algorithm and Jaccard similarity measurement. The SLPMiner algorithm finds all the frequent sequential patterns at each site. To get the most frequent sequential patterns from all the sites we use the Jaccard Similarity measurement in a privacy preserving way. The proposed algorithm is presented in Algorithm 4.1. Assuming the number of participating sites are n and each party has a sequence database D_i, then at each site we apply the SLPMiner algorithm to find the frequent sequential patterns. In the process of finding frequent sequential patterns, the SLPMiner algorithm creates a prefix tree in depth first order. During the backtracking of the prefix tree, that is, traversing from the leaf node to the root node we encrypt all the nodes of a branch. Since all the nodes of the prefix tree represent the frequent patterns, we get all the frequent patterns in encrypted format. To encrypt the patterns any standard public key encryption scheme can be used where the public key – private key pair is

generated (line 1, Algorithm 4.1) by a Trusted Third Party (TTP), e.g., Certificate Authority (CA) and only the public key is distributed among all the parties. Among all the sites the algorithm chooses one random site S_j, where $j \in [1, n]$ (line 5, Algorithm 4.1). All the other sites send the encrypted frequent patterns to site S_j. Site S_j calculates the Jaccard Similarity coefficient between each and every patterns. It also checks if the Jaccard Similarity Coefficient for a particular encrypted pattern is equal to 1 (line 11–16, Algorithm 4.1). If the number of occurrences (i.e., σ) of the Jaccard Similarity coefficient = 1 for a particular pattern is greater than some threshold value τ then the pattern is a most frequent sequential pattern. Another site S_r, where $S_r \neq S_j$ and $r \in [1, n]$ is chosen (line 17, Algorithm 4.1) and all the most frequent sequential encrypted pattern will be sent to site S_r. The TTP sends the private key to site S_r and it decrypts the patterns and publishes the most frequent sequential patterns.

4.5.1. ALGORITHM

INPUT: Assume there are n sites with n sequence databases, $D_1, D_2, D_3, \ldots, D_i, \ldots, D_n$ one database at each site. A trusted third party (TTP) will generate the public key–private key pair (x, y). A threshold value τ.

OUTPUT: The most frequent sequential patterns, $P_a, P_c, P_i, P_k, \ldots$, among all the sites n.

1. Suppose there are n number of sites. The TTP generates public key–private key pair (x, y) and distributes the public key x among the n sites.
2. for each site $i = 1\ to\ n\ do$
3. Apply the SLPMiner algorithm on the available sequential database. During backtracking of the prefix tree encrypt each itemset I_i of every sequential pattern $P_i = < (I_1), (I_2), \ldots, (I_k) >$. Therefore the encrypted sequential pattern becomes $E(P_i) = < E(I_1), E(I_2), .., E(I_k) >$
4. $end\ for$
5. Randomly choose one site S_j

6. for each site $i = 1\ to\ n\ do$
7. $for\ each\ sequential\ pattern\ j = 1\ to\ m\ at\ each\ site\ S_i$
8. Send the encrypted sequential pattern $E(P_j)$ to S_j
9. $end\ for$
10. $end\ for$
11. At site S_j, $for\ each\ pattern\ \ E(P_k)\ where\ k = 1\ to\ m\ \times n$
12. Calculate Jaccard Similarity coefficient
13. if Jaccard Similarity coefficient = 1
14. Count number of occurrences σ of Jaccard Similarity coefficient = 1
15. $end\ if$
16. $end\ for$
17. Randomly choose a site S_r where $S_r \neq S_j$
18. $for\ each\ pattern\ E(P_k)\ \ k = 1\ to\ m\ \times n\ at\ site\ S_j$
19. $if\ \sigma >$ some threshold value τ
20. Consider $E(P_k)$ as the most frequent sequential pattern
21. Send $E(P_k)$ to S_r
22. $end\ if$
23. $end\ for$
24. TTP sends the private key y to S_r
25. S_r decrypts the sequential patterns and publishes the most frequent sequential patterns.

Algorithm 4.1.

4.6. EXPERIMENTAL ANALYSIS

4.6.1. EXPERIMENTAL RESULTS

We evaluate our algorithm using different datasets generated by the IBM Quest research group [90]. We used two datasets with different numbers of sequences (2,000 rows of data to 10,000 rows of data). Each sequence of data consists of eight itemsets on average. All our experiments are performed on two computers with a 3.33 GHz Intel Core i5 processor with 4 GB RAM and a 64 bit operating system. We have named our datasets as DataSet1 and DataSet2. Since our algorithm works on encrypted frequent itemsets, we encrypt the most frequent itemsets for each

data set. The size of data, time to get frequent itemsets using the SLPMiner algorithm, number of frequent itemsets and time to encrypt the frequent itemsets for DataSet1 and DataSet2 are given below in Table 2 and Table 3.

Size of Data (in rows)	Time to get frequent items (in seconds)	No. of frequent itemsets	Time to encrypt frequent itemsets (in seconds)
2,000	9.5	503	0.09
4,000	18	1206	0.18
6,000	33	1600	0.19
8,000	44	1725	0.2
10,000	50	1826	0.2

Table 2: DataSet 1

Size of Data (in rows)	Time to get frequent itemsets (in seconds)	No. of frequent itemsets	Time to encrypt frequent itemsets(in seconds)
2,000	6.9	445	0.07
4,000	12	502	0.08
6,000	18	1488	0.18
8,000	20	1940	0.27
10,000	25	2470	0.29

Table 3: DataSet 2

To get the common, most frequent itemsets between DataSet1 and DataSet2 we use Algorithm 2. Although we have considered two sites only, we can use our algorithm for multiple number of sites. The experimental results, namely, size of data, number frequent datasets in two sites, common frequent itemsets between two sites, time to get common frequent itemsets and decryption time to get actual data from encrypted datasets are shown in Table 4 below.

Size of Data (in rows)	No. of frequent items in Dataset 1	No. of frequent items in Dataset 2	Common frequent items between Dataset 1 and Dataset 2	Time to get common frequent items (in seconds)	Decryption time (in seconds)
2,000	503	445	276	0.35	0.04
4,000	1206	502	317	0.5	0.05
6,000	1600	1488	634	1.6	0.08
8,000	1725	1940	631	2.3	0.10
10,000	1826	2470	910	2.5	0.12

Table 4: Experimental results

4.7. ANALYSIS

We analyze our algorithm in terms of correctness analysis, privacy analysis, and performance analysis.

4.7.1. CORRECTNESS ANALYSIS

In Algorithm 4.1 we calculate the Jaccard Similarity Coefficient between the encrypted frequent sequential patterns. From the definition of Jaccard Similarity Coefficient (Equation 7) we know if the two sets are same then the Jaccard Similarity Coefficient between two sets are 1. Assume two parties $i \; and \; k$ are participating. Any sequential pattern P_i (at site i) consists of itemsets, $P_i =< (I_1), (I_2), .., (I_m) >$. Instead of encrypting the entire pattern we encrypt each itemset of a sequential pattern, $E(P_i) =< E(I_1),\ E(I_2), \dots\dots, E(I_m) >$ where $E(P_i)$ acts as the set and $E(I_1), E(I_2), \dots\dots, E(I_m)$ are the elements of the set. The algorithm calculates the Jaccard Similarity Coefficient between two encrypted patterns $E(P_i)\ and\ E(P_k)$ at some other site r, where $E(P_k) = < E(I_1),\ E(I_2), \dots\dots, E(I_m) >$. Since $P_i = P_k$ therefore,

$$J\big(E(P_i), E(P_k)\big) = \frac{|E(P_i) \cap E(P_k)|}{|E(P_i) \cup E(P_k)|} = 1 \qquad (10)$$

Equation (10) proves if the patterns are same the Jaccard similarity coefficient between two encrypted patterns are 1.

4.7.2. PRIVACY ANALYSIS

In our proposed algorithm each site i encrypts their patterns and sends them to other randomly selected site k. Since only the encrypted frequent patterns are shared with the other site k and private key x is not available, knowing the real frequent patterns is not possible. The TTP sends the private key to some other randomly selected site S_r which has only the most frequent sequential patterns in encrypted format. This selected site S_r publish the common, most frequent sequential patterns. Since all the shared patterns are in encrypted format and sites are chosen randomly, any private or sensitive data is not compromised.

4.7.3. EFFICIENCY ANALYSIS

From the experimental results we see that the time to get the frequent sequential patterns from each of the data sets using the SLPMiner algorithm is at most 50 seconds. The time to encrypt the frequent itemsets is less than 0.3 seconds for 10,000 rows of data and the decryption time is less than 0.15 seconds for 10,000 rows of data. The total time to get the most frequent sequential patterns between the two participating sites using the Jaccard Similarity Coefficient is at most 2.5 seconds. The time measurements of our experiments reflect the fact that our algorithm performs well and is efficient.

4.8. CONCLUSIONS

Privacy preserving data mining ensures analysis of large volumes of data among different participating sites without revealing details of private, sensitive data. We propose a privacy preserving sequential pattern mining algorithm which publishes only common, most frequent sequential patterns among participating sites in a privacy preserving manner. The analysis of our

algorithm in terms of privacy and correctness imply that our algorithm is correct and it protects sensitive information from unwanted or unsolicited disclosure. The performance analysis proves the efficiency of our algorithm.

CHAPTER V

MULTILEVEL THRESHOLD SECRET SHARING IN DISTRIBUTED CLOUD

5.1. LITERATURE REVIEW

The Distributed cloud [95] makes use of resources provided by users in a P2P manner. In the distributed cloud resources are virtualized. The existing cloud uses huge data centers where resources are provided using virtualization techniques. Some of the voluntary computing systems such as BOINC [96], SETI [97] and Planetlab [98] uses resources provided by users, but these are managed by a central entity and are completely different from either a centralized cloud or the distributed cloud. Unlike users of the existing cloud, who don't have control over the resources, in the distributed cloud users can choose resources of their choice. Moreover, in the distributed cloud, resources are provided by users. Since users of the distributed cloud have more control over resource selection, they can perform strong encryption to secure their data. The only issue will be managing these keys. We use a secret sharing mechanism to protect secret keys in the distributed cloud. Secret sharing schemes can mitigate the risks of managing these keys. Secret sharing schemes are used in many cryptographic protocols. Threshold secret threshold secret sharing schemes was introduced by Shamir [99] and Blakley [100]. In the threshold secret sharing mechanism, information is split into multiple shares and these shares are distributed among multiple users. The only way to retrieve the information is to have all the shares or a qualified number of shares available. This qualified number is the threshold. This means that any number of shares less than the threshold cannot retrieve the information. Shamir's secret sharing scheme is based on polynomial interpolation whereas Blakley's secret sharing scheme is based on

geometry. Several secret sharing schemes has been introduced, e.g., threshold secret sharing scheme, ideal secret sharing, linear secret sharing, multi linear secret sharing, [99, 100, 101, 102] etc. The Asmuth and Bloom secret sharing scheme [103] uses the Chinese reminder theorem to reconstruct shares. Multi linear secret sharing schemes [104] uses monotone span programs. In these schemes, secret is a sequence of elements from a finite field and the share is derived using the secret and some random elements from the field. If the secret is of size one i.e. has only one element it is called linear. We propose a multilevel threshold secret sharing scheme to provide security of secret key in the distributed cloud. In the first level of the proposed secret sharing scheme we split the secret key into multiple shares and distribute them among multiple resource providers. The second level consists of splitting each share into multiple numbers of sub-secrets at each resource provider. To make our scheme secure we make the determination of threshold value at the second level dynamic. We generate a share pool which is used to determine the number of shares at each resource provider using the Chinese Remainder Theorem (CRT) [105]. The advantage of our proposed scheme is that in the distributed cloud, users provide resources and therefore do not need to invest in the commercial cloud such as Amazon AWS [106], EC2 [107], Windows Azure [108] etc. Security in the distributed cloud has not been studied much and our scheme provides a secure way to protect the secret key. We perform the multilevel threshold secret sharing mechanism on the secret key itself as the key plays an important role in encrypting or decrypting the file content. In our proposed scheme the secret is distributed over multiple resource providers therefore an attacker cannot have the original secret until and unless all the participating resource providers are compromised. We generate the replica of each secret share to ensure the fact that if a provider is compromised then that particular share is available from other providers. Moreover the user can revoke the access of any resource provider if the provider is compromised. We increase the chances of detecting a compromised provider by introducing dummy key shares at each resource provider. We use Shamir's threshold secret sharing scheme [99]. This is an ideal secret sharing scheme as the size of the secret is identical with the size of shares. A lot of work

[102, 109, 110] has been developed based on [99]. A threshold ideal secret sharing has been proposed in [109] which uses only XOR operations to make shares and to recover the secret. The authors also show that this scheme is secure and faster with reduced storage usage and computational cost compared to [99]. In [110] a multilevel threshold secret sharing scheme has been proposed with two modifications of [99]. The first modification allows the shareholders to keep both the x coordinate and y- coordinate of a polynomial as a private share and in the second modification a polynomial degree larger than the threshold value is used by the dealers to generate the share. In our scheme we perform splitting the secret twice, once by the user and the other by the resource provider. At resource provider along with the shares we also store dummy shares which improves security. Another type of secret sharing scheme, Linear secret sharing scheme [99, 100, 102] is based on linear algebra and it contains super polynomial lower bounds on the share size. Though [99] can only realize the access structure of the secret sharing scheme, [102] can design a secret sharing scheme from any given access structure. A multi linear secret sharing scheme which uses super polynomial bounds on the share size has been proposed in [101]. In [111] the authors introduce a hierarchical threshold secret sharing scheme. In this scheme the secret is shared in a hierarchical structure among groups of participants which is further partitioned into levels. In our multi-level threshold secret sharing scheme we provide security to protect secret key in the distributed cloud and we generate dummy keys and replica of each secret share to detect existence of a compromised provider.

5.2. DESIGN CONSIDERATION

In the distributed cloud model [95, 112], computation and data is stored in resources provided by other users as shown in Fig. 3. Since users use resources provided by other users, there are obvious security concerns. Dependable Storage in the Intercloud [71], provides reasons why the single cloud is not secure, namely, we need to have more than one cloud to make the system more secure. The Distributed cloud model by itself solves the problem of single domain cloud as it has more than

one user who will be acting as a cloud provider. In the distributed cloud, protecting data stored on other users' resources poses security issues, which is handled using standard encryption techniques, which in turn leads to key management issues and insider attacks. A resource provider can get access to the data stored on his resources with ease and can use brute force attacks on it. Security of data storage currently depends on the strength of the encryption keys and the effectiveness of the key management schemes use. So instead of having a single key that gives access to the encrypted files, we propose using multiple shares. In our proposed secret sharing mechanism in the distributed cloud we use secure mechanisms to enhance the security of secret key shares. We have a share pool that determines the number of shares for a particular key share. In the distributed cloud, since users can join and leave the cloud, we need to make sure this resource pool is always available. So we make duplicates of this pool and store them for multiple resource providers in the distributed cloud.

5.3. PROPOSED SOLUTION

5.3.1. PRELIMINARIES

In this section we discuss the basic preliminaries we use in this work, namely, Shamir's threshold secret sharing scheme and the Chinese Remainder Theorem.

5.3.1.1. SHAMIR'S THRESHOLD SECRET SHARING

We use Shamir's (k, n) threshold secret sharing scheme [99]. Shamir's (k, n) threshold secret sharing scheme is based on polynomial interpolation. A secret S can be shared by n number of users and can be reconstructed as well, if the number of shares to reconstruct exceeds some threshold value k. This scheme uses a polynomial function of order $(k-1)$ which can be constructed as,

$$f(x) = d_0 + d_1 x + d_2 x^2 + \ldots + d_{k-1} x^{k-1} \quad (11)$$

Here d_0 is the secret S.

