2016

NEW SECURE SOLUTIONS FOR PRIVACY AND ACCESS CONTROL IN HEALTH INFORMATION EXCHANGE

Ahmed Fouad Shedeed Ibrahim
University of Kentucky, ahmed.uky@gmail.com
Digital Object Identifier: http://dx.doi.org/10.13023/ETD.2016.307

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Ahmed Fouad Shedeed Ibrahim, Student

Dr. Mukesh Singhal, Major Professor

Dr. Miroslaw Truszczynski, Director of Graduate Studies

NEW SECURE SOLUTIONS FOR PRIVACY AND ACCESS CONTROL IN HEALTH INFORMATION EXCHANGE

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Engineering at the
University of Kentucky

By
Ahmed Fouad Shedeed Ibrahim
Lexington, Kentucky

Director: Dr. Mukesh Singhal, Professor of Computer Science
Co-Director: Dr. Dakshnamoorthy Manivannan, Professor of Computer Science
Lexington, Kentucky 2016

ABSTRACT OF DISSERTATION

NEW SECURE SOLUTIONS FOR PRIVACY AND ACCESS CONTROL IN HEALTH INFORMATION EXCHANGE

In the current digital age, almost every healthcare organization (HCO) has moved from storing patient health records on paper to storing them electronically. Health Information Exchange (HIE) is the ability to share (or transfer) patients' health information between different HCOs while maintaining national security standards like the Health Insurance Portability and Accountability Act (HIPAA) of 1996. Over the past few years, research has been conducted to develop privacy and access control frameworks for HIE systems. The goal of this dissertation is to address the privacy and access control concerns by building practical and efficient HIE frameworks to secure the sharing of patients' health information.

The first solution allows secure HIE among different healthcare providers while focusing primarily on the privacy of patients' information. It allows patients to authorize a certain type of health information to be retrieved, which helps prevent any unintentional leakage of information. The privacy solution also provides healthcare providers with the capability of mutual authentication and patient authentication. It also ensures the integrity and auditability of health information being exchanged. The security and performance study for the first protocol shows that it is efficient for the purpose of HIE and offers a high level of security for such exchanges.

The second framework presents a new cloud-based protocol for access control to facilitate HIE across different HCOs, employing a trapdoor hash-based proxy signature in a novel manner to enable secure (authenticated and authorized) on-demand access to patient records. The proposed proxy signature-based scheme provides an explicit mechanism for patients to authorize the sharing of specific medical information with specific HCOs, which helps prevent any undesired or unintentional leakage of health information. The scheme also ensures that such authorizations are authentic with respect to both the HCOs and the patient. Moreover, the use of proxy signatures simplifies security auditing and the ability to obtain support for investigations by providing non-repudiation. Formal definitions, security specifications, and a detailed theoretical analysis, including correctness, security, and performance of both frameworks are provided which demonstrate the improvements upon other existing HIE systems.

KEYWORDS: Health Information Exchange, Security, Privacy, Authentication, Authorization, Integrity

Author's signature: Ahmed Fouad Shedeed Ibrahim

Date: July 22, 2016

NEW SECURE SOLUTIONS FOR PRIVACY AND ACCESS CONTROL IN HEALTH INFORMATION EXCHANGE

By
Ahmed Fouad Shedeed Ibrahim

Director of Dissertation:	Dr. Mukesh Singhal
Co-Director of Dissertation:	Dr. Dakshnamoorthy Manivannan
Director of Graduate Studies:	Dr. Miroslaw Truszczynski
Date:	July 22, 2016

Dedicated to the soul of my father.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Mukesh Singhal, for his constructive comments, guidance, support, counseling, and advice throughout my graduate study. Thanks to Dr. Dakshnamoorthy Manivannan, Dr. Zongming Fei, and Dr. Ibrahim Jawahir for serving on my dissertation committee, and providing critical and constructive comments on my dissertation. I would also like to express my gratitude and appreciation of Dr. Santosh Chandrasekhar for his support and guidance that helped me get through several topics in my research, specially proxy signatures security proof.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

In order to promote widespread adoption and meaningful use of Health Information Technology (HIT), the US federal government provided financial incentives to healthcare providers under the 2009 Health Information Technology for Economic and Clinical Health (HITECH) Act. The HITECH supports the development of electronic health information management systems and secure health information exchange.

1.1 Health Information Exchange

Health Information Exchange (HIE) refers to the electronic transmission of health care data among facilities and professionals within a particular community, area, or hospital. In other words, HIE is the ability to appropriately access and securely share patients' health information between different healthcare organizations (HCOs) according to national standards. It involves healthcare and government institutions, health information organizations, and qualified health care providers. The purpose of HIE development is to improve healthcare delivery by offering reliable and secure ways to access and retrieve health information among diverse systems. It is an inherent part of the health information technology infrastructure as it aims to improve data gathering and medical care.

While now-a-days most medical information is still gathered through written records and is stored in paper form, HIE managed to develop three main methods to innovate current healthcare procedures. These methods are defined as: consumer-

mediated exchange, directed exchange, and query-based exchange [4]. Consumer-mediated exchange is done by providing patients with access to their own electronic records, thus allowing them to track their health conditions, determine whether there is erroneous billing or medical data, and update their self-reports. Directed exchange is conducted when a healthcare organization transfers such vital information as lab test results and medication dosage to other specialists involved in the care of the same patient. Query-based exchange usually occurs in unplanned medical care when a healthcare organization needs the previous health records of a new patient. This is done by requesting access to these records through the HIE system.

A patient's medical records should follow him or her wherever and whenever needed, despite barriers that may occur due to the involvement of multiple facilities in different geographic areas. Thus, current HIE governance should be done through effective collaboration between entities while considering implementation costs. Unfortunately, not all facilities are able to afford electronic exchange, and this is why this issue should be further addressed by both reducing the overall cost of HIE and searching for substantial financial support for the development of the system [5]. Furthermore, governance policies and models should be introduced and constantly updated to efficiently manage the system and solve arising issues. Finally, since health-related information is too private to be retrieved by non-professionals, HIE relies on secure data transfer among various electronic systems [6]. This is why HIE should be done with regards to privacy policies, strictly limiting the access to patients' records in order to prevent any outsider from gaining unauthorized access to the system. The information should be exchanged among entities without leaking out

of the system, which may occur because of the faults of the system itself or liability issues.

1.2 Health Information Exchange Concerns

Privacy and security management already create problems in today's HIE. Specifically, there are three main issues which need to be addressed. The first problem occurs when insiders of the system abuse their access rights [7]. This usually happens when health care providers share medical records of their patients with unauthorized individuals, either out of irresponsibility, for personal reasons, or in exchange for some kind of gain. For instance, medical records of celebrities and politicians frequently leak out of HIE system into the media. The second problem concerns unauthorized insiders, who may have access to the system itself but not to the records [8]. For instance, hospital employees who do not provide direct patient care or former employees who have not yet been electronically restricted from data retrieval. The former group can use the existing access to hack the private informational database while the latter may decide to seek vengeance on their former employers by undermining HIE security. The third problem arises when an unauthorized intruder attempts to enter the system either by attacking it directly or by pretending to be part of the health care team [9]. Either way, this problem has already shown itself as most harmful because outside intruders risk more than the insiders, and thus they tend to have ambitious plans that are worth their risk. Therefore, it is evident that HIE needs to address the problem of privacy and security before further development and application in health care.

As the healthcare industry starts relying more heavily on electronic storage of health records, the need to secure such records has become essential. As shown in Figure 1.1, the Secretary of the U.S. Department of Health and Human Services (HHS) developed two rules to satisfy the requirements of the Health Insurance Portability and Accountability Act of 1996 (HIPAA): the privacy rule and the security rule.

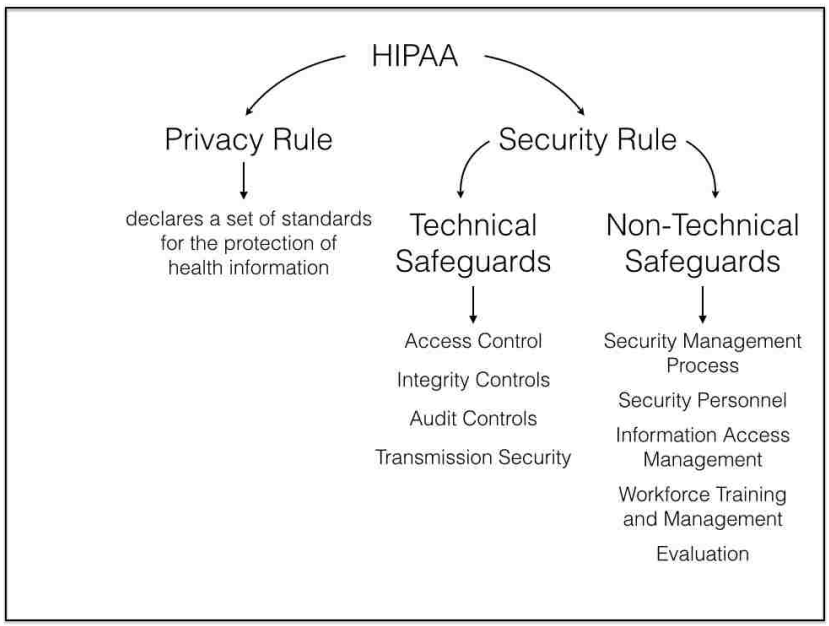

Figure 1.1: HIPAA Rules

The privacy rule [10] declares a set of standards, which describe how healthcare providers keep, store, protect, and share patients' health information. The HIPAA security rule [11] defines security standards to protect the privacy of individuals' electronic Protected Health Information (e-PHI). It specifically mentions the requirements of ensuring the confidentiality, integrity, and availability of all e-PHI created, received, maintained or transmitted. It also requires the identification and protection of e-PHI against anticipated threats to the security or integrity of the information.

An Electronic Health Record (EHR) refers to a patient's paper chart stored in a digital format. EHRs are increasingly becoming the way of storing information about patients (like medication, allergies, laboratory test results, immunization statuses, vital signs, radiology images, personal statistics like age, weight, and billing information). EHRs encompass patient's medical and treatment histories. Such records are usually stored locally within a certain infrastructure, e.g., at medical institutions, and are only accessible from within the infrastructure. To increase efficiency in medical services and to provide complete and accurate medical information, inter-institutional EHRs are more and more used to accumulate and keep information on patients. EHRs instantly and securely make information to authorized users only in order to automate and streamline providers' workflow [12]. In addition, EHRs facilitate medical decision-making concerning patients' care by allowing access to evidence based tools. EHR systems are developed to increase the efficiency of medical services and to provide complete and accurate medical information on patients. The U.S. Government is encouraging healthcare providers to deploy EHR systems by providing "EHR Incentive Programs" [13].

1.3 Electronic Health Records Benefits

Electronic Health Records (EHRs) are useful to both providers and patients. The benefits associated with EHRs include:

Improved health care quality and accessibility: EHRs improve the quality and ease of accessibility of health care for both the patients and providers. The

electronic system allows providers to gain quick access to the records of patients from inpatient and remote locations for easy coordination and efficiency of health care [14]. Also, it supports improved decision making, clinical alerts, medical information, and reminders. Furthermore, it provides tools for improving performance through real-time quality reporting, offers intelligible and comprehensive documentation for precise billing and coding and links with other EHRs, registries, and laboratories. Finally, it supports more steadfast and safer prescription of medication (electronic prescribing). For the patients, it reduces the repetitive process of filling out forms at each office they visit, provides convenient electronic prescriptions which are sent to the pharmacies directly, offers online patient portals to enable them to interact with the providers and provides the patients with electronic referrals for easier access to continued care with providers and specialists [15].

Increased patient's participation: Collaboration in informed decision making can be facilitated when patients and providers share access to electronic health information. Patient participation is enhanced through creation of an avenue for communication between providers and patients [16]. Personal health record applications can be provided to patients to enable them maintain and manage their own health information.

Enhanced care management: EHRs has the ability to integrate and organize the health information of patients in order to enable its immediate dissemination among all the authorized providers involved in patient care. Every provider has access to up-to-date and accurate information about each patient to improve care quality [17].

Value added diagnostics and patient results: Complete and accurate information enables the providers to enhance patient medical care. EHRs improve disease diagnostic abilities and minimize medical errors. This leads to enhanced patient results and improved patient safety [18]. Risk management can be improved through providing clinical reminders and alerts to providers.

Improved medical management through increased practice efficiencies and cost savings: This is made possible through enhanced communication with all other providers such as labs, technicians, and health plans by making patient information available anywhere. Amount of time spent on paperwork is reduced, leading to reduced costs [19]. Paperwork tasks can be streamlined to increase efficiencies and which in turn further reduces time and monetary costs. Introduction of electronic prescribing replaces paper prescriptions. E-prescribing lowers costs and improves health care. EHRs reduce the duplication of testing by providing a central place for the storage of patient information.

1.4 Electronic Health Records Standards

Standards in representing EHRs are important in that they provide a common language and a set of anticipations to enable interoperability among systems and devices. The standards for EHRs are aimed at increased coordination of care. There are groups formed to harmonize the challenges faced by shared Electronic Health Record systems [15]. The groups include: HL7 [20] [21] [22], which is responsible for developing a standard for the clinical document architecture and templates, openEHR [23], which

deals with the technical activities like architecture, implementation projects, and clinical activities like archetypes, standard activities, and CEN (a standardization committee for Europe dealing with EHR communication standards). The following standards derived from the above groups should be observed when presenting Electronic Health Record systems:

1. EHRs should have the ability to generate, send, obtain and present standardized evolution of care documents.

2. EHRs should have the ability to offer electronic prescription, reconciliation between patients' past and present medical information, incorporation of laboratory test outcomes, and creation of patients' care summaries.

3. EHRs should be capable of working with standardized documents, which is referred to as interoperability.

4. EHRs must have the ability to use the consolidated clinical document architecture (CCDA) for the transfer of care documents summaries.

1.5 Health Information Exchange Architectures

Since the role of HIE is to provide care facilities with the ability to circulate EHRs among varied medical information systems, HIE systems can have federated, centralized, or hybrid architectures. The federated structure requires local patient information storage at each healthcare organization to ensure a higher level of data security and privacy. If an outside organization wants to access patient information, it must

retrieve it from the HCO holding the information. In addition, to access the information the entity must be a member of an association, and at the same time, must commit itself to sharing the information with other members of the network. The participants in this model are often held responsible for ensuring that information is accessed by the authorized members only.

The centralized HIE system, also referred to as the consolidated model, involves storing all health information in a single data repository or warehouse (e.g., cloud). Each member of the centralized HIE system is expected to transmit patients' health information to the remote repository, where the information is securely stored [24]. The health information is continuously updated through interfaces connected directly to each healthcare organization's information repository in order to improve security and confidentiality. These interfaces usually allow for unaltered patient information flow to the central authority. Whenever a member organization requests access, it is subjected to pre-defined unique patient identifiers before being authorized.

The hybrid HIE system combines the elements of centralized and federated systems. This system holds significant record identifiers along with requests for patient data distributed across a network. To ensure security and privacy, access to the information is often subject to stern measures. As such, a record locator key is habitually used not only to gather health information but also to transfer it to the healthcare organization. The system uses algorithms within the network applications to ensure positive trends in gathering the patient information stored in the remote repository.

1.6 Problem Statement

This section presents a case study of the problem we are considering. HOMEBASE is a provider of medical services and is the medical home for George. FARAWAY is a provider of medical services in another community. George is traveling and finds that he must seek services at the FARAWAY clinic.

1. FARAWAY calls and asks the health information management (HIM) department at HOMEBASE for George's summary clinical record.

2. The HOMEBASE's HIM department asks the requesting physician at FARAWAY about George to confirm his patient identity and consults the national provider database (or another resource that the relying party trusts) to verify that FARAWAY is a valid provider of services.

3. The HOMEBASE's HIM department then calls the FARAWAY management services organization (MSO) department to verify that the requesting physician actually works at the FARAWAY clinic and verifies the fax number of FARAWAY.

4. Having assured that the request is legitimate (established trust), the HOMEBASE's HIM department faxes the records requested by FARAWAY.

To clarify the problems and drawbacks of the current HIE process, we present a scenario of the problem we are considering, as shown in Figure 1.2.

Tim lived in two states before moving to California. In the first state, Virginia, Tim had a terrible accident when he was 25 years old and received a blood transfu-

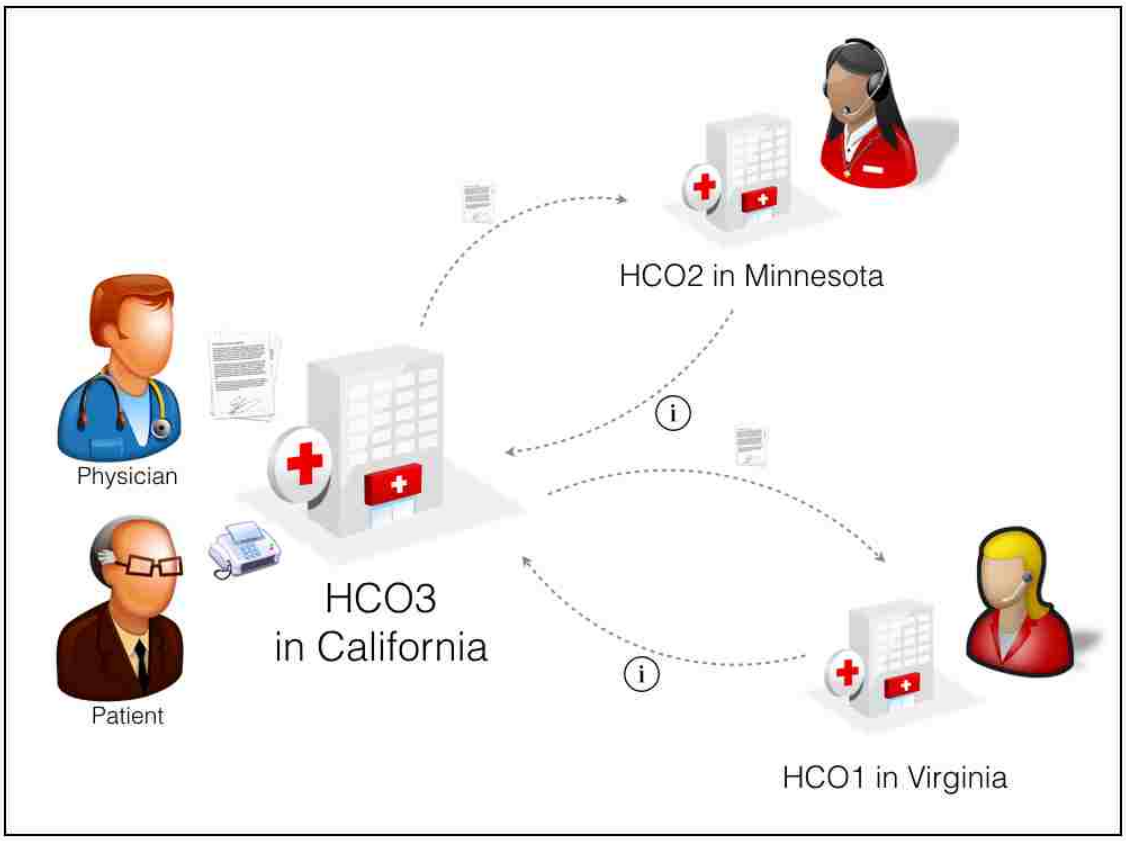

Figure 1.2: Current Health Information Exchange Process

sion at HCO1. Healthcare organization HCO1 uses EHR system "S1". Tim visited HCO1 for various treatments while he was living in Virginia. Tim then moved to Minnesota and discovered that he had a liver problem. After being seen several times at healthcare organization HCO2 in Minnesota and doing blood work, Tim's physician informed him that he needed a liver transplant. Healthcare organization HCO2 uses EHR system "S2". Tim received a successful liver transplant at HCO2 when he was 45 years old. He continued his care at HCO2 for several years to follow-up on the transplant surgery. He was also seen by physicians from other specialities unrelated to the transplant surgery. Tim considers his visits to the other speciality physicians confidential and he does not want anyone, including other physicians in

HCO2 and his family members, to know about the visits. The EHR system in HCO2 automatically prevents physicians from viewing health information unrelated to their area of expertise.

Now, Tim is visiting California and had an accident. He is admitted to organization HCO3 which wants to collect all medical history to have clear and accurate health information before starting a treatment plan. Tim's medical history includes all treatments, surgeries, x-rays, blood work, and allergies. Healthcare provider HCO3 uses EHR system "S3". The typical process to retrieve Tim's health information from HCO1 and HCO2 is to obtain a written consent, signed by Tim, to release his health information stored at both HCO1 and HCO2. The written consent is a general form that does not specify which health information is being requested. HCO3 has to fax the signed written consent to HCO1 and HCO2 and wait for the reply to be faxed back. The process may also include phone calls between involved parties to authenticate the initiating healthcare provider.

This process of health information retrieval from several healthcare organizations has three main drawbacks. First, even if the requesting healthcare provider and the provider holding requested medical information are operating normally, the patient must wait until the medical information is received back.

Second, the requesting healthcare provider could face a huge delay in getting the response back if there exists a time difference between communicating healthcare providers or if the healthcare provider holding requested medical information is not operating during the time of request. In this case, a further appointment may need to be arranged when all medical information is gathered and available.

Lastly, the signed written consent does not specify which medical information should be retrieved. Subsequently, disclosure of health information that the patient did not want to share with the requesting healthcare provider is more likely to occur. Also, the consent can be used to retrieve health information from healthcare organizations other than those specified by the patient.

There is a range of high-level consent models a Health Information Exchange might choose to implement [25]. These include no consent, opt-in, opt-in with limitations, opt-out, and opt-out with exceptions [26]. The no consent usually does not necessitate any form of agreement on the patient's part to take part in the HIE. The opt-out model tends to allow for a prearranged information set to be included by design in the HIE. However, patients have the right to restrict or deny access to their information in the HIE system. The opt-out with exceptions model allows personal health information to be available in a HIE system. However, it provides the patient with the freedom to exclude information from an exchange in a selective manner. The patient is also allowed to limit information exchange for certain uses. The opt-in model necessitates that patients make their desires known about the exposure of their information within an exchange. The opt-in with restriction model requires the patients to specify the data and amount of information to be availed in the HIE system. To automate HIE, HCOs must have common terms and attribute values that would govern the exchange and maintain trust requirements.

1.7 Motivation

A healthcare system breach refers to an impermissible security occurrence where data that is sensitive, confidential, or protected is accessed, altered, or used by an unauthorized person. Breaches are caused by theft, disclosure or unauthorized access, loss, improper disclosure or hacking into the electronic health record systems. Employees of an organization pose the greatest threat to information security. Adequate security measures and mechanisms should be enforced to safeguard information against breaches [15]. When a breach occurs, the affected individuals should be notified. Also, internal actions should be taken to minimize the damage like corrective activities and measures to safeguard against recurrence. There has been a gradual increasing trend of breaches into healthcare systems. The rise in the number of incidents is attributed to the advancements in technology. Hackers are using more sophisticated technologies to attack systems. Also, the health sector is more vulnerable to attacks. Security controls in the health sector are not fully enforced.

Moreover, this new digital environment is more vulnerable to cyber criminal exploits. The availability and even distribution of healthcare associated cybercrime is a concern and an emerging threat to healthcare systems [27]. Major breaches have occured in organizations like Anthem, CareFirst, Premera and UCLA Health systems. As a result, a total of 143 million patient records were exploited by cyber-attacks, which amounted to 45% of the American population [28]. A cyber security assessment by the Healthcare Information and Management Systems Society (HIMSS) in 2015 indicates that in the last 12 months, 64% of medical centers had been exposed to

external cyber-attacks [29]. Bloomer News claimed that in the previous 2 years, of all healthcare providers, 90% have been attacked [30]. Furthermore, most data breaches occur in healthcare and medical industries as compared to financial, governmental, or educational sectors [31]. In 2014, the healthcare and medical sectors were identified to have been victims of 46% of the described breaches [29].

The Saint Joseph Health System in Texas suffered a security breach in 2014 [32] [33]. A server was hacked and 405,000 records of both past and current patients, employees, and beneficiaries to the employees were compromised. Medical information of patients' and employees' bank account data were accessed without authorization.

Rural Illinois hospital was attacked in 2014 [34]. The hospital received an email from an unknown sender containing patient names, addresses Social Security numbers and birth dates. The sender demanded a substantial payment and threatened that if the hospital failed to pay, the information of all the patients would be made public. As a result, the private health records of 12,621 persons were compromised [34].

Boston-based Massachusetts General Hospital suffered a data breach when employees replied to a phishing email, believing it was legitimate, providing protected health information in 2014 [35]. As a result, 3,300 patients were affected and notified by the hospital for the security compromise of their health information.

The Saint Vincent health system in Indianapolis faced a breach in 2014 [36]. Social Security numbers and clinical data for medical group patients were compromised. The breach occurred through an email phishing event. 760 patients were affected and notified. Employees' network usernames and passwords were compromised. The hospital was also faced by other two breaches which involved the theft of unencrypted

laptops containing medical records for 2,341 patients. The breach corrupted the health records for all involved patients.

In April 2015, OhioHealth based in Columbus, Ohio discovered that an unencrypted flash drive containing 1000 sensitive health information records for patients was missing and notified the patients of a breach [37] [38]. The missing flash drive could have been a theft by one of the employees for financial gain.

In July 2015, University Hospitals Elyria Medical Center suffered a data breach. An employee accessed unauthorized medical records for patients [39] [40]. The breach affected 297 patients. The employee was fired and the affected patients notified.

Texas Children's Clinics faced a data breach in October 2015 [41] [42]. An employee took business documents home from the office. The employee gained unauthorized access to patient health records, took screenshots, and showed them to a third party. The employee was fired immediately. 16,000 patient records were affected. Patients' names, birth dates, diagnostic information, and treatment information were compromised.

In February 2016, The Hollywood Presbyterian Medical Center in Southern California was targeted by a ransomware attack [43] [44] [45]. The hospital's network was hacked and disabled, rendering all computers useless. The computer network data was encrypted and held hostage. To restore their systems and administrative functions, the hospital had to pay the hackers $17,000 for the decryption key.

Whereas the financial and retail sectors have had sufficient time to develop reliable security standards, the recent transition to EHRs has placed healthcare security behind the security of other industries [46]. Health information contains details like

prescriptions for controlled substances which makes it more valuable to hackers than credit card numbers [47]. The FBI indicates that cyber criminals sell partial medical information at a rate of $50 per record on the black market, compared to $1 for a stolen social security number or credit card number [47]. Attackers aim to exploit the patients' records in the healthcare and medical sectors since the records contain very critical data [48]. The value of medical records is sustained compared to other documents. For instance, it may be difficult to address individual medical identity theft but a credit card theft is easy to deal with and further harm can be prevented. There are no available remediation strategies for dealing with medical identity theft. This shows that the threats in healthcare industry has reached acute level [49].