Given any k shares $< x_0, f(x_0) >, \ldots, < x_{k-1}, f(x_{k-1}) >$ the secret S can be reconstructed using the Lagrange Interpolation formula as follows:

$$S = \sum_{i=0}^{k-1} (\prod_{i \neq j} \frac{x_j}{x_j - x_i}) f(x_i) \quad (12)$$

5.3.1.2. CHINESE REMAINDER THEOREM

We use the Chinese Remainder Theorem (CRT) [105] to generate the threshold value at each resource provider at the second level of the threshold secret sharing scheme. Given the following system of equations,

$$x = a_1 \, mod \, p_1$$

$$x = a_2 \, mod \, p_2$$

$$\ldots\ldots\ldots\ldots$$

$$x = a_k \, mod \, p_k$$

There is one unique solution as

$$x = \sum_{i=1}^{k} \frac{P}{p_i} y_i a_i mod \, P \quad (13)$$

Where $P = p_1 p_2 \ldots p_k$ and $\frac{P}{p_i} y_i mod p_i = 1$, if all moduli are pairwise coprime, i.e., $\gcd(p_i, p_j) = 1$ for every $i \neq j$.

5.3.2. PROPOSED SCHEME

The Distributed cloud is formed using the resource provided by users and can provide resources for free to everyone who is part of the system. The Distributed cloud makes use of virtualization to allocate resources to multiple users and shares available resources efficiently and it also avoids single point of failure. In this work we propose a scheme to protect the secret key using a multilevel

threshold secret sharing mechanism as shown in Fig. 20 in the distributed cloud environment. Our proposed scheme encrypts the file using the secret key before distributing the key shares among participant resource providers whom we assume to be honest. At the first level the user splits the key and distributes the shares among resource providers. Instead of attaching key shares as metadata to the pieces of data [71] we split the key shares at each resource provider again into multiple shares in the second level. The second level of our mechanism improves the security since to get the original secret the attacker has to have all the shares from the two levels. We generate the threshold value in the second level dynamically which enhances the security as the attacker cannot know about the threshold value beforehand. In addition to that at each resource provider we create dummy keys which increases the probability of knowing if a resource provider is compromised by any attacker. In Algorithm 5.1 we ensure the security of the secret key in a distributed cloud environment. To do that we use the multilevel threshold secret sharing scheme (t, n) in which the first level uses the threshold $t < n$ and the second level uses the threshold $t = m$.

First the user splits the secret key into n number of shares S_i,where $i \in (1, n)$ and distribute them among all resource providers (Algorithm 5.2. line 1-7). At this level the threshold value $t < n$ is set which indicates that to reconstruct the secret key at least t number of shares would be required. Each share of secret S_i is replicated into k numbers so that if one resource provider goes offline or is compromised then that share can be accessed from other resource provider. At each resource provider we further split the share S_i into m number of shares S_{ij} (Algorithm 5.2. line 12). At this level, i.e., second level the threshold value m is generated dynamically. To determine the number of shares m each resource provider R_i, $i \in (1, n)$ selects a (P_i, N_i) pair from the share pool SP (Algorithm 5.2 line 9-12). The share pool is created beforehand (Algorithm 5.1. Generate share pool SP) .The user saves the pair (i, P_i) for each provider R_i and the provider saves N_i. We intend to have multiple share pools and place one or two of them in each cluster of the distributed cloud. The CRT solution generates a number m which decides the number of shares to split and reconstruct

in the second level. At each resource provider at least j number of dummy shares S_{ij}^{D}, where i $\in$ {1, n} and $j \geq m$, are generated (Algorithm 5.2 line 13). Whenever a resource provider is compromised, the user revokes the access of that particular resource provider. We intend to have a greater number of dummy shares than secret shares,i.e.,$j \geq m$, so that if any outside attacker tries to get the share, the probability that he ends up with the dummy share instead of a real share is greater than or equal to 0.5. This helps the user to take action (e.g., revoke the access of that resource provider) accordingly when some attacker selects a dummy key. To reconstruct the key sub shares, each resource provider need to have P_i from the user to generate the threshold value m (Algorithm 5.3. line 2). The user reconstructs the secret S from t numbers of S_i shares.

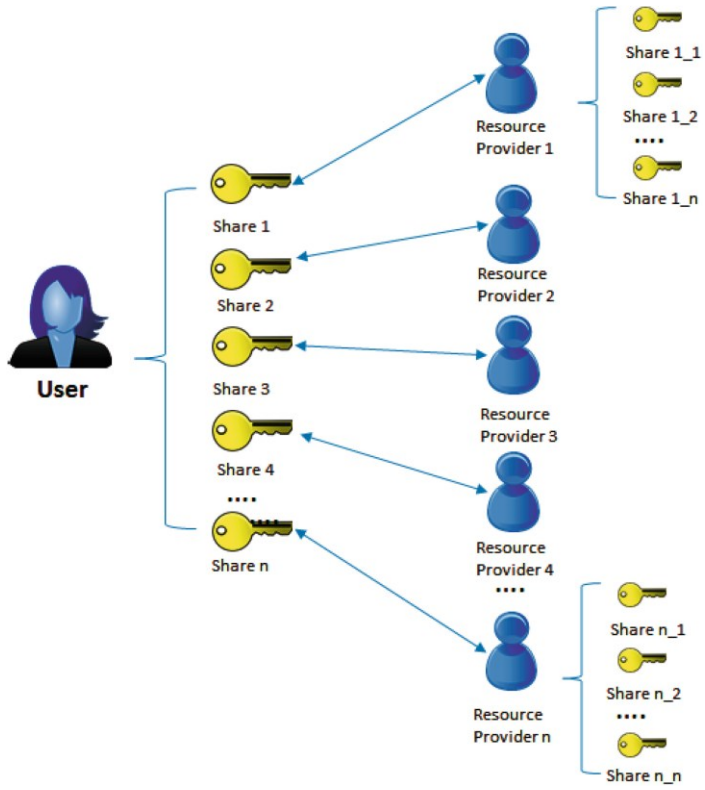

Figure 20: Proposed multilevel threshold secret sharing scheme

5.3.2.1. ALGORITHM

INPUT: Secret key S, n number of resource providers in the distributed cloud, threshold value $t \leq n$.

ALGORITHM 5.1.GENERATE SHARE POOL *SP*

From this share pool each resource provider generates a number out of a selected pair.

1. $for\ i = 1\ to\ atleast\ n$
2. Generate a random series of pairwise relatively prime positive integers, $P_i = \{p_{i1}, p_{i2}, \ldots\ldots, p_{im}\}$.
3. Generate a random series of m arbitrary integers $N_i = \{n_{i1}, n_{i2}, \ldots\ldots, n_{im}\}$.
4. Place these two series P_i along with N_i , represented as (P_i, N_i) in SP.
5. $end\ for$

ALGORITHM5. 2. SPLIT ALGORITHM

1. User U encrypts the file f with secret key S.
2. Split the secret key S into n number of shares, $S_1, S_2, S_3, S_4, \ldots\ldots, S_n$.
3. To reconstruct S at least t number of shares are required.
4. $for\ each\ share\ i\ of\ S\ , where\ i\ \in \{1, n\}$
5. Replicate the share S_i into $k\ (\geq 1)$ number of replicas.
6. Distribute the replicas among n number of resource providers.
7. $end\ for$
8. $for\ each\ resource\ provider\ R_i, i \in \{1, n\}$
9. Select a pair (P_i, N_i) from SP.
10. User U saves (i, P_i) and R_i saves N_i.
11. Get a unique solution $m\ = x_i$ from (P_i, N_i) using Chinese Remainder Theorem (CRT).
12. Split the share of secret S_i into m number of shares, $S_{i1}, S_{i2}, S_{i3}, S_{i4}, \ldots\ldots, S_{im}$.
13. Generate j number of dummy shares S_{ij}^{D}, where $j\ \geq m$.
14. $end\ for$

ALGORITHM 5.3.RECONSTRUCTION ALGORITHM

1. $for\ each\ resource\ provider\ R_i, i \in \{1, n\}$
2. R_i asks for (i, P_i) pair from user U and R_i has N_i.
3. Get the unique solution $m = x_i$ from (P_i, N_i) using Chinese Remainder Theorem (CRT).
4. Reconstruct the share of secret S_i from m number of shares, $S_i = S_{i1}, S_{i2}, S_{i3}, S_{i4}, \ldots\ldots, S_{im}$.
5. $end\ for$
6. $for\ each\ resource\ provider\ R_i, i \in \{1, t\}$
7. Collect the share S_i from each resource provider.
8. $end\ for$
9. Reconstruct the secret S from S_i where $i = \{1, \ldots, t\}$.

5.4. RESULTS

5.4.1. EXPERIMENTAL ANALYSIS

We performed simulations using a distributed cloud model that is based on the P2P overlay Kademlia [114]. We implemented our algorithm for secret sharing on the distributed cloud. We assumed that resource providers have a minimum of 2 GB RAM up to 16 GB, 2 to 8 cores. First a node identifies available resource providers near it and then divides the key into multiple shares and distributes each share to an available resource provider. The resource provider again creates more shares by using its share as the secret and stores it. The user will maintain the list of providers who store the secret shares. When a user requires the key, she contacts other resource providers who have the shares. After resource providers receive the request, they combine the shares they have and send the actual share to the user. Once the threshold numbers of shares are received by the user, she will reproduce the original key and use it.

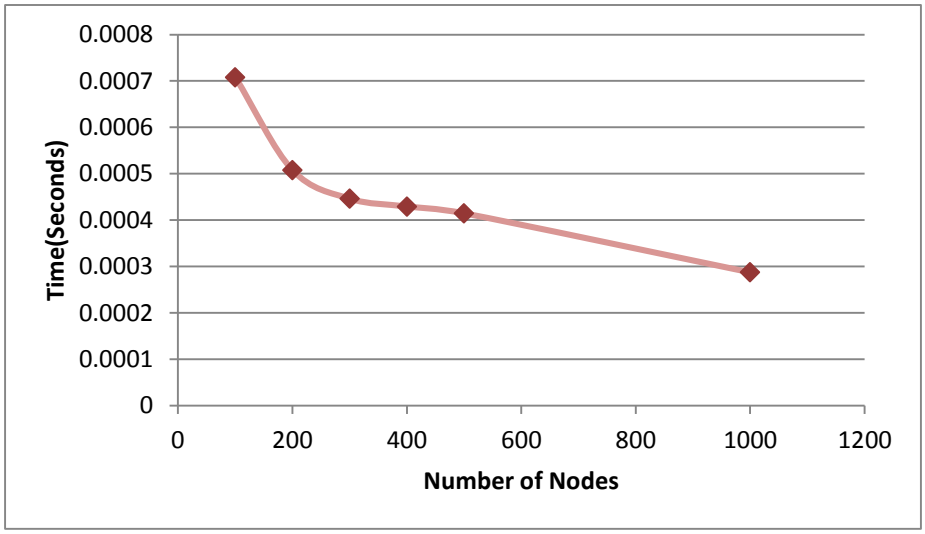

Figure 21: Time to split shares and distribute them to other nodes

Fig. 21 shows the average total time to split the secret into shares at the first level, find resources, distribute the shares to resource providers and split the shares at the second level. We can see that as the number of nodes increases, the time to find nodes and distribute shares to nearby nodes decreases. Since the distributed cloud is formed by many users we can see our mechanism for secret sharing is feasible and efficient.

Fig. 22 shows the time to retrieve the shares from resource providers and combine them to reproduce the secret. We see that the average time to retrieve shares from resource providers and to combine them takes significantly less time when compared to distributing the shares. In order to make sure that there are shares available to the user at any time replicas of shares are made and stored on multiple resource providers in the distributed cloud. This increases the chances of retrieving the share efficiently even when some of the nodes leave the network.

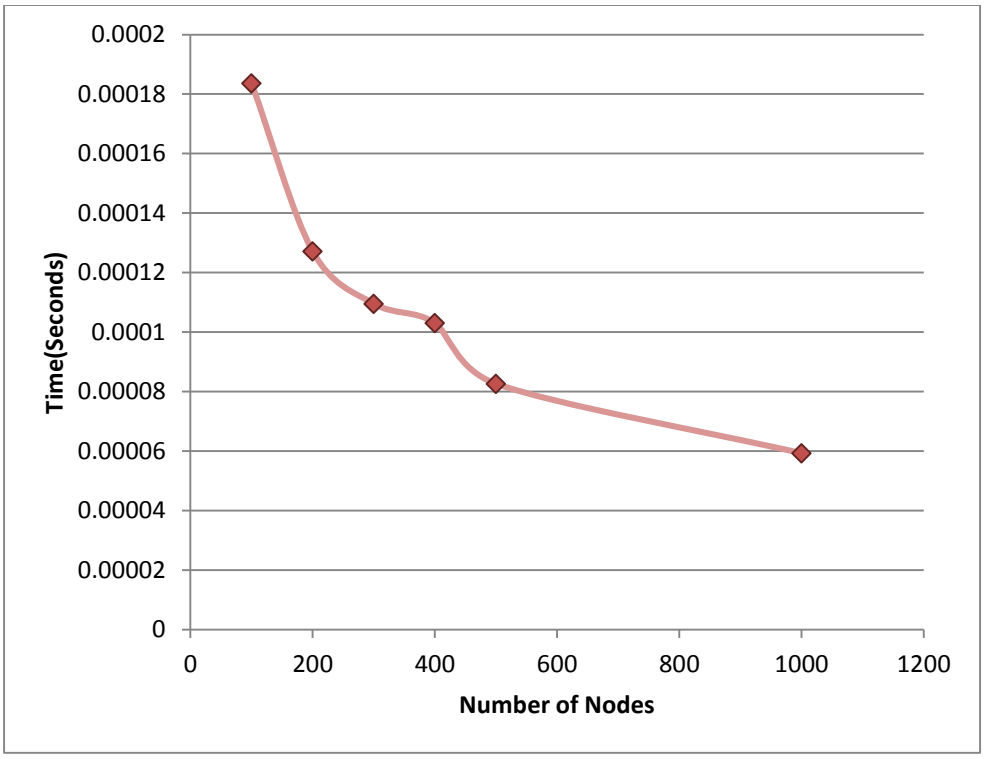

Figure 22: Time to retrieve shares and combine them

5.4.2. SECURITY ANALYSIS

In this work since we are using a threshold secret sharing scheme and we are storing the shares and their replicas on multiple resource providers, compromising one or more resource providers will not disclose the secret. At the second level at each resource provider we generate equal or more dummy shares which an attacker may access as actual shares. If the resource provider is compromised by an outside attacker then the probability of using the dummy shares is greater than or equal to 0.5. This increases the chances for a resource provider to react to any security breach immediately. We create a share pool to reduce the user overhead in deciding the number of shares to be split. Since the resource provider selects (P_i, N_i) pairs randomly no eavesdropper can make a guess about the number of shares beforehand.

5.5. CONCLUSIONS

We propose a multilevel threshold secret sharing scheme to ensure the security of secret key in the distributed cloud environment. Our scheme can also be used for other distributed and P2P systems. Our findings show that a secret key, which plays a significant role to encrypt and decrypt the file content stored in cloud storage, can be stored securely in a distributed cloud environment. The increasing number of providers in the distributed cloud decreases the time to split and distribute the shares. Experimental and security analysis shows that our scheme is feasible and secure. Data replication is another issue in distributed systems and we will look into new strategies to perform data replication to increase response time for retrieving data and making the distributed cloud more robust.