When designing HIE systems, security is paramount. The following security requirements should be taken into consideration:

Authenticity and Authentication: This refers to the process of establishing the veracity of claims made by someone or about a subject [50]. The designer of an HIE system should ensure that only authenticated users have access to the specific information. The credentials provided by users should be compared with the database records and if they match, access may be granted to the healthcare record system.

Non-repudiation: This is the guarantee that someone cannot deny the authenticity of their signature or sending of a message. For example, one party cannot deny having sent or received a transaction. Once a patient record is inserted into the system, the employee who performed the entry cannot deny having performed the transaction. Digital signatures and encryption should be used to establish non-repudiation [51].

Ownership of Information: The designer of the HIE system should take into consideration the owner of the information so as to apply the necessary measures to protect against unauthorized access or use of the patient's medical information. The patient should be able to communicate their consent to the sharing of their health information to authorized healthcare specialists [36]. HIE systems should give the patients the privilege of granting rights to the authorized users on the basis of what role the user plays.

Integrity: This refers to the process of ensuring the accuracy and consistency of data. The data should be protected against unauthorized tampering through the use of access control and encryption techniques.

Confidentiality: This means safeguarding information against unauthorized access. Only authorized users should have access to the health and identity information. This can be accomplished through the use of access control and encryption techniques.

Availability: Information should be available when needed. During design, the security controls used to protect the information and the communication channels should be tested to ensure that they are functioning correctly [14]. Ensuring that information is available at any time makes decision-making more efficient.

1.8 Contributions of the Dissertation

The development of HIE protocols is an important milestone for the future of monitoring patients. It can also save lives, enhance disease management, and help medical research achieve more efficient and effective results. The goal of this dissertation is to

address two major concerns in building practical and efficient HIE solutions: privacy and access control, in order to secure the sharing of patient health information.

The first solution presents a system model and protocol to secure HIE among different healthcare providers while focusing primarily on the privacy of patients' personal and health information. The system model specifies all of the major physical and logical components to allow secure HIE between different healthcare providers. The protocol provides healthcare providers with the capability of mutual authentication between communicating organizations and patient authentication to his/her home healthcare organization. The privacy is maintained through a unique way of generating a symmetric key, at both healthcare organizations involved in the HIE, which corresponds to the patient and is valid for the authorized session only. The solution allows patients to authorize a certain type of health information to be retrieved, which helps prevent any unintentional leakage of information. It also ensures the integrity and auditability of the health information exchanged. A security analysis and a performance study of the first protocol show that it is efficient for the purpose of HIE and offers a high level of security for such exchanges.

The second solution presents a cloud-based HIE system model and an authorization protocol that fills the gap between cryptographic and non-cryptographic approaches, with the former lacking explicit authorization enforcement mechanism that is cryptographically secure, and the latter being complex, computationally expensive, and limited in policy specification. The protocol combines authentication with authorization rather than combining encryption with authorization as done by the majority of existing cryptographic approaches (ABE-based schemes). A novel

proxy signature-based protocol is developed to enable authenticated and authorized selective sharing of patient health information via a cloud-based HIE. The protocol exhibits several desirable features that include non-interactive and on-demand operation, flexible specification of access control policies, audit support, and compliance with agreements between healthcare providers and patients. The security analysis for the protocol shows that it is secure against forgery under the well-known discrete log assumption. The performance of the protocol using the developed trapdoor hash-based proxy signature scheme achieves the best all-round performance while being provably secure compared to other well-known proxy signature schemes.

1.9 Organization of the Dissertation

The rest of this dissertation is organized into five chapters. Related work is presented in Chapter 2. A background on cryptographic techniques used in the solutions is presented in Chapter 3. The privacy protocol and its security analysis are presented in detail in Chapter 4. The access control protocol and its security analysis are presented in detail in Chapter 5. Finally, Chapter 6 concludes this dissertation.

Chapter 2 Related Work

In this chapter, we review recent related work and solutions for the privacy and access control of Health Information Exchange. The work presented is the most relevant to the research presented in this dissertation.

2.1 Research on the Privacy of Health Information Exchange

Yu et al. 2007. An Electronic Health Record Content Protection System Using SmartCard and PMR. [52]

Yu et al. [52] presented a patient-centric content protection system that is based on temper-resistant hardware and widely adopted security protocols for identification, EHR encryption, and communication. A Personalized Media Recorder (PMR) is used to achieve high availability and interoperability of EHR content protection systems. High density SmartCards [53] were used for the identification and authentication of patients, healthcare providers, and support personnel. A patient's SmartCard (P-SmartCard) is used to encrypt the patient's EHR content and to enforce rule-based access control to a patient's EHRs.

Every healthcare provider (e.g. physician, nurse, laboratory worker, etc.) who seeks access to and creates new EHRs would use the Healthcare Provider SmartCard (HP-SmartCard) to undeniably sign initiated transactions, other than those for identification and authentication. A specially designed hardware, called a Personalized

Media Recorder (PRM), is used to interface with a P-SmartCard and HP-SmartCards in order to store patients' protected EHR transactions at the HP site and on patients' recordable media. A secure mobile agent called V-SmartCard can emulate the functionality of the patient's real SmartCard (P-SmartCard) if it is not present. These components run under a hardware backbone which provides public-key infrastructure (PKI), distribution of CRLs, EHR backup, and V-SmartCard management.

Adequate cryptographic analysis of the overall scheme was not provided. The proposed work requires that each patient has a SmartCard at the time of service. The proposed system requires the PRM hardware to be available at every healthcare provider. V-SmartCards can be used to impersonate the use of a real P-SmartCard, which can create security flaws. Another concern is that at a certain threshold, patients' EHR privacy can be compromised if most of the patients' information is not signed and encrypted by the public/private key pair.

Hu et al. 2010. A hybrid public key infrastructure solution (HPKI) for HIPAA privacy/security regulations. [54]

Hu et al. [54] proposed that patients sign a fixed time-period contract with the healthcare provider and make sure every access to the PHI during the medical treatment process is securely authorized by the hospital and recorded for non-repudiation purposes. This work presented a contract-oriented e-health security architecture which delegates the trust and security management to the medical service provider during the contract period of the treatment. A hybrid security scheme based on public-key infrastructure (PKI) and smartcards is used. A PKI scheme is deployed for the mu-

tual authentication and distribution of sensitive yet computationally non-intensive data. Their architecture consists of a smartcard trust center (STC) that issues medicare smartcards and a medical center server (MCS) that belongs to the place where a patient registers as the home medical service provider.

The MCS has its own database that stores all relevant data including its patients PHI data. Patients are issued a smart medical care card by the STC which contains the patient's private-public key pair and other basic data. The STC keeps a copy of the patients' private-public key pair. Each MCS also needs to register with the STC to obtain a private-public key pair. Medical personnel register with their organization and are issued an organizational smartcard containing their private-public key pair.

At the time of treatment, the patient negotiates a contract key with the MCS which allows the medical center unlimited access to the PHI data during medical service period and sets an expiry date for such key. At the end of the treatment, the PHI data and a hash of the PHI data signed by the medical personnel are encrypted using the contract key and sent to the patient to be stored on the patient's smartcard. Also, the PHI data and a hash of the PHI data signed by MCS's private key are encrypted using STC's public key and sent to the STC for storage. At a foreign MCS, the patient must provide the contract key and PHI data stored on the smartcard. If the foreign MCS wants to retrieve the patient's PHI data from home MCS, the patient's contract key must be provided.

Storage of the encrypted PHI data and contract keys on the patient's smartcard is infeasible due to the limited storage capacity of smartcards. In the case that the patient loses the smartcard, all stored PHI data and contract keys are lost which

may compromise the privacy of the patient. A new smartcard must be issued by the STC, including the patient's key pair stored at the STC. No patient authentication mechanism at the home MCS was proposed for when PHI data are being requested by a foreign MCS. In fact, if a contract key is compromised, MCS can still access the corresponding PHI data encrypted with such a key. It was proposed to use patients' PHI data stored at the STC for emergency access, but no identification and privacy mechanisms were proposed.

Haas et al. 2011. Aspects of Privacy for Electronic Health Records. [55]

Haas et al. [55] proposed a scheme to allow user-controlled disclosure of health data to third parties which is divided into two subsystems: data service and patient service. Medical systems use the data service for storing and retrieving medical data. Medical data are indexed with a data ID and stored in an encrypted form, using a persistent random value unique for every patient and a consecutive number for every piece of data stored about this patient. Digital watermarking is used to tag each instance of medical data access. The patient service offers policy management, as well as logging and verification services. It is used by patients to check the correct functioning of the data service, which puts a lot of burden on patients and requires patient training and continuous monitoring.

The authors proposed a scheme to allow user-controlled disclosure of health data to third parties without forcing patients to trust the EHR system provider. Their scheme offers a privacy-management system that offers patients informational self-determination including usage control with the implicit possibility to trace data flows

after sensitive data has been legitimately disclosed. The scheme offers an access control mechanism with audit capabilities concerning access to health data and its disclosure to third parties.

The system for EHRs is divided into two subsystems: patient service and data service. The patient service is the communication interface for patients. It offers policy management, a logging service, and a verification service which allow patients to check the correct functioning of the data service. The data service controls the disclosure and storage of EHRs. Medical systems use the data service for storing and retrieving medical data. Before storage, medical data D_{med} is encrypted using a persistent random value $R_{Patient,Key}$ that is unique for every patient and a consecutive number for every stored piece of data about this patient.

The encrypted data is indexed via a data ID_{data}, which is derived from a hash of a second patient-related random number $R_{Patient,Index}$, and the consecutive number. Digital watermarking is used to tag each instance of medical data access. The tag includes the medical data provider's identity and the requester's identity. The sequence of tags for the same medical data constitutes a so-called delegation chain. It is assumed that every medical provider will provide a valid digital watermark with every medical data request (read/write). Digital watermarks are validated by combining an asymmetric fingerprinting scheme with the author's previous work: the protocols for a non-linkable delegation of rights by anonymous credentials [56].

The paper did not present a preventive scheme as expected. The patient service provides nothing new, and only trivial methods were used to offer policy management and the logging and verification services. The authors' previous work [56] used in the

digital watermark validation has not be tested nor used before. There is no way to prove that the link between the medical provider and its digital watermark is valid and legitimate. The system presented is not privacy-protecting and does not enforce obligations regarding health data disclosure. Instead, an after fact method was proposed to detect disclosures which did not meet patients' policy access rights. It is not clear how illegitimate disclosures can happen. The proposed scheme does not include a way to keep track of changes in patients' policy access rights and their effects on tracing previous disclosures. Neither security measures no a security study were presented. Confidentiality of the medical data outside of the proposed system and during transmission was not considered.

Vergara et al. 2015. Chains of Trust for On-Demand Requests of Electronic Health Records. [57]

Vergara et al. [57] proposed creating on-demand trust relationships among medical institutions' Health Record Servers (HRS) to authenticate requesters based on chains of trust. This is approached by contacting several parties to validate the identity of the requester and deliver the requested records. The authors propose to create an on-demand trust relationship among participating medical institutions' servers. As there may not be prior relationships established between medical providers, the proposed protocol uses two primary types of servers: Health Record Servers and Trust Servers, that allow parties to interact with some degree of trust for the exchange of basic (emergency information, but not complete) EHRs. Health Record Servers (HRS) are the medical provider servers storing patients' EHRs. Upon access requests, Trust

Servers (TS) validate the requester (via SSL) and query other nearby trust servers to retrieve the appropriately signed token corresponding to the HRS holding the patient's EHR for the requester. Finally, a signed identity token is returned to the requester which can be presented to the HRS to access the needed records.

Trust Servers (TS) validate the requester and query other nearby Trust Servers to retrieve a signed token corresponding to the HRS holding patient's EHR. A signed Identity Token is returned to the requester which can be presented to the HRS to access the needed records. Identity tokens can then be sent to HRS holding the patient's EHR which validates them and sends the patient's requested EHR to the requester encrypted using a symmetric key agreed upon beforehand.

Identity tokens include the following information: patient ID, expiration, record location, and signature (by a TS known to the desired HRS). These fields ensure the integrity of the token, prevent malicious users from impersonating a requester, and prevent the requester from obtaining the records of a different person. Identity tokens can then be sent to the HRS holding the patient's EHRs. HRS validates the received identity token and then sends the requested patient's EHRs encrypted (using a symmetric key agreed upon beforehand) to the requester.

There are many issues with this work. It operates on mobile devices only and no protocol was proposed to operate in any existing healthcare provider's system. The protocol is designed to handle emergency access only, so the access is limited to basic information and no approach to gain full access was presented. It has no privacy policy, no patient consent requirement, and no process of patient digital approval. Other trust mechanisms should be studied to handle different roles and

access restrictions rather than trusting requesters with full access to basic health records immediately.

The communication between the requester and the Health Record Server is always done using the same unique symmetric key generated by the system for all communication to decrease the cost of encryption and decryption, which is not a secure solution for long term implementation. Tests of the proposed protocol were performed using only three Trust Servers in extremely close proximity. This approach contacts several parties to validate the identity of the requester which will require much longer TS hops to locate the correct TS responsible for signing the Identity token. A real world deployment of the protocol may require many more Trust Server hops to locate the correct Trust Server responsible for signing the identity token. This may become prohibitively expensive to set up or practically impossible to implement. Clearly, there are several weaknesses in the protocol that need to be addressed before it can be used in the real world.

2.2 Research on the Access Control of Health Information Exchange

Wang et al. 2012. Implementing a Personal Health Record Cloud Platform using Ciphertext Policy Attribute Based Encryption. [58]

Wang et. al. [58] propose a model that can be used to overcome the challenges associated with Personal Health Records (PHRs). The proposal was based on the argument that information privacy issues arise during data movement from the PHR systems to the clouds, which leads to the exposure of certain data that is confidential and if

released, could result in patient embarrassment. To avoid this, the authors propose the application of a Cipher-text Policy ABE (CP-ABE) which helps maintain privacy by encrypting data before it is sent over the PHR. This is necessary since conventionally, data on the PHR can be downloaded by anyone. The objective of the proposed CP-ABE system is to ensure that although this data can be downloaded by anyone, its decryption is on an access controlled basis. The proposed system architecture operates based on various key security requirements which include: maintenance of transit health data confidentiality, integrity of health data, health data authenticity, patient centric control of data access, and flexibility in data revocation.

Based on the provided security requirements, the key participants of the proposed CP- ABE include the health care providers, patients, and providers of cloud services, as well as the PHR viewers and attribute authorities. These participants use the system architecture which is defined through actions such as the set-up of the systems, conversion of the EHR data into more manageable PHR data, data encryption via the PHR, sending requests, generation of identity key and data decryption at the recipient end. The system set-up procedure involves the organization of both hardware and software components into structures that support the running of the proposed system. The facility-generated EHRs are then converted into PHR systems which enable the data to be handled on a patient centric basis. This allows the patient to define the attributes associated with health records and to encrypt the data in the PHR. The encrypted data is then transferred into the public cloud proxy. As highlighted previously, the key objective of the proposed system is to prevent privacy losses during data transfer. The generated identity key can only be used by the intended

data recipient to decrypt the data from the PHR.

This architecture operates through a procedure that involves provision of a set of attributes which go beyond the conventional occupation and work place status requirements. The attributes provided by the patient include the name, gender, date of birth, marital status, work place, and patient address, as well as the desired expiration of the PHR. Of these attributes, only the name is regarded by the system as a non-numerical value while all other data are classified as numerical and are modified to fit the assigned category. For instance, the date of birth is translated to age while the marital status is encoded as a numeric value of 0 or 1. Only the data considered general, such as marital status, gender, and address, are stored in the server, while the remaining attribute data are stored in the private cloud system.

The access policy used by the system is designed by the owner and embedded in the Cipher-text used for the CP-ABE. In comparison with other previous work that applies ABEs in the healthcare sector for EHRs, the proposed system was evaluated based on its capability to reduce privacy concerns during data transfer and was found to be feasible. While other studies focus on the use of Identity Based Encryption, this proposal works through the differences between conventional ABEs. The authors therefore conclude that the CP-Based ABE system is better in terms of privacy achievement during data transfer as compared to Identity Based ABEs. Moreover, the system also helps to avoid unwarranted decryption due to the presence of access keys, which are designed to be used to access information related to the case in question.

Li et al. 2013. Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption. [59]

Another study that focuses on the use of ABE systems in the management of EHRs was carried out by [59]. According to these authors, the key issues related to the use of PHRs include exposure of user privacies, flexibility of information access, and the efficiency of user revocation in accessing information on patient health records. To address these issues, the authors propose a model for using Attribute-Based Techniques in PHRs. In particular, the authors propose a system that addresses the concern about multi-user capabilities in the handling of PHRs. The applicability of the system involves situations where multiple users are required to access the required information without having to expose the health records to the general public or unauthorized personnel. According to these authors, privacy issues in PHRs arise mainly during data outsourcing, especially where there is multi user access to information. To address this concern, a patient centric system is recommended for the incorporation of access keys for use by different recipients. The authors' proposal involves the use of multi-authority data access ABE to achieve privacy as well as to enable dynamics for data modification through access policies.

The proposed ABE architecture is described as a Multi Authority ABE (MA-ABE) and is designed to operate based on key performance requirements which include: data confidentiality, revocation on demand capabilities, scalability, efficiency, usability, flexibility, and control of access. Data confidentiality is planned to be such that any unauthorized persons lacking sufficient attributes matching the demands of

the system are not allowed to access the users' data under any conditions. Similarly, authorized persons whose attributes are no longer valid can have their access authorities revoked. The users who have access to the patients' data should also be restricted in their potential operations through control of activities such as modification of patient details. This framework operates using a multi authority ABE. The architecture involves several data users, multiple owners, as well as numerous public/private domains (PSDs) and attribute authorities (AAs). This implies that for each patient, the operations must involve system set-ups and data encryption at the first stage of the information management system. The encryption is carried out based on common data attributes shared across all PSDs with emergency attributes assigned across the break glass strategy.

The proposed framework allows multi-user access to health information. The patient is referred to as the PHR owner and is tasked with creating the patient file using various labels. The data is then encrypted according to the patient's data labels through the use of Key-Policy ABE (KP-ABE) and a role-based access policy for file access. The encryption applied using the patient's own labels aims to restrict access to files considered confidential and the access can only be achieved after authorization by the patient through provision of a break glass key. The role-based access is defined through the use of recommended settings or based on the patient's definition settings. For instance, the patient may describe the role-based access to be restricted only to recipients with attributes that align to the terms 'physician and doctor'. From this point of view, the physician and/or doctor cannot access the patient's private information encrypted with own data labels. The break glass key is provided for

use in Emergency departments by those who have access to the key and/or the first labels.

In the proposed multi-model framework, each patient is supposed to have control over his/her own PHR and is considered a trusted authority over the data. The accessibility of the data by outside recipients is controlled by the data owner through the use of the KP- ABE system which helps to create the first key. The combination of the patient-centric design through which KP-ABE is used and the multi-user interface enables the PHR to have a self-protecting attribute which makes it possible to protect information from unauthorized access through the break glass key. Moreover, the enhancements made to the conventional ABE have made it possible to prevent data infiltration even by semi-trusted recipients. However, the data owners are allowed to make changes to the PHR documents through attribute changes in the Cipher-text policy. The key modifications allowed in the cipher text include addition, deletion, and editing.

Tong et al. 2014. Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption. [60]

Tong et al. [60] examined the possibility of providing role-based access control and auditability for PHRs through the integration of ABE with threshold signing for data. The objective of this proposal was to prevent misuse of PHRs in both normal and emergency contexts. The paper begins the introduction to the proposed ABE framework through a discussion of the various concepts associated with ABE use in PHRs. The proposed system model is comprised of four key parts, which include a private

cloud, public cloud, PHR user, and EMT. In this context, the EMT refers to the Emergency Medical Technician who is the physician tasked with offering emergency treatment services to the patient.

The interactions between these four components in the system model are such that each user collects their own data through recording devices that are either worn or carried by the user. Each of the users is supported on a private cloud. Many such private clouds are then supported through a physical server. The private clouds are always present online and can be accessed in case of emergency. Before storing the data in the public clouds, the private clouds add security encryptions to the data. While proposing this system model, the authors assume that the connection between the user and their private cloud is secure and that it offers a secure opportunity to negotiate the network key to be used in the system.

The system proposed functions based on a threats model described by the private cloud's actions and the suspicions of the public clouds. The private clouds are trusted by the user while the public clouds are treated as honest but also suspected to be curious. This is based on the argument that the public cloud does not attempt to delete the user's data or modify it in any way, but does try to compromise the privacy of the user. To avoid such occurrences, public clouds are not allowed to access the user's sensitive health information. This makes it difficult for the user's information to be jeopardized. Similarly, the EMT is considered honest in their dealings in that they do not access information not pertaining to health records or information that is not role-based. The only information accessed by the EMT is that which is relevant to the case solution and thus applicable for the treatment of the patient.

The system proposed operates using various security requirements which include data privacy and auditability. The privacy concerns are resolved through various requirements such as confidentiality, anonymity, unlink ability, keyword privacy, and search pattern privacy. While implementing the proposed ABE system with threshold signing, the authors began by constructing the secure index for the PHR, meant to be used for data preservation; encrypting the data files with a time tag inference; hiding the data patterns in the private clouds, and retrieving data files based on user request. During data retrieval, the private cloud is used to compute a trap door, and then sends it to the required public cloud, which provides the address relevant to the required data. This operation procedure involves online operations and can only be achieved by meeting the system security requirements.

In comparison to other systems such as CP-ABE, IBE, and the threshold secret sharing, the proposed ABE plus threshold signing provides an efficient interface for the management of patient data. While the other forms of PHR management achieve key features of maintaining the privacy of the data, the proposal by these authors maintains privacy while also enhancing the auditability of the data in the system. On the other hand, the proposed framework differs from the MA-ABE proposed by Li et al. [59] in that while the latter provides an opportunity for multi-user access to data, the paper by Tong et. al. [60] provides only a framework that achieves auditability while still maintaining the single user access protocol. When evaluating the proposed framework for performance, the authors used an approach based on communication and storage efficiency, in addition to the auditability of the entered records and computational efficiency. These were the key measures of effective performance when

evaluated from the viewpoint of the authors.

In conclusion, the authors stated their study had accomplished the design objectives of achieving auditability through the use of techniques relevant to the situation. Through the use of a redundancy-based pattern hiding technique and the incorporation of access control in emergency situations, the authors managed to achieve their user-centric PHR design plans. Recommendations for future research include devising mechanisms that can be used to detect whether there have been leakages of patient data and to identify the possible sources.

Fabian et al. 2015. Collaborative and Secure Sharing of Healthcare Data in Multi-Clouds. [61]

The use of ABE is also discussed and implemented by [61] through the proposal to disperse data across several clouds through an integrated ABE with a Cryptographic secret sharing procedure. The basis for the proposal is the understanding that PHR systems are often faced with the challenge of data privacy and security, particularly during information sharing. The authors propose a combined method for the achievement of inter-organizational information sharing. The proposed system uses an architecture based on the application of a client-centered design to share information across various organizations. According to the authors, the big data era has led to an increasing need for information sharing across various health care organizations which results in privacy concerns and data security issues. The proposed architecture functions through interactions across various security and privacy requirements which include: confidentiality of medical records and their existence,

patient anonymity in the medical records, unlinkability between the records and patients, integrity and availability of records, user authentication, fine-grained access control and authorization, and anonymity of user capabilities, among others.

The architecture proposes data signing and encryption by patients at different health facilities, and that the encrypted data is then sent to multiple cloud systems in which the data is split and stored. The data from the various health facilities stored in independent cloud servers are then connected with data from various other cloud proxies. The summary of the architecture involves the signing and encryption of medical information by the doctor's software. The encrypted information is then sent to the local proxy of multiple clouds, which splits the information according to the scheme for secret sharing. In the third step, external identification indices are constructed by the multi-cloud proxies so that access to the data is possible from both ends. The data saved in the multi-cloud proxies is accessible by any authority given the identifier index for the particular patient. This enables the patient's information to be shared across different healthcare facilities, given a patient authorization. As such, the PHRs are made available across the board. The retrieval process involves access of information from the patient's health records through the proxies. The cloud proxies corresponding to the particular patient's access information are based on that client's health records, which are authenticated via the proxies and decrypted. The decrypted data is then sent to the doctor who confirms its authenticity in the particular context.

While addressing the security concerns for the data, the authors assume that all parties with access to the PHRs have common interest in upholding the privacy of the

patient information. However, the authors also recognize that in spite of formulation of common security objectives, procedural challenges may also arise during implementation, which calls for further system refinement since the authors believe that this is beyond the scope of the present study. In implementing the ABE system, the authors suggested a combination of a role-based access policy with CP-ABE would ensure that patient data are only accessed using user-defined attributes belonging to the patient. In using the CP-ABE, the party that encrypts the data incorporates an access control policy, based on pre-defined role-based terminology, into the encryption to ensure that only those recipients whose descriptions match the attributes in the user defined data can access the data.

Similarly, the application of secret sharing algorithms is based on the need for privacy through the use of a threshold scheme which involves the sharing of documents across a given number of parties. Access to information stored through threshold secret sharing is limited since the access can only be achieved through the availability of at least a given number of participants in the framework. A combination of two algorithms referred to as a space efficient algorithm, which allows the storage of data in transit and the secret sharing scheme, allows for encryption of the data. The space efficient algorithm is considered favorable for the storage of data that crosses over multiple users while the secret sharing scheme enhances the privacy maintenance of the system. The implementation of the entire system proposal requires that various concerns be resolved. The evaluation of the performance of the proposed architecture was based on experimentation which led to a realization of efficient performance and feasibility. Despite being limited in terms of addressing the issues of several open

tasks and failure to support multiple diverse interfaces, the study proved conclusive in obtaining a response from the PHR.

A comparison between various proposed frameworks for the use of PHRs through ABE has shown that despite being on par with recommendations such as MA-ABE and CP-ABE, the proposed combination CP-ABE and cryptographic secret sharing makes it a strong enough system for use in the multi-proxy settings where the applied data is usable across different health facilities. The authors thus conclude that the proposed framework addresses the issues of good performance and feasibility. Future research, however, is recommended for aspects of key management and role-based access control (RBAC) [62] policy management.

Guo et al. 2015. PAAS: A Privacy-Preserving Attribute-Based Authentication System for eHealth Networks. [63]

Guo et al. [63] present another framework for the use of ABE in PHR which involves only two end users. In the proposed scheme, the framework aims to maintain the conventional accuracy and convenience associated with PHRs while also addressing the need for sustainable PHR privacy through the incorporation of a Privacy Preserving Attribute-based Authentication System (PAAS). The PAAS proposed uses the verifiable user attributes to authenticate PHR users and thus protect their privacy. Through a combination of the ABE system with authentication procedures that have progressive privacy capabilities and requirements between the patients and physicians, the proposed system provides a sustainable privacy-preserving feature which is applicable in the health care sector for the handling of patient health records. The

proposed method relies heavily on the use of ABE and the authentication of user data to enhance record keeping security and data privacy.

As an operational framework, the authors suggest a system model based on ABE rather than the conventional identity-based Encryption procedure that is used in several PHRs. The system framework uses a Trust Authority (TA) for the verification of patient information and privacy preservation. This TA is responsible for the distribution of the PHR key to the users and serves as a semi-trusted center contact. The semi-trusted registration center (RC) functions as the generator in the system, whose task is to generate user credentials based on the particular attributes of the user as recorded by the system. The RC verifies the physician's professional attributes and then issues the matching credentials to the physician. As a result, the RC can frequently update the credentials as needed. In this system model, the patients can also update their credentials based on given attributes such as observed symptoms. The key advantage of this system model is that it allows the physicians and patients to interact on an anonymous basis, with the patients only giving information about their symptoms and past medical history and the doctor being assigned to cases based on their professional credentials. There is continuous interaction between patients and physicians through this system model, which enables the change of attributes due to change in observations. The patient and doctor are assumed to be within the transmission ranges for cloud data whenever required.

The proposed framework is also built on the basis of a security preservation model based on the need to preserve each user's confidential information on four key fronts. According to the design proposed by the authors, the four privacy levels include:

PAAS0, PAAS1, PAAS2 and PAAS3, which require physicians to show their valid professional qualifications in the first phase. The second phase involves the preservation of the patients' privacy through the valuation of user attributes while the third phase helps to maintain privacy between two or more interacting patients. The last phase involves the protection of all patients' information from unwarranted exposure. The operation of the proposed system involves the generation of a parameter that is used as the user's public key. This parameter is generated by the TA and can be used by any number of users. The generated parameter is thus distributed across the board to relevant professional recipients. The generated key is used for data encryption at the patient end and for the subsequent decryption of the data at the recipient's end.

While analyzing the system for performance effectiveness, the authors based their evaluation on the aspects of security and efficiency in the operations of the proposed system. Through numerical analyses, it became clear that potential security attacks on the various phases of privacy preservation demand protection of each phase independently. The effectiveness of the proposed PAAS system is based on the availability of several privacy protection phases. While the study findings are relevant and highly supported, the vulnerability associated with multiple user systems is not addressed.

Chapter 3 Preliminaries

This chapter discusses cryptographic techniques which are used in the development of solutions presented in this dissertation. In the context of computer science, cryptography is the study of securing electronic communication between parties. It involves developing techniques and protocols to prevent eavesdropping, impersonation, and message modification.

Before 1976, cryptography focused solely on message secrecy. Several algorithms and techniques were developed to offer message confidentiality using symmetric keys, where both the sender and receiver share the same key. Symmetric key cryptography techniques are described in section 3.1. In 1976, new directions in cryptography were introduced by W. Diffie and M. Hellman [64]: public key cryptosystem and one-way authentication. Public key cryptography (asymmetric cryptography), which use two keys (public and private), are described in Section 3.2.

A brief history of Attribute-Based Encryption (ABE) and its two types, along with the strengths and weaknesses of this encryption method are presented in Section 3.3. Recent cryptographic hash functions and their weaknesses are discussed in Section 3.4. The concept of trapdoor hash functions and their advantages are presented in Section 3.5.

3.1 Symmetric Key Cryptography

Symmetric key cryptography, also known as conventional encryption, refers to an encryption method in which the cryptographic key used for encrypting plaintext is the same key used for decrypting the ciphertext. This type of encryption demonstrates good performance; it is simple and fast to use, and thus used frequently. Symmetric key encryption is suitable for use where there is bulk encryption. This encryption method is classified into stream ciphers or block ciphers.

Many advantages are attributed to symmetric key encryption. In fact, symmetric key encryption uses less computer resources compared to asymmetric encryption and protects against widespread message security compromise One compromised key affects only a single pair of parties, leaving other communications secure. Also, symmetric key encryption is relatively fast to encrypt and decrypt messages [65]. On the other hand, symmetric key encryption has its disadvantages. Key sharing is the biggest problem. A secure method is needed to exchange the shared key with the other party. Thus, symmetric key encryption is more useful when encrypting one's own information than when encrypting shared data.

Additionally, when the symmetric key encryption is compromised, more damage is caused. Both parties using the shared key are affected when the encryption is compromised. The existence of too many keys poses a security challenge in managing and securing all keys. It offers privacy through the secret key sharing but cannot be used for authentication since the key can be shared by more than one person and cannot be linked to an identity. Authentication must be dealt with separately before

a symmetric key can be used. There is no guarantee of the authenticity and origin of a message. This becomes a problem when a dispute arises since the origin of the message cannot be verified due to the use of a common key between the receiver and sender [66].

The security strength of a symmetric key encryption can be measured by the time and cost needed to decrypt the message. A strong symmetric key encryption should ensure that the time taken to decrypt the message is longer than the expected lifetime of the message. Upon successful decryption, the attacker would realize that the information has already been used for the intended purpose. The decryption process will be regarded as a waste of time. Also, the cost involved in decrypting the message should be greater than the expected value of the information. The attacker will tend to avoid undertaking the attack because the process of cracking the ciphertext would have no benefit, rather, it would be a loss [67].

Attacks on symmetric key encryption include frequency analysis, where language is analyzed for the pattern of words to gather clues for attacking the ciphertext, and brute force, which occurs when one attempts to decrypt the encrypted message using every possible key. Although many attempts fail, one key usually works. This kind of attack cannot be prevented. Choosing a long key makes the attack impossible. Cryptanalysis occurs when an encrypted message is deciphered without knowing the key, but rather using a combination of sophisticated mathematics and computing power. The cryptanalyst can aim to discover the plaintext using the ciphertext or discovering the encryption key using the ciphertext.

A man-in-the-middle attack occurs when the attacker taps a secure communication and can intercept, or even substitute, the key. System based attacks refer to a type of attack where the cryptographic system is attacked instead of attacking the algorithm itself. Reverse engineering can be used to deconstruct the encryption device in order to extract the plaintext. A side-channel attack is where the cryptanalyst gains information about the physical implementation of a particular cryptographic system [68]. Next, we discuss some popular symmetric key encryption methods.

3.1.1 Rivest Cipher (RC4)

RC4 is a stream cipher type of symmetric key encryption designed in 1987 by Ron Rivest of RSA Security. It is widely used in most popular protocols like SSL (to protect internet traffic) and WEP (to secure wireless networks). The RC4 generates a pseudo-random key-stream that is combined with the plaintext using XOR operation for encryption and decryption. RC4 was used for a period of 28 years, between 1987 and 2015. The key size ranges between 64 and 256 bits.

RC4 is weak due to the inadequacies in its structure and key generation. Currently, RC4 is considered unsafe at any key size. Attacks to break RC4 began in 2013 and improved in 2015 to make defeating RC4's security practical. Statistical biases in the RC4 table were discovered in 2013, which were used, along with a large number of TLS encryptions, to recover parts of the plaintext. RC4 is not used today because many modern browsers have been designed to defeat the BEAST attacks which affected the CBC ciphers of the RC4 [65]. Its prohibition was due to attacks that weakened SSL/TLS (discussed in Subsection 3.2.4) using RC4.

3.1.2 Data Encryption Standard (DES)

The Data Encryption Standard (DES) is a symmetric key encryption block cipher developed by International Business Machines (IBM) in the early 1970's. DES was used beginning in the early 1970's and up until 1998. Its block size is 64-bit. 8-bits of the 64 are used during parity checks; thus, the key is 56-bits. For encryption, the plaintext is used as the input for the DES algorithm and the keys K_i, where $i \in \{1, ..., 16\}$ used throughout the iterations. For decryption, the input to the DES algorithm is the ciphertext, and the keys K_i are used in reverse order starting with K_{16} and going to K_1.

An attack on DES was made possible by the presence of weaknesses in the Feistel structure used in its design. DES was broken in 1998 by Electronic Frontier Foundation by trying all the possible keys using known input and output. It took 22 hours and 15 minutes to break a DES key. The maximum number of attempts to find the correct key is 2^{56}. DES is not used today because it is regarded as insecure for most applications due to its 56-bit key being too small [69].

3.1.3 Triple Data Encryption Standard (3DES)

3DES refers to a block cipher type of symmetric key encryption that is an advancement of the DES cipher algorithm. It processes the data three times. The DES algorithm is applied three times to every 64-bit block of data using three keys (K1, K2, and K3). Encryption and decryption are similar to that of DES, however, in 3DES, they are done three times. The encryption process includes encryption-decryption-

encryption while the decryption process involves decryption-encryption-decryption. The encryption algorithm is:

$$ciphertext = E_{K3}(D_{K2}(E_{K1}(plaintext)))$$

where DES uses K1 to encrypt the plaintext (first instance) and then K2 to decrypt the second instance then K3 to encrypt the third instance. The decryption is done in reverse:

$$plaintext = D_{K1}(E_{K2}(D_{K3}(ciphertext)))$$

where DES uses K3 to decrypt the ciphertext (first instance) then K2 to encrypt the second instance then K1 to decrypt the third instance.

The keys used for the encryption process are also used for the decryption process. Both the plaintext and ciphertext are 64-bits each. The key sizes can be 56,112, or 168 bits. There are three keying options for 3DES. The first keying option uses the same key (56-bits) for all encryption/decryption processes. The second keying option uses two keys (2 x 56-bits = 112-bits), where the first key is used for encrypting the first and last instances while the second key is used for decrypting the middle instance. The third keying option uses three different keys (3 x 56-bits = 168-bits), where the first key is used for encrypting the first instance, the second key is used for decrypting the middle instance, and the third key is used for encrypting the last instance.

3DES was established in 1998 and has been in use since. However, due to its poor performance in software implementations, it is not widely used. Each key is 64-bits including parity bits. The longer the key, the stronger and more secure the algorithm is. 3DES is more secure than DES. For 3DES with 2 keys, it would take 2^{112} attempts

to find the correct key, while for 3DES it would take 2^{168} attempts. Breaking 3DES is not practical and is considered more secure but very slow for software implementation, and is thus considered inefficient [70].

3.1.4 Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) encryption technique executes data in groups of 128 bits. The technique was first developed in 2001 and has since become the standard for symmetric encryption [71]. The algorithm uses a substitution-permutation network principle to eliminate the Feistel structure's weakness. The principle combines permutations and substitutions on 128-bit blocks and has key sizes of 128, 192, or 256 bits. The algorithm runs the plaintext through a number of transformation rounds to compute the ciphertext. The number of rounds is influenced by the key size. The algorithm, using 128, 192, or 256 bit keys, performs 10, 12, or 14 rounds, respectively. As the number of rounds increase, it becomes more difficult to crack the encrypted data.

Each round consists of four processes known as SubBytes, ShiftRows, MixColumns, and AddRoundKey. The SubBytes process involves the use of a lookup table to check the value that replaces each byte. The ShiftRows process contains a specific number of rows in which individual rows of the condition are shifted by a certain offset in a cyclical manner, leaving the initial row unmodified. In the MixColumns process, combination operations take place through invertible linear changes where the four bytes in each column are mixed in individual columns. Finally, in the AddRoundKey process, round keys are generated based on Rijndael's key technique and added to

individual bytes of the condition by combining round key bytes with the corresponding state byte. The final round in the algorithm does not involve the MixColumns process.

AES is used to secure many critical systems and information in many organizations all over the world. The size of the key is used to determine the time required for a successful crack of the algorithm. A large key size completely hinders the break of AES by any attack including a brute-force attack. As indicated in table 3.1, it would take much more than a billion year to crack a 128-bit AES key.

Table 3.1: Time Needed to Crack AES Keys

Key size	Time to crack
128-bits	$1.020 * 10^{18}\ years$
198-bits	$1.872 * 10^{37}\ years$
256-bits	$3.310 * 10^{56}\ years$

The effectiveness of an AES increases with the increased size of the encryption key [72]. Moreover, the AES encryption method has never been breached by brute force attacks. In summary, AES is able to deter attacks and is currently impractical to crack [73]. Thus, the Advanced Encryption Standard is the best symmetric encryption to implement. However, researchers are currently working to determine the potential possibility of side-channel attacks [68].

3.2 Public Key Encryption

Public key encryption uses a pair of keys, a public key and a private key, generated in such a way that the private key cannot be derived from the public key. The public key is distributed publicly while the private key is kept secret. The two keys are related in that the message encrypted using a public key can only be decrypted using the corresponding private key. Public key cryptography achieves confidentiality, authentication, and non-repudiation. To achieve confidentiality on this platform, data is encrypted using the receiver's public key and can only be decrypted by the party with the corresponding private key. In order to attain authentication, the data is encrypted with the sender's private key such that the sender is authenticated when the receiver successfully decrypts the data with the corresponding public key. Non-repudiation can be achieved by digital signatures (discussed in Subsection 3.2.2).

The security strength of the public key cryptography depends on the difficulty of calculating certain mathematical problems, such as factoring the product of two large prime numbers or computing discrete logarithms. Public key cryptography has advantages over other methods of encryption. It offers increased security and convenience. There is no need to exchange private keys and thus, key distribution problems are eliminated. Some disadvantages attributed to public key cryptography are that the speed of encryption and decryption are relatively low. Additionally, this method of cyptrography can be vulnerable to impersonation and man-in-the-middle attacks.

3.2.1 Rivest-Shamir-Adleman (RSA)

RSA is a public key cryptography developed by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977 [74]. Since then, it has been widely used in insecure networks, like the Internet, for securing the transmission of sensitive data. The Rabin-Miller primality test algorithm is used to generate two large prime numbers, p and q. The two prime numbers are multiplied to produce the modulus n which links both the public and private keys. The modulus is expressed in bits and forms the key length. The public key is comprised of the modulus and a public exponent e. The private key is composed of the modulus n and the private exponent d. The extended Euclidean algorithm is used to calculate the private exponent e. The multiplicative inverse, with respect to the totient of n, can be computed through this calculation. Here are the public-private key generation steps:

1. Select two prime numbers p and q.

2. Obtain the modulus n by multiplying the two prime numbers (p and q).

3. Calculate the totient of n: $\phi(n) = (p-1) * (q-1)$.

4. Choose the public key exponent e as an integer where $1 < e < \phi$ such that $gcd(\phi(n), e) = 1$.

5. Calculate the value for private exponent d using the extended Euclidean Algorithm where $1 < d < \phi(n)$ such that $ed \equiv 1 \, mod \, \phi(n)$.

The public key is (e, n) and the private key is (d, n). The values of d, p, q and $\phi(n)$ are kept secret. To encrypt a message (M) using the receiver's public key (for

confidentiality), the ciphertext is calculated as:

$$C \ = \ M^e \ mod \, n$$

and the receiver can decrypt the ciphertext (C) to retrieve M using its private key (d, n) as follows:

$$M \ = \ C^d \ mod \, n$$

Fortunately, the calculations work due to the property of modular arithmetic:

$$[\ (a \ mod \, n) * (b \ mod \, n)\ ] \ mod \, n \ = \ (a * b) \ mod \, n$$

Therefore, when M is encrypted by being raised to the power e mod n, it can be retrieved on the receiver's side as follows:

$$M \ = \ C^d \ mod \, n \equiv (M^e)^d \ mod \, n \equiv M^{ed} \ mod \, n \equiv M \ mod \, n$$

Figure 3.1 shows how the RSA encryption/decryption works over insecure channels to achieve confidentiality.

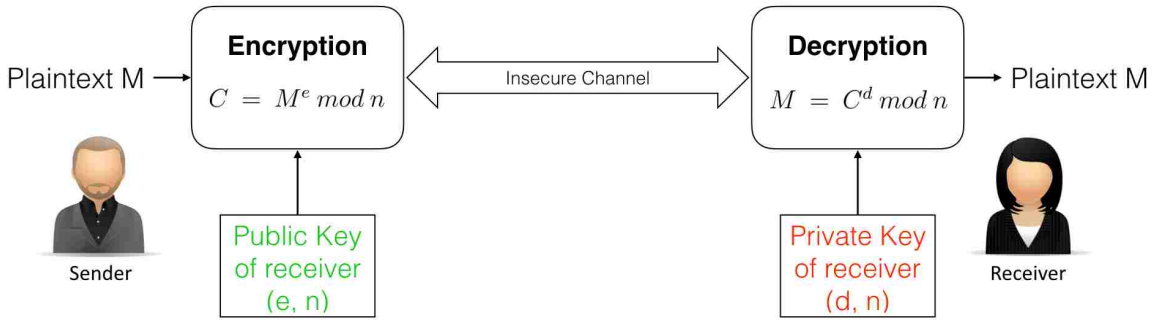

Figure 3.1: RSA Encryption/Decryption

To achieve authentication, a message M can be encrypted using the sender's private key (d, n) and the receiver (or anyone who knows the corresponding public key) can use the sender's public key to decrypt the message and ensure the authenticity of the sender. However, it is very time consuming to encrypt the whole message,

so, a more efficient way of achieving authentication is digital signatures (where the signature is applied to a digest of the original message) which is discussed in detail in Subsection 3.2.2.

Breaking an RSA that is using a sufficient key size is not practical due to the difficulty of factoring very large prime numbers. As mentioned in [1], in order to factor a 1024 bit key, it would take the Special Number Field Sieve (SNPS) and General Number Field Sieve (GNFS) factoring algorithms about 10^7 and 10^{11} MIPS-years (a million-instructions-per-second processor running for one year), respectively. In order to factor a 2048 bit key, it would take the two algorithms about 10^{14} and 10^{20} MIPS-years, respectively, as shown in Figure 3.2. In 2010, Kleinjung et al. [75] announced that they were able to factor a 768 bit key using about 1000 cores in two years. They also anticipated that, using the same hardware, it would take 7481 years to crack a 1024 bit key. To date, there is no known algorithm that can factor two large randomly chosen prime numbers, like p and q, from their product. Factoring a 1024 bit key is quite complex and would require a long amount of time to break. The current recommended key length for a secure RSA transmission is 2048 bits [76].

3.2.2 Digital Signatures

A digital signature refers to a technique that is used to validate the integrity and authenticity of software, digital documents, or a message. Digital signatures bind a document with its signer. It is used to eradicate the problem of impersonation and tampering in communications, which take place digitally. Digital signatures are used to fulfill the fundamental intentions of data security which include integrity,

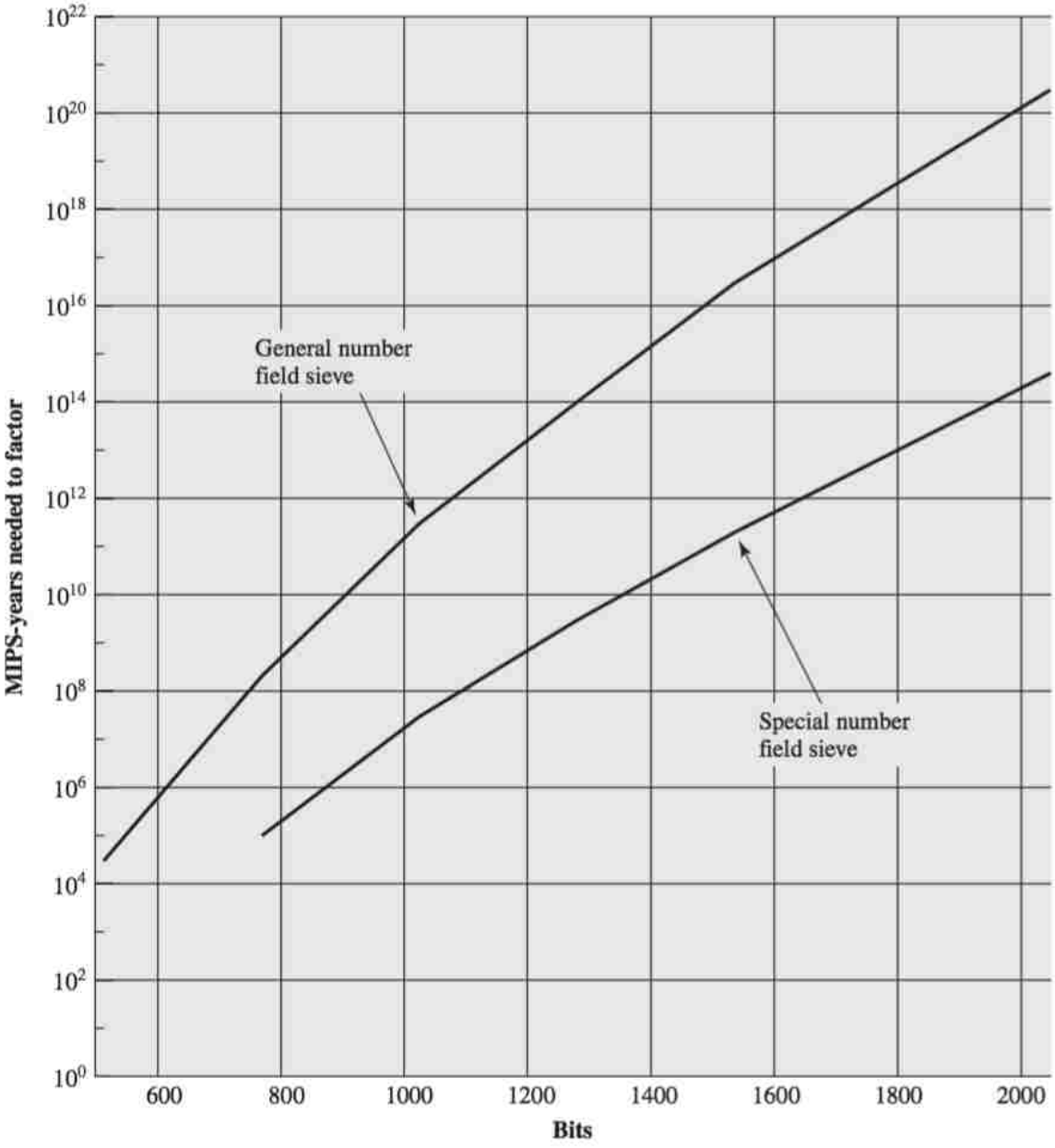

Figure 3.2: MIPS-years Needed to Factor RSA Keys [1]

authentication, and non-repudiation. Authentication is achieved through the use of corresponding private and public keys. Only the receiver with the corresponding public key can successfully decrypt the message. Integrity is achieved by comparing whether or not the hash values match to ensure that the message has not been tampered with during transmission. Non-repudiation is when the signer cannot deny having signed the document or message if their keys have not been compromised.

In order to create a digital signature, as shown in Figure 3.3, a hash (message digest) of the document to be signed is first computed. The hash value is then encrypted using the signer's private key to produce the signature. Both the message and signature are then sent to the verifier (receiver). In order to verify a digital

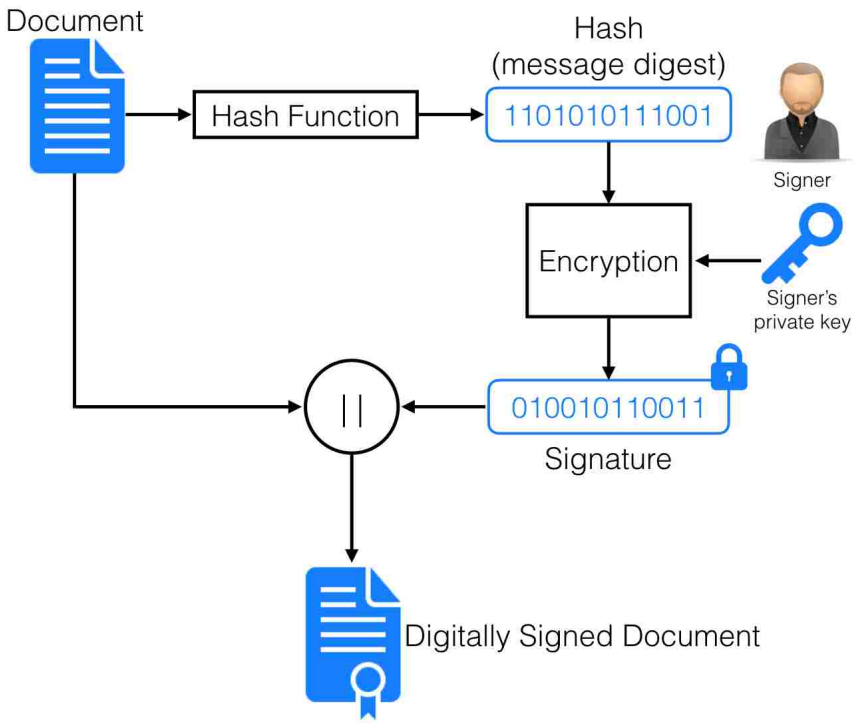

Figure 3.3: Digital Signatures Creation Process

signature, as shown in Figure 3.4, the verifier computes the hash (message digest)

of the received document and decrypts the signature using the signer's public key to verify the authenticity of the signature and retrieve the signed hash. The digital signature is considered valid if the computed hash matches the decrypted hash.