CHAPTER VI

PRIVACY PRESERVING DISTRIBUTED CLOUD STORAGE

6.1. LITERATURE REVIEW

Cloud computing [8, 9, 115] is an innovative model which has attracted plenty of attention from both industry as well as academia. Cloud computing allows users to manage resources in a flexible and scalable manner. There is a substantial increase in number of IT firms, schools and regular users using the cloud resources. Cloud resources have been replacing desktops, laptops, large storage devices, servers, etc. In order to effectively use these available resources, the distributed cloud computing model [112, 116, 95] has been introduced. Virtualization [117] is key concept for making cloud computing possible. Xen is one of the open source standard for hardware virtualization used by many cloud providers. Hyper-V, KVM, and Sun xVM are some of the other key virtualization management tools used. Cloud providers provide resources in form of various instances based on user needs. For example Amazon provides for different sizes of virtual machine instances – Small, Medium, Large and Extra Large. Moreover these resources are scalable and can be increased or decreased according to needs. These virtualization tools can be run on small machines like simple desktops and laptops.

The distributed cloud uses virtualization techniques to effectively use resources which are lying idle. Cloud computing uses a centralized architecture. With the increase in the number of users network traffic has increased tremendously at data center locations. Moreover it requires huge amounts of energy to maintain these data centers. Distributed cloud can be used to mitigate these

problems of a centralized cloud and use unused resources effectively thereby providing a green computing model. The distributed cloud provides all the services that a cloud can provide. Since these resources are provided by individual users located in a wide geographical area, the network traffic is also widely distributed. In addition there isn't a single point of failure as it uses a completely decentralized architecture. Security and privacy is a concern in both the distributed cloud as well as the existing cloud models. Since users store and do computation on resources provided by other users we need a secure and privacy preserving mechanism to store and retrieve data. Confidentiality, integrity and availability are three major security concerns along with privacy when resources are provided by others. Data confidentiality and integrity can be ensured using encryption mechanisms. Availability depends on the resource management model. A novel resource management mechanism for distributed cloud using Multi-valued hash table mechanism and game theoretic model [116, 62] has been proposed. However there has not been any research on preserving privacy of data in the distributed cloud. In this work we propose a secure and privacy preserving mechanism to share, store and retrieve data in a distributed cloud model.

6.2. DESIGN CONSIDERATION

The resource provider is a user who provides the services in a distributed cloud. Resources can be any kind of service offered by a resource provider. Software as a Service, Platform as a Service, Infrastructure as a Service and Storage as a Service can be provided by the resource provider. The distributed cloud provides storage and computation resources in a peer-to-peer fashion over an open network. This leads to security and privacy issues that need to be resolved. Transferring data and processing information to another user providing the service requires security and privacy. If we send data and information in an encrypted format, the resource needs to have the key in order to update it. Providing the security key to the resource provider leads to security and privacy issues. Moreover when we need to search the data using keywords, all the information regarding the data

provider, data content, and the details of queried data can be monitored by an outsider or malicious party.

The security model we propose encrypts the data and its names using a homomorphic encryption scheme. To retrieve the data the file name is encrypted to avoid advertising the real keywords we are searching for over the network. In order to preserve privacy of the contents of the file, i.e., the actual data, we split them and perform a homomorphic encryption on the data and store it on the resource providers. We use homomorphic properties to query and validate the resultant data. In this work we use homomorphic encryption to store or share the files in a privacy preserving manner among different resource providers. We use the multiplicative homomorphic encryption scheme. RSA, ElGamal [118, 119] encryption schemes support multiplicative homomorphic property. According to the multiplicative homomorphic encryption scheme if $E(x)$denotes the cipher text of plain text x then it satisfies $E(x_1) \cdot E(x_2) = E(x_1 \cdot x_2)$.

Privacy preserving is one of the main issues in both distributed and centralized cloud computing systems. In [120], authors used public key homomorphic authenticator mechanism to achieve privacy preserving public cloud data auditing systems. In the cloud since all the resources are in one location, auditing information should be preserved. In [15], the author uses public key encryption based keyword search in cloud so that we need not decrypt the entire message. But in distributed cloud we only search for exact keywords to find the file name and we don't want to decrypt the data to find out the value of the keyword. This method of searching encrypted data over an encrypted pool of messages is computationally more expensive. In [17, 121], authors proposed homomorphic encryption to protect sensitive data stored in a cloud data center and shared by multiple clients. Homomorphic encryption mechanism is used to preserve security and privacy of data by using additive or multiplicative properties. In order to check the similarity between two encrypted files, Euclidean distance can be used to measure the distance between them [122].

6.3. SECURITY AND PRIVACY PRESERVING MECHANISM

In our proposed solution we consider security and preserving privacy of data to be stored on resource providers. Since we are using a distributed cloud, the computation and storage are on resource providers. In order to retrieve the data we use file names or key words to search in the distributed cloud. The key idea is to split data/file into multiple partitions and encrypt those partitions using a multiplicative homomorphic encryption scheme as shown in Fig. 23. For the file name of a data or keywords we do not use the split mechanism.

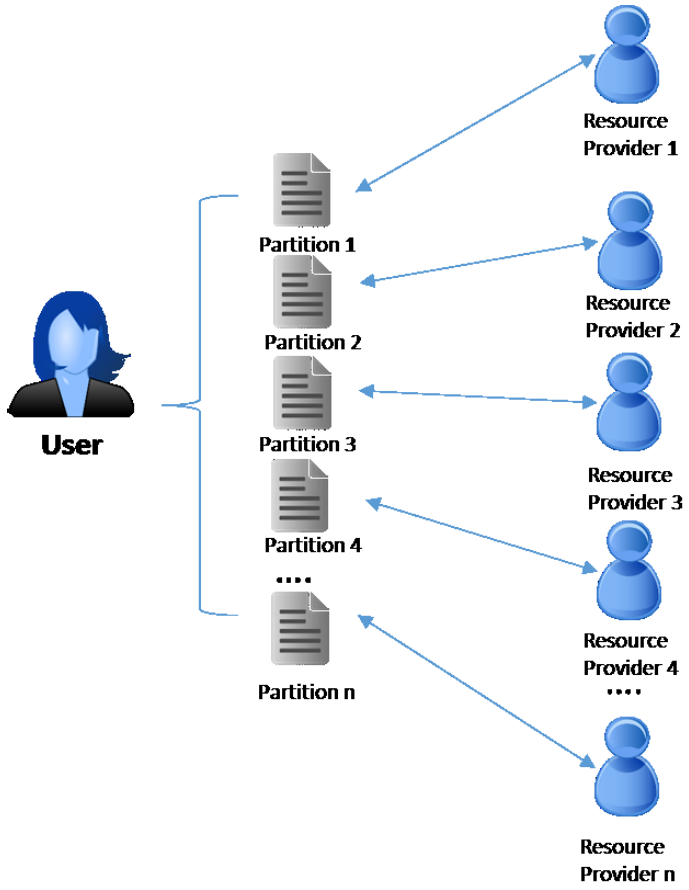

Figure 23: Overall view of proposed scheme

Preserving security and privacy of data is a twofold process. The first step is to encrypt the name of the data and in the second step encrypt the actual data. To store and retrieve data in a distributed cloud we propose two algorithms. Algorithm 6.1 and Algorithm 6.2 describe the procedure to store data on resource providers and retrieve data from resource providers respectively. To store data on resource providers we encrypt the data name/file name (Algorithm 6.1 Step 1) and distribute the data content among different resource providers (Algorithm 6.1 Step 2). Encrypting the data name/file name is a simple one step process. We split the original data content into multiple random partitions and encrypt each partition using a multiplicative homomorphic encryption scheme. We then calculate the Euclidean distance between different random encrypted data partitions before distributing them among different resource providers. Retrieving the data from different resource providers includes validating the data name/file name (Algorithm 6.2 Step 1) and retrieving the data partitions (Algorithm 6.2 Step 2) from different resource providers. To validate the data name/file name we calculate the distance between encrypted data/ file name. If the distance is zero then we proceed to retrieve the data partitions of that particular data/ file. After all the partitions have been retrieved the user measures the distances between the encrypted random partitions and compare them with distances stored. If the same distance values are the same, this indicates that the file has not been tampered with.

6.3.1. ALGORITHM

ALGORITHM 6.1. STORING DATA ON RESOURCE PROVIDERS

INPUT: File name f, Data m , Public key–private key pair (x, y).

Step 1: Encrypt the data name/file name

1. Encrypt the file name f using a multiplicative homomorphic encryption scheme.

Step 2: Distribute the data among different Resource Providers

1. Split the data m into n number of partitions, $m_1, m_2, \ldots, m_n$.
2. $for\ each$ data partitions $m_i\ \forall i\ \in \{1, n\}$
3. Encrypt the data partitions using a multiplicative homomorphic encryption scheme, $E(m_1), E(m_2), \ldots, E(m_n)$.
4. $end\ for$
5. Choose a random number $r \mid 1 < r < n$.
6. Calculate the Euclidean distance d between r random pair of encrypted data partitions.
7. $for\ each\ r$ random pair of encrypted data
8. Store the calculated Euclidean distances $d_1, d_{2,\ldots}, \ldots, d_r$ in$< distance\ (d_i), E(m_j), E(m_k) >$ format. Here d_i is the distance between $E(m_j)\ and\ E(m_k)$.
9. $end\ for$
10. Distribute the encrypted data partitions among resource providers.

ALGORITHM 6.2: RETRIEVE THE DATA FROM RESOURCE PROVIDERS

Step 1: Validate the name

1. Search using the encrypted file name $E(f)$.
2. $for\ all$ possible resource providers R_i
3. $for\ all$ available encrypted file names $E(f_{rpi})$ in the resource provider R_i
4. Calculate the Euclidean distance $D(C)$ between $E(f)$ and $E(f_{rpi})$
5. $if\ D(C) = 0$
6. Retrieve the data partitions using Step 2.
7. $end\ if$
8. $end\ for$
9. $end\ for$

Step 2: Retrieving the data

1. $for\ each\ r$ random pair of encrypted data partitions $E(m_i')$ retrieved
2. Calculate the distances, $d_1', d_2', \ldots, d_r'$, between encrypted pair of random data partitions.
3. Compare $if\ d_i = d_i'$ is true
4. Data partition is not tampered.

5. Decrypt the Data partition with private key y.
6. $end\ if$
7. $end\ for$

6.4. ANALYSIS

In this work we analyze our proposed solution in terms of correctness analysis, privacy analysis and performance analysis.

6.4.1. CORRECTNESS ANALYSIS

In our solution we calculate the Euclidean distance between two encrypted file names $E(f)$ and $E(f_{rpi})$ (Algorithm 6.2 Step 1 line 4). Since we encrypt the data using a multiplicative homomorphic encryption scheme we can perform arithmetic operations on those encrypted data. To prove correctness, the decrypted result of the Euclidean distance $D(C)$ (Algorithm 6.2 Step 1 line 5) between two encrypted data yields the Euclidean distance between the two original data.

$$
\begin{aligned}
&D(C)\\
&= D(E(f) * E(f) + E(f_{rpi}) * E(f_{rpi}) - 2 * E(f) * E(f_{rpi}))\\
&= D(E(f - f_{rpi})^2)\\
&= (f - f_{rpi})^2
\end{aligned}
$$

A zero Euclidean distance between f and f_{rpi} data denotes that they are same. Similarly to retrieve the data partitions we calculate the Euclidean distances between them.

6.4.2. PRIVACY ANALYSIS

In this subsection we explain how we preserve the privacy of data. We distribute the data partitions in encrypted format among different resource providers (Algorithm 6.1 Step 2 line 3) and the key pair (x, y) is only known to the user. Therefore no other party can learn anything

but $E(m_1), E(m_2), \ldots, E(m_n)$. We calculate the distance between encrypted file names and retrieve the data partitions only when the distance between the encrypted file names is zero. We also calculate the distance between the random pairs of encrypted data partitions to verify the integrity of data. Our proposed scheme preserves privacy and is secure since no third party or adversary can know about the original data as we have distributed the data partitions in encrypted format and compute operations on encrypted data where the key pair is only known to the user.

6.4.3. PERFORMANCE ANALYSIS

We implement our algorithm for different sizes of file storage in a simulated distributed cloud environment. To evaluate the performance of our algorithm we vary the sizes of files and perform a privacy preserving homomorphic encryption scheme on the data partitions and file names. To query encrypted file names our algorithm takes 0.2 milliseconds. To store different sizes of files in the distributed cloud environment and to retrieve them back using our proposed solution takes 0.2 second for 10KB of data. Fig. 24 depicts the evaluation times of our algorithm for varying file sizes. From the performance evaluation we can see that the elapsed time increases linearly with increasing file sizes. In order to meet time constraints the partition size can be varied accordingly.

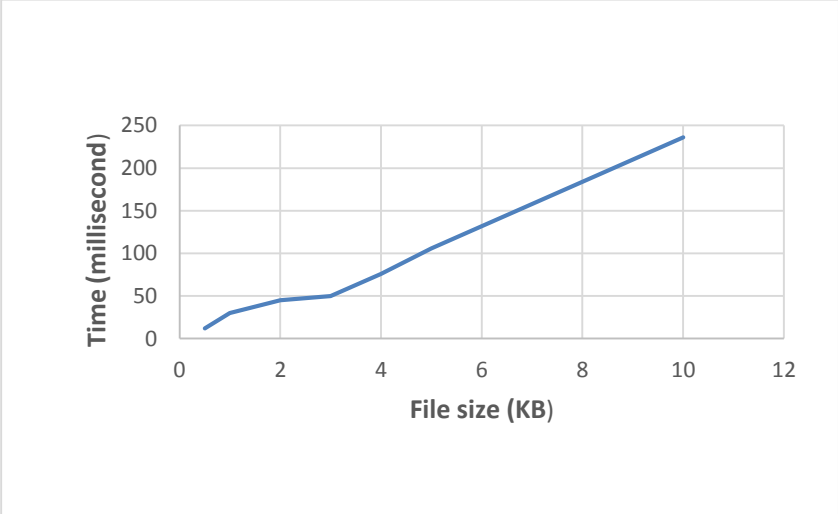

Figure 24: File Size - Elapsed Time

6.5. CONCLUSIONS

We use homomorphic encryption mechanism and its multiplicative property to preserve security and privacy of data in a distributed cloud. Our method works on encrypted data without revealing the actual data. We also show that our algorithm is correct and preserves the privacy of data. The performance analysis of our algorithm shows its efficiency and feasibility. This work shows that we can preserve privacy and security of data in the distributed cloud. In future we intend to work on preserving privacy of database records in the distributed cloud.

CHAPTER VII

SECURE AND PRIVACY PRESERVING BIOMETRIC AUTHENTICATION USING WATERMARKING TECHNIQUE

7.1. LITERATURE REVIEW

To authenticate the identity of any user, biometric authentication is one of the most trustworthy techniques as it involves verification of users biometric traits. Biometric authentication systems authenticate users based on their physical traits, such as, fingerprint, face image, iris, signature etc. which are unique. Different biometric authentication mechanism has been used extensively in various domains. In [123] the authors proposed a framework for three–factor authentication scheme in a distributed system. A template protecting biometric authentication scheme to fingerprint data has been implemented in [124]. In this finger print based authentication mechanism the authors have presented an algorithm which identifies the reliable components of Gabor filtered fingerprint and applies quantization to make a binary representation of the fingerprint. Noise correction is applied on the quantized binary representation. A multimodal biometric authentication scheme for adapting score–level fusion functions based on quality measures has been proposed in [125]. In [126] a novel and efficient facial image representation based on local binary pattern (LBP) has been proposed which extracts the LBP feature distributions by dividing the face image into several regions and concatenating them into an enhanced feature vector to be used as a face descriptor. Instead of using two different biometric traits a personal identification scheme using palm print and hand geometry which can be acquired from the same image has been proposed in [127]. The authors

have integrated the hand geometry on the palm print to improve the performance of the verification system.