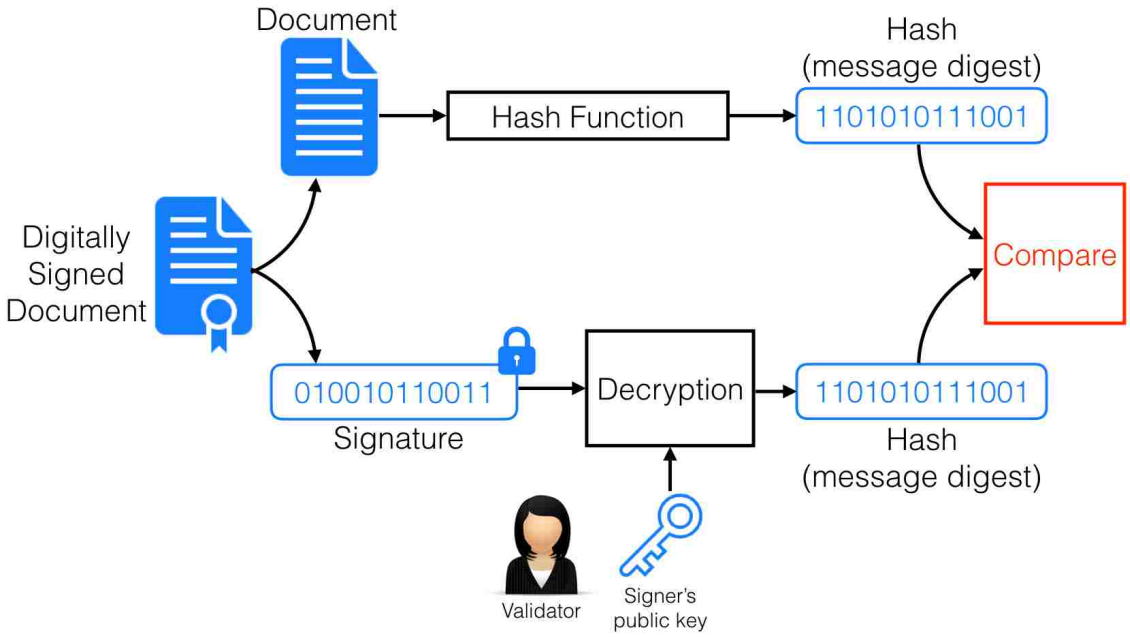

Figure 3.4: Digital Signatures Verification Process

3.2.3 Public Key Infrastructure (PKI)

Public Key Infrastructure (PKI) refers to the set of policies, roles, and procedures that have been designed to create, manage, use, store, and revoke digital certificates and manage public-key encryption. This infrastructure aims to provide authentication, non-repudiation, confidentiality, integrity, and access control to a particular set of digital data. It enables secure data exchange over networks and identity verification between the computer and user by supporting the distribution and identification of public encryption keys. It ensures authentication by assuring identity and is used for ensuring security for any sensitive data exchange.

PKI consists of software, hardware, standards, and policies used for managing the creation, administration, distribution, and revocation of digital certificates and keys. The typical elements of PKI include: a certificate authority (CA), which is a trusted party that serves as the root of trust and provides authentication services for the identities of entities like computers and individuals, a registration authority which is certified by the root CA for issuing certificates for specific uses that have been permitted by the root CA, and a certificate database for storing requests for certificates. PKI also possesses the ability to issue and revoke certificates and has a certificate cache for storing issued certificates and private keys.

A digital certificate refers to an attachment that is applied to an electronic message to ensure security. It is used to authenticate the message sender's identity and also provides a mechanism that enables the receiver to encode a reply. The sender applies for the digital certificate from a certificate authority. A CA issues digital certificates to the applicants containing their public key and other information meant for identification purposes. Each digital certificate is only valid for a specific period of time and is issued by a particular CA. A public key belonging to the CA is made available through print or over the Internet and is used by recipients to verify the digital certificate. The recipient obtains the sender's public key and identification information after verifying the digital certificate. This information is used by the recipient to encrypt messages to the certificate holder.

A chain of trust refers to the trust relationship between identities when using Subordinate Certificate Authorities to allow easy delegation of digital certificates. The signing process in a chain of trust entails signing of the intermediate certificate by

the root CA using its private key. This shows that the root CA trusts the intermediate certificate. On the next level, the Subordinate CA signs the identity certificate using its private key to authenticate that it trusts the identity certificate. A Chain-of-Trust has several problems or weaknesses. If one entity like a CA is compromised, the entire public key infrastructure security is at risk. Also, the user must assume that all entities in the chain are honest, which may not be always true.

Chain of Trust Limitations

In the overall PKI trust model, all aspects, including users, administrators, the key management server, and any third-party key management entity must be able to trust one another so that the keys and identities of those using the keys are trusted. Usually, communicating parties develop trust over time according to their communication progress. You can trust a certificate only if you can trust the CA that issued it, and you can trust that CA only if you can trust a CA above it in the chain. The validated chain then implies the authenticity of all the certificates. However, PKI authentication is vulnerable when used in initial contact due to the "Chain of Trust" concept and a lack of prior trust and communication.

As stated in [77], a certificate hardly implies the level of trust in a party's identity. It is possible to have a suspicious intermediate CA which threatens a party's authenticity. It is difficult to establish trust for the sets of public and private keys of geographically dispersed entities on intranets and the Internet. Subsequently, it is hard to completely trust the relationship between users and the CA that certified their public key and to specify the relationship in their certificate.

3.2.4 Secure Sockets Layer (SSL) and Transport Layer Security (TLS) Protocols

The Secure Sockets Layer (SSL) protocol refers to the secure communication protocol used to secure any transmission over the Transfer Control Protocol (TCP). It aims to provide privacy and reliability between two communicating applications. Privacy is achieved by encrypting the connection. Identity identification is ensured through the use of certificates. Reliability is maintained through message integrity checks by performing dependable maintenance of the secure connection.

SSL was originally developed by Hickman and Elgamal at Netscape [78]. Due to security flaws in version 1.0, it was never publicly released. Version 2.0 was released in 1995, though it too had flaws. These flaws led to the development of version 3.0 which was released in 1996. Version 3.0 was considered insecure due to its vulnerability to poodle attacks [79] [80] affecting all block ciphers in SSL. SSL version 2.0 was prohibited in 2011 [81] due to its lack of protection of handshake messages and usage of Message Digest 5 (MD5) authentication, while the prohibition of SSL 3.0 began in 2015 [82].

The Transport Layer Security (TLS) protocol guarantees data integrity and privacy between communicating applications over a network like the Internet. This protocol is comprised of two layers. The first layer is the TLS Record Protocol, which is at the top of a reliable transport protocol to ensure that a connection is secure. It encapsulates the higher-level protocols. The second layer is the TLS Handshake Protocol that authenticates the client and server. It also permits negotiation

of an encryption algorithm and cryptographic keys before data is sent or received by the application protocol.

TLS 1.0 was designed as an upgrade to SSL version 3.0 in 1999 by Christopher Allen and Tim Dierks. TLS 1.0 weakened security by including a means through which a TLS implementation could downgrade the connection to SSL 3.0. This difference prevents interoperability between the TLS 1.0 and SSL 3.0 protocols. TLS 1.1 was an upgrade of TLS 1.0 and was defined in 2006. This version added protection against poodle attacks. It replaced the implicit initialization vector (IV) with explicit IV. TLS 1.2 was defined in 2008 as an upgrade to TLS 1.1. This version introduced SHA-256 with an option of using a cipher-suite specified pseudorandom function (PRFs). TLS 1.2 was enhanced with an ability to allow the client or server to specify which hash and signature to accept. The version expanded the support for authenticated encryption ciphers. A TLS 1.3 draft has been developed to improve upon TLS 1.2.

The authentication process involves an SSL / TLS client sending a message to the SSL / TLS server. The server, in turn, responds with the information that the server needs to authenticate itself. Both the server and the client exchange session keys, which marks the end of the dialog. In mutual SSL / TLS authentication, both the client and server authenticate each other through digital certificates so that both parties are assured of the other's identity. The process of authenticating and establishing an encrypted channel using certificate-based mutual authentication, shown in Figure 3.5, involves the following steps:

1. A client requests access to a protected resource.

2. The server presents its certificate to the client.

3. The client verifies the server's certificate.

4. If successful, the client sends its certificate to the server.

5. The server verifies the client's credentials.

6. If successful, the server grants access to the protected resource requested by the client.

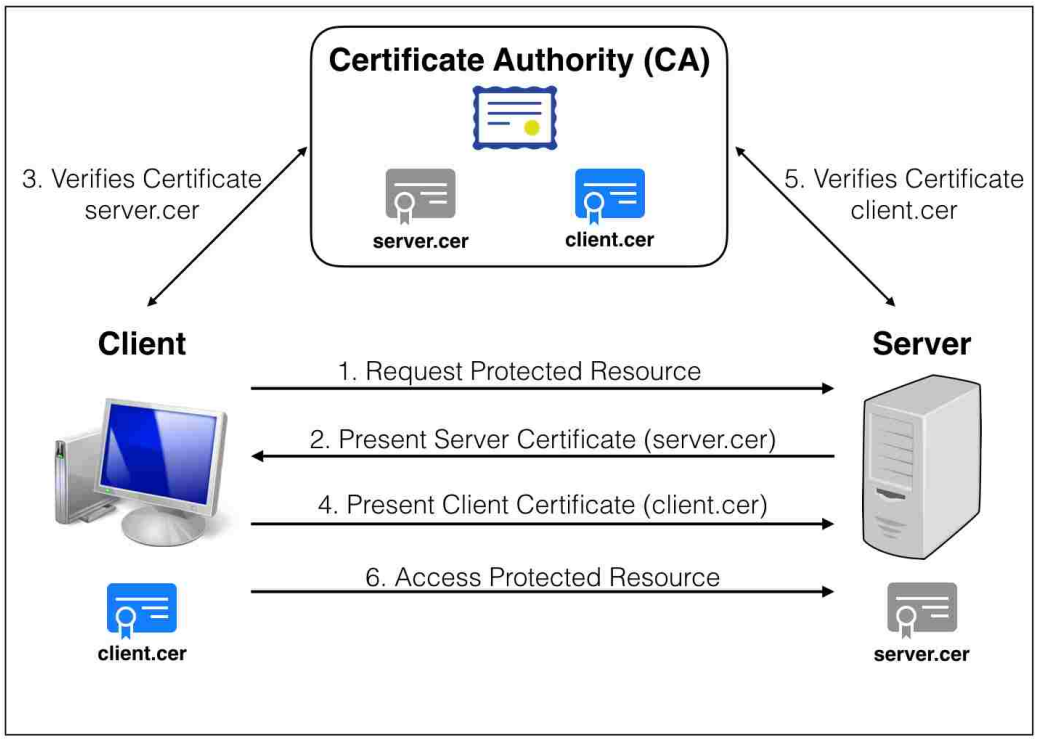

Figure 3.5: Mutual SSL / TLS Authentication Process

Man-in-the-middle Attack

In 2009, Marlinspike created two tools called "sslstrip" [83] and "sslsniff" [84] which were presented at Black Hat DC [85]. These tools do not attack the SSL protocol itself, but the transition from non-encrypted to encrypted communications. They transparently hijack HTTP traffic on a network, watch for HTTPS links and redirects, and then map those links into either look-alike HTTP links or homograph-similar HTTPS links. Marlinspike was able to make use of the previously mentioned "Chain of Trust" limitations and vulnerabilities by obtaining a valid digital certificate for a legitimate URL that he owned, and then issued a new digital certificate to an internal URL which he called "PayPal". A fake PayPal link can be created and sent to victims. When the fake PayPal link is used, the browser searches the chain of CAs until it finds the trusted issuing CA which validated Marlinspike's legitimate URL. The browser then considers the fake PayPal link valid and proceeds. This allows any attacker to read all data passing through by being a man-in-the-middle who acts as the server for the client and acts as the client for the server, reading all data passing through.

3.3 Attribute-Based Encryption

Attribute-based encryption (ABE) refers to a mode of data security where information is encoded for a particular user via various features like the country of residence of the user. This type of encryption is secure because it is collusion-resistant [86]. In ABE, ciphertexts and users' keys are tagged with diverse descriptive characteristics. For a

key to decode a certain ciphertext, the features of the user's key and ciphertext must correspond. This section will provide a brief history of ABE and will also discuss the various types of ABEs and the strengths and weaknesses of this encryption method.

ABE is a relatively a new mode of information safety. Waters and Sahai [87] were the pioneers of this system of data encryption. Later, other computer scientists like Goyal [86] built on what Sahai and Waters had established. Sahai and Waters developed an attribute-based encryption system that focused on the articulation of access threshold guidelines [88]. Later, Goyal enhanced the effectiveness of the system by introducing a mechanism that made it possible for a private key to operate with diverse attributes.

ABE can either be a key-policy (KP) or ciphertext-policy (CP). The difference between key-policy and ciphertext-policy encryption systems lies in the type of ciphertexts or secret keys attributed to the access guidelines [89]. In the KP-ABE, the admission policies are related to secret attribute keys. In the CP-ABE, the access policies are attributed to ciphertexts and both the sender and recipient have a common secret key.

3.3.1 Key-Policy Attribute-Based Encryption (KP-ABE)

In key-policy attribute-based encryption (KP-ABE), the correspondent labels the ciphertexts with a collection of expressive attributes [90]. Conversely, a dependable attribute authority generates the user key. This mode of data encryption is used mostly in structured institutions that wish to conceal information from certain parties. KP-ABE is frequently used in forensic investigations. The system helps to ensure

that the forensic analysts have the right to use data that concerns only their areas of investigation. Figure 3.6 below shows how KP-ABE systems work.

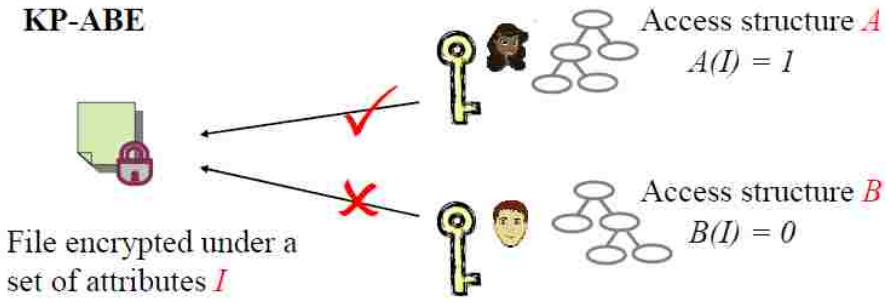

Figure 3.6: How KP-ABE Systems Work [2]

3.3.2 Ciphertext-Policy Attribute-Based Encryption (CP-ABE)

In ciphertext-policy ABE systems, the sender sets the rules regarding the access of varied attributes included in the ciphertext. The sender determines the receivers who have the authority to decipher the ciphertext. In this system, the users have a collection of attributes and the attribute authority issues matching secret attribute keys to the appropriate users [91]. To decode a ciphertext, the attributes must correspond to the admission guidelines associated to the ciphertext. Figure 3.7 below shows how CP-ABE systems work.

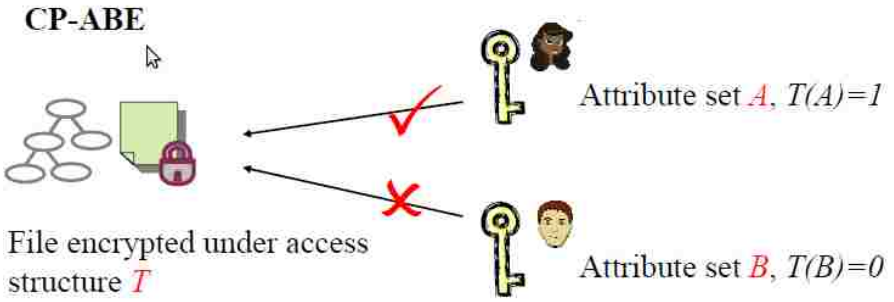

Figure 3.7: How CP-ABE Systems Work [2]

3.3.3 Attribute-Based Encryption Strengths

The main strength of attribute-based encryption is that it minimizes data loss in the event of an attack. The encryption system makes it difficult for an attacker to compromise the stored information [92]. Therefore, the user is guaranteed that the data remains secure and accurate even after the attack. Another strength of attribute-based encryption is that it helps to establish the missing link. Whenever the duties of a user change, the ABE system changes his/her attributes. Therefore, ABE systems ensure that a user does not access unauthorized information [93]. The system guarantees the consistency of information access and safety of data by matching user attributes with access policies.

3.3.4 Attribute-Based Encryption Weaknesses

In spite of the ABE system being an authoritative and promising technique, it is associated with numerous shortcomings. The weaknesses include, but are not limited to, inefficiency, key revocation and coordination challenges, and a lack of attribute revocation methods. The opponents of attribute-based encryption systems claim that the method is not efficient due to the nature of the decryption algorithm. The decryption process demands "double pairings for each leaf of the access tree that is matched by a private key attribute and at most one exponentiation for each node along a path from such a leaf to the root" [94]. Besides inefficiency in the decryption algorithm, an ABE system does not have attribute revocation techniques. There are no mechanisms to determine the expiration date of the attributes. Consequently, it

is difficult for the key authority to control or revoke attributes that are no longer in use. Furthermore, it becomes challenging for some users to decrypt information when they lack important private keys.

As stated above, an attribute-based encryption system does not have a mechanism for key revocation. The system does not allow a sender to determine whether the receiver has been withdrawn. Moreover, numerous recipients may have identical decryption policies, making it difficult for the key authority to determine the correct key to retract [95]. The fact that multiple users may have similar decryption policies makes it challenging for the ABE system to have an efficient key coordination technique. The current ABE system uses attributes that do not support numerical values. Furthermore, the access policies do not execute integer assessments. Hence, it is challenging for the system to efficiently manage users' keys.

Numerous factors affect the performance of attribute-based encryption systems. They include the security level, number of attributes, and the quality of the device being used. The execution time of ABE relies on the number of attributes [96]. On average, the ABE encryption operation takes about four seconds. On the other hand, the key generation process takes about two seconds. An increase in security level results in an increase in execution time. The attribute-based encryption system involves a pairing calculation that helps to match private keys with particular attributes.

The system entails various operations. They include encryption, key generation, and decryption operations [97]. Encryption is comprised of an algorithm that transforms a message, public parameters, and a collection of attributes into ciphertexts. The key generation process constitutes an algorithm that uses public parameters, a

master key, and an access structure to generate the decryption key. On the other hand, the decryption operation consists of an algorithm that uses the decryption key to decode a ciphertext to obtain the original message. The operations of attribute-based encryption are complex in that they entail trade-offs according to private key sizes and ciphertexts [98]. Additionally, the operations entail exponentiations that depend on the number of attributes.

3.4 Hash Functions

A hash function takes a variable string of characters and turns it into a fixed length string called the digest. It is used to preserve data integrity either where the data is stored or when it is traveling through computer networks. Cryptographic hash functions are non-invertible (1-way), which means that given a hash value, one cannot find the original message. The simplest hash function is one that takes any string and returns a 1 if the input has an even number of characters or a 0 otherwise. The main application of the hash function is to promote integrity. Messages or hash values are verified to ensure that no unauthorized modifications have occurred. Integrity is a necessary quality of any organization's sensitive data. The integrity verification involves comparing the hash values before sending and after any event that occurs. Digital signatures make use of cryptographic hash functions to verify the validity of hashed messages before being signed.

The problem with this hash function is that two different messages can produce the same digest. This is called a collision. To correct the problem, many popular and more useful cryptographic hash functions have emerged. The security strength

of cryptographic hash functions is evaluated using its collision resistance, properties needed by the application, and preimage resistance. If it is difficult to create a collision, then the hash function becomes collision resistant [5]. The following is an explanation of the hash functions adopted in recent history, as well an overview of their advantages and disadvantages.

3.4.1 Message Digest, Version 5 (MD5)

MD5 is a message digest algorithm developed in 1991 by Ronald Rivest. MD5 works through a series of steps. The input is first divided into blocks of 512 bits each. At the end of the final block, 64 bits are added. All blocks are separated into 16 words containing 32 bits each. Processing of the blocks then follows, which involves mixing buffers with words from the input for a specific number of rounds. The final output of size 128 bits is produced after all rounds are complete.

The algorithm used by MD5 for computation is fast and therefore high speed is achieved in this method. Also, collision resistance was considered the algorithm's main strength. Moreover, MD5 is popularly known and thus receives wide application and acceptance. Furthermore, it makes use of a one-way hash, making it difficult to reverse the process. However, several flaws and vulnerabilities in MD5 give threats a chance to exploit the system. In 2005, Wang and Yu [99] showed that it is theoretically possible to find hash collisions in MD5 and other hash functions. In 2007, Stevens et al. [100] were able to construct two X.509 certificates [101] with different name fields which have identical digital signatures. Later, in 2009, it was estimated to take one day to complete an MD5 collision using a cluster of PlayStation 3 consoles [102].

A malware called "Flame" was discovered in 2012 [103] [104], which was used to attack Middle Eastern Governments' computers running Microsoft Windows for several years. Flame exploited the weaknesses in MD5 to fake a Microsoft digital signature in order to gather data files, remotely change settings on computers, turn on computer microphones to record conversations, take screen shots, and copy instant messaging chats [105]. It is believed that Flame had been operational for five years before it was discovered.

In 2014, McHugh [106] was able to come up with two PHP scripts which behave differently but have the same MD5 hash. He edited a PHP script with a subtle difference in the first chunks of binary data. This produced the same MD5 hash when passed though the whole MD5 algorithm, including padding. Afterwards, he was able to create a chosen prefix collision attack to come up with two totally different images which have exactly the same MD5 hash [107]. A few weeks later, McHugh was able to find a third image which had exactly the same MD5 hash [108]. As a result, attackers can create many input sources which map to the same hash value through the collision weakness. MD5 security is highly compromised and hence is not recommended for use in securing a system or sensitive data.

3.4.2 Secure Hash Algorithm, Version 0 (SHA-0)

The requirements of SHA-0 were established in 1993 and the algorithm was subsequently utilized by the NSA. It was later revised in 1995. The SHA-0 algorithm receives a message (input) of size of $<= 2^{64}$ bits. The input undergoes several transformations or rounds where the bits are altered. The final output is a message digest

of 160-bits. SHA-0 has weaknesses associated with its vulnerability to attacks. The attacks were due to the existence of flaws in the initial code. Moreover, the algorithm faces collision issues.

In 1998, the first collision attack was announced, and it was shown that collisions were evident in 2^{61} processes [109]. Later, in 2005, another attack was announced that established collisions in 2^{36} processes [110]. In 2005, SHA-0 was termed inappropriate for application in any cryptography transaction [111]. More attacks were announced afterwards. In 2008, it was announced that SHA-0 collisions could be found in about one hour using a standard PC [112]. The recent attacks are classified as existential forgery [113]. As a result, any application of the SHA-0 algorithm is not recommended because of its vulnerability to attacks.

3.4.3 Secure Hash Algorithm, Version 1 (SHA-1)

SHA-1, a revision of the much weaker SHA-0, is similar in principle to MD5, but with a more conservative implementation. It was published in 1995 by the National Institute of Standards and Technology (NIST). The algorithm was different from SHA-0 in that the message schedule became complex. However, the round function of the two algorithms was similar. NSA revised the SHA-0 to SHA-1 with the intent to correct the errors of the SHA-0 algorithm.

The SHA-1 algorithm begins by creating sub-registers of the initial 160-bit register. It then proceeds through a sequence of iteration of the individual 512-bit message digest. A series of rounds relating to four intervals then follows. Twenty iterations are involved in the process with the input and blocks being operated by the rounds

throughout. After the rounds, the other 160-bit register is included in the initial 160-bit register. The outcome for SHA-1 is a 160-bit hash value.

There have been several attacks against SHA-1. The first attack was announced in early 2005, which explained a theoretical attack to find collisions in 2^{80} operations [114]. In the same year, collisions were witnessed in 2^{69} processes [115]. In 2013, Microsoft announced that it will not accept SSL certificates using SHA1 after January 1^{st}, 2017 [116] [117]. The SHA-1 algorithm was fully broken in February 2015 after researchers concluded that they found collisions in the algorithm [118]. The developers immediately concluded that use of the algorithm was not advised.

3.4.4 Secure Hash Algorithm, Version 2 (SHA-2)

The SHA-2 algorithms were introduced in 2001 as a draft and were approved in 2002 [119]. Several changes and updates to SHA-2 were published in 2004, 2008, 2011, and 2012. SHA-2 is a family of four similar hash functions with varying digest lengths: SHA-224, SHA-384, SHA-256 and SHA-512. The SHA-2 algorithm accepts input with 64-bit or 128-bit sizes. Three operations take place in the SHA-2 algorithm. They include right rotation and exclusive and modular addition. The output message digest (hash) size can be 224, 256, 384, or 512 bits, according to the SHA-2 version used.

The algorithm is not compatible with the older versions of browsers and other applications. For instance, early versions of Windows like XP are not compatible with this algorithm. As a result, the SHA-2 algorithm has not achieved a wide application. Moreover, the SHA-2 application gives rise to problems in websites.

Many customers make use of the algorithm although they may face the associated compatibility issues.

In 2009, a pre-image attack on 46 steps of the hash function, which employed the meet-in-the-middle technique, was published [120]. Indesteege et al. [121], as well as Sanadhya and Sarkar [122], published a 24 steps practical collision attack on the SHA-512 hash function. Last year, Dobraunig et al. [123] were able to practically demonstrate collision examples in SHA-2. Adoption of the algorithm has been challenging, although advancements are being designed and tested to ensure that all future attacks are prevented.

The need for an alternative algorithm led to the establishment of SHA-3. In late 2012, NIST announced that Bertoni et al. [124] won the SHA-3 competition. NIST released the final standard version of SHA-3 in August 2015 [125]. The algorithm is different from SHA-0 and SHA-2 in that its design is unique. It employs a sponge construction design, wherein data is absorbed, and then the output is squeezed out and has an arbitrary size [124]. The new SHA-3 is assumed to be secure and it is hoped that the arbitrary size message digest can stop, or slow, collision attacks.

3.5 Trapdoor Hash Functions

A trapdoor hash function [126] is a collision resistant hash function with a trapdoor (a secret key) for finding collisions. The concept of a trapdoor hash function was originally derived from the notion of trapdoor commitments proposed by Brassard et al. [127]. More formally, a trapdoor hashing scheme `TH` consists of the tuple ($ParGen$, $KeyGen$, TH, $TrapColGen$), which is described next.

ParGen: A probabilistic polynomial time (PPT) algorithm for parameter generation that takes a security parameter 1^λ as input and outputs system public parameters params.

KeyGen: A PPT key generation algorithm that takes params as input and outputs a trapdoor and hash key pair (TK, HK).

TH: A trapdoor hash function that takes params, HK, a message m, and a random element r as inputs and outputs the digest of m denoted as $TH_{HK}(m, r)$.

TrapColGen: A collision-finding algorithm that takes params, TK, m, r, and an additional message $m' \neq m$ as inputs and outputs a collision parameter r' such that $TH_{HK}(m, r) = TH_{HK}(m', r')$.

Figure 3.8 [128] shows the computation of the trapdoor hash of m to get a digest h, along with the computation of the collision parameter r' for m', which, when used for computing the trapdoor hash of m', leads to the same digest h.

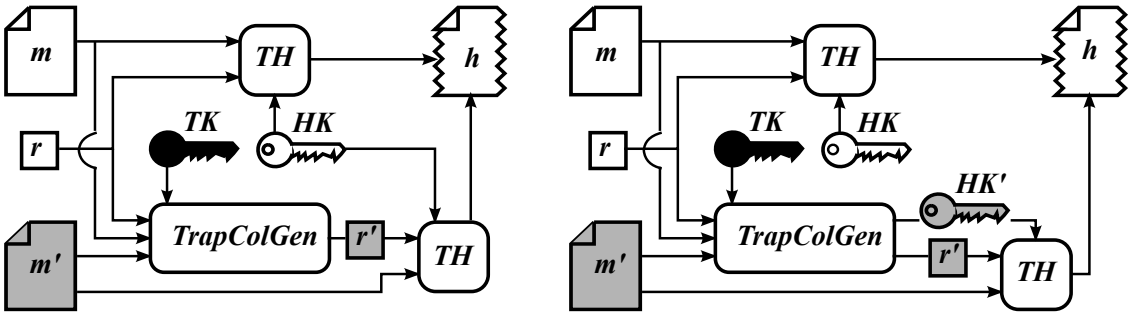

Figure 3.8: Single- and double- trapdoor hash functions. For the case of double-trapdoor hash function, $HK = (HK_l, HK_e)$ and $HK' = (HK_l, HK'_e)$

Krawczyk et al. [126] used trapdoor hash functions (referred to as chameleon hash functions) to construct a non-interactive non-transferable signature scheme, called

chameleon signatures (closely related to undeniable signatures), under the hash-and-sign paradigm. Chameleon signatures allow a signer to undeniably commit to the contents of a signed document, but do not allow the recipient of the signature to disclose the signer's commitment to a third party without the signer's consent. The trapdoor hash function employed by Krawczyk et al. suffers from the key exposure problem that allows anyone with the knowledge of a collision to compute the private trapdoor key. Ateniese et al. [129] partially addressed this problem by introducing an identity-based trapdoor hashing scheme that uses a new key pair for each collision computation. Using this scheme, a collision only leads to the to the exposure of a single trapdoor key that was used for computing that particular collision, thus preventing the exposure from affecting other collisions. Later, Chen et al. [130] and Ateniese et al. [131] proposed full constructions of trapdoor hash functions without key exposure, and provided several applications for trapdoor hashing. Additionally, double-trapdoor hashing schemes were proposed [132], which, as the name suggests, use pairs of hash keys, one long-term and the other ephemeral (or one-time).

In a double-trapdoor hashing scheme DTH, the trapdoor key TK now contains two components (TK_l, TK_e), where TK_l is long-term and TK_e is ephemeral. Similarly, the hash key is given as $HK = (HK_l, HK_e)$. The trapdoor hash of a message is computed in a fashion similar to conventional trapdoor hash functions, except that the computation also uses the ephemeral component of the hash key. Differences in collision computation are more evident. In double trapdoor hashing schemes, $TrapColGen$ takes params, $TK = (TK_l, TK_e)$, m, r, and an additional message $m' \neq m$ as inputs, and outputs a collision parameter r' and $HK' = (HK_l, HK'_e)$ such

that $TH_{HK}(m, r) = TH_{HK'}(m', r')$. Analogous to single-trapdoor hashing schemes, in double-trapdoor hashing schemes, computing the digest of a message using TH and collisions using $TrapColGen$ must be achievable in polynomial time. The function TH is said to be part of a double-trapdoor hash family $\mathcal{TH}$ described by params, where each TH is associated with a hash key $HK = (HK_l, HK_e)$.

Similar notions for collision and key-exposure resistance also exist for trapdoor hash functions that use ephemeral (or double) trapdoors [132]. Informally, collision forgery resistance implies that given params and $HK = (HK_l, HK_e)$, it is computationally infeasible to find a tuple $\langle m, r, HK, m', r', HK' \rangle$, where $HK' = (HK_l, HK'_e)$ and $TH_{HK}(m, r) = TH_{HK'}(m', r')$. Key-exposure resistance implies that given params and a tuple $\langle m, r, HK, m', r', HK' \rangle$, such that $TH_{HK}(m, r) = TH_{HK'}(m', r')$, it is computationally infeasible to find the long-term trapdoor key, TK_l corresponding to HK_l. Figure 3.8 shows the operation of a double trapdoor hash function. The function hashes a message m to get a digest h. During collision computation with a message m', the function outputs a collision parameter r' and an ephemeral hash key HK', which, when used for computing the trapdoor hash of m', lead to the same digest h.

For a trapdoor hash function to be practical, computing the digest of a message using TH, and collisions using $TrapColGen$, must be achievable in polynomial time. The function TH is said to be part of a trapdoor hash family $\mathcal{TH}$ described by params, where each TH is associated to a hash key HK. Well-known security notions associated with trapdoor hashing schemes include collision forgery resistance and key-exposure resistance [126] [131] [133]. Collision forgery resistance [126] [133] im-

plies that given params and HK, it is computationally infeasible to find a tuple $\langle m, m', r, r' \rangle$ such that $TH_{HK}(m, r) = TH_{HK}(m', r')$. Key-exposure resistance [131] implies that given system parameters, params and a tuple $\langle m, m', r, r', HK \rangle$ such that $TH_{HK}(m, r) = TH_{HK}(m', r')$, it is computationally infeasible to find the trapdoor key TK corresponding to HK.

Trapdoor hash functions find applications in the development of several novel signature schemes that include chameleon [126], online/offline [133], threshold [132], proxy [134] [135], sanitizable [136] [137], and amortized [138] signatures. More recently, Chandrasekhar et al. [139] introduced the concept of multi-trapdoor hash functions, which allow multiple entities to compute a collision with a given trapdoor hash value, with applications in query authentication of cloud-based storage systems involving multiple entities [140] and aggregate signcryption schemes [128].

3.6 Proxy Signatures

The concept of a proxy signature was introduced by Mambo et al. [141] in 1996. Proxy delegation is a process by which an entity, the delegator, transfers its signing rights and capabilities to another entity, the proxy. Following delegation, the proxy can generate signatures, called proxy signatures, on behalf of the delegator. Mambo et al. [141] classified proxy delegation into partial delegation, full delegation and delegation by warrant, presented possible constructions for proxy signatures, and provided an informal security analysis. As opposed to full and partial delegation which require a secure channel and absolute trust on the proxy, the delegation-by-warrant approach eliminates these impractical requirements [142] [143]. This is typically done by using

the warrant as a message space descriptor, and requiring all messages that the proxy signs to comply with the warrant.

In a typical delegation-by-warrant, the delegator generates a signature on the warrant and sends the signed warrant to the proxy. The proxy generates signatures on messages that comply with the warrant, and includes the signed warrant in the resulting proxy signatures. Any entity wanting to verify a proxy signature must check the validity of the proxy signature as well as delegator's signature on the warrant (indicating the agreement on the signed message). Later classifications of proxy signatures included partial delegation by warrant [144] and strong/weak proxy signatures [145].

Since the introduction of proxy signatures, researchers have focused on developing new schemes that enhance the security and/or efficiency of previous schemes [146] [147] [148] [149] [150] [151] and developing extension/variations of proxy signatures [143]. Proxy signatures have been found to have numerous applications in distributed systems, grid computing, mobile agent applications, etc. For an extensive list of related work see [143]. More recently, researchers have focused on formal security notions for proxy signatures, as well as on the development of provably secure proxy signatures [143] [152] [153].

Chandrasekhar et al. [134] proposed a technique to construct proxy signatures using trapdoor hashes, along with a specific discrete log-based scheme that improves upon prior schemes in terms of complexity, security, and efficiency. The technique for generating proxy signatures using trapdoor hashes is as follows. The delegator generates a trapdoor hash of the warrant, signs the hashed warrant, and sends the (warrant, signature) pair to the proxy over an insecure channel. The proxy does not

generate a signature on the chosen message in the traditional sense on behalf of the delegator. Instead the proxy uses its trapdoor key, known exclusively to itself, to find a collision between the trapdoor hash of the warrant and the given message. The proxy tags the result of the collision along with the delegators signature, warrant and message to collectively generate the proxy signature. The technique guarantees strong unforgeability, verifiability, strong identifiability, strong undeniability, and prevention of misuse [134].

More formally, a trapdoor hash-based proxy signature scheme `TPS` is the tuple ($ParGen$, $KeyGen$, $SigGen$, $SigVer$, $\langle Delegate, Accept\rangle$, $PSigGen$, $PSigVer$) that is described as follows:

ParGen: A PPT algorithm that takes a security parameter 1^λ as input and outputs system public parameters params.

KeyGen: A PPT algorithm that takes params as input and outputs a (private, public) key pair (SK, PK) and a (trapdoor, hash) key pair (TK, HK).

SigGen: A PPT algorithm that takes params, SK, and a message m_p as inputs and outputs a signature σ on m_p.

SigVer: A deterministic algorithm that takes params, PK, σ and m_p as inputs and outputs *Valid* if σ was generated on m_p using SK and *Invalid* otherwise.

Delegate, *Accept*: A pair of interactive algorithms for proxy delegation that are defined as follows:

Delegate: A PPT algorithm that takes params, SK, and a warrant w (that contains a message space descriptor as defined in [143] along with the identities of delegator and proxy) as inputs, and computes a delegation certificate *cert* containing the following: (1) A random element r; (2) A signature σ generated using SK on $TH_{HK}(w, r)$. *Delegate* has no local output. *Delegate* will interact with *Accept* for delegation of signing rights.

Accept: A deterministic algorithm that takes params, *cert*, w, PK and HK as input, and outputs $\langle \mathit{Valid}, \mathit{cert} \rangle$ if: (1) w conforms to agreement between delegator and proxy, and (2) σ is a valid signature on $TH_{HK}(w, r)$ under PK; and $\langle \mathit{Invalid}, \perp \rangle$ otherwise.

PSigGen: A PPT proxy signature generation algorithm that takes params, *cert*, w, a message m, and TK as inputs and outputs a proxy signature $\sigma_P = \langle \mathit{cert}, r', HK' \rangle$ on m complying with warrant w, where $TH_{HK}(w, r) = TH_{HK'}(m, r')$.

PSigVer: A deterministic algorithm that takes params, PK, HK, w, m and σ_P as inputs and outputs *Valid* if: (1) *Accept*(params, *cert*, w, PK, HK) returns $\langle \mathit{Valid}, \mathit{cert} \rangle$; (2) m complies with w, and (3) $TH_{HK}(w, r) = TH_{HK'}(m, r')$; and $\langle \mathit{Invalid}, \perp \rangle$ otherwise.

Chapter 4 A Secure Solution to Maintain Patients' Privacy in Health Information Exchange

The objective of this chapter is to discuss the development of a secure solution for HIE that ensures patients' privacy. The solution enables patients, at a healthcare organization, to retrieve their health information, in entirety or partially, from a remote healthcare organization. It allows the healthcare organization holding the health information to authenticate patients and requesting healthcare organizations before processing an information exchange request. The proposed HIE solution would allow patients to authorize the requesting healthcare organizations to retrieve certain type of health information (e.g., Cardiovascular) located at their home healthcare organization. Requested health information is encrypted using a symmetric key generated simultaneously at the involved healthcare organizations and sent to the requesting healthcare organization.

In Section 4.1, the system model for the solution is presented. Section 4.2 discusses the protocol to exchange health information while maintaining patients' privacy. The security analysis of the privacy protocol is discussed in Section 4.3. Section 4.4 describes the evaluation of the message exchange overhead in the protocol.

4.1 The System Model

This section presents the conceptual model of the HIE privacy system and describes the representation and organization of the system. The general architecture, shown

in Figure 4.1, consists of three main elements: the National Medical Certification Authority (NMCA), Healthcare Organizations (HCOs), and Patients. The structure and capabilities of the National Medical Certification Authority (NMCA) are described and discussed in subsection 4.1.1. Healthcare organizations' structure and capabilities are described and discussed in subsection 4.1.2.

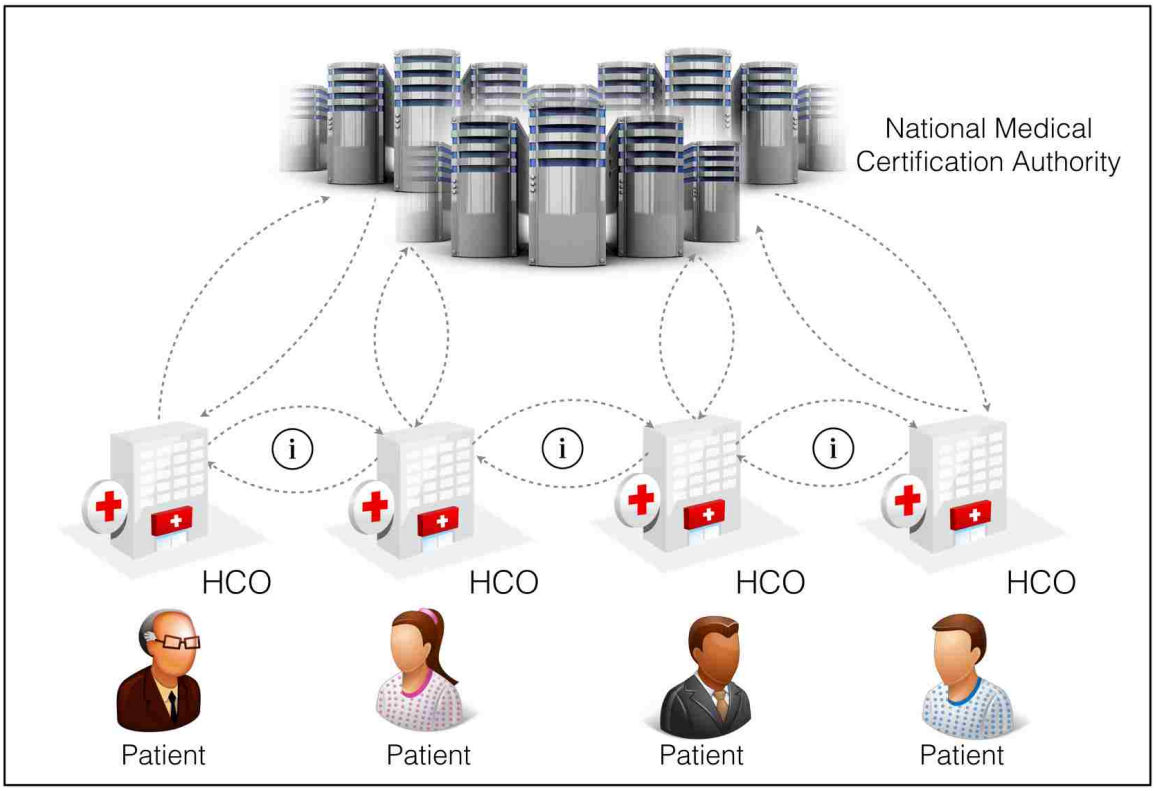

Figure 4.1: The General Architecture for the HIE Privacy Solution

4.1.1 National Medical Certification Authority (NMCA)

The NMCA infrastructure, shown in Figure 4.2, consists of:

- A *registration authority* (RA) that acts as the verifier for the certificate author-

ity before a digital certificate is issued to a requestor.

- A *certificate authority* (CA) that issues and verifies X.509 digital certificates.
- *X.509 digital certificates* which contains identity information of entities.
- A *certificate repository database* where the digital certificates are stored.
- A *certificate checker* that runs the Online Certificate Status Protocol (OCSP) [154] for obtaining the revocation status of issued digital certificates.
- A *certificate management system* which provides the tools for the NMCA to perform.

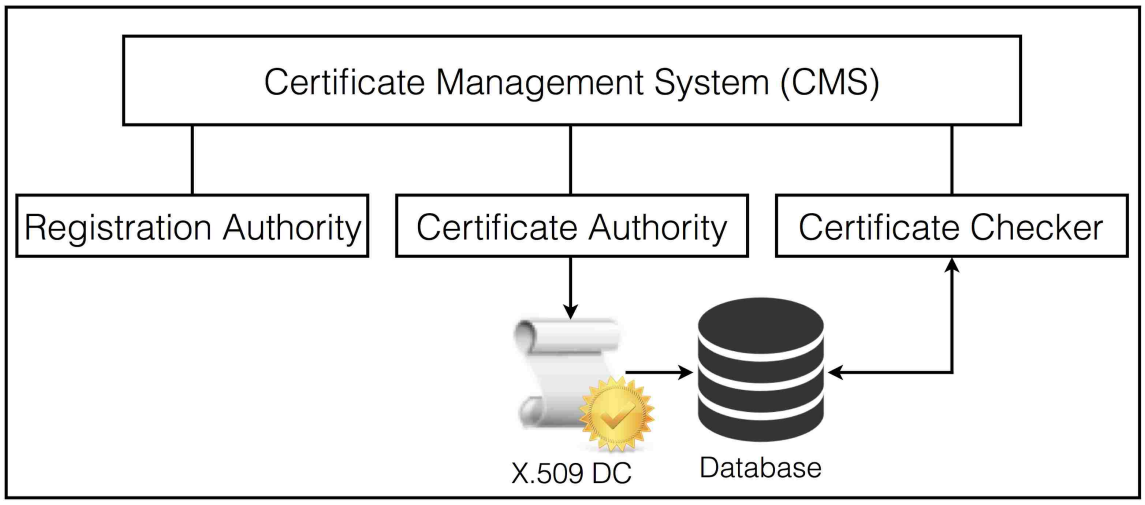

Figure 4.2: The National Medical Certification Authority (NMCA) Components

The **National Medical Certification Authority (NMCA)** is capable of validating legitimate healthcare providers and issuing X.509 digital certificates [101], with all corresponding information for the registering HCO. For the highest level of security, NMCA does not generate the public/private key pair for HCOs. Instead, the

HCOs are required to submit a valid public key during their registration and must store the corresponding private key in a highly secure place. The National Medical Certification Authority (NMCA) issues X.509 digital certificates to each healthcare organization (e.g., hospitals). The X.509 digital certificates is used by HCOs to establish authentication (discussed in subsection 4.2.1) before any patient or health information is exchanged. The X.509 digital certificates are formatted as shown in Figure 4.3.

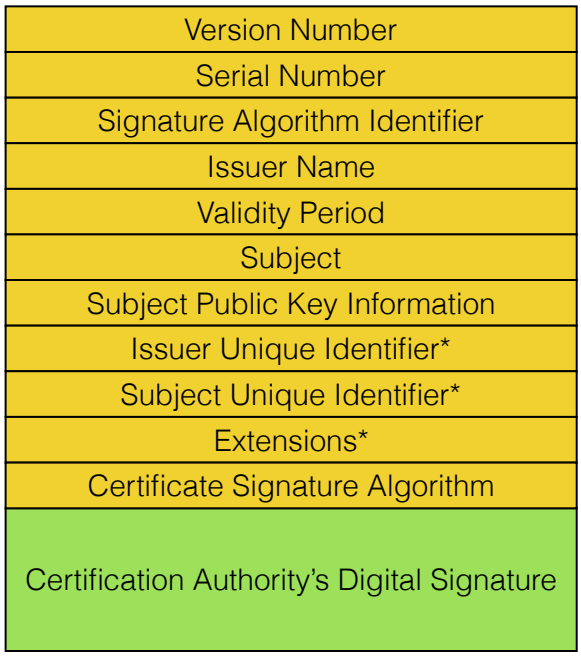

Figure 4.3: X.509 Digital Certificate Format; *: Optional Fields

4.1.2 Healthcare Organizations (HCOs)

Healthcare organizations are the most important piece of the system model as they retain patients' EHRs as well as information about people treating those patients. Each HCO's system architecture consists of a communication gateway (CG), an integration system (IS), an EHR data system (DS), and an audit system (AS) as shown

in Figure 4.4.

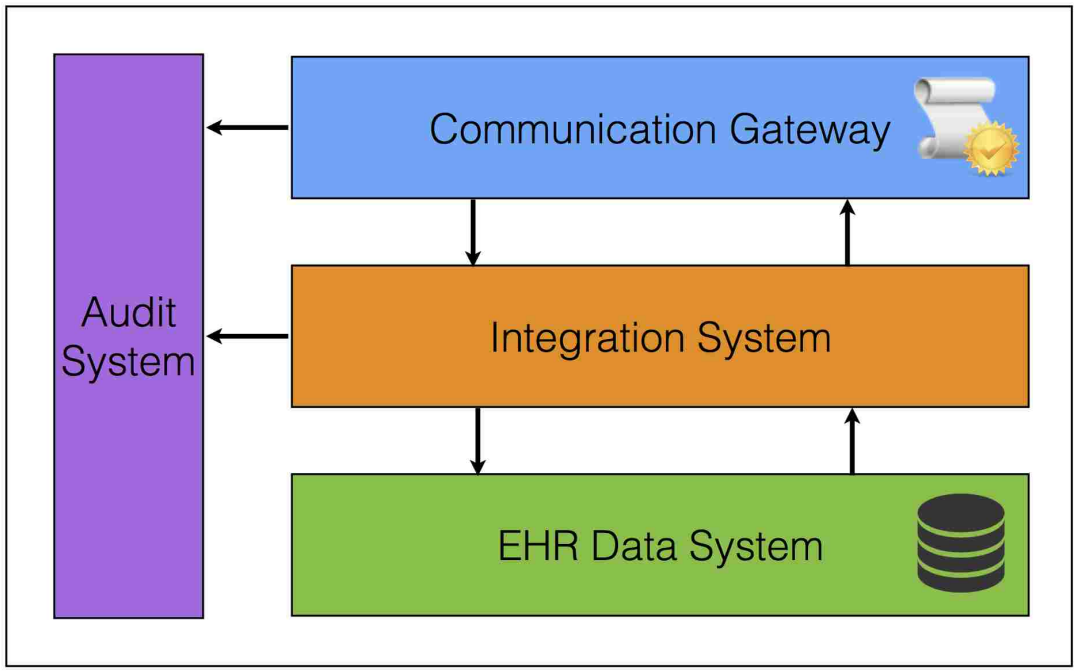

Figure 4.4: Healthcare Organizations Architecture

The **Communication Gateway** (CG) stores the HCO digital certificate registered with the NMCA. It is responsible for establishing secure communication with other HCOs. It handles mutual authentication with other HCOs, identifies patients, sets the type of health information to retrieve, and performs message exchange. The CG also updates the records in the audit system with corresponding actions.

The **Integration System** (IS) processes requests after the communication gateway completes the authentication process. On the sender side, it removes all of the patient's identity information from the requested EHRs of the requested type and then integrates the remaining data into one medical information record. On the receiver side, it splits the medical information received back into separate EHRs and adds the patient's identity information (which the receiver has) back to the EHRs.

The IS also updates the records in the audit system with corresponding actions.

The **EHR Data System** (DS) is responsible for running queries to retrieve selected patients' EHRs, adding new EHRs for patients, and handling internal encryption/decryption of stored EHRs.

The **Audit System** (AS) keeps a complete record for every incoming and outgoing request to ensure auditability in case of a breach. It receives data from the communication gateway and the integration system to build complete transaction logs. The audit system retains a log for all received records from the communication gateway and the integration system. Records are indexed using a **Request ID** which is generated by the communication gateway. The **Request ID** is agreed upon by the communication gateways for the HCOs involved in the process, using a communication protocol discussed later.

4.2 The Proposed Protocol

The protocol specifies the set of rules and measures that govern the interactions between the communicating entities in the system. This section describes the details of the protocol developed to securely exchange certain types of health information (e.g., Cardiovascular) for patient P from the current healthcare organization (Current) to the new healthcare organization (New).

We assume that patients have valid existing credentials (username/password) which they use for their accounts with every healthcare organization they use. Patients' usernames at every healthcare organization are unique, which means that no two patients have the same username. Also, each healthcare organization uses its

own hash algorithm to store patients' passwords. That is, healthcare organizations do not have to change the system they already use for maintaining their patients' credentials (username/password). Thus, any healthcare organization is eligible to run the protocol presented in this section.

First, all healthcare organizations have to register with the NMCA and obtain a valid digital certificate. Typically, when patient (**P**) is at a new healthcare organization (**New**) and wants to retrieve his/her health information, in entirety or partially, from his/her current healthcare organization (**Current**), **New** and **Current** exchange their digital certificates in order to authenticate each other. Both HCOs verify the NMCA's digital signature in received digital certificates and run the Online Certificate Status Protocol (OCSP) [154] with the NMCA to check the validity of the certificate.

Also, the patient identifies the type of health information (HI_{type}) to be exchanged and then **Current** authenticates the patient and retains a digital consent for the exchange request. **Current** prepares **P**'s health information of type HI_{type}, encrypts the health information using a symmetric key (simultaneously generated at both HCOs) and then sends it to **New**.

Table 4.1 summarizes the notations used in the following sections of this chapter. Subsection 4.2.1 presents the protocol's mutual authentication, patient authentication by **Current**, and symmetric key generation at both HCOs. The process of health information preparation and exchange is discussed in Subsection 4.2.2.