Watermarking is a technique to embed specific data in a digital content. The watermarking scheme has enormous diversity. It can be categorized [128, 129, 130] based on the embedding domain (Spatial domain, Transform domain, Feature domain), watermarking host signal (video signal, audio signal, IC design, etc.), availability of original signal during extraction (blind, semi–blind, non–blind) etc. The watermarking technique is advantageous because, to embed the watermark into the original data this technique does not create any separate file to store authentication information. Besides all the advantages of the watermarking technique, any modification on the embedded data can be manipulated easily. Therefore one important requirement of the watermark technique is to make it almost unrecognizable and robust so that watermark cannot be removed or modified by any attack. To make the biometric watermarking technique more secure, encryption based privacy preserving techniques can be applied. Some watermarking techniques (e.g., fragile, robust) becomes invalid or detectable if a slight modification or some image processing operations such as image scaling, cropping, bending is done on the watermarked image. As any individual's biometric features are unique therefore these biometric features should not be disclosed under any circumstances to any adversary. In our work, we focus on the privacy preserving secure biometric watermarking scheme for authentication purposes. We preserve the privacy of users' watermarked biometrics using cryptography. To overcome the vulnerability of biometric authentications and to protect the privacy of a user's watermarked biometrics, we use biometric authentication in conjunction with digital watermarking and cryptography. We embed the fingerprint on the facial image as a watermark to improve the security of user authentication. To achieve the privacy of a user's biometric traits we encrypt the watermarked biometric before user verification.

A number of biometric watermarking techniques has been proposed. In [128] the authors proposed a semi–blind biometric watermarking scheme using both a watermarking technique and face image

recognition. The face features are embedded in the non-overlapping blocks of the host image using Singular Value Decomposition (SVD) .A robust hybrid biometric watermarking approach for offline handwritten signature has been proposed in [129]. The authors amalgamate the lifting wavelet transform (LWT) and SVD to make the biometric watermarking technique robust. In [130] a watermarking scheme in spatial domain for color image using the SVD technique has been introduced. Three dimensions of color iris images are used to embed into color face image to make the technique robust and reliable. An image ownership and tampering authentication scheme based on watermarking techniques has been proposed in [131]. The watermarking technique is used here to detect malicious manipulation over embedded images and to protect the rightful ownership. In [132] the authors proposed an efficient and secure encryption scheme to transmit the biometric data over unsecured data channel. An asymmetric watermarking technique has been proposed in [133]. In [134] a layered architecture for watermarking technique has been proposed which provides security by using cryptography primitives on top of the watermarking algorithm. Many biometric watermarking techniques have been proposed but watermarked biometric can be a threat to security if it is attacked by a determined attacker. In this work we propose a biometric authentication scheme while ensuring privacy of the users' watermarked biometrics.

7.2. DESIGN CONSIDERATION

We verify the authentication of a user in a privacy preserving secure manner using the user's biometrics, watermarking scheme and cryptography. The problem can be framed formally as follows. There are n number of users $U_1, U_2, \ldots, U_i, \ldots, U_n$ and a server S which holds a database D_{fv} with all users' encrypted watermarked feature vector $WSDB_{fv}$. Given a user $U_i's$ biometrics, i.e., face image (I) and fingerprint(F), the user's face image 'I' will be watermarked with fingerprint $'F'$ and will be denoted as WB_{fv}. The Euclidean distance ∂ between the user's encrypted

watermarked feature vector WB_{fv} and the encrypted watermarked feature vectors $WSDB_{fv}$, in D_{fv} decides the user's authentication (Eqn. 14).

$$\partial\left(E\left(WB_{fv}\right), E\left(WSDB_{fv}\right)\right) \leq \tau \tag{14}$$

τ denotes the threshold value and if $\partial \leq \tau$, the algorithm determines that the user is authentic.

7.3. PRELIMINARIES

7.3.1. HOMOMORPHIC ENCRYPTION

We calculate the Euclidean distance between two cipher texts and to do that we use a homomorphic property based encryption scheme. In this work we use multiplicative homomorphic property. For example, RSA and ElGamal [15, 17] supports the multiplicative homomorphic property, i.e., if $E(x)$denotes the ciphertext of plain text x then according to multiplicative homomorphism, $E(x_1) \cdot E(x_2) = E(x_1 \cdot x_2)$.

7.3.2. SINGULAR VALUE DECOMPOSITION

In Linear Algebra, Singular Value Decomposition (SVD) [135] is a factorization technique of a real or complex matrix. In signal processing SVD has been used in applications, e.g., watermarking, noise reduction, image compression, etc. According to the SVD theorem any rectangular matrix A of $m \times n$ dimension can be decomposed into the product of three matrices: an orthogonal matrix U of size $m \times m$, a diagonal matrix S of size $m \times n$ and the transpose of a orthogonal matrix V of size $n \times n$.

$$\mathrm{A_{m\times n}} = \mathrm{U_{m\times m}}\,\mathrm{S_{m\times n}}\,\mathrm{V_{n\times n}}$$

The original matrix can be obtained by multiplying $\mathrm{U, S, V^T}$.

$$\mathrm{A} = \mathrm{U_{m\times m}}\,\mathrm{S_{m\times n}}\,\mathrm{V^T_{n\times n}}$$

SVD based watermarking techniques can be categorized into many categories [9], such as, by modification right/left singular vectors of the host images, modification of all singular vectors, hybrid transformation techniques which combines SVD with other transformations(DWT, DFT, DCT, etc.), etc.

7.4. PROPOSED SOLUTION

We propose a solution for privacy preserving secure biometric authentication scheme using a watermarking technique. We consider two biometric traits, face image (I) and fingerprint(F) for biometric watermarking authentication purpose and extract the feature vectors from both of these biometrics. To make the entire authentication procedure secure we perform some specific transformations on the real biometric feature vectors and use transformed biometrics instead of the real biometrics. The reason behind distorting the feature vectors is to make the authentication more secure. Therefore even if any adversary manages to get the watermarked feature vector, they will not be able to get the transformed matrices. Since anyone's face image can be captured by any adversary we watermark the transformed face image with the transformed finger print image to make it more secure. We use a linear transformation, i.e., rotation on the actual feature vectors and denote it as transformed feature vector. We watermark the transformed feature vector of face image (TSI) with transformed feature vector of fingerprint (TSF) using the SVD based watermarking technique. Further computation for authentication is performed on the watermarked biometric. The watermarked feature vector is stored in the server to authenticate any user. To preserve the privacy of the user's watermarked biometrics we encrypt the watermarked biometrics using homomorphic encryption at both the server end and user end. A user is authenticated by calculating the Euclidean distance between the encrypted watermarked biometric provided from the user end and the encrypted watermarked biometric from the server end. Since the Euclidean distance between cipher texts will be calculated we use a homomorphic encryption technique as it allows computation on

cipher texts. If the distance less than some threshold value τ, this indicates that the user is authenticate. The overall flow of the entire authentication procedure is given in Fig. 25.

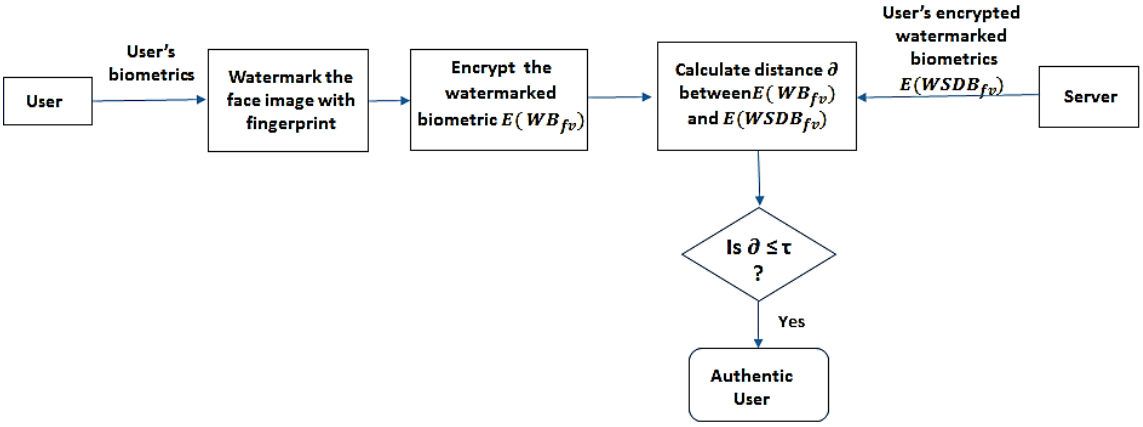

Figure 25: Overall flow of the authentication mechanism

7.4.1. ALGORITHM

In this section we describe the algorithm (Algorithm 7.1) for the proposed solution. First the feature vectors of the user's face image and finger print will be extracted. The feature vectors will be distorted using some linear transformation, e.g., rotation (Algorithm 7.1 line 5-6). To watermark the transformed face image, RI_{fv} with the transformed finger print, RF_{fv} we use SVD based watermarking technique [21] (Algorithm 7.1 line 7). The rest of the computations will be performed on the watermarked feature vector WB_{fv}.

We use the ElGamal encryption scheme to encrypt the watermarked feature vector WB_{fv}. Algorithm 7.1 at the user end computes $E(g^{WB_{fv}[i,j]})$and $E(g^{WB_{fv}[i,j]^2})$ and sends them to the server (Algorithm 7.1 line 8-12). The server already has the watermarked transformed feature vector for all users. To check the authentication the server computes the Euclidean distance between the encrypted feature vectors $WSDB_{fv}$ and WB_{fv} for a particular user (Algorithm 7.1 line 13). The server computes

$$C = \prod_{i,j=1}^{m,n} E(g^{WB_{fv}([i,j]^2)})\, E(g^{WB_{fv}[i,j]})^{-2(WSDB_{fv}[i,j]} E(g^{WSDB_{fv}[i,j]^2}) \qquad (15)$$

which is the Euclidean distance between the encrypted feature vectors. The decrypted Euclidean distance $D(C) = \partial \leq \tau$ denotes that the user is authentic.

INPUT: User's finger print F, user's face image I

Steps:

1. Generate private key–public key pair (x, y) where x is the private key and y is the public key.
2. The server S keeps a database DB for all users watermarked biometrics $WSDB_{fv}$
3. Get the feature vector matrix F_{fv} of finger print F
4. Get the feature vector matrix I_{fv} of face image I
5. Rotate on face image feature vector, $RI_{fv} = \text{Rotation}(F_{fv})$
6. Rotate on finger print feature vector, $RF_{fv} = \text{Rotation}(I_{fv})$
7. Run SVD based watermarking method to watermark RI_{fv} with RF_{fv}. The resulting biometric for user U_i becomes WB_{fv}
8. $for\ i = 1\ to\ n$
9. $for\ j = 1\ to\ m$
10. Compute $E(g^{WB_{fv}[i,j]})$and $E(g^{WB_{fv}[i,j]^2)})$ and send to server
11. $end\ for$
12. $end\ for$
13. Server computes $C = \prod_{i,j=1}^{m,n} E(g^{WB_{fv}([i,j]^2)})\, E(g^{WB_{fv}[i,j]})^{-2(WSDB_{fv}[i,j]} E(g^{WSDB_{fv}[i,j]^2})$
14. Server decrypts C with private key x to obtain $D(C) = g^{\sum_{i,j=1}^{m,n}(WB_{fv}[i,j] - WSDB_{fv}[i,j])^2}$
15. $if\ D(C) \leq \tau$
16. User is authentic
17. $else$
18. Discard user
19. $end\ if$

Algorithm 7.1.

7.5. SYSTEM ANALYSIS

In this section we analyze the proposed solution in terms of correctness, privacy and performance.

7.5.1. CORRECTNESS ANALYSIS

To prove the correctness of Algorithm 7.1 it should be shown that the server and the user compute the Euclidean distance without knowing the plain text of the feature vectors. Line no. 14 of Algorithm 7.1 can be proved by the homomorphic property of the ElGamal algorithm.

$$
\begin{aligned}
& D(C) \\
&= D\Big(\prod_{i,j=1}^{m,n} E\left(g^{WB_{fv}([i,j]^2)}\right) E\left(g^{WB_{fv}[i,j]}\right)^{-2(WSDB_{fv}[i,j])} E\left(g^{WSDB_{fv}[i,j]^2}\right)\Big) \\
&= D\Big(E\Big(\prod_{i,j=1}^{m,n} g^{(WB_{fv}([i,j])^2}\, g^{(WSDB_{fv}[i,j])^2)}\Big) \prod_{i,j=1}^{m,n} \left(E\left(g^{(WB_{fv}[i,j])}\right)^{-2(WSDB_{fv}[i,j])}\right)\Big) \\
&= D\Big(E\Big(\prod_{i,j=1}^{m,n} g^{(WB_{fv}([i,j])^2}\, g^{(WSDB_{fv}[i,j])^2} g^{(WB_{fv}[i,j])^{-2(WSDB_{fv})}}\Big)\Big) \\
&= D\left(E\left(g^{\sum_{i,j=1}^{m,n}(WB_{fv}[i,j]-\, WSDB_{fv}[i,j])^2}\right)\right) \\
&= g^{\sum_{i,j=1}^{m,n}(WB_{fv}[i,j]-\, WSDB_{fv}[i,j])^2}
\end{aligned}
$$

7.5.2. PRIVACY ANALYSIS

In this section we discuss the privacy of the proposed solution. We assume that the server is secure. Algorithm 7.1 is secure in a semi-honest model [31]. The user sends the encrypted biometrics to the server. Therefore any adversary will obtain nothing but the encrypted watermarked feature vector. Moreover the server does the computation on the cipher texts to obtain the distance, which implies that Algorithm 7.1 is secure in a semi honest model. The only decrypted text is the encrypted distance from which no one can learn about the user's biometrics which indicates privacy is preserved in our proposed solution.

7.6. EXPERIMENTAL ANALYSIS

7.6.1. EXPERIMENT SETUP

To test the performance we considered different face images sizes varying from 3KB–7KB and embed different finger prints on them using the watermarking technique. Before watermarking the face image we extracted the feature vectors of the face image and finger print and transformed them by a linear transformation technique. To extract the feature vector of biometrics we used the PCA - based Face Recognition System package using Matlab [60]. For watermarking, we used the SVD based watermarking technique proposed in [21]. To implement our technique we used the Java programming language and JAMA package [136], since feature vector comes in the form of a matrix. We used the SVD based watermarking technique, but because of the constraint of the JAMA package for SVD calculation we modified the library so that it would work for any type of feature vector matrix. The existing JAMA package works correctly only for a full rank $m \times n$ matrices with $m \geq n$.

7.6.2. PERFORMANCE ANALYSIS

We evaluate the performance of our technique in terms of time for different modules, namely, watermarking time, encryption time and distance calculation time. Fig. 26 shows the computation time to watermark the face image with finger print image using SVD based watermarking at the user end for different file sizes. We vary the file sizes from 3KB to 7 KB with 1KB intervals. Fig. 26 shows that if we increase the face image size then the time to watermark the biometrics increases almost linearly.