Table 4.1: Notations Used

Notation	**Description**
P	Patient
Current	Healthcare Provider Holding Patient's Health Information
New	New Healthcare Provider where patient is currently present
HI	Requested health information
HI_{type}	The type of requested health information
Current_{CG}	Communication Gateway at Current Healthcare Provider
New_{CG}	Communication Gateway at New Healthcare Provider
Current_{IS}	Integration System at Current Healthcare Provider
New_{IS}	Integration System at New Healthcare Provider
Current_{DS}	Data System at Current Healthcare Provider
New_{DS}	Data System at New Healthcare Provider
Current_{AS}	Audit System at Current Healthcare Provider
New_{AS}	Audit System at New Healthcare Provider
ID_i	Unique identifier for entity i (from digital certificate)
K_{pr}^{i}	Private key for entity i (where i can be Current or New)
K_{pu}^{i}	Public key for entity i (where i can be Current or New)
N_i	Nonce issued by entity i (where i can be Current or New)
M_n	Message with sequence number n
OCSP	Online Certificate Status Protocol
Req_{ID}	Request ID
$P_i^{username}$	Patient's username for entity i (where i can be Current or New)
$P_i^{H(password)}$	Patient's password hash for entity i (where i can be Current or New)
H_{alg}	Hash function algorithm used to store password hash
H()	Cryptographic one-way hash function
$\text{Sig}_i(M)$	Signature of message M signed using entity i's private key
$E_K(M)$	Encrypting message M with key K

4.2.1 Mutual Authentication & Simultaneous Key Generation

In this protocol, **New** begins the mutual authentication with **Current**, generates the one-time symmetric key with **Current**$_{CG}$ simultaneously, and provides the patient's consent as follows:

1. **New**$_{CG}$ generates a new request ID (Req$_{ID}$), a fresh nonce (N$_{new}$), and asks P to enter its username ($P_{current}^{username}$) for **Current** (but does not save it). **New**$_{CG}$ creates a new record in the **New**$_{AS}$, indicating the outgoing request, and saves the $P_{new}^{username}$, Req$_{ID}$, ID$current$, and N$_{new}$ for this request. **New**$_{CG}$ encrypts (Req$_{ID}$, ID$_{new}$, N$_{new}$, and $P_{current}^{username}$) using **Current**$_{CG}$'s public key $K_{pu}^{current}$ and sends it to **Current**$_{CG}$ as message M_1 as follows:

$$New \rightarrow Current : M_1$$

$$where\ M_1 = E_{K_{pu}^{current}}(Req_{ID} \,||\, ID_{new} \,||\, N_{new} \,||\, P_{current}^{username})$$

2. **Current**$_{CG}$ decrypts M_1 using its private key $K_{pr}^{current}$ to retrieve the Req$_{ID}$, N$_{new}$, and $P_{current}^{username}$. **Current**$_{CG}$ creates a new record in the **Current**$_{AS}$, indicating the incoming request, and saves the $P_{current}^{username}$, Req$_{ID}$, ID$_{new}$ and N$_{new}$ for this request. **Current**$_{CG}$ generates a fresh nonce N$_{current}$, appends it to the record at **Current**$_{AS}$, then encrypts (Req$_{ID}$, ID$_{current}$, N$_{current}$, N$_{new}$, and the hash algorithm H_{alg} it uses to store patients' passwords) using **New**$_{CG}$'s public key K_{pu}^{new} and sends it to **New**$_{CG}$ as message M_2 as follows:

$$Current \rightarrow New : M_2$$

$$where\ M_2 = E_{K_{pu}^{new}}(Req_{ID} \,||\, ID_{current} \,||\, N_{current} \,||\, N_{new} \,||\, H_{alg})$$

3. **New**$_{CG}$ decrypts M_2 using its private key K_{pr}^{new} to retrieve Req$_{ID}$, ID$_{current}$, N$_{current}$, N$_{new}$, and H_{alg}. The nonce N_{new}, included in M_2, authenticates **Current**$_{CG}$. **New**$_{CG}$ appends the corresponding record in the **New**$_{AS}$ with N$_{current}$. **New**$_{CG}$ asks P to enter its password for **Current** and hashes it using H_{alg} to create ($P_{current}^{H(password)}$) but does not save it. **New**$_{CG}$ uses the nonce received from **Current** along with the hash of the patient's password to compute the consent symmetric key K_{cons} as follows:

$$K_{cons} = (N_{current} \,||\, P_{current}^{H(password)})$$

New$_{CG}$ then creates a digital consent $Consent$ by encrypting the Req$_{ID}$ and the health information type requested HI_{type} using the consent key K_{cons} as follows:

$$Consent = E_{K_{cons}}(Req_{ID} \,||\, HI_{type})$$

and appends the corresponding record in the **New**$_{AS}$ with HI$_{type}$. **New**$_{CG}$ sends the concatenations of: (1) the Req$_{ID}$ and the N$_{current}$ encrypted using **Current**$_{CG}$'s public key $K_{pu}^{current}$ and (2) the digital consent ($Consent$) to **Current**$_{CG}$ as message M_3 as follows:

$$New \rightarrow Current : M_3$$

$$where\ M_3 = E_{K_{pu}^{current}}(Req_{ID} \,||\, N_{current}) \,||\, Consent$$

4. **Current**$_{CG}$ decrypts the first part of M_3 using its private key $K_{pr}^{current}$. The nonce $N_{current}$ authenticates **New**$_{CG}$ as it was able to retrieve the nonce using its private key. **Current**$_{CG}$ retrieves the stored hash of P's password $P_{current}^{H(password)}$) corresponding to $P_{username}$ to compute the consent symmetric key K_{cons} simul-

taneously as follows:

$$K_{cons} = (N_{current} \,||\, P_{current}^{H(password)})$$

Current$_{CG}$ then decrypts the second part of M_3 using the consent key K_{cons} to retrieve the HI_{type} which the patient **P** authorizes **New** to retrieve. **Current**$_{CG}$ appends the corresponding record in the **Current**$_{AS}$ with HI$_{type}$ then sends the $P_{current}^{username}$ and HI_{type} to **Current**$_{IS}$.

4.2.2 Health Information Retrieval (Exchange)

The protocol proceeds by preparing the requested health information type(s) and performing the exchange as follows:

5. **Current**$_{IS}$ asks **Current**$_{DS}$ to prepare all Electronic Health Records (EHRs) of type HI_{type} belonging to $P_{current}^{username}$.

6. **Current**$_{DS}$ queries all requested EHRs belonging to $P_{ID}^{current}$ from its secure database and sends them to **Current**$_{IS}$.

7. **Current**$_{IS}$ removes $P_{current}^{username}$ and P's personal information from the EHRs and integrates the remaining parts in one health information record HI. **Current**$_{IS}$ updates the corresponding record in the **Current**$_{AS}$ with the health information record HI and then sends it to **Current**$_{CG}$.

8. **Current**$_{CG}$ digitally signs the health information ($Sig_{current}(HI)$) for patient P and updates the corresponding record in the **Current**$_{AS}$ with its signature. **Current**$_{CG}$ then encrypts (Req$_{ID}$, HI, and $Sig_{current}(HI)$) using K_{cons} and sends it to **New**$_{CG}$ as message M_4 as follows:

$$Current \rightarrow New : M_4$$

$$where\ M_4 = E_{K_{cons}}(Req_{ID} \,||\, HI \,||\, Sig_{current}(HI))$$

9. **New**$_{CG}$ decrypts M_4 using the symmetric key K_{cons} and verifies the integrity and authenticity of the received HI using the $Sig_{current}(HI)$, updates the corresponding record in the **Current**$_{AS}$ with the HI and signature, then sends HI to **New**$_{IS}$.

10. **New**$_{IS}$ creates EHRs, including the patient personal information it has at **New**, from the received HI then sends the EHRs to **New**$_{DS}$.

11. **New**$_{DS}$ stores the complete EHRs to be available for internal use.

Figure 4.5 shows the flow of the protocol. Notice that the patient username ($P_{current}^{username}$) entered in step 1 and the password entered in step 3 are never stored at the **New** healthcare organization. The *Consent* generated simultaneously at each healthcare organization can be used for this session only, since the *Consent* depends on K_{cons} which is calculated using the fresh nonce $N_{current}$ issued for this session by **Current**. Also, the transferred health information reveals no personal information whatsoever about the patient's identity.

Upon completion of the protocol, both healthcare organizations have successfully authenticated each other, and the **Current** healthcare organization has authenticated Patient **P** and the type of health information requested. **Current** prepares the requested health information of type HI_{type} corresponding to $P_{current}^{username}$ and sends it

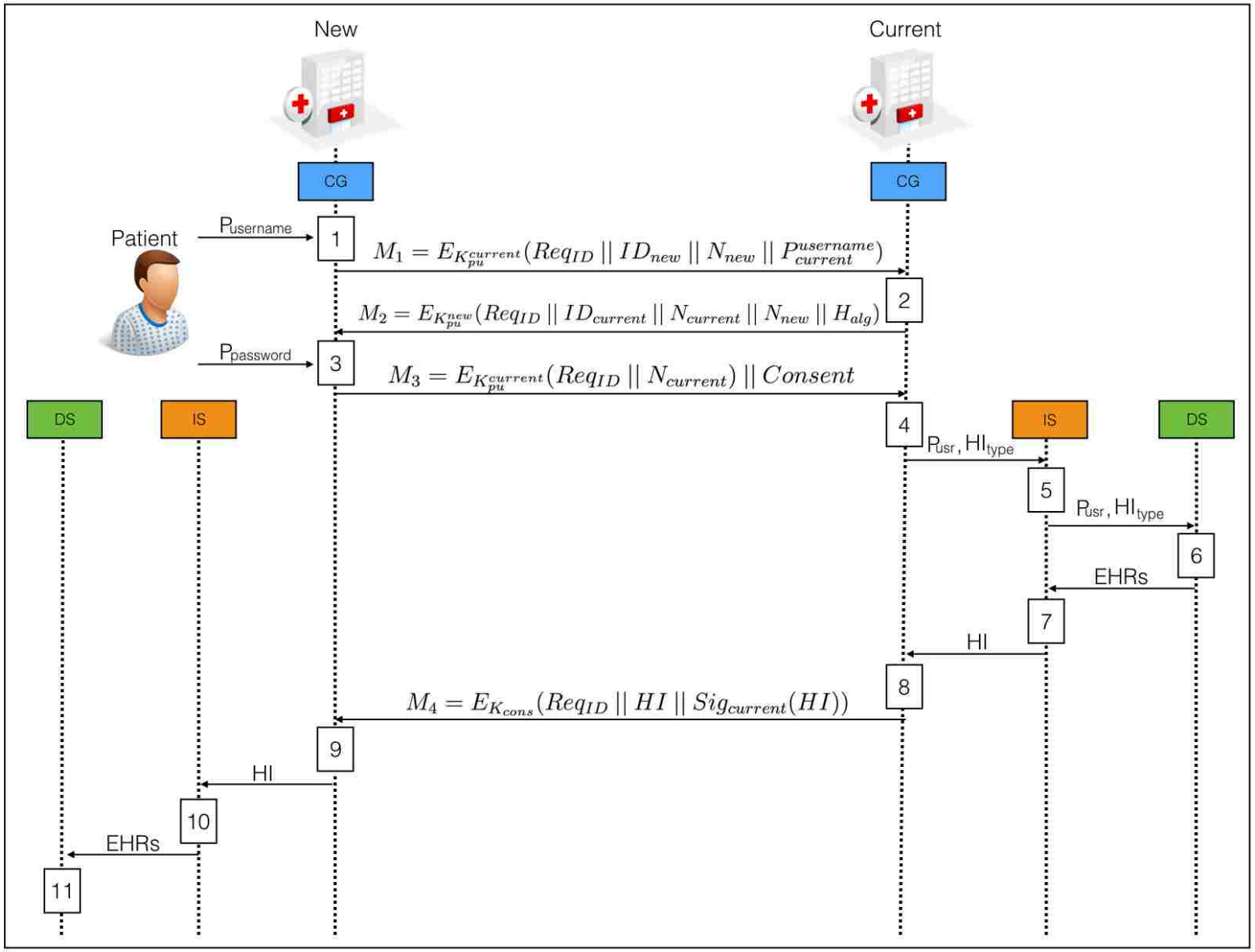

Figure 4.5: Health Information Exchange Privacy Protocol Flow

securely to Current.

The Audit System (AS) at every healthcare organization receives useful information from the Communication Gateway (CG) and the Integration System (IS). The AS at each communicating healthcare organization stores corresponding information for every incoming/outgoing request ID (Req_{ID}) for every communication to allow the matching of records in $\mathbf{New}_{AS}$ and $\mathbf{Current}_{AS}$ when needed.

A full record at $\mathbf{New}_{AS}$ is a tuple that includes the following:

"outgoing", $P_{new}^{username}$, Req_{ID}, $ID_{current}$, N_{new}, $N_{current}$, HI_{type}, HI, $Sig_{current}(HI)$

which can be read as an outgoing request number Req_{ID} to $ID_{current}$ to retrieve $P_{new}^{username}$'s health information of type HI_{type}. A full record at $\mathbf{Current}_{AS}$ is a tuple that includes the following:

"incoming", $P_{current}^{username}$, Req_{ID}, ID_{new}, N_{new}, $N_{current}$, HI_{type}, HI, $Sig_{current}(HI)$

which can be read as an incoming request number Req_{ID} from ID_{new} to retrieve $P_{current}^{username}$'s health information of type HI_{type}.

4.3 Security Analysis

In this section, we describe the security features which our protocol provides to maintain patients' privacy and comply with the technical safeguards of the HIPAA security rule. The protocol is designed to provide mutual authentication between healthcare organizations and patient authentication for current healthcare organizations. In addition, it allows only the health information of specified type (e.g., Cardiovascular) to be exchanged according to the patient's consent. It also prevents man-in-the-middle attacks and can detect replayed messages.

4.3.1 Validity of Certificates

Before authentication starts, communicating parties exchange their digital certificates (issued by NMCA) and verify NMCA's digital signature in received digital certificates. Certificate Revocation Lists (CRL) [155] have traditionally been used to check the validity of digital certificates. CRLs demonstrate a severe scalability problem as directories grow large over time and pose communication delays [156]. Our protocol uses the OCSP, as mentioned in subsection 4.2.1, rather than CRLs to check the validity of certificates since the OCSP consumes less network bandwidth and enables near real-time status checks even for high volume operations. Running the OCSP at the start of communication will detect any invalid certificates.

4.3.2 Authentication

The protocol in our framework mutually authenticates communication parties before preparing the patient's EHR for exchange. At the beginning of the protocol, nonces are exchanged between both parties to ensure that each party holds the private key corresponding to their published public key at the time of authentication. **New** authenticates **Current** when it validates that the N_{new} received in step 3 (in subsection 4.2.1) matches the N_{new} it generated and sent to **Current** in step 1 (in subsection 4.2.1). **Current** authenticates **New** when it validates that the $N_{current}$ received in step 4 (in subsection 4.2.1) matches the $N_{current}$ it generated and sent to **Home** in step 2 (in subsection 4.2.1). If any of the communicating party's nonce validation process fails, the authentication process fails and the protocol stops.

Additionally, the protocol authenticates the patient to its current healthcare provider using the patient's assigned username and the stored hash of the patient's password. **Current** identifies the patient P who should be physically present at **New** when it receives $P_{current}^{username}$ in message M_1 and then sends the hash algorithm (H_{alg}) used to store P's password to **New**. **New** asks P to enter the password for **Current**, which is hashed using the hash algorithm received. The password entered by P is never stored by **New**. The hash of P's password ($P_{current}^{H(password)}$)) is then used by **New** to compute K_{cons} as described in step 3 (in subsection 4.2.1) to create the patient's consent ($Consent$). In step 4, **Current** computes K_{cons} using the hash of P's password ($P_{current}^{H(password)}$)) stored in its system. **Current** then uses K_{cons} to decrypt the digital consent ($Consent$) in order to validate that the hash of the password used at **New** matches the one stored in **Current**'s system. **Current** authenticates the patient only after successfully decrypting the patient's consent ($Consent$).

4.3.3 Authorization

The authorization requirement aims to ensure that the patient P has authorized **New** to retrieve a certain type (e.g., Cardiovascular) of his/her own health information from **Home**. No other type of health information should be exchanged other than the type authorized by the patient. Both **New** and **Current** use the hash of the patient's password ($P_{H(password)}$) and **Current**'s nonce ($N_{Current}$) to compute K_{Cons}. **New** creates the digital consent ($Consent$) by encrypting HI_{type} using K_{Cons}. When **Current** successfully decrypts the received digital consent ($Consent$) using K_{Cons}, HI_{type} implies the patient's authorization for **New** to retrieve the patient's health

information of type HI_{type} only. In addition, the usage of $N_{Current}$ in computing K_{Cons} guarantees that the authorization is only for the current request since **Current** knows the nonce being used for each communication session.

4.3.4 Privacy

Our protocol maintains patient's privacy in two ways. First, patients authorize a certain type of health information (e.g. Cardiovascular) to be retrieved, which helps prevent an undesired or unintentionally leakage of health information. Our protocol uses K_{Cons} to encrypt the health information type (HI_{type}), which forms a digital consent ($Consent$) specifying the type of health information requested by the patient physically at **New**. When **Current** decrypts $Consent$, it prepares only the patient's health information of type HI_{type} and sends it back to **New**. No other type of health information is sent to **New** other than the type authorized by the patient.

Second, the protocol does not link patients' identity to their health information during the transmission of the exchanged health information. When a patient's health information is prepared by **Current**, the Integration System ($\mathbf{Current}_{IS}$) removes all patient identity information from the requested EHRs and then integrates the remaining data into one health information record HI. As a result, the patient's health information (HI) in M_4 (Subsection 4.2.2) reveals nothing about the patient's identity.

4.3.5 Confidentiality

To ensure the confidentiality of the health information (HI) during transmission, the symmetric key K_{Cons} is used to encrypt M_4 (Subsection 4.2.2). The symmetric key K_{Cons} is computed using the hash of the patient's password ($P_{current}^{H(password)}$) and **Current**'s nonce ($\mathrm{N}_{Current}$), which is randomly generated by **Current** for each communication session. The patient has a username and password which are used when he/she wants to communicate with **Current**. Our protocol uses the username ($P_{current}^{username}$) to identify the patient for the $\mathbf{Current}_{CG}$. The $\mathbf{Current}_{CG}$ fetches the hash of the patient's password and then generates a fresh nonce and computes the session key K_{Cons} as the concatenation of the patient's hash of the password stored in the system and the generated nonce.

The $\mathbf{Current}_{CG}$ sends the generated nonce to the $\mathbf{New}_{CG}$ in order for the $\mathbf{New}_{CG}$ to be able to generate the same session key simultaneously, but does not send any information about the patient (e.g., the hash of the password). After the $\mathbf{New}_{CG}$ receives the nonce, it asks the patient to enter his/her password corresponding to his/her account with **Current**. The $\mathbf{New}_{CG}$ hashes the patient's password and computes the session key K_{Cons} by concatenating the patient's hash of the password and the received nonce.

The password supplied by the patient in step 3 is used only to compute the hash and is never stored by **Current**. Since both ($P_{current}^{H(password)}$ and $\mathrm{N}_{Current}$) are exchanged in an encrypted form, the symmetric key K_{Cons} is simultaneously generated by **Current** and **New** and is never exchanged. Thus, it is impossible for an adversary

to find a key (K_{Cons}) that can successfully decrypt M_4.

4.3.6 Integrity and Auditability

In this solution, the protocol checks the integrity of the requested health information and can detect any possible alteration or distortion of information. **Current** computes a signature ($Sig_{current}(HI)$) of the health information sent in M_4. The integrity of the retrieved HI in M_4 is checked by **New** by comparing the hash of HI to the hash in the signature attached. If they match, it proves the integrity of the health information **HI** received.

Auditability is achieved in our framework by keeping records for all incoming and outgoing exchange requests in the Audit System (AS) for each healthcare organization. The Audit System (AS) maintains adequate information (mentioned in subsection 4.2.2) about each transaction which allows auditing at any time.

4.3.7 Man-in-the-middle Attacks (MITM)

In a man-in-the-middle attack (MIMT), an attacker impersonates each endpoint to their satisfaction as expected from the legitimate other end. The attacker can simply eavesdrop or secretly relay and alter the communication between the parties who believe they are directly communicating with each other. A MITM attack on a mutually authenticated connection can succeed only when the attacker can satisfy each party's authentication requirements while impersonating the other party. In our protocol, MITM attacks are prevented because the HCOs first exchange digital certificates issued and verified by the NMCA at the beginning of the protocol while

the rest of the protocol is designed to detect and prevent such possible attacks.

After the HCOs validate received digital certificates, an attacker cannot alter message M_1 sent from a legitimate **New** because it is encrypted using **Current**'s public key. If an attacker forges message M_1 with an invalid ID for **New**, **Current** will detect that **New**'s ID is invalid, discard that message, and stop the communication. If an attacker forges message M_1 with a fake nonce ($\mathrm{N}_{attacker}$) instead of the original N_{new}, **Current** will send the fake nonce ($\mathrm{N}_{attacker}$) to **New** in M_2 which will detect that the received $\mathrm{N}_{attacker}$ is invalid, discard that message, and stop the communication. If an attacker was able to prevent a legitimate message M_1 from reaching **Current** and tries to confuse **Current** by impersonating **New** by replaying message M_1 to **Current**, **Current** will send the reply (M_2) back to **New** because M_1 includes **New**'s ID and the attacker will gain no benefit.

Furthermore, an attacker cannot alter message M_2 sent from a legitimate **Current** because it is encrypted using **New**'s public key. If an attacker forges message M_2 with an invalid ID for **Current**, **New** will detect that **Current**'s ID is invalid, will discard that message, and will stop the communication. If an attacker forges message M_2 with a fake nonce ($\mathrm{N}_{attacker}$) instead of the original $\mathrm{N}_{current}$, **New** will send the fake nonce ($\mathrm{N}_{attacker}$) to **Current** in M_3 which will detect that $\mathrm{N}_{attacker}$ received is invalid, discard that message, and stop the communication. If an attacker was able to prevent a legitimate message M_2 from reaching **New** and tries to confuse **New** by impersonating **Current** by replaying message M_2 to **New**, **New** will send the reply (M_3) back to **Current** because M_2 includes **Current**'s ID and the attacker will gain no benefit.

In case of an insider attack, where the attacker is using a legitimate HCO trying to retrieve a patient's health information, the attacker will not be able to proceed beyond step 2 of the authentication process because the patient's password is required in order to compute the hash used to authenticate the patient.

In case of eavesdropping, an attacker capturing M_1 and/or M_2 will not gain any useful information that can help in any future MITM attacks as both messages are encrypted using the receiver's public key. Also, if an attacker was able to spoof the communication between **New** and **Current** after steps 1 and 2 to intercept M_3, the attacker will not be able to gain any information from the first part of M_3 as it is encrypted using **Current**'s public key and the second part *Consent* is encrypted using K_{Cons} (the value of which the attacker has no knowledge). The same applies for M_4 as it is encrypted using K_{Cons} as well.

4.3.8 Replay Attacks

In our protocol, nonces are generated by the communicating parties and exchanged for authentication as discussed in subsection 4.2.1. The nonces used in our protocol also allow each party to detect a replayed message. If an attacker replays an older message M_1 to **Current**, **Current** will detect that the nonce N_{new} received in M_1 has already been used and will discard M_1. If an attacker replays an older message M_2 to **New**, **New** will detect that the nonce N_{new} received in M_2 is not the same as the one sent in M_1 and will discard M_2. Similarly, if an attacker replays an older message M_3 to **Current**, **Current** will detect that nonce $\mathrm{N}_{current}$ received in M_3 is not the same as the one sent in M_2 and will discard M_3.

Since the computation of K_{Cons} depends on $N_{current}$ and $P_{current}^{H(password)}$, the $Consent$ is unique for each request. This implies that **Current** can easily detect a replayed message M_3 if the computed K_{Cons} does not successfully decrypt the $Consent$ received in M_3. If an adversary or an insider tries to attach an older $Consent$ to a legitimately encrypted nonce $N_{current}$ in M_3, **Current** will detect that K_{Cons} does not successfully decrypt the $Consent$ received in M_3 which indicates that **New** is not legitimate, is being compromised, or is trying to retrieve health information other than what it was authorized to retrieve.

4.4 Evaluation

In this section, we evaluate the performance of public/private key encryption used in the protocol and the symmetric key encryption/decryption process on messages of different size. A prototype was built using the Java SE Development Kit 8 with Java Cryptography Architecture (JCA) and Java Cryptology Extension (JCE) libraries to support the required cryptographic operations. We assume that each HCO ID (e.g., ID_{new} and $ID_{current}$), generated nonce (e.g., N_{new} and $N_{current}$), patient username, and password hash is represented with 128-bits. The hash algorithm (H_{alg}) is represented with 8-bits and HI_{type} is represented with 16-bits.

We identified the size of each message in the protocol and then ran the protocol 25 times to measure the maximum, minimum, and average execution times. We used a machine with an Intel Quad Core 2.6GHz processor as our benchmark which demonstrated that the protocol uses encryption/decryption techniques which boast quick speeds.

The well-known RSA public key cryptography algorithm was used to generate 2048-bit public-private keys for each healthcare organization. After running 25 encryption processes of message M_1's payload (384-bits) using $\mathbf{Current}_{CG}$'s public key $K_{pu}^{current}$ (2048-bits), the minimum execution time was 125 milliseconds, the maximum execution time was 187 milliseconds, and the average execution time for the 25 runs was $\approx$ 145 milliseconds as shown in Figure 4.6. After running 25 decryption processes on the encrypted M_1 using $\mathbf{Current}_{CG}$'s private key $K_{pr}^{current}$ (2048-bits), the minimum execution time was 31 milliseconds, the maximum execution time was 62 milliseconds, and the average execution time for the 25 runs was $\approx$ 41 milliseconds as shown in Figure 4.7.

After running 25 encryption processes of message M_2's payload (392-bits) using $\mathbf{New}_{CG}$'s public key K_{pu}^{new} (2048-bits), the minimum execution time was 125 milliseconds, the maximum execution time was 172 milliseconds, and the average execution time for the 25 runs was $\approx$ 146 milliseconds as shown in Figure 4.8. After running 25 decryption processes on the encrypted M_2 using $\mathbf{New}_{CG}$'s private key K_{pr}^{new} (2048-bits), the minimum execution time was 31 milliseconds, the maximum execution time was 47 milliseconds, and the average execution time for the 25 runs was $\approx$ 36 milliseconds as shown in Figure 4.9.

The concatenation of the password hash and nonce results in a consent key (K_{Cons}) with a total length of 256-bits. The Advanced Encryption Standard (AES) was used for encryption/decryption using the generated 256-bit consent key. The encryption process of message M_3's payload (144-bits) using $\mathbf{Current}_{CG}$'s public key $K_{pu}^{current}$ (2048-bits) to encrypt the $\mathrm{N}_{current}$ and using K_{Cons} to encrypt the Req_{ID} and HI_{type},

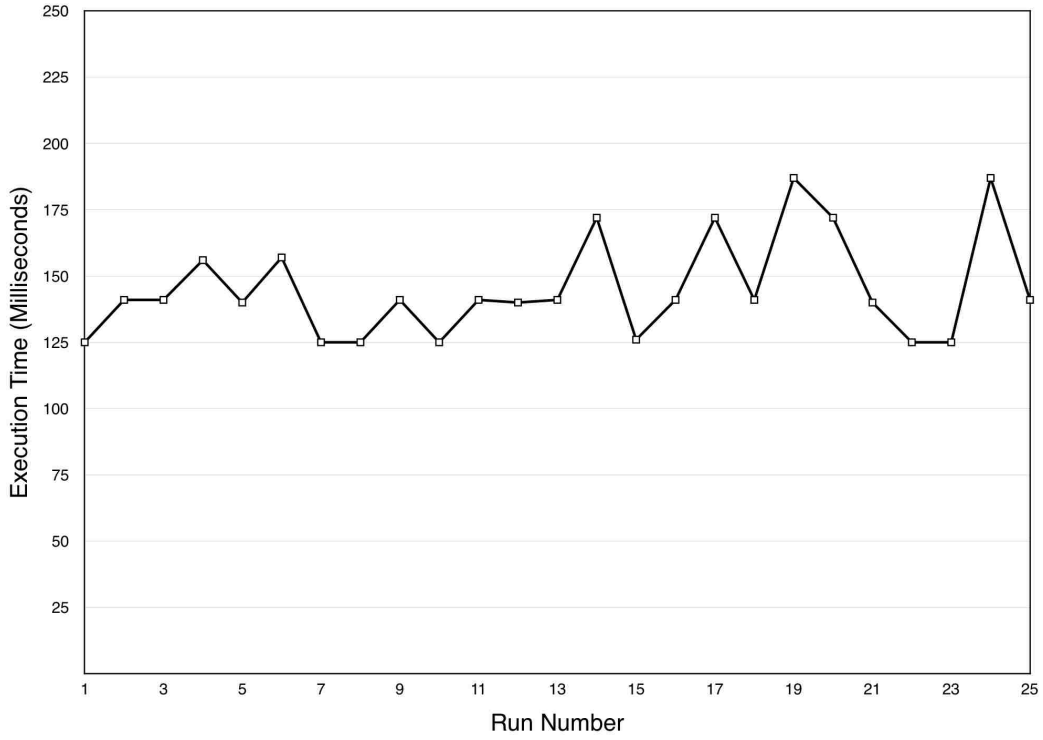

Figure 4.6: M1 Encryption Execution Time

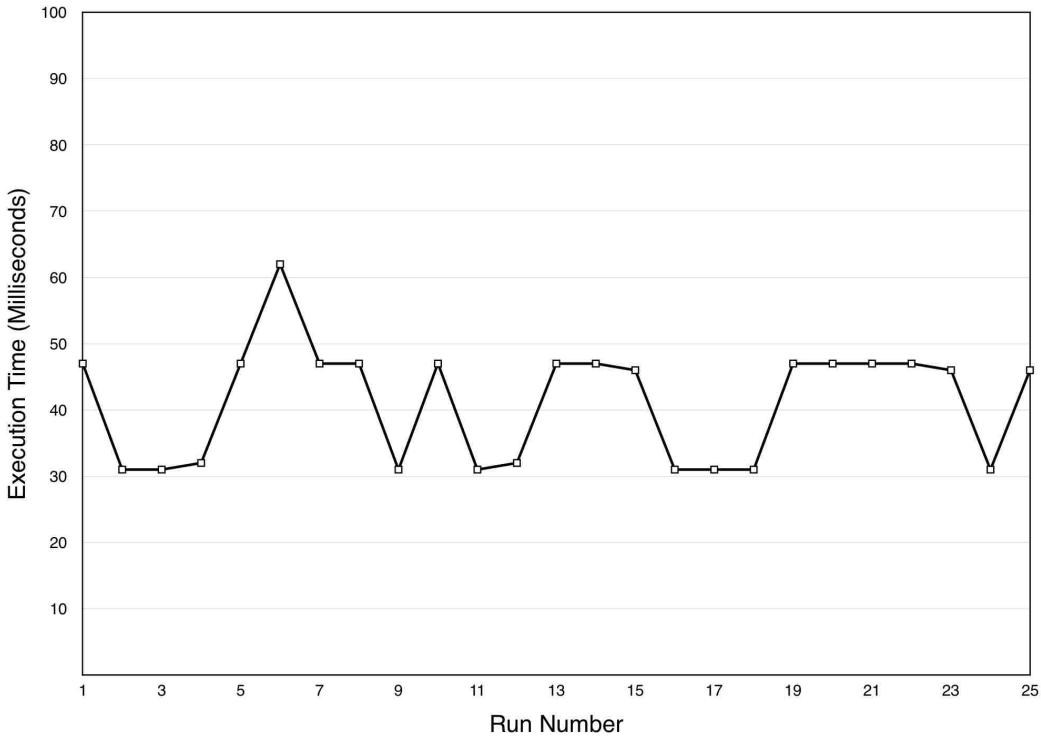

Figure 4.7: M1 Decryption Execution Time

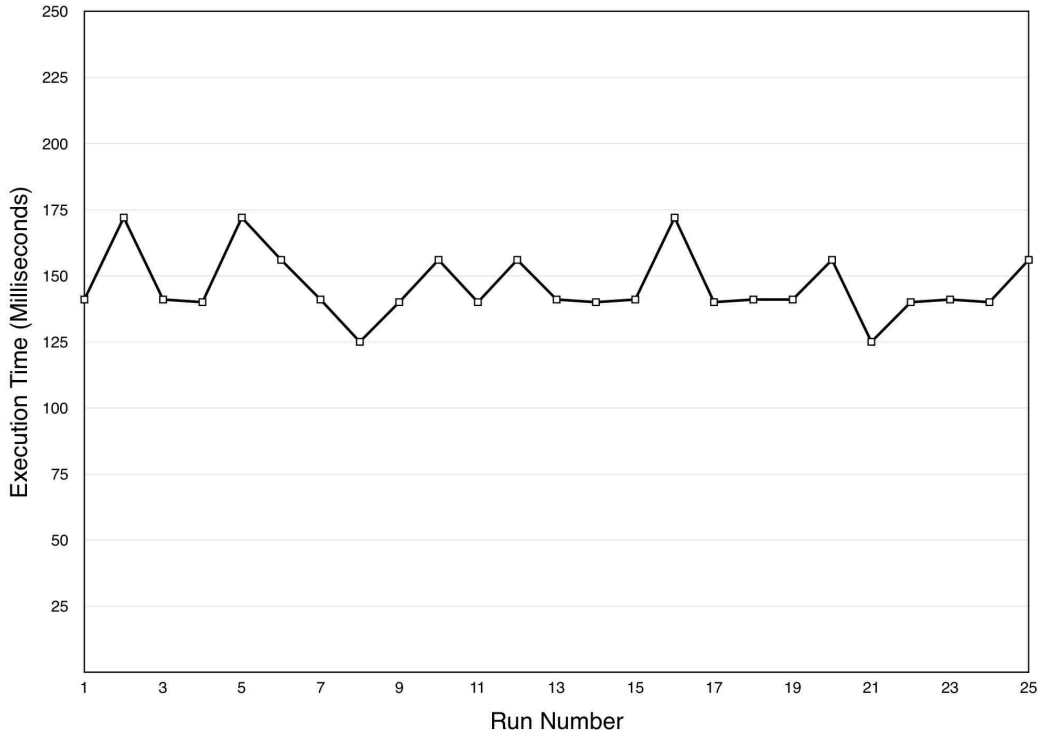

Figure 4.8: M2 Encryption Execution Time

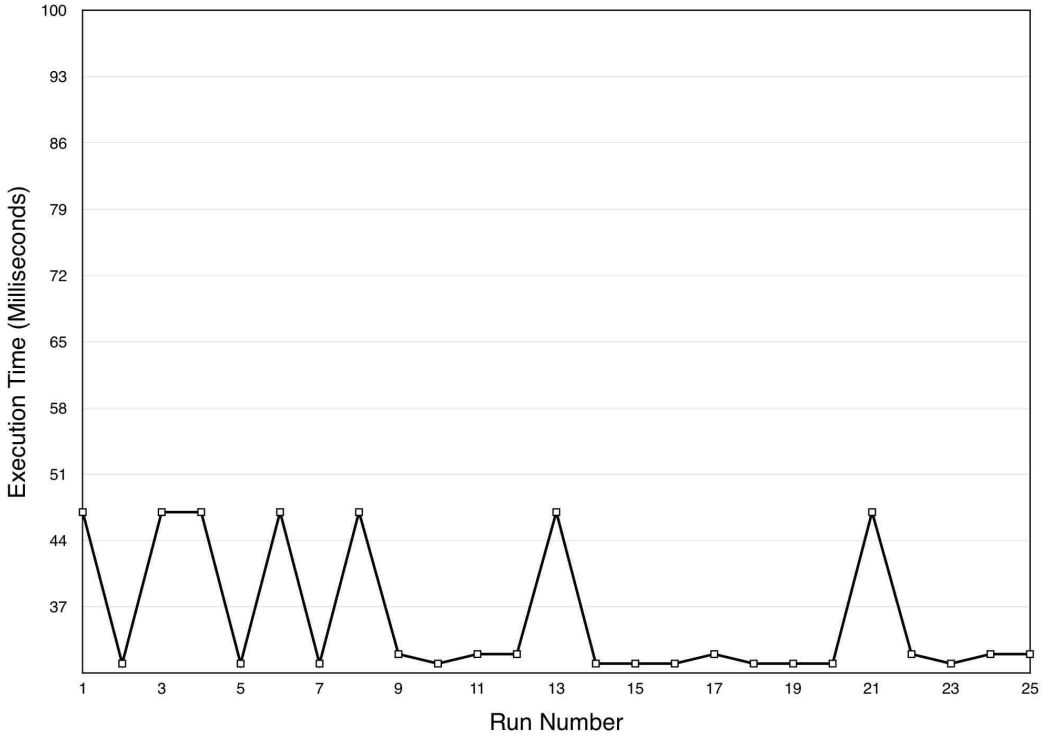

Figure 4.9: M2 Decryption Execution Time

the minimum execution time was 135 milliseconds, the maximum execution time was 235 milliseconds, and the average execution time for the 25 runs was $\approx$ 164 milliseconds as shown in Figure 4.10. After running 25 decryption processes on the encrypted M_3 using the $\mathbf{Current}_{CG}$'s private key $K_{pr}^{current}$ (2048-bits) and using K_{Cons}, the minimum execution time was 32 milliseconds, the maximum execution time was 47 milliseconds, and the average execution time for the 25 runs was $\approx$ 42 milliseconds as shown in Figure 4.11.

The AES was used to encrypt/decrypt health information of sizes 1MB, 10MB, 50MB, 100MB, 500MB, and 1GB. After running 25 symmetric encryption processes using K_{Cons} (simulating M_4) on health information of size 1MB, the minimum execution time was 15 milliseconds, the maximum execution time was 47 milliseconds, and the average execution time for the 25 runs was $\approx$ 26 milliseconds as shown in Figure 4.12. After running 25 symmetric decryption processes using K_{Cons} (simulating M_4) on encrypted health information of size 1MB, the minimum execution time was 15 milliseconds, the maximum execution time was 16 milliseconds, and the average execution time for the 25 runs was $\approx$ 15 milliseconds as shown in Figure 4.13.

After running 25 symmetric encryption processes using K_{Cons} (simulating M_4) on health information of size 10MB, the minimum execution time was 62 milliseconds, the maximum execution time was 79 milliseconds, and the average execution time for the 25 runs was $\approx$ 69 milliseconds as shown in Figure 4.12. After running 25 symmetric decryption processes using K_{Cons} (simulating M_4) on encrypted health information of size 10MB, the minimum execution time was 46 milliseconds, the maximum execution time was 63 milliseconds, and the average execution time for the

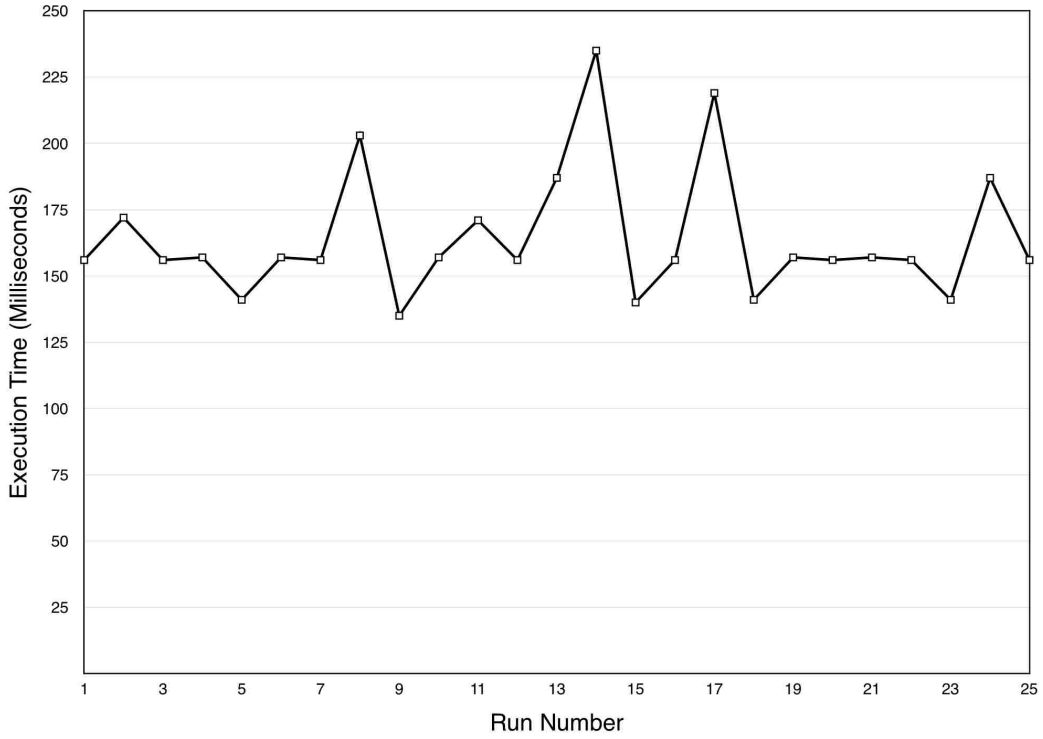

Figure 4.10: M3 Encryption Execution Time

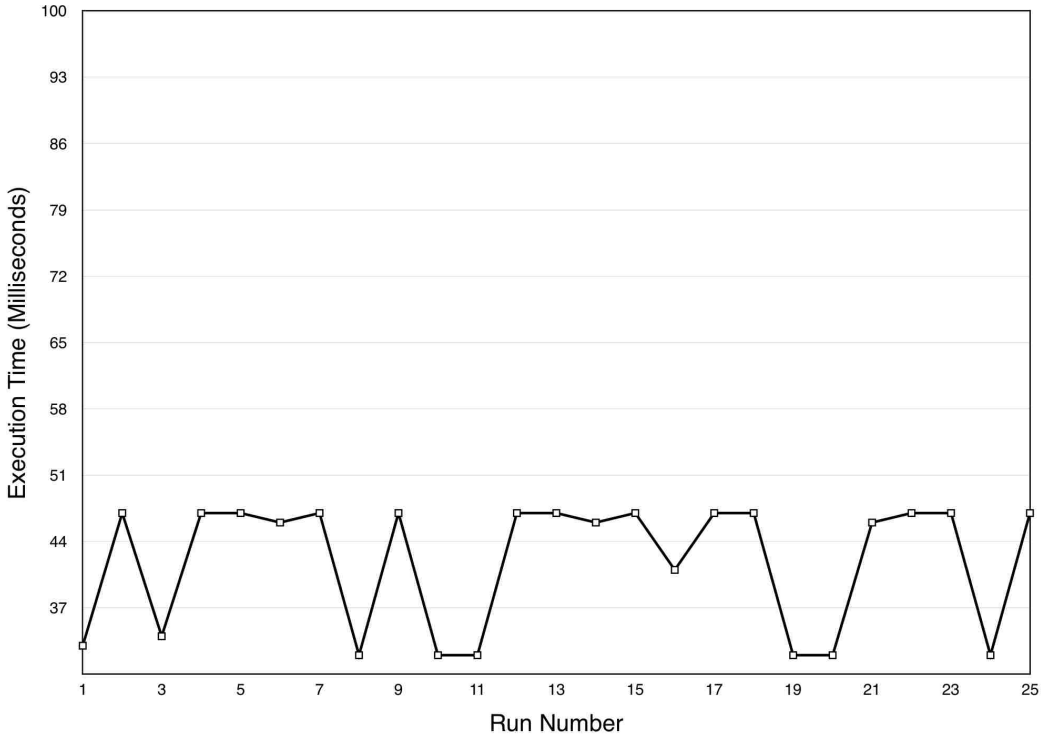

Figure 4.11: M3 Decryption Execution Time

25 runs was ≈ 57 milliseconds as shown in Figure 4.13.

After running 25 symmetric encryption processes using K_{Cons} (simulating M_4) on health information of size 50MB, the minimum execution time was 297 milliseconds, the maximum execution time was 375 milliseconds, and the average execution time for the 25 runs was ≈ 320 milliseconds as shown in Figure 4.12. After running 25 symmetric decryption processes using K_{Cons} (simulating M_4) on encrypted health information of size 50MB, the minimum execution time was 250 milliseconds, the maximum execution time was 313 milliseconds, and the average execution time for the 25 runs was ≈ 264 milliseconds as shown in Figure 4.13.

After running 25 symmetric encryption processes using K_{Cons} (simulating M_4) on health information of size 100MB, the minimum execution time was 625 milliseconds, the maximum execution time was 922 milliseconds, and the average execution time for the 25 runs was ≈ 661 milliseconds as shown in Figure 4.12. After running 25 symmetric decryption processes using K_{Cons} (simulating M_4) on encrypted health information of size 100MB, the minimum execution time was 500 milliseconds, the maximum execution time was 812 milliseconds, and the average execution time for the 25 runs was ≈ 561 milliseconds as shown in Figure 4.13.

After running 25 symmetric encryption processes using K_{Cons} (simulating M_4) on health information of size 500MB, the minimum execution time was 3469 milliseconds (3.469 seconds), the maximum execution time was 9871 milliseconds (9.871 seconds), and the average execution time for the 25 runs was ≈ 7196 milliseconds (7.196 seconds) as shown in Figure 4.14. After running 25 symmetric decryption processes using K_{Cons} (simulating M_4) on encrypted health information of size 500MB, the minimum

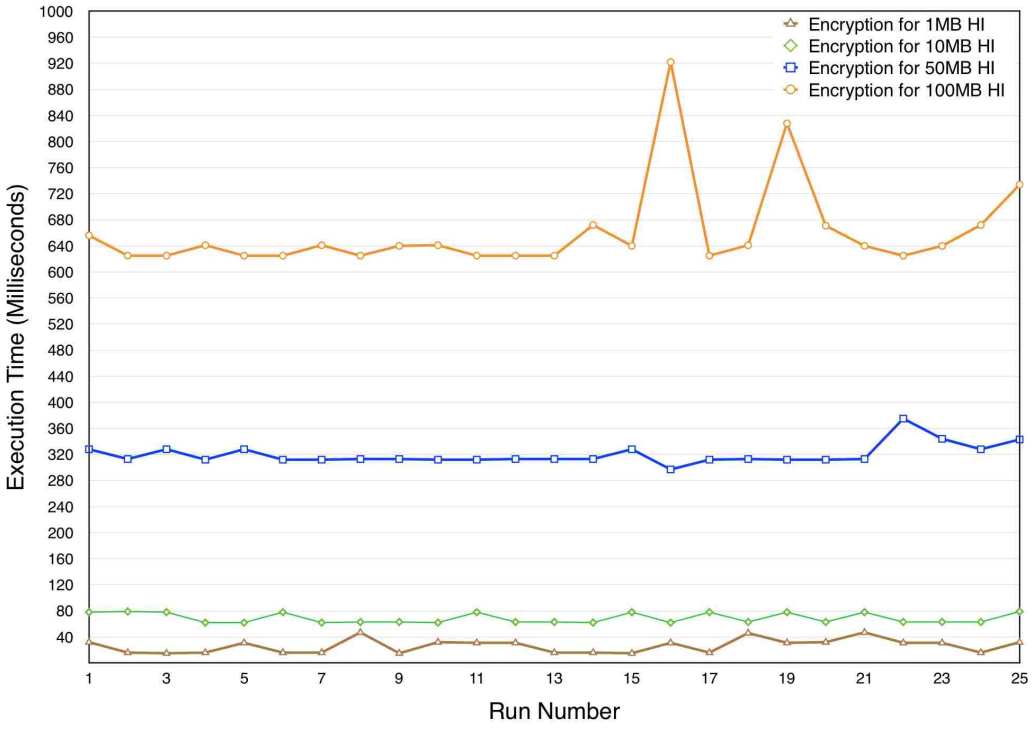

Figure 4.12: M4 Encryption Execution Times for 1MB, 10MB, 50MB, and 100MB of Health Information

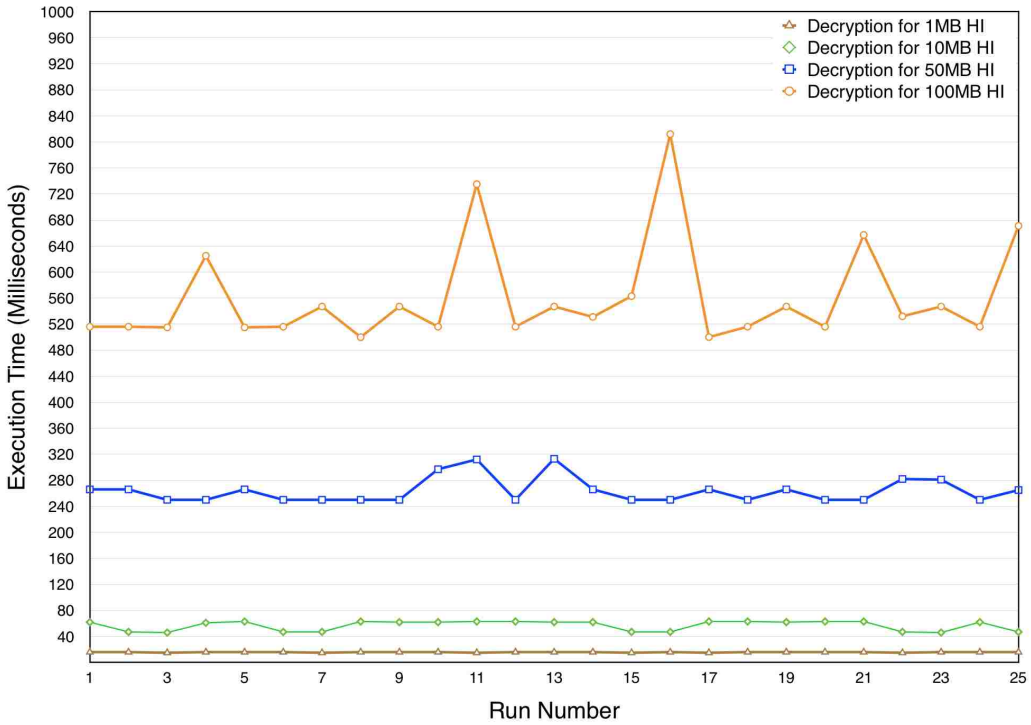

Figure 4.13: M4 Decryption Execution Times for 1MB, 10MB, 50MB, and 100MB of Health Information

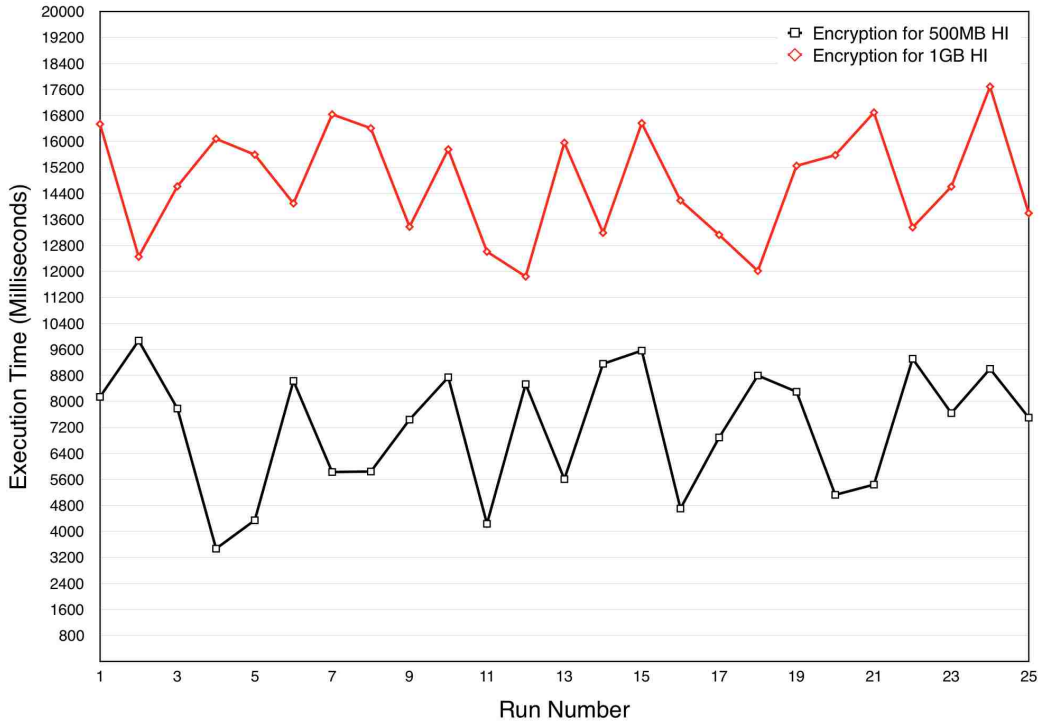

Figure 4.14: M4 Encryption Execution Times for 500MB and 1GB of Health Information

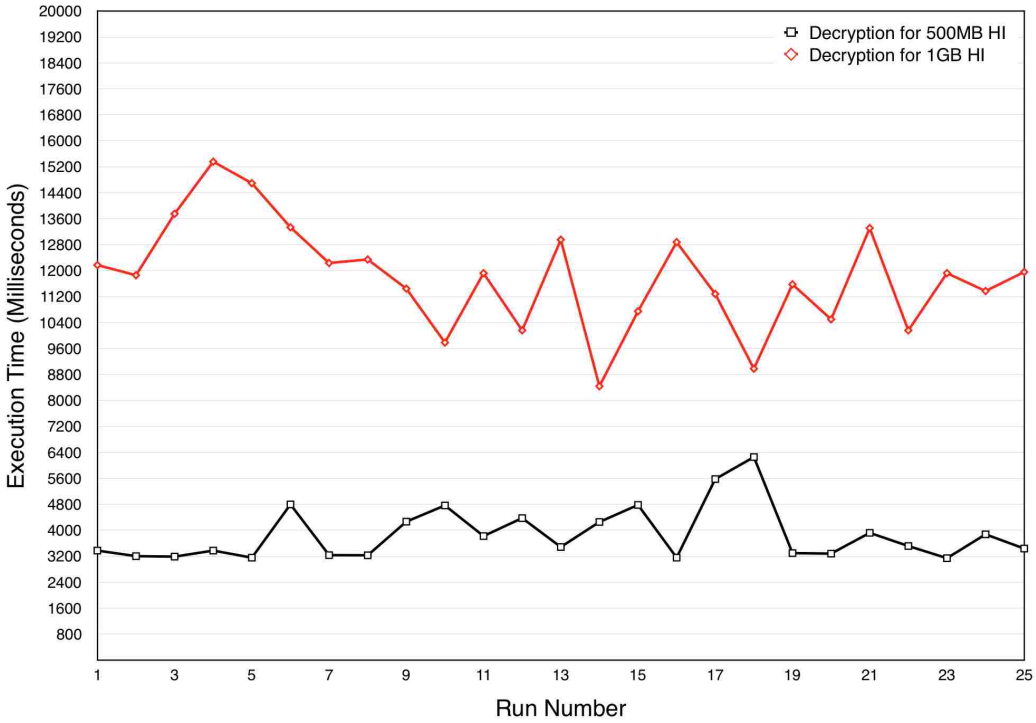

Figure 4.15: M4 Decryption Execution Times for 500MB and 1GB of Health Information

execution time was 3141 milliseconds (3.141 seconds), the maximum execution time was 6256 milliseconds (6.256 seconds), and the average execution time for the 25 runs was ≈ 3871 milliseconds (3.871 seconds) as shown in Figure 4.15.