Figure 26: Computation time of watermarking for different file sizes

We encrypt the watermarked face image with ElGamal encryption. Fig. 27 depicts the time to encrypt the watermarked feature vector at the user end for different sizes of face images. The time to encrypt the watermarked face images increases linearly with increasing face image sizes. The user is authentic if the distance between the encrypted watermarked biometric at the user end and at the server end is below some threshold value τ.

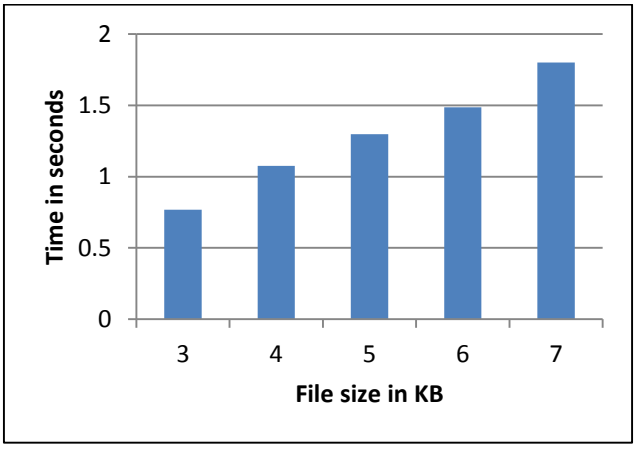

Figure 27: Computation time of encryption for different file sizes

Fig. 28 shows the computation time to calculate distance for a particular user at the server side. From Fig. 28 it is evident that with the increasing face image size the time to calculate distance increases linearly.

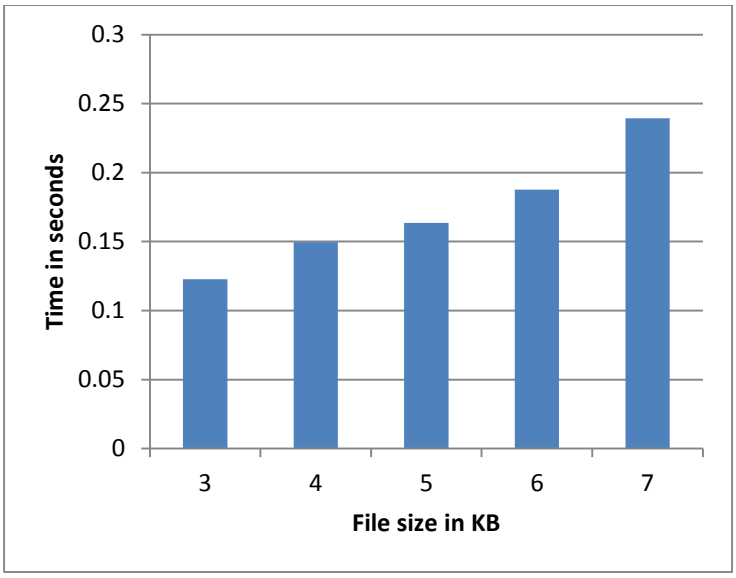

Figure 28: Computation time to calculate distance for different file sizes

7.8. CONCLUSIONS

We propose a secure privacy preserving technique to authenticate a user in a system using user biometrics. We transform the user biometric and watermark the face image with finger print to secure the authentication technique. To preserve the privacy of users watermarked biometric we encrypt it using homomorphic encryption. The analysis of our system proves the correctness and privacy of our scheme. The performance analysis shows the efficiency of our technique since time to authenticate a particular user is less than 3 seconds where the face image size is 7KB.

CHAPTER VIII

PRIVACY PRESERVING BIOMETRIC AUTHENTICATION FOR SECURE MOBILE PAYMENTS

8.1. LITERATURE REVIEW

Mobile devices face various challenges when it comes to M-commerce applications because of low bandwidth, low security, low computation power and highly complex device configurations. With the increase in mobile applications, security and privacy becomes a concern for mobile payments. There has been a lot of focus on mobile payments architectures in recent years. All the architectures revolve under how to perform transactions securely. Some of the technologies used for mobile commerce are NFC [137] a short range communication protocol, Mobile wallet [138] etc. Mobile wallet is an application residing on the mobile device which holds the information regarding the credit cards and can be used for payments as well. Authenticating these applications using existing mechanism is not sufficient; we need a more secure mechanism to authenticate mobile payment applications which resists theft of devices and other known attacks where user credentials are compromised.

Authentication is major requirement to secure any application, device or account. Biometric authentication mechanisms have gained popularity as they are more secure and offer non repudiation. A biometric authentication mechanism is proposed in [53] to secure mobile payments. It uses the public key of the bank to encrypt the credit card information and hashes of finger print to send to the bank. The bank server has information regarding all the users and their finger print

information. This method doesn't protect the privacy of the user. In [54], the authors have specified an identity based encryption (IBE) and biometric scheme for secure data access in the cloud. The biometric scheme used here authenticates the user and the IBE scheme was used to encrypt and decrypt the data. Ross and Othman [55] used visual cryptography to protect the privacy of a user. They decompose a user's biometric template into two noise like images using visual cryptography and store them in two distinct databases. During authentication these images are laid together to create a temporary biometric template for verification. The main advantage of this mechanism is that the biometric information is never exposed to an attacker as there is more than a single database. This method is practically very expensive as it requires maintaining two databases and during authentication the search process is doubled. Barni et al. [56] designed a privacy preserving protocol for finger print identification using finger codes. This protocol uses a multi-party computation approach and makes use of the homomorphic properties of the pallier cryptosystem. Feng et al. [57] used cancellable biometrics and proposed a three step hybrid approach for face template protection. All the biometric authentication mechanisms used are computationally very expensive because of the low computation power of mobile devices.

As mentioned in [16], the Dutch Company Moxomo uses Personal Identification Number (PIN) to confirm the purchase and the payment is directly debited to the merchant's mobile wallet. The wallet can be charged again to pay a merchant's bank account or pay to another mobile phone. Our application uses a direct debit solution for mobile payments. Though our transaction procedure uses a direct debit method, it still authenticates the user before a transaction. Our application incorporates the RSA scheme with a direct debit method, i.e., it uses the RSA scheme for user authentication before transferring the payment amount from the user's bank account to the merchant's bank account. The proposed biometric authentication mechanism makes use of cloud services to perform computations thereby reducing load on mobile devices. Moreover biometrics data are very sensitive, and we need to protect the privacy of the user. So we propose a privacy

preserving secure biometric authentication which is not only very fast compared to traditional password based authentication mechanisms, but is also more secure and protects the privacy of the user.

8.2. DESIGN CONSIDERATION

To formalize the problem statement, there is a cloud server S_I which maintains and keeps track of database a D_I. If there are n authorized registered users $< U_a, U_b, \ldots, U_n >$ then D_I consists of n number of entries, $< D_{U_a}, D_{U_b}, \ldots, D_{U_n} >$. Each entry D_{U_i} in D_I contains user $U_i's$ templates t. For secure mobile payments the application uses feature vector f_c of a user U_i where $i \in [1, n]$. We design a privacy preserving algorithm to decide whether a user's biometric feature vector belongs to at least one of the templates stored in the server database D_I. Our system checks whether

$$E < D_{U_k} > = E < D_{U_i} > \tag{16}$$

Here $E < D_{U_k} >$ is the encrypted template that belongs to server and $E < D_{U_i} >$ is the encrypted feature vector data that belongs to user i. Our algorithm decides whether there exists a user U_i and a biomteric template t stored in D_I such that the distance between f_c and t is below a threshold ϵ. We write this requirement as

$$\delta(f_c, t) < \epsilon \tag{17}$$

Suppose that the biometrics feature vector has m dimensions. For any feature vector f_c, and any $j \in [1, m]$, we use $f[j]$ to denote the j^{th} component of f_c. Hence equation (17) is equivalent to

$$\sum_{j=1}^{m} (f_c[j] - t_{i,k}[j])^2 < \epsilon^2 \tag{18}$$

8.3. SCHEMES

Preserving privacy is very important in a mobile application when it deals with mobile payments which include bank transactions. In our system, a registered user logs in to their application and pays for their purchases via the bank. To make the entire procedure secure we focus on

authenticated user login and secure bank transactions. For an authenticated user login we propose a new scheme named Privacy Preserving Biometric based Authentication Algorithm and for secure bank transactions we incorporate RSA encryption scheme on the direct debit method. In the following two subsections we discuss about these two schemes.

8.3.1. PRIVACY PRESERVING BIOMETRIC BASED AUTHENTICATION ALGORITHM

In this subsection we discuss details about the Privacy–preserving Biometric–based Authentication Algorithm. We design the Privacy–preserving Biometric–based Authentication Algorithm using Paillier encryption [4]. We use $E_y(\,)$ to denote Paillier encryption using public key y. That is, for any clear text m, $E_y(m)$ means encrypting m using public key y. Similarly, we use $D_x(\,)$ to denote Paillier decryption using private key x. We use the homomorphic property of Paillier encryption in our algorithm. For any clear texts m_1 and m_2, for any public key y, $E_y(m_1)E_y(m_2)$, is equal to an encryption of clear text $m_1 + m_2$ using public key y.

We use a user's biometric (e.g., face image, finger print, etc.) feature vector which is input to our algorithm. The server database stores the templates of all the users biometric feature vectors. The goal of the algorithm is to decide whether the user's biometric feature vector matches any biometric template stored in server, without revealing the user's feature vector and the templates stored in the server. The main idea is that we encrypt the user's feature vector using the Paillier scheme and send the cipher text to the server for matching. The details of our Privacy–preserving Biometric–based Authentication Algorithm are shown in Algorithm 8.1.

1. Mobile Application randomly generates a pair of keys (x, y) where x is the private key and y is the public key.
2. Generate some random numbers r and save them in a list R.
3. for each element r and each row of the feature vector matrix do
4. Go to the particular r^{th} element of the row and generate hash value H of that element.
5. if r^{th} and $(r+1)^{th}$element of H is equal to some predefined pattern (e.g."11") $then$

6. Save the location of that element in a list L
7. $end\ if$
8. $end\ for$
9. m is the length of L
10. Mobile application encrypts all the elements in L and sends it to the server
11. $for\ j = 1\ to\ m\ do$
12. Mobile application encrypts $f_c[j]$ using $y : F_c[j] = E_y(f_c[j])$
13. Mobile application encrypts the square of $f_c[j]$ using $y : G_c[j] = E_y((f_c[j])^2)$
14. $end\ for$
15. $for\ i = 1\ to\ n\ do$
16. Mobile application sends y, F_c, G_c to server.
17. $end\ for$
18. Server decrypts L
19. for each template $t_{i,k}$ where $(i,k) \in L$ in server database do
20. Server computes $SqDist = \prod_{j=1}^{m} \frac{E_y\left((t_{i,k})^2\right) G_c[j]}{(F_c[j])^{2t_{i,k}[j]}}$
21. $end\ for$
22. Server decrypts each $SqDist_{i,k}$ one by one
23. if any clear text obtained from the decryption is less than $\epsilon^2\ then$
24. Server accepts the user.
25. $else$
26. Server rejects the user.
27. $end\ if$
28. Server sends the message to the mobile application whether to accept or reject the user.

Algorithm 8.1: Privacy – Preserving Biometric – based Authentication Algorithm

First the Mobile application captures the user's feature vector and generates a public key–private key pair *(x,y)* to use in the Paillier encryption scheme, line 1 Algorithm 8.1.To make the entire procedure efficient in terms of time complexity we randomly sample the input data and save the location of these randomly selected data in a list *L*. Algorithm 8.1 line 2-8 do the random sampling. The main objective behind the random sampling on the input data is to make our system more efficient, light weight and reduce the time complexity for user log in without losing

correctness. L is of size $\leq m$ if the feature vector matrix is of size $m\ X\ n$. The mobile application encrypts each element $F_c[j]$ of L and its square $G_c[j]$ using the public key y and send y along with F_c and G_c to the server, line 10-17 Algorithm 8.1. The server encrypts the square of each template $t_{i,k}$ and compute $(F_c[j])^{2t_{i,k}[j]}$ and $SqDist_{i,k}$, line 19-21 Algorithm 8.1. Server also decrypts $SqDist_{i,k}$one by one. Any plain text less than the threshold value ϵ^2 indicates that the user is authentic. If all the decrypted squared distances are above the threshold ϵ^2; it implies that the user's feature vector does not match any template in the server and rejects the user.

8.3.2. SECURE TRANSACTION USING RSA ENCRYPTION SCHEME

For secure bank transactions we incorporate the RSA scheme on the direct debit method. In the RSA scheme if the public key and private key are (n, e) and (d, e) respectively, where $n = p \times q$ and p, q are prime numbers, then the cipher text c for a message m is

$$c = m^e (mod\ n) \tag{19}$$

The decryption equation to get the original message m from c is

$$m = c^d (mod\ n) \tag{20}$$

Here we assume that the public key-private key pair (n, e) and (d, e) to be used for the RSA scheme is known beforehand to the user bank and the user. The secure transaction procedure is shown in Fig. 29 below. The mobile application creates a connection to the user bank when the user proceeds check out. The mobile application sends the payment amount to the user bank as well. To authenticate the user the user bank sends an encrypted challenge (e.g., a personal question) using the public key and the RSA encryption scheme to the user. During a secure transaction we use the RSA scheme for each encryption and decryption. The user decrypts the challenge using the private key and encrypts the answer. The mobile application sends the encrypted answer to the user bank. The user bank checks if the decrypted answer is correct or not. If the answer is correct, then the

user bank will proceed with the direct debit payment procedure and will create a connection to the merchant bank. The user bank requests the merchant bank for merchant authorization. If the merchant is authorized then the user bank will execute the transaction from the user bank account to the merchant bank account.

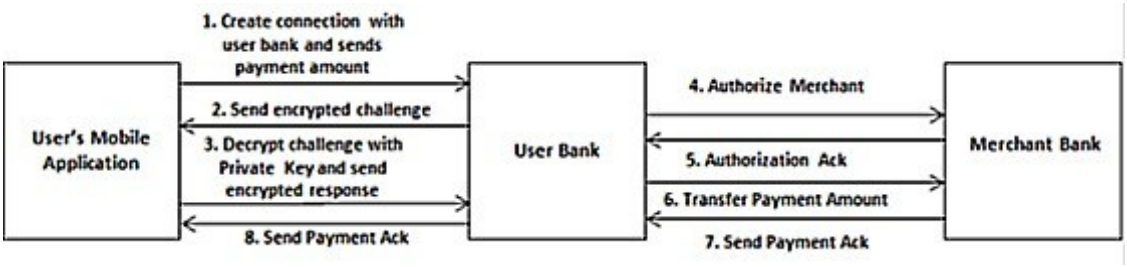

Figure 29: Secure Transaction using RSA encryption scheme

8.4. ANALYSIS

We analyze our Privacy–preserving Biometric–based Authentication Algorithm in terms of correctness, privacy and complexity. In the following subsections we discuss about the Correctness Analysis, Privacy Analysis and Complexity Analysis respectively.

8.4.1. CORRECTNESS ANALYSIS

We now show that our algorithm is correct. From the description above, we can see that the key point is to show that $D_x(SqDist_{i,k}) = \sum_{j=1}^{m}(f_c[j] - t_{i,k}[j])^2$, $\forall i,k$. Given the homomorphic property of Paillier scheme $\forall i,k$ we have

$$D_x(SqDist_{i,k})$$

$$= D_x\left(\prod_{j=1}^{m} \frac{E_y((t_{i,k}[j])^2)G_c[j]}{(F_c[j])^{2t_{i,k}[j]}}\right)$$

$$= D_x\left(\prod_{j=1}^{m} \frac{E_y((t_{i,k}[j])^2)G_c[j]}{(E_y(f_c[j])^{2t_{i,k}[j]})}\right)$$

$=\sum_{j=1}^{m}((t_{i,k}[j])^2 + (f_c[j])^2 - (f_c[j]^{2t_{i,k}[j]})$

$= \sum_{j=1}^{m}(f_c[j] - t_{i,k}[j])^2$

This proves the correctness of Algorithm 8.1.

8.4.2. PRIVACY ANALYSIS

In this subsection we explain that our algorithm is privacy preserving in the semi-honest model. In Algorithm 8.1 we can see that the mobile application encrypts the user biometric feature vector and sends it to the server. The server receives only $F_c[j]$ and $G_c[j]$ which are cipher texts. Therefore any adversary or the server cannot learn $f_c[j]$, i.e., the biometric feature vector but the cipher texts. Here we assume that the server is secure. The server calculates the $SqDist_{i,k}$ which is composed of multiplications and divisions of encrypted feature vectors. After calculating $SqDist_{i,k}$ the server sends the approval or disapproval of user authentication to the application which does not involve an exchange of the feature vector. Our algorithm is secure in the -honest model since no third party can learn anything about the original message.

8.4.3. COMPLEXITY ANALYSIS

We analyze the computation cost of our Algorithm 8.1. The operations on the mobile application include encryption of $f_c[j]$ and its square $\forall\, j \in [1, m]$. Therefore the computation cost at mobile application becomes $2m \times T_E$ where T_E is the encryption cost and m is the length of final selected sampled list L of feature vector elements to be encrypted. The cloud server computes $SqDist_{i,k}$ for each template $t_{i,k}$ where $i, k \in L$ and decrypts each $SqDist_{i,k}$ in the worst case. The server also decrypts all the elements in the final selected list L and this decryption takes $l \times T_D$. The computation of $SqDist_{i,k}$ requires m Paillier encryptions and $O(m)$ multiplications. The computation cost at the server becomes $l \times m \times T_E + l \times O(m) \times T_M + l \times T_D, where\ l$ is the

number of templates belonging to the final selected random sampled list and T_M is the cost of one multiplication.

8.5. IMPROVING OUR SYSTEM USING CLOUD

As the number of stores and users increase, there will be huge amounts of data for store details and users. In order to accommodate these increasing users and handle large amounts of data, it would be beneficial to incorporate cloud computing with our application. Cloud computing because of its characteristics such as on demand self-service, broad network access, resource pooling, rapid elasticity and measure service, would be a perfect fit for our application model. Since we don't know the exact increase or decrease in data and number of users, on demand self-service would be useful as the cloud can adjust to these situations without human intervention. Users can use various kinds of devices to access this service as cloud provides a broad network access. This can also accommodate multiple stores or companies to manage their data at the same time using resource pooling and also can manage their resources to scale upwards or downwards using rapid elasticity. Since cloud is a measured service the stores or the companies participating will pay only for the services they used. Because of the cloud's advantages, the authors in [6] have specified a identity based encryption (IBE) and biometric scheme for secure data access in cloud. The biometric scheme was used here to authenticate the user and IBE scheme was used to encrypt and decrypt the data. One of the major benefits of using the cloud would be to execute the privacy preserving authentication algorithm more efficiently using computation resources available in the cloud. With a large number of users, an authentication mechanism using a single server will result in traffic bottlenecks and there will be delays in scheduling them. Using the cloud we can avoid this scenario by efficiently scheduling the jobs on multiple instances of cloud.

8.6. EXPERIMENTAL ANALYSIS

8.6.1. EXPERIMENTAL SETUP

We have designed and developed an Android mobile application for secure mobile payments. To implement this application we used the Java programming language on computers with 3.33 GHz Intel Core i5 processor, 4 GB RAM, 64 bit operating system. To test our application we used Google Nexus 7 with Android version 4.2.1. We implemented our system by developing an Android application using Android SDK and Android Developer Tools (ADT). The main system components are connected to each other by the Internet. The communication between the server and the application is realized by Java Platform, Enterprise Edition (Java EE) and Java EE APIs. We use the javax.servlet package and HTTP requests which includes Java Server Pages (JSP) specification. To implement the communication between two banks and with the user we use javax.servlet packages. MySQL was used as the database. For extracting the feature vector from user biometrics we used the PCA-based Face Recognition System package in Matlab [60]. This package implements a PCA-based face recognition method which considers a face image as an input and checks whether the input matches with any face image entry from the database or not. To implement the method this package generates the feature vector matrix for the input face image and applies further implementation steps on that. Our algorithm requires only the feature vector matrix as input and therefore we use the intermediate result, i.e., feature vector matrix for our algorithm and do not need to implement the entire package. To test the efficiency of our system we use real world data, i.e., original face image as input to the application.

8.6.2. METRICS

To test the scalability of our system we check the time the mobile application takes in its different modules. Since the RSA scheme has already been extensively studied in terms of efficiency we focus on the efficiency of our Privacy - Preserving Biometric–based Authentication Algorithm. We

use our algorithm for secure log in and hence check the time efficiency for the log in module only. The entire log in time consists of the encryption time and the computation time in the cloud server.Using the Privacy-Preserving Biometric–based Authentication Algorithm we test time efficiency for encryption, computation in cloud server and log in procedure.

8.6.3. RESULTS

We test the scalability of our application in terms of time efficiency. To test the efficiency of our system we consider real-world datasets. Since the input to our mobile application is a face image we considered a set of real face images as the input. To test the time efficiency we vary the size of the input, i.e., the size of the face image. In Algorithm 8.1 instead of doing computation on all of the elements in the user's feature vector matrix we sample it randomly (as mentioned in Algorithm 8.1 line no. 2 - 8) and this makes Algorithm 8.1 efficient in terms of time without losing its correctness. To show efficiency we first check how long the application takes if the application does the computation on all the elements of the feature vector matrix and we name this approach as the *primitive* approach of our experiment. After that we compare this result with our two efficiency enhancement efforts, namely, using multi-threading and then random sampling. We label our efficiency enhancement efforts for algorithm 8.1 as *efficiency enhanced approach using multi-threading* and *efficiency enhanced approach using random sampling* in this chapter. As mentioned in the previous subsection we test the efficiency for encryption, computation in the cloud server and log in procedure. Therefore we test the efficiency of encryption time at the end user, computation time in the cloud server, elapsed time for the log in procedure using the primitive approach, multithreading approach and random sampling approach in the next three sub-subsections.

8.6.3.1. ENCRYPTION TIME AT END-USER

Fig. 30 shows the encryption time of Algorithm 8.1, i.e., results from the primitive approach, efficiency enhanced approach using multithreading, and efficiency enhanced approach using random sampling with variable sizes of data. Here size of data/image varies at 3KB, 4KB, 5KB, 6KB and 7KB. For the maximum size of data we tested, i.e., 7KB the primitive approach, efficiency enhanced approach using multithreading, efficiency enhanced approach using random sampling took 3.7, 3.01 and 0.02 seconds respectively.

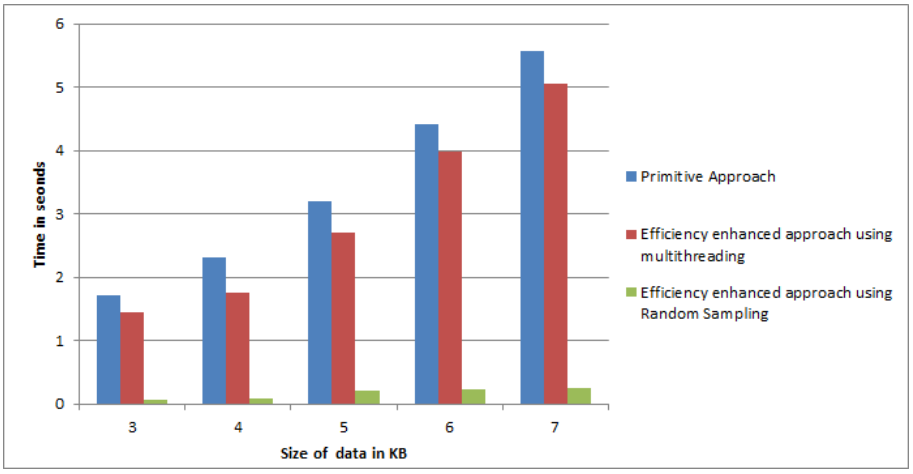

Figure 30: Encryption time at end – user

8.6.3.2 COMPUTATION TIME IN SERVER

Fig. 31 shows the computation time in the server for the three approaches, that is, the primitive approach and efficiency enhanced approaches. Here computation time of the server includes the Square distance calculation time in the server, checking user authentication and sending the response back to the mobile application. To test the time efficiency of computation in the server we vary the size of the input data at 3KB, 4KB, 5KB, 6KB and 7KB. For the maximum size of data we tested, i.e., 7KB the primitive approach, efficiency enhanced approach using multithreading,

efficiency enhanced approach using random sampling took 1.97, 1.83 and 0.19 seconds respectively.

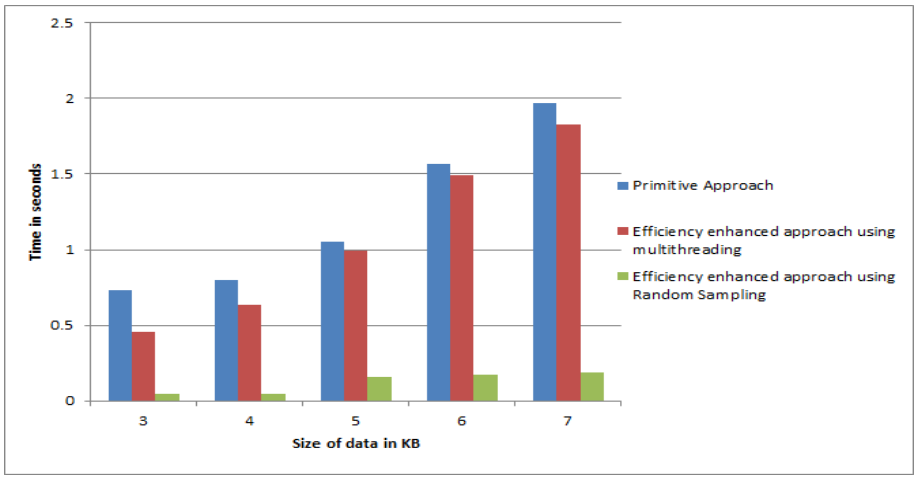

Figure 31: Computation time in Server

8.6.3.3. ELAPSE TIME IN LOG IN PROCEDURE

Fig. 32 shows the elapsed time for Log in to our mobile application. Here Log in time mainly depends on the encryption time and computation time in the cloud server. To test the time efficiency for Log in to the mobile application we vary the size of the input data at 3KB, 4KB, 5KB, 6KB and 7KB. For the maximum size of data we tested, i.e., 7KB the primitive approach, efficiency enhanced approach using multithreading, efficiency enhanced approach using random sampling took 5.58, 5.05 and 0.26 seconds respectively.

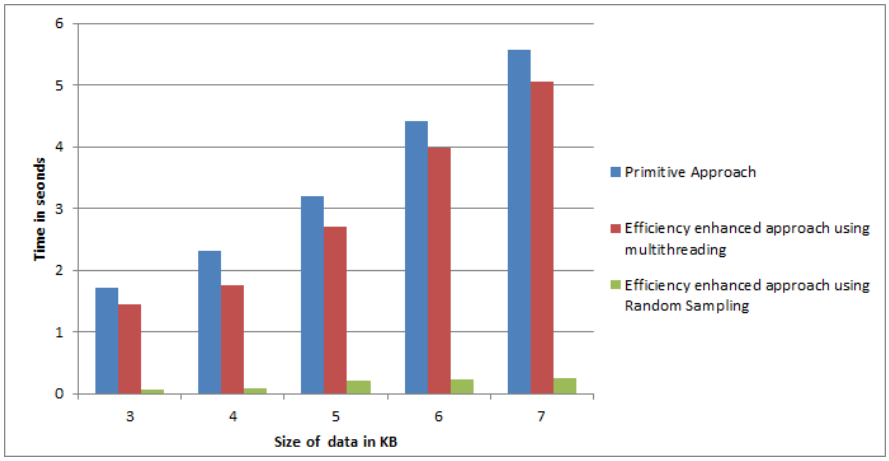

Figure 32: Elapse time in Log In Procedure

8.7. CONCLUSIONS

In this chapter, we have designed and developed a mobile application for secure mobile payments. To make this application secure we focus on secure user login to the mobile application and secure transaction from the bank. For secure login we use biometrics and have proposed a new algorithm named Privacy preserving Biometric based Authentication Algorithm. For secure transaction we used RSA encryption scheme on direct debit method. We analyzed our novel Privacy preserving Biometric based Authentication Algorithm in terms of time. Further we tested the efficiency of our algorithm on real world data sets and achieved expected results. The rigorous experiments on different approaches of our algorithm reflect the efficiency of our algorithm in a real world scenario.

CHAPTER IX

CONCLUSION AND FUTURE WORK

9.1. CONCLUSION

Data can contain sensitive information about its entities and when sharing with others it can be a threat to privacy. With the rapid growth of online world, electronic data, cyberspaces data become vulnerable to privacy and security threats. Sharing data in a privacy preserving way has become a real challenge as it involves sharing data among two to multiple number of parties without revealing confidential information. Data privacy is important in a wide range of industries such as, healthcare, aviation industry, education system, federal law enforcement agencies, retail, financial services, etc. In this thesis we propose privacy preserving algorithms for emergent computing environments, namely, health care, data mining, biometrics, mobile payments, and distributed cloud computing environment.

Linking medical records across different medical service providers is important to the enhancement of health care quality and public health surveillance. In records linkage, protecting patients' privacy is a primary requirement. In real-world medical records may contain errors such as typos. Linking the error-prone data and preserving data privacy at the same time are very difficult. Existing privacy preserving solutions for this problem are only restricted to textual data. To enable different medical service providers to link their error-prone data in a private way, we provided a holistic solution by designing and developing a medical record linkage system for medical service providers. Identifying frequent patterns in medical records is important as it helps to find disease pattern,

patient path, effective treatment, etc. Extracting sequential patterns collaboratively and privately on the medical data sets held across the multiple medical sites is a challenging task. In this thesis we proposed a privacy preserving sequential pattern mining algorithm across multiple medical sites.

In the distributed cloud storage and computation on resources are provided to users in a distributed peer-to-peer fashion. Data protection is a concern in the distributed cloud as resources are provided by the other users. Confidentiality, integrity and availability are three major security concerns' along with privacy when resources are provided by others. Data confidentiality and integrity can be achieved using encryption mechanisms. To preserve security and privacy of data in a distributed cloud, we proposed a privacy preserving algorithm using a cryptography technique. Managing secret key is difficult in the distributed cloud. To provide security of the secret key we proposed a multilevel threshold secret sharing scheme in the distributed cloud.

Biometric authentication ensures user identity by means of users' biometric traits. Multimodal biometric authentication uses more than one biometric trait of a user. Though biometrics are unique and secure, it can be still stolen or misused by any adversary. Protecting biometrics is important as it is a unique trait of any user. In our work we provide a secure and privacy preserving biometric authentication scheme using watermarking technique. Mobile payments needs privacy and security as it involves bank transaction. We proposed a privacy preserving biometric based authentication algorithm for secure mobile payments.

9.2. FUTURE WORK

Our future work will include proposing privacy preserving techniques in different computing environments such as big data, distributed cloud environment, social networking sites, wireless sensor networks etc.

Big data refers to the collection of massive electronic data collected by different organizations. The mainstream definition of big data is defined by high volume, velocity and variety. Different

software and platforms, large scale of cloud infrastructures, large computer networks are required to work with these data which increases privacy and security threats to these huge collection of data. Big data is collected by different organizations across different domains, such as, health care, social networking etc. As the data contains sensitive, private data, privacy becomes a serious issue for big data. Several issues of privacy in big data are secure data storage and transaction logs, real time security monitoring, privacy preserving data mining and analysis, etc.

Cloud computing delivers computing as a service. It is the set of resources and services offered through the internet and delivered from data centers located throughout the world. It is easy to manipulate data in the cloud and easy to lose control of the data at the same time. As privacy is a concern in the online world and storing private data somewhere in the cyberspace leads to major threats to privacy of the sensitive data. Vulnerability of the security/privacy of data leads to the need for privacy preserving techniques in the cloud computing field.

Social networking sites such as, Facebook, Google+, LinkedIn, Instagram, Twitter, etc. play significant role in our everyday life. These networking sites contain a lot of information, personal as well as professional. As these online social networking sites become an integral part of our life there is a lot of opportunity for the information available on these sites to be exposed. Risks for social network users and the information shared on these networks are identity theft, advertising harassment/spam, unauthorized physical and network access, misuse of computer, sensitive information or social profile, etc. Considering the various threats and security issues in social networking sites, preserving privacy is a real need in this field.

Wireless sensor networks are used to monitor physical and environmental conditions such as temperature, sound, vibration, pressure etc. Security is a challenge in wireless sensor networks because of sensor node sizes, density of networks, physical attacks to unattended sensors, attacks on the security mechanism or routing mechanisms, attacks on information transmitted over the network etc. Keeping into account of all the security vulnerabilities privacy preserving techniques are needed in wireless sensor networks.

Besides all the above mentioned computing fields, privacy threats are real concern in various other computing fields too, such as, Internet, software defined networks, bioinformatics, password management, computer networks, etc.

REFERENCES

1. U.S. Department of Health and Human Services. The privacy rule URL: http://www.hhs.gov/ocr/privacy/hipaa/administrative/privacyrule/. [November 25, 2015]
2. The Family Educational Rights and Privacy Act (FERPA), http://www2.ed.gov/policy/gen/guid/fpco/ferpa/index.html. [November 25, 2015]
3. U.S. Department of Homeland Security, http://www.dhs.gov/. [November 25, 2015]
4. Inan, A., Kantarcioglu, M., Bertino, E., & Scannapieco, M, "A hybrid approach to private record linkage", in Proceedings 24th IEEE International Conference on Data Engineering, ICDE 2008, pp. 496-505, 2008.
5. Hjaltason, Gísli R., and Hanan Samet, "Properties of embedding methods for similarity searching in metric spaces.", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, No. 5, pp 530-549, 2003.
6. Varshney, Upkar, "Mobile payments", Computer 35, No. 12, pp 120-121, 2002.
7. Sadeh, Norman, "M-commerce: technologies, services, and business models", John Wiley & Sons, 2003.
8. Buyya, Rajkumar, Chee Shin Yeo, and Srikumar Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities", in Proceedings 10th IEEE International Conference on High Performance Computing and Communications, HPCC'08, pp. 5-13., 2008.
9. Mell, Peter, and Tim Grance, "The NIST definition of cloud computing", National Institute of Standards and Technology 53.6, 50, 2009.

10. Jain, Anil K., Arun Ross, and Sharath Pankanti, "Biometrics: a tool for information security", IEEE Transactions on Information Forensics and Security, Vol. 1.No. 2, pp 125-143, 2006.
11. Clarke, Nathan L., and Steven M. Furnell, "Authentication of users on mobile telephones–A survey of attitudes and practices", Computers & Security, Vol 24, No. 7, pp 519-527, 2005.
12. Wayman, James, Anil Jain, Davide Maltoni, and Dario Maio, "An introduction to biometric authentication systems.", Springer London, 2005.
13. Prabhakar, Salil, Sharath Pankanti, and Anil K. Jain, "Biometric recognition: Security and privacy concerns.", IEEE Security & Privacy 2, pp 33-42, 2003.
14. Zheng, Xiaolin, and Deren Chen, "Study of mobile payments system.", in Proceedings IEEE International Conference on E-Commerce, CEC 2003,. IEEE, 2003.
15. Rivest, Ronald L., Adi Shamir, and Len Adleman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM Vol. 21, No. 2, pp 120-126, 1978.
16. Mallat, Niina, Matti Rossi, and Virpi Kristiina Tuunainen, "Mobile banking services." Communications of the ACM Vol. 47. No. 5, pp 42-46, 2004.
17. Elgamal, T, "A public key cryptosystem and a signature scheme based on discrete logarithms." IEEE Transactions on Information Theory, Vol. 31, No. 4, pp 469-472, 1985.
18. Paillier, Pascal, "Public-key cryptosystems based on composite degree residuosity classes", Advances in cryptology, EUROCRYPT99, Springer Berlin Heidelberg, 1999.
19. Gentry, Craig, "Fully homomorphic encryption using ideal lattices", In STOC, Vol. 9, pp. 169-178. 2009.
20. Yao, Andrew Chi-Chih, "How to generate and exchange secrets", Foundations of Computer Science, IEEE 27th Annual Symposium, 1986.
21. Agrawal, Rakesh, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu, "Order preserving encryption for numeric data", in Proceedings 2004 ACM SIGMOD international conference on Management of data, pp. 563-574. ACM, 2004.

22. Boldyreva, Alexandra, Nathan Chenette, Younho Lee, and Adam O'neill, "Order-preserving symmetric encryption", in Advances in Cryptology-EUROCRYPT 2009, pp. 224-241, Springer Berlin Heidelberg, 2009.
23. Binnig, Carsten, Stefan Hildenbrand, and Franz Färber, "Dictionary-based order-preserving string compression for main memory column stores", in Proceedings 2009 ACM SIGMOD International Conference on Management of data, ACM, 2009.
24. Antoshenkov, Gennady, "Dictionary-based order-preserving string compression", The VLDB Journal—The International Journal on Very Large Data Bases, Vol. 6, No.1, pp 26-39, 1997.
25. Wang, Cong, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou, "Secure ranked keyword search over encrypted cloud data", in Proceedings 30th IEEE International Conference on Distributed Computing Systems (ICDCS), pp. 253-262. IEEE, 2010.
26. Goldreich, Oded, and Yair Oren, "Definitions and properties of zero-knowledge proof systems." Journal of Cryptology, Vol 7, No. 1, pp 1-32, 1994.
27. Rackoff, Charles, and Daniel R. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack", Advances in Cryptology—CRYPTO'91, Springer Berlin Heidelberg, 1992.
28. Bellare, Mihir, Ran Canetti, and Hugo Krawczyk, "Keying hash functions for message authentication", In Advances in Cryptology—CRYPTO'96, pp. 1-15, Springer Berlin Heidelberg, 1996.
29. Quantin, Catherine, Hocine Bouzelat, F. A. A. Allaert, Anne-Marie Benhamiche, Jean Faivre, and Liliane Dusserre, "How to ensure data security of an epidemiological follow-up: quality assessment of an anonymous record linkage procedure", International Journal of Medical Informatics, Vol 49, No. 1, pp 117-122, 1998.
30. Alhaqbani, Bandar, and Colin Fidge, "Privacy-preserving electronic health record linkage using pseudonym identifiers", in Proceedings 10th IEEE International Conference e-health Networking, Applications and Services, HealthCom 2008, pp. 108-117, 2008.

31. Lindell, Yehuda, and Benny Pinkas, "Privacy preserving data mining." In Advances in Cryptology—CRYPTO 2000, pp. 36-54, Springer Berlin Heidelberg, 2000.
32. Agrawal, Rakesh, and Ramakrishnan Srikant, "Privacy-preserving data mining", In ACM Sigmod Record, Vol. 29, No. 2, pp. 439-450, 2000.
33. Churches, Tim, and Peter Christen, "Some methods for blindfolded record linkage", BMC Medical Informatics and Decision Making Vol. 4, No. 19, 2004
34. Goldreich, Oded, “Foundations of cryptography: volume 2, basic applications”, Cambridge university press, 2004.
35. Mohammed, Noman, Xiaoqian Jiang, Rui Chen, Benjamin CM Fung, and Lucila Ohno-Machado, "Privacy-preserving heterogeneous health data sharing", Journal of the American Medical Informatics Association, Vol 20, No. 3, pp 462-469, 2013.
36. Wang, Jixian, and Peter T. Donnan, "Adjusting for missing record linkage in outcome studies", Journal of Applied Statistics, Vol 29, Issue 6, pp 873-884, 2006.
37. Quantin, Catherine, Hocine Bouzelat, and Liliane Dusserre, "A computerized record hash coding and linkage procedure to warrant epidemiological follow-up data security", Studies in Health Technology and Informatics, Vol 43, pp 339-342, 1996.
38. Bachteler, Tobias, Rainer Schnell, and Jörg Reiher, "An empirical comparison of approaches to approximate string matching in private record linkage", in Proceedings of Statistics Canada Symposium, pp. 290-295, 2010.
39. Schnell, Rainer, Tobias Bachteler, and Jörg Reiher, "Privacy-preserving record linkage using Bloom filters", BMC Medical Informatics and Decision Making, Vol 9, No.1: 41.2009
40. Zhang, Weizhe, Xuehui Wang, Bo Lu, and Tai-hoon Kim, "Secure encapsulation and publication of biological services in the cloud computing environment", BioMed research international, 2013.
41. Gkoulalas-Divanis, Aris, and Grigorios Loukides, "Privacy-Preserving Medical Data Sharing", SIAM Data Mining. 2012.

42. Kum, Hye-Chung, Ashok Krishnamurthy, Ashwin Machanavajjhala, Michael K. Reiter, and Stanley Ahalt, "Privacy preserving interactive record linkage (PPIRL)", Journal of the American Medical Informatics Association, Vol 21, No. 2, pp 212-220, 2014.
43. Mohammed, Noman, et al., "Privacy-preserving heterogeneous health data sharing", Journal of the American Medical Informatics Association, Vol 20, No. 3, pp 462-469, 2013.
44. Kungpisdan, Supakorn, Bala Srinivasan, and Phu Dung Le, ”A secure account-based mobile payment protocol.”, in Proceedings IEEE International Conference on Information Technology: Coding and Computing, ITCC 2004. Vol. 1. 2004.
45. Fourati, Alia, Ben Ayed, Hella Kaffel, Farouk Kamoun, and Abdelmalek Benzekri, "A SET based approach to secure the payment in mobile commerce." In Proceedings 27th Annual IEEE Conference on Local Computer Networks, LCN 2002, pp. 136-137, 2002.
46. Tiwari, Ayu, Sudip Sanyal, Ajith Abraham, Svein Johan Knapskog, and Sugata Sanyal, "A multi-factor security protocol for wireless payment-secure web authentication using mobile devices", arXiv preprint arXiv:1111.3010, 2011
47. Harb, Hany, Hassan Farahat, and Mohamed Ezz, "SecureSMSPay: secure SMS mobile payment model", in Proceedings 2nd International IEEE Conference on Anti-counterfeiting, Security and Identification,. ASID 2008, pp. 11-17, 2008.
48. Pourghomi, Pardis, and Gheorghita Ghinea, "Managing NFC payment applications through cloud computing", in Proceedings IEEE International Conference on Internet Technology And Secured Transactions, pp. 772-777, 2012.
49. Scannapieco M., “Privacy preserving schema and data matching”, in Proceedings ACM SIGMOD International Conference on Management of Data, 2007.
50. Zhang W, Wang X, Lu B, Kim TH, “Secure encapsulation and publication of biological services in the cloud computing environment”, BioMed Research International, 2013.

51. Fung, Benjamin, Ke Wang, Rui Chen, and Philip S. Yu, "Privacy-preserving data publishing: A survey of recent developments", ACM Computing Surveys (CSUR), Vol 42, no. 4: 14. 2010
52. Hall, Rob, and Stephen E. Fienberg, "Privacy-preserving record linkage", In Privacy in statistical databases, pp. 269-283, Springer Berlin Heidelberg, 2010.
53. Gordon, Michael, and Suresh Sankaranarayanan, ”Biometric security mechanism in Mobile paymentts”, in Proceedings 7th IEEE International Conference on. Wireless and Optical Communications Networks (WOCN), 2010.
54. Cheng, Hongbing, Chunming Rong, Zhenghua Tan, and Qingkai Zeng, "Identity based encryption and biometric authentication scheme for secure data access in cloud computing", Chinese Journal of Electronics, Vol 21, no. 2, pp 254-259, 2012.
55. Ross, Arun, and Asem Othman, “Visual cryptography for biometric privacy”, IEEE transactions on information forensics and security, Vol 6, Issue 1, pp 70-81, 2011.
56. Barni, Mauro, Tiziano Bianchi, Dario Catalano, Mario Di Raimondo, Ruggero Donida Labati, Pierluigi Failla, Dario Fiore et al., "Privacy-preserving fingercode authentication", in Proceedings 12th ACM workshop on Multimedia and security, pp. 231-240, 2010.
57. Feng, Y. C., Pong C. Yuen, and Anil K. Jain, "A hybrid approach for face template protection", In SPIE Defense and Security Symposium, pp. 694408-694408, International Society for Optics and Photonics, 2008.
58. Laukkanen, Tommi, and Jari Lauronen, "Consumer value creation in mobile banking services", International Journal of Mobile Communications, Vol 3, no. 4, pp 325-338, 2005.
59. UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/datasets.html. [November 25, 2015]
60. PCA-based Face Recognition System, http://www.mathworks.com/matlabcentral/fileexchange/ 17032-pcabased-face-recognition – system. [November 25, 2015]

61. Gupta, Manish, and Jiawei Han, "Applications of pattern discovery using sequential data mining", Pattern Discovery Using Sequence Data Mining: Applications and Studies, pp 1-23, 2011.
62. Barham, Paul, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield, "Xen and the art of virtualization", ACM SIGOPS Operating Systems Review, Vol 37, no. 5, pp 164-177, 2003.
63. Riha, Zdenek, "Toward reliable user authentication through biometrics", IEEE Security & Privacy (3), pp 45–49, 2003.
64. Jain, Anil K., Arun Ross, and Salil Prabhakar, "An introduction to biometric recognition", IEEE Transactions on Circuits and Systems for Video Technology, Vol 14, no. 1, pp 4-20, 2004.
65. Katzenbeisser, Stefan, "On the integration of watermarks and cryptography", Digital Watermarking, Springer Berlin Heidelberg, pp. 50-60, 2004.
66. Erkin, Zekeriya, Martin Franz, Jorge Guajardo, Stefan Katzenbeisser, Inald Lagendijk, and Tomas Toft, "Privacy-preserving face recognition", Privacy Enhancing Technologies, Springer Berlin Heidelberg, pp. 235-253, 2009.
67. Haghighat, Mohammad, Saman Zonouz, and Mohamed Abdel-Mottaleb, "CloudID: Trustworthy cloud-based and cross-enterprise biometric identification", Expert Systems with Applications, Vol 42, No. 21, pp 7905-7916, 2015.
68. Snelick, Robert, et al., "Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 27, No.3, pp 450-455, 2005.
69. Wang, Fenghua, and Jiuqiang Han, "Robust multimodal biometric authentication integrating iris, face and palmprint", Information Technology and Control, Vol 37, No. 4, 2015.

70. Alsolami, Fahad, and Terrance E. Boult, "CloudStash: Using Secret-Sharing Scheme to Secure Data, Not Keys, in Multi-clouds", in Proceedings 11th IEEE International Conference on Information Technology: New Generations (ITNG), pp. 315-320, 2014.
71. Cachin, Christian, Robert Haas, and Marko Vukolic, "Dependable storage in the intercloud", Research report RZ 3783, pp 1-6, 2010.
72. Alsolami, Fahad, and C. Edward Chow, "N-Cloud: Improving performance and security in cloud storage", in Proceedings 14th IEEE International Conference on High Performance Switching and Routing (HPSR), pp. 221-222, 2013.
73. Bessani, Alysson, Miguel Correia, Bruno Quaresma, Fernando André, and Paulo Sousa, "DepSky: dependable and secure storage in a cloud-of-clouds", ACM Transactions on Storage (TOS), Vol 9, no. 4, Article No. 12, 2013.
74. Xiong, H., Zhang, X., Zhu, W., Yao, D., "CloudSeal: end-to-end content protection in cloud-based storage and delivery services", In: Rajarajan, M., Piper, F., Wang, H., Kesidis, G. (eds.) SecureComm. LNICST, Springer, Heidelberg, Vol. 96, pp. 491–500, 2012.
75. Agrawal, Rakesh, and Ramakrishnan Srikant, "Mining sequential patterns", in Proceedings 11th IEEE International Conference Data Engineering,, pp. 3-14, 1995.
76. Srikant, Ramakrishnan, and Rakesh Agrawal, "Mining sequential patterns: Generalizations and performance improvements", Springer Berlin Heidelberg, 1996.
77. Pei, Jian, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu, "Mining sequential patterns by pattern-growth: The prefixspan approach." IEEE Transactions on Knowledge and Data Engineering, Vol 16, No.11, pp 1424-1440, 2004.
78. Ayres, Jay, Jason Flannick, Johannes Gehrke, and Tomi Yiu, "Sequential pattern mining using a bitmap representation.", In Proceedings 8th ACM SIGKDD international conference on Knowledge Discovery and Data Mining, pp. 429-435, 2002.

79. Zaki, Mohammed Javeed, "SPADE: An efficient algorithm for mining frequent sequences.", Machine Learning, Vol 42, No. 1-2, pp 31-60, 2001.
80. Zaki, Mohammed Javeed, "Fast mining of sequential patterns in very large databases." Rapport Technique, University of Rochester Computer Science Department, New York, 1997.
81. Seno, Masakazu, and George Karypis, "Slpminer: An algorithm for finding frequent sequential patterns using length-decreasing support constraint.", in Proceedings 2002 IEEE International Conference on Data Mining, ICDM 2002., pp. 418-425, 2002.
82. Clifton, Chris, and Don Marks, "Security and privacy implications of data mining", In Proceedings ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, pp. 15-19, 1996.
83. Verykios, Vassilios S., Elisa Bertino, Igor Nai Fovino, Loredana Parasiliti Provenza, Yucel Saygin, and Yannis Theodoridis, "State-of-the-art in privacy preserving data mining", ACM Sigmod Record 33, no. 1, pp 50-57, 2004.
84. Kapoor, Vishal, Pascal Poncelet, François Trousset, and Maguelonne Teisseire, "Privacy preserving sequential pattern mining in distributed databases", in Proceedings 15th ACM international Conference on Information and Knowledge Management, pp. 758-767, 2006.
85. Ouyang, Weimin, and Qinhua Huang, "Privacy preserving sequential pattern mining based on secure multi-party computation", in Proceedings IEEE International Conference on Information Acquisition, pp 149-154, 2006.
86. Gorawski, Marcin, and Pawel Jureczek, "An Efficient Algorithm for Sequential Pattern Mining with Privacy Preservation", Advances in Systems Science, Springer International Publishing, pp 151-161, 2014.
87. Zhan, Justin Z., LiWu Chang, and Stan Matwin, “Privacy-preserving collaborative sequential pattern mining”, Ottawa Univ (Ontario) School Of Information Technology, 2004.

88. Agrawal, R., V. Crestana, and V. Parasad, “A Tree Projection Algorithm for Generation of Large Itemsets for Association Rules”, New York: IBM, 1998.
89. Guralnik, Valerie, Nivea Garg, and George Karypis, “Parallel tree projection algorithm for sequence mining”, Springer Berlin Heidelberg, pp 310-320, 2001.
90. Frequent Itemset Mining Dataset Repository, http://fimi.ua.ac.be/data/. [November, 2015]
91. Yang, Zhenglu, Yitong Wang, and Masaru Kitsuregawa, "LAPIN: effective sequential pattern mining algorithms by last position induction for dense databases", Advances in Databases: Concepts, Systems and Applications, Springer Berlin Heidelberg, pp 1020-1023, 2007.
92. Han, Jiawei, Jian Pei, Behzad Mortazavi-Asl, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu, "FreeSpan: frequent pattern-projected sequential pattern mining", In Proceedings sixth ACM SIGKDD international conference on Knowledge Discovery and Data Mining, pp. 355-359, 2000.
93. Mooney, Carl H., and John F. Roddick, "Sequential pattern mining--approaches and algorithms", ACM Computing Surveys (CSUR), Vol 45, No. 2: 19, 2013
94. Bonomi, Luca, and Li Xiong, "Mining frequent patterns with differential privacy", Proceedings of the VLDB Endowment, Vol 6, No. 12, pp 1422-1427 , 2013.
95. Khethavath, Praveen, Johnson Thomas, Eric Chan-Tin, and Hong Liu, "Introducing a Distributed Cloud Architecture with Efficient Resource Discovery and Optimal Resource Allocation", in Proceedings IEEE Ninth World Congress on Services (SERVICES), pp. 386-392, 2013.
96. Anderson, David P., "Boinc: A system for public-resource computing and storage", in Proceedings Fifth IEEE/ACM International Workshop on Grid Computing, pp. 4-10, 2004.
97. Anderson, David P., Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer, "SETI@ home: an experiment in public-resource computing", Communications of the ACM 45, no. 11, pp 56-61, 2002.

98. Chun, Brent, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman, "Planetlab: an overlay testbed for broad-coverage services", ACM SIGCOMM Computer Communication Review 33, no. 3, pp 3-12, 2003.

99. Shamir, Adi, “How to share a secret”, Communications of the ACM 22, no. 11, pp 612-613, 1979.

100. Blakley, George Robert, "Safeguarding cryptographic keys", In Proceedings of the National Computer Conference, Vol. 48, pp. 313-317, 1979.

101. Beimel, A., Ben-Efraim, A., Padr´o, C., and Tyomkin, I, “Multi-linear secret-sharing schemes” In: Lindell, Y. (ed.) TCC 2014, LNCS, Vol. 8349, pp. 394–418, Springer, Heidelberg (2014).

102. Ito, Mitsuru, Akira Saito, and Takao Nishizeki, "Secret sharing scheme realizing general access structure", Electronics and Communications in Japan (Part III: Fundamental Electronic Science) 72, no. 9, pp 56-64, 1989.

103. Asmuth, Charles, and John Bloom, "A modular approach to key safeguarding", IEEE transactions on Information Theory 29, no. 2, pp 208-210, 1983.

104. Beimel, Amos, "Secret-sharing schemes: a survey", In Coding and cryptology, pp. 11-46, Springer Berlin Heidelberg, 2011.

105. Ding, C., “Chinese Remainder Theorem”, World Scientific, Singapore, 1996.

106. Amazon AWS, http://aws.amazon.com/. [July, 2014]

107. Amazon EC2, http://aws.amazon.com/ec2/. [July, 2014]

108. Microsoft Azure, http://azure.microsoft.com. [July, 2014]

109. Kurihara, Jun, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka, "A new (k, n)-threshold secret sharing scheme and its extension", In Information Security, pp. 455-470, Springer Berlin Heidelberg, 2008.

110. Lin, Changlu, Lein Harn, and Dingfeng Ye, "Ideal perfect multilevel threshold secret sharing scheme", in Proceedings 5th International IEEE Conference on Information Assurance and Security, IAS'09, Vol. 2, pp. 118-121, 2009.

111. Tassa, Tamir, "Hierarchical threshold secret sharing", Journal of Cryptology, Vol 20, No. 2, pp 237-264, 2007.

112. Endo, Patricia Takako, Andre Vitor de Almeida Palhares, Nadilma Nunes Pereira, Glauco Estacio Goncalves, Djamel Sadok, Judith Kelner, Bob Melander, and Jan-Erik Mångs, "Resource allocation for distributed cloud: concepts and research challenges", IEEE Network, Vol. 25, No. 4, pp 42-46, 2011.

113. Grozev, Nikolay, and Rajkumar Buyya, "Inter-Cloud architectures and application brokering: taxonomy and survey", Software: Practice and Experience, Vol 44, No. 3, pp 369-390, 2014

114. Maymounkov, Petar, and David Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric", In Peer-to-Peer Systems, pp. 53-65, Springer Berlin Heidelberg, 2002.

115. Michael, A., Armando, F., Rean, G., Anthony, J., Randy, K., et al. (2009), "Above the Clouds: A Berkeley View of Cloud Computing", Technical Report EECS-2009-28, UC Berkeley.

116. Babaoglu, Ozalp, Moreno Marzolla, and Michele Tamburini, "Design and implementation of a P2P Cloud system.", In Proceedings 27th Annual ACM Symposium on Applied Computing, pp. 412-417, 2012.

117. Liu, Q., Wang, G.-J., & Wu, J., "An efficient privacy preserving keyword search scheme in cloud computing", Proceedings International Conference on Computational Science and Engineering: Vol. 2, pp. 715-720, 2009.

118. Maha, T., & Said, E. H., "Secure cloud computing through homomorphic encryption", International Journal of Advancements in Computing Technology, Vol 5, No. 16, 2013.

119. Li, J., Chen, S.-C., Song, D.-J, “Security structure of cloud storage based on homomorphic encryption scheme”, Proceedings 2nd IEEE International Conference on Cloud Computing and Intelligent, Vol. 1, pp. 224-227, 2012.

120. Praveen, K., Johnson, T., & Hong, L., “Game theoretic approach to resource provisioning in a distributed cloud”, Proceedings of International Conference on Data Science & Engineering, 2014.

121. Cong ,W., Qian, W., Kui, R., & Lou, W.-J, “Privacy-preserving public auditing for data storage security in cloud computing”, Proceedings of INFOCOM, pp. 1-9, 2010.

122. Pal, Doyel, Tingting Chen, Sheng Zhong, and Praveen Khethavath, "Designing an Algorithm to Preserve Privacy for Medical Record Linkage With Error-Prone Data.", JMIR medical informatics Vol. 2, No. 1, 2014.

123. Huang, Xinyi, Yang Xiang, Ashley Chonka, Jianying Zhou, and Robert H. Deng, “A generic framework for three-factor authentication: preserving security and privacy in distributed systems”, IEEE Transactions on Parallel and Distributed Systems, Vol. 22, No. 8, pp 1390-1397, 2011.

124. Tuyls, P., Akkermans, A.H., Kevenaar, T.A., Schrijen, G.-J., Bazen, A.M., and Veldhuis, R.N., “Practical biometric authentication with template protection”, LNCS, Vol. 3546, pp. 436–446. Springer, Heidelberg, 2005.

125. Fierrez-Aguilar, Julian, Javier Ortega-Garcia, Joaquin Gonzalez-Rodriguez, and Josef Bigun, "Discriminative multimodal biometric authentication based on quality measures", Pattern Recognition Vol. 38, No. 5, pp 777-779, 2005.

126. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: application to face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, No. 12, pp. 2037–2041, 2006.

127. Kumar, Ajay, David CM Wong, Helen C. Shen, and Anil K. Jain, "Personal verification using palmprint and hand geometry biometric", In Audio-and Video-Based Biometric Person Authentication, pp. 668-678, Springer Berlin Heidelberg, 2003.

128. Inamdar, V.S., Rege, P.P., "Face features based biometric watermarking of digital image using singular value decomposition for fingerprinting", International Journal of Security and Its Applications, Vol. 6, No. 2, pp 47-60, 2012.

129. Arya, Meenakshi, and Rajesh Siddavatam, "A Novel Biometric Watermaking Approach Using LWT-SVD.", In Information Technology and Mobile Communication, pp. 123-131, Springer Berlin Heidelberg, 2011.

130. Dogan, Sengul, Turker Tuncer, Engin Avci, and Arif Gulten, "A robust color image watermarking with Singular Value Decomposition method.", Advances in Engineering Software, Vol. 42, No. 6, pp 336-346, 2011.

131. Chang, Chin-Chen, Yih-Shin Hu, and Tzu-Chuen Lu, "A watermarking-based image ownership and tampering authentication scheme.", Pattern Recognition Letters, Vol. 27, No. 5, pp 439-446, 2006.

132. Mehta, G., Dutta, M.K., Kim, P.S., "An efficient & secure encryption scheme for biometric data using holmes map & singular value decomposition", In: Proceedings IEEE International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MedCom), 2014.

133. Furon, Teddy, and Pierre Duhamel, "An asymmetric watermarking method.", IEEE Transactions on Signal Processing, Vol. 51, No. 4, pp 981-995, 2003.

134. Cox, I.J., Doërr, G., Furon, T., "Watermarking is not cryptography", LNCS, Springer, Heidelberg, Vol. 4283, pp. 1–15, 2006.

135. Golub, Gene H., and Christian Reinsch, "Singular value decomposition and least squares solutions", Numerische mathematik 14, No. 5, pp 403-420, 1970.

136. JAMA: A Java Matrix Package, http://math.nist.gov/javanumerics/jama/. [November, 2015]

137. Want, Roy, "Near field communication.", IEEE Pervasive Computing 3, pp 4-7, 2011.

138. Shin, Dong-Hee, "Towards an understanding of the consumer acceptance of mobile wallet.", Computers in Human Behavior, Vol. 25, No. 6, pp 1343-1354, 2009.

139. Deza, Michel Marie, and Elena Deza, "Encyclopedia of distances", Springer Berlin Heidelberg, 2009.

140. Chandra, DV Satish, "Digital image watermarking using singular value decomposition", in Proceedings 45th IEEE Midwest Symposium on Circuits and Systems, Vol. 3, pp. III-264, 2002.

141. National Institute of Standards and Technology (NIST). Computer Security Division Computer Security Resource Center. Federal Information Processing Standards (FIPS) Publications, http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf. [November 25, 2015]

142. Rivest R. Internet Engineering Task Force. 1992. The MD4 message-digest algorithm, http://www.ietf.org/rfc/rfc1186.txt. [November 25, 2015]

VITA

Doyel Pal

Candidate for the Degree of

Doctor of Philosophy

Thesis: PRIVACY PRESERVING ALGORITHMS FOR NEWLY EMERGENT COMPUTING ENVIRONMENTS

Major Field: Computer Science

Biographical:

Education:

Completed the requirements for the Doctor of Philosophy in Computer Science at Oklahoma State University, Stillwater, Oklahoma in December, 2015.

Completed the requirements for the Master of Technology in Computer Science and Engineering at University of Calcutta, Calcutta, West Bengal, India in 2011.

Completed the requirements for the Bachelor of Technology in Information Technology at University of Calcutta, Calcutta, West Bengal, India in 2009.

Completed the requirements for the Bachelor of Science in Computer Science at University of Calcutta, Calcutta, West Bengal, India in 2006.