After running 25 symmetric encryption processes using K_{Cons} (simulating M_4) on health information of size 1GB, the minimum execution time was 11844 milliseconds (11.844 seconds), the maximum execution time was 17688 milliseconds (17.688 seconds), and the average execution time for the 25 runs was ≈ 14736 milliseconds (14.736 seconds) as shown in Figure 4.14. After running 25 symmetric decryption processes using K_{Cons} (simulating M_4) on encrypted health information of size 1GB, the minimum execution time was 8437 milliseconds (8.437 seconds), the maximum execution time was 15358 milliseconds (15.358 seconds), and the average execution time for the 25 runs was ≈ 11805 milliseconds (11.805 seconds) as shown in Figure 4.15.

Table 4.2 shows the minimum, maximum, and average execution times for encrypting and decrypting M_1, M_2, M_3, and M_4 (with 1MB Health Information). Accordingly, the expected minimum, maximum, and average execution times for the protocol (excluding communication delays) are 509 milliseconds, 813 milliseconds, and 615 milliseconds, respectively.

Using 2048-bit RSA keys ensures the infeasibility of cracking the cipher because it requires factoring a 617-digit number which would take a standard desktop computer 6.4 quadrillion years. Brute-force attacks are impossible on AES with a 256-bit key because they would require the generation of 1.1 x 10^{77} key combinations which would require about 3.31 x 10^{56} years to crack the cipher. Thus, the communication will

Table 4.2: Execution Times for Encrypting/Decrypting M_1, M_2, M_3, and M_4 (with 1MB Health Information)

Message	Process	Minimum	Maximum	Average
M_1 (384-bits)	Encryption	125 ms	187 ms	145 ms
	Decryption	31 ms	62 ms	41 ms
M_2 (392-bits)	Encryption	125 ms	172 ms	146 ms
	Decryption	31 ms	47 ms	36 ms
M_3 (144-bits)	Encryption	135 ms	235 ms	164 ms
	Decryption	32 ms	47 ms	42 ms
M_4 (1MB HI)	Encryption	15 ms	47 ms	26 ms
	Decryption	15 ms	16 ms	15 ms

maintain the privacy of the health information exchanged during each session as there are no known attacks on RSA with 2048-bit keys or AES with 265-bit keys.

Also, table 4.3 shows a comparison between the four existing solutions [52] [54] [55] [57] and our protocol according to several requirements/features. Compared to [52] [54] [55] [57], our protocol avoids unnecessary requirements such as costly smart cards, patient training, and the involvement of third parties. Unlike [52] [54] [55] [57], our protocol offers patients the capability to specify the type of medical information to be transferred.

Table 4.3: Comparison of the proposed HIE privacy protocol with related protocols

Requirement/Feature	[52]	[54]	[55]	[57]	Our HIE Privacy Protocol
Special hardware with patients	Yes	Yes	No	No	No
Special hardware at HCPs	Yes	Yes	No	No	No
Patient training	N/A	N/A	Yes	No	No
Uses third parties	Yes	Yes	Yes	Yes (Trust Servers)	No
Specifies medical info. type	No	No	No	Basic info. only	Yes
Authenticates healthcare providers	Home only	Home only	Digital Watermarking	SSL	Yes (Mutual)
Way of checking revocations	CRLs	N/A	N/A	CRLs	OCSP
Authenticates patient	Yes	Yes	N/A	No	Yes
Patient authentication way	Digital Certificate	Digital Certificate	N/A	N/A	Username / Password
Medical info. encryption key type	Asymmetric	Asymmetric	Blinded Committments	Symmetric	Symmetric
Medical information stored in	Recordable Media	Patients' smartcard	Home HCPs' site	Home HCPs' site	Home HCPs' site
Number of message exchange	N/A	9	N/A	Number of hops + 2	4
Has audit controls	Yes	No	No	No	Yes
Privacy preserving	No	Yes	N/A	No	Yes
Provides adequate security analysis	No	No	No	No	Yes
Prevents man-in-the-middle attacks	No	No	No	No	Yes
Detects replay messages	No	No	No	No	Yes

Chapter 5 A Proxy-Signature-Based Access Control for Health Information Exchange

Access control (or authorization) is a critical and integral requirement in the development of a secure HIE. Given that the primary purpose of an HIE is the exchange of health information, it is essential that a request for any such exchange contains an explicit authorization to prevent any undesired or unintentional leakage of health information. Moreover, the process of authorization must be accompanied by the authentication of the entities making the access request. In this chapter, *we address the problem of enabling secure (authenticated and authorized) on-demand access to patient records in a cloud-based HIE.*

Existing methods of authorization in electronic health information systems can be divided into cryptographic and non-cryptographic approaches [62] [157]. The non-cryptographic approaches for authorization mainly focus on the development of a policy-based authorization infrastructure, where access to health information is governed by novel access control policies specifically developed to secure electronic health information. The predominant access control model proposed in the majority of existing literature is the role-based access control (RBAC) model [62]. However, non-cryptographic approaches also elude the enforcement of the access control policies to standard off-the-shelf mechanisms or commercial standards that can be prone to problems such as misconfiguration, policy corruption, forgery, and other technical errors or limitations [62] [158]. Any use of the existing non-cryptographic approaches

for secure health information access in HIEs requires an explicit and secure mechanism for policy enforcement. To this end, cryptographic approaches provide just such an explicit and secure mechanism for enforcing authorization. The primary cryptographic approach for providing access control in health information systems is the use of ABE [157]. Although ABE and its variants can be an effective mechanism for providing access control in HIE, they suffer from several limitations [157]. Being pairing-based, ABE-based schemes typically suffer from significantly high computational overhead. In addition, ABE-based schemes can be limited in terms of specifying access policies. ABE-based schemes also exhibit high complexity, which can lead to implementation errors.

5.1 Objectives

The goal of this chapter is to address the issues with existing approaches for authorization in health information systems, and to develop a simple, effective, and efficient protocol suitable for securing HIE access control. The authorization protocol fills the gap between cryptographic and non-cryptographic approaches, with the former lacking an explicit authorization enforcement mechanism that is cryptographically secure, and the latter being complex, computationally expensive, and limited in policy specification. Moreover, rather than combining encryption with authorization, as has been the method of most existing cryptographic approaches (ABE-based schemes), a more universal approach is followed to combine authentication with authorization.

More specifically, we develop a secure and efficient trapdoor hashing scheme, and employ it in a novel manner to construct a proxy signature-based protocol for authen-

ticated and authorized on-demand access to patient records. The protocol facilitates a patient-centric approach for controlling access to and exchange of a patient's health information in a selective manner that complies with policies agreed upon by healthcare providers and patients. The mechanism allows patients to authorize the sharing of specific medical information with specific healthcare providers for a specific period of time, which helps prevent any undesired or unintentional leakage of health information. The scheme also ensures that such authorizations are authentic with respect to both the healthcare providers and the patient, and comply with the established access control policies. Moreover, the use of proxy signatures simplifies security auditing and the process of obtaining support for investigations by providing non-repudiation. In summary, the contributions of this Chapter are as follows:

1. Developing a novel discrete log-based trapdoor hashing scheme that is efficient and secure against collision forgery and key-exposure.

2. Using the trapdoor hashing scheme to develop a novel proxy signature-based protocol that enables authenticated and authorized selective sharing of patient health information via a cloud-based HIE. The protocol exhibits several desirable features that include non-interactive and on-demand operation, flexible specification of access control policies, audit support, and compliance with agreements between healthcare providers and patients.

3. Performing a security analysis of the proposed protocol which shows that the trapdoor hashing scheme is secure against collision forgery and key-exposure,

and that the protocol is secure against forgery under the well-known discrete log assumption.

4. Performing a performance analysis of the proposed protocol using the developed trapdoor hash-based which shows that our proxy signature scheme achieves the best all-round performance (while being provably secure) compared to other well-known proxy signature schemes in the literature.

5.2 The System Model

This section describes the system model associated with the protocol for secure information access and exchange in HIE. The system consists of two main components: the HIE cloud and healthcare organizations (HCOs) which offer health services to patients as shown in Figure 5.1.

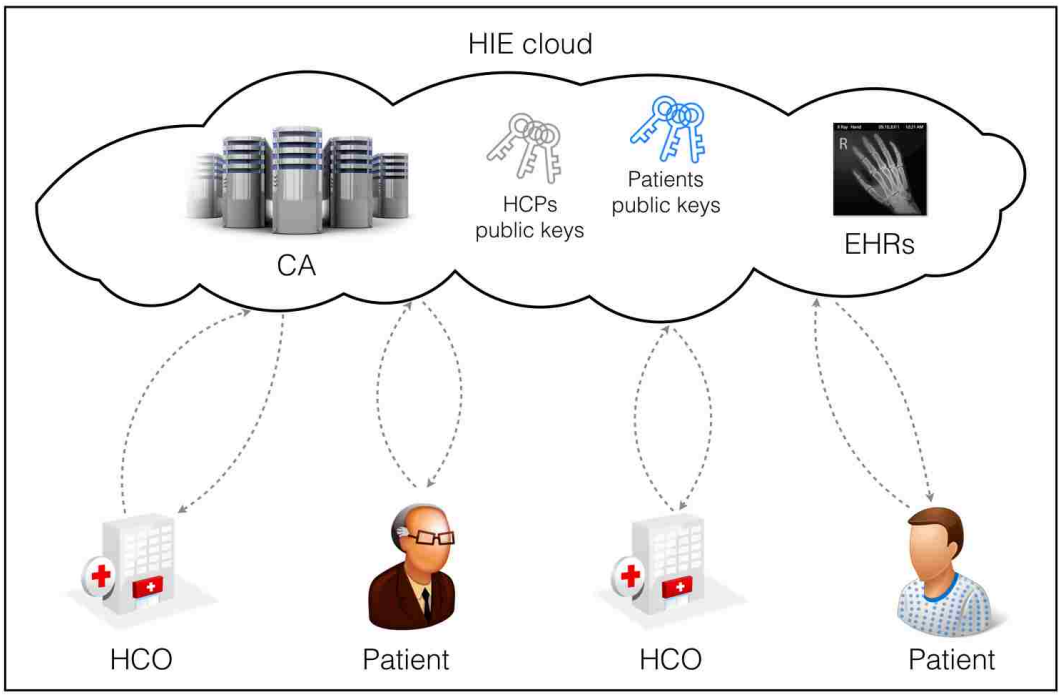

Figure 5.1: HIE Access Control System Components

5.2.1 The Health Information Exchange Cloud

The HIE cloud consists of the (layered) hardware and software components (such as servers, storage, communication and virtualization software) that are used by various HCOs for storing, sharing, processing, and managing healthcare data. The HIE cloud is built as a hybrid cloud maintained by a (trusted) external cloud service provider and the HCOs, and functions as the principal point of interaction between the patients and various HCOs. It is assumed that the HIE cloud implements mechanisms that allows for the integration and composition of data from disparate sources [159] into a single uniform database. All patient data is maintained in an encrypted form during storage using homomorphic or searchable schemes [157], while all communication is secured using well-known mechanisms like the TLS protocol [62].

5.2.2 Healthcare Organizations

Any entity that offers health services is considered an HCO. All HCOs generate valid public-private key pairs, register with the HIE cloud, and obtain a valid digital certificate. Each HCO has the appropriate tools to allow the processing and storing patients' health information in the HIE cloud and the ability to access patients' health information created by other HCOs according to the access permissions given by patients. Patients obtain their credentials from the HCO that first creates their health information record. Patients' devices (e.g., laptop or smart phone) contain all the necessary software to communicate with various HCOs and the HIE cloud, and to access and manage their health information.

Patients also generate their public-private key pair and register with the HIE cloud, obtaining valid certificates. The proposed authorization protocol is designed to provide patients full control over access to their health information, which we assume will be implemented as software components installed and maintained on the patient-side, HCO-side, as well as the HIE cloud-side. Patients use their credentials to grant access permissions to other HCOs and control the type of medical information that each HCO is authorized to access.

It is assumed that both patients and HCOs have the necessary mechanisms to support the use of public key cryptography. To support this, the HIE cloud implements a public key infrastructure (PKI) that validates legitimate HCOs and patients, and issues X.509 digital certificates with all corresponding information for registering HCOs and patients. The HIE cloud also implements the necessary revocation mechanisms and maintains a publicly available directory to make the public keys and certificates available to requesting entities.

The design of the protocol takes advantage of the way trapdoor hash-based proxy signatures are generated, along with the use of a message space descriptor, in the form of a warrant, that is used in the delegation process for controlling various parameters, like the validity period, identities of delegator and proxy, message format, etc. The current HCO of a patient wanting to share a patient's health information, say H_c, acts as a delegator and generates a standard signature on the trapdoor hash of a warrant, along with some additional information, like its own and the patient's identities. This warrant acts as an *authorization template* that is used by the patient, acting as a proxy, to specify the new HCO that needs to be granted access to patient data,

say H_n, the validity period of the authorization, and the types of records that can be accessed. To do this, the patient generates a trapdoor collision between the warrant and these specific authorization parameters. This process, in effect, is a legitimate modification of the warrant into the authorization parameters without invalidating the original HCO's signature, and requires no interaction with the original HCO, H_c. The resulting proxy signature on the authorization parameters, along with the HCO's signature on the warrant, act as an *authorization certificate*, which is then transferred to H_n.

Now, H_n's request to the HIE cloud for access to the patient records is accompanied by this authorization certificate, which can be verified by the HIE cloud using the public keys of the patient and H_c. A successful verification not only indicates to the HIE cloud that H_n's request is authorized by the patient, but also that this authorization is in agreement with the patient's current HCO, H_c. In this way, we are able to provide an explicit mechanism for patients to authorize the sharing of specific medical information with specific HCOs, which helps prevent any undesired or unintentional leakage of health information. The protocol also ensures that such authorizations are authentic with respect to both the current HCO H_c and the patient granting access.

The protocol governs the interaction between the various architectural components and entities in direct and indirect ways to perform various tasks that can be divided into three phases, namely initialization, certificate generation, and certificate verification. We assume a patient P wanting to provide record access to a new HCO, H_n with the cooperation of its current HCO H_c. For simplicity, it is assumed that

the identities of patients and HCOs are unique and unforgeable.

In the initialization phase, all entities compute and agree on the common cryptographic parameters, and generate and register their public keys. In the certificate generation phase, P and H_c interact with each other to generate an authorization certificate for H_n. Finally in the certificate verification phase, H_n presents the HIE cloud with the authorization certificate as part of the request to access patient data, followed by the HIE cloud verifying the certificate to check whether the authorization is valid.

5.3 The Proposed Protocol: The Formal Description

Before describing the access control protocol for HIE systems, we present a secure discrete log (DL)-based trapdoor hashing scheme, called `DL-mDTH` in Subsection 5.3.1. The `DL-mDTH` is used to construct a proxy signature scheme-based protocol for secure access and exchange of patient health information. Afterwards, we describe the proposed protocol formally.

5.3.1 The `DL-mDTH` Trapdoor Hashing Scheme

Chandrasekhar et al. [128] [140] proposed an efficient DL-based instantiation of a double-trapdoor hashing scheme, called `DL-DTH`, and also performed a detailed security analysis that proves the `DL-DTH` is secure against key-exposure and collision forgery under the DL assumption. The `DL-mDTH` scheme is a variant of the `DL-DTH` [128] [140] scheme that does not require use of a zero-knowledge proof of knowledge (ZKPoK) for the secret ephemeral trapdoor key while maintaining security and

efficiency of DL-DTH.

The common system public parameters of the DL-mDTH scheme are $\mathsf{params} = \langle p, q, g, H \rangle$, where p and q are primes such that $q \mid (p-1)$, g is an element of order q in $\mathbb{Z}_p^*$ and $H : \{0,1\}^* \mapsto \mathbb{Z}_q^*$ is a cryptographic hash function. The long-term trapdoor and hash key pair of an entity is $(TK_l, HK_l) = (y \in_R \mathbb{Z}_q^*, Y = g^y \in \mathbb{Z}_p^*)$, and the ephemeral trapdoor and hash key pair of an entity is $(TK_e, HK_e) = (z \in_R \mathbb{Z}_q^*, Z = g^z \in \mathbb{Z}_p^*)$. An entity generates a trapdoor hash of a message $m \in \{0,1\}^*$ using the hash key $HK = (Y, Z)$ by choosing an element $r \in_R \mathbb{Z}_q^*$ and computing the hash as $TH_{HK}(m, r) = g^{H(m||Y||Z)}(YZ)^r \mod q$. Given system parameters params the trapdoor key $TK = (y, z)$, message $m \in \{0,1\}^*$, $r \in \mathbb{Z}_q^*$ and an additional message $m'(\neq m) \in \{0,1\}^*$, an entity computes a collision as follows:

1. Chooses an ephemeral trapdoor key $z' \in_R \mathbb{Z}_q^*$ and computes the corresponding ephemeral hash key $Z' = g^{z'} \mod q$.

2. Computes r' by solving $r' = y + z'^{-1}(H(m||Y||Z) - H(m'||Y||Z') + (y+z)r) \mod q$.

3. Outputs $\langle r', HK' = (Y, Z') \rangle$, such that $g^{H(m||Y||Z)}(YZ)^r = g^{H(m'||Y||Z')}(YZ')^{r'}$

In Section 5.4, a detailed security analysis of the proposed DL-mDTH trapdoor hashing scheme is presented, proving its resistance to collision forgery and key exposure.

5.3.2 Initialization Phase of the Protocol

The initialization phase begins with all entities choosing and agreeing on the common system public parameters $\mathsf{params} = \langle p, q, g, H, G \rangle$, where p, q, g, and H are

as described in subsection 5.3.1 for the `DL-mDTH` scheme, along with an additional cryptographic hash function $G : \{0,1\}^* \mapsto \mathbb{Z}_q^*$. Each HCO chooses its long-term private (signing) key $x \in_R \mathbb{Z}_q^*$ and computes the corresponding long-term public key as $X = g^x \in \mathbb{Z}_p^*$. In addition, each patient P chooses his/her long-term trapdoor key $TK = (y, z)$, where $y, z \in_R \mathbb{Z}_q^*$, and computes the corresponding long-term hash key $HK = (Y, Z)$, where $Y = g^y \mod p$ and $Z = g^z \mod p$. As mentioned in Section 5.2, we assume the existence of a public key infrastructure where all public keys are published in a publicly available directory, with certificates that identify and validate key ownership.

5.3.3 Certificate Generation Phase of the Protocol

In this phase, a patient P interacts with its current HCO, H_c, to generate an authorization certificate that will allow a new HCO, H_n, to access P's records from the HIE cloud. The process begins with the patient P requesting H_c for a signed authorization template to allow P to grant H_n access to his/her health information. Upon receiving this request, H_c creates a message M.

M is partitioned into the following two parts: a) the component m that contains the identities of the current HCO and patient, and b) the message space descriptor (warrant) m_w that serves as an authorization template. Figure 5.2 shows the format of Message M. The patient uses this warrant to fill in the remaining parameters (fields) pertaining to the desired authorization, which include the new HCO's identity, type(s) of records that can be accessed, and the validity period for the authorization.

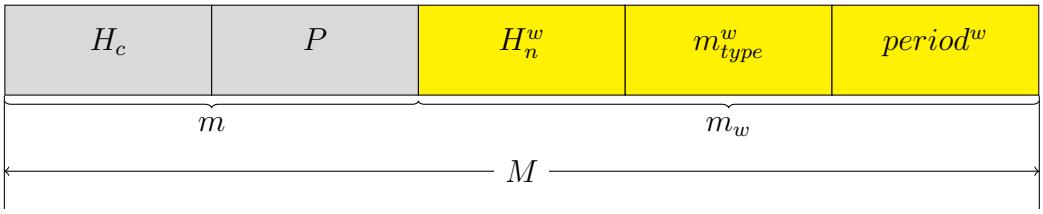

Figure 5.2: Message M Format

The HCO H_n retrieves the long-term trapdoor hash key pair $HK = (Y, Z)$ of the patient P and generates a signed authorization template (contained within M) as follows:

1. Chooses a $k \in_R \mathbb{Z}_q^*$ and computes $K = g^k \mod p$, where (k, K) are used as the ephemeral (private, public) key pair for H_c's signature on M.

2. Computes $h_w = H(m_w||Y||Z)$ and $r = K \mod q$, and generates the trapdoor hash on m_w as $th_w = TH_{HK}(m_w, r) = g^{h_w}(YZ)^r \mod q$.

3. Computes $h = G(M||th_w||K)$ and solves for t in $t \equiv k + xh \mod q$ (Schnorr [160]-type signing).

4. Creates the signed authorization template $A_T = \langle M, \sigma \rangle$, where $\sigma = \langle t, h \rangle$, and sends A_T to P.

After receiving the signed authorization template A_T, the patient P retrieves the long-term public key X of H_c, verifies H_c's signature σ on M (containing the warrant) under X, and generates an authorization certificate as follows:

5. Parses M as $(m||m_w)$, and computes $K = g^t X^{-h} \mod p$, $r = K \mod q$, $h_w = H(m_w||Y||Z)$, and $th_w = g^{h_w + (y+z)r} \mod q$.

6. Checks whether $h = G(M||th_w||K)$, and if so, σ is a valid signature by H_c on M.

7. Generates the authorization parameters $m_p = (H_n||m_{type}||period)$ complying with m_w.

8. Chooses an ephemeral trapdoor key $z' \in_R \mathbb{Z}_q^*$ and computes the corresponding ephemeral hash key $Z' = g^{z'} \mod p$.

9. Computes $h_p = H(m_p||Y||Z')$ and solves for r' in $r' = (y + z')^{-1}(h_w - h_p + (y + z)r) \mod q$, resulting in a collision between the trapdoor hashes of the warrant m_w and the authorization parameters m_p, i.e., $th_w = g^{h_w}(YZ)^r = g^{h_p}(YZ')^{r'} = th_p$.

10. Generates the authorization certificate $A_C = \langle M', \sigma_P \rangle$, where $\sigma_P = \langle t, h, r', h_p \rangle$ and $M' = M||m_p$, and sends A_C to H_n.

We observe that after generating an authorization certificate A_C for an HCO H_n, the patient can store the authorization template A_T to grant authorizations to other HCOs, or modify existing authorizations for a previously authorized HCO. The patient generates new (or modifies existing) authorization certificates simply by creating new authorization parameters m'_p and computing a trapdoor hash collision with m_w (following steps 7 - 10). The patient need not re-verify σ (using steps 5 and 6), which provides significant savings in computational costs.

5.3.4 Certificate Verification Phase of the Protocol

After receiving the authorization certificate A_C from the patient, H_n sends a request to the HIE cloud for access to the patient records by attaching A_C to the request. The HIE cloud retrieves the long-term public key X of H_c and long-term hash key (Y, Z) of P, and verifies H_n's authorization as follows:

11. Parses M' as $(m||m_w||m_p)$ and σ_P as $\langle t, h, r', h_p \rangle$, and checks whether m_p complies with m_w.

12. Computes $K = g^t X^{-h} \mod p$, $r = K \mod q$, $h_w = H(m_w||Y||Z)$, and $th_w = g^{h_w}(YZ)^r \mod q$.

13. Checks whether $h = G(M||th_w||K)$, and if so, σ is a valid signature by H_c on M.

14. Computes $Z' = Y^{-1}(g^{(h_w - h_p)r'^{-1}}(YZ)^{rr'^{-1}}) \mod p$ and checks whether $h_p = H(m_p||Y||Z')$. If so, then the authentication parameters are valid.

The new HCO H_n now has access to the patient P's records created by H_c of type m_{type} for a period of time (*period*). Once again, just as a patient can re-use a signed authorization template to save on computation, the HIE cloud can also save the signed authorization template and avoid the cost of re-verifying H_c's signature on the warrant. Every time the HIE cloud receives a new authorization certificate A'_C generated by the same combination of H_c and P, the HIE cloud can simply compare the t and h parameters in A'_C with those in the previously received A_C, and, in case of a match, executes step 14 to verify the collision between trapdoor hashes of the

warrant and the new authorization parameters. Figure 5.3 shows the flow of the HIE access control protocol.

5.4 Security Analysis

This section provides a detailed security analysis of the DL-based double trapdoor hashing scheme `DL-mDTH` and the resulting protocol for secure (authenticated and authorized) access and exchange of patient health information.

The difficulty of forging collisions and key exposure in `DL-mDTH` and the security of the proposed authorization protocol are based on the difficulty of solving the well-known discrete logarithm problem (DLP) in group $\mathbb{Z}_p^*$ [134]. We begin by proving the following two theorems that establish the security of `DL-DTH`.

Theorem 1 *The trapdoor hashing scheme* `DL-mDTH` *[cf. subsection 5.3.1] is collision-forgery-resistant.*

Proof: We prove the forgery resistance property of the proposed trapdoor hashing scheme by showing that the discrete log problem in group $\mathbb{Z}_p^*$ reduces to collision forgery, thus violating the well known discrete log assumption.

Assume that there exists a PPT collision forger $\mathcal{F}$ against the proposed trapdoor hashing scheme with non-negligible advantage. Given a hash key $HK = (Y, Z)$ and parameters $\langle p, q, \alpha, H, G\rangle$, $\mathcal{F}$ runs in polynomial time and outputs the tuple $\langle m, r, m', r', HK'\rangle$, where $HK' = (Y, Z')$, such that $m \neq m'$, $r \neq r'$, and $h = TH_{HK}(m, r) = TH_{HK'}(m', r')$ with non-negligible probability. Given $\mathcal{F}$, we can construct a PPT algorithm $\mathcal{D}$ that breaks the subgroup DLP assumption [134] as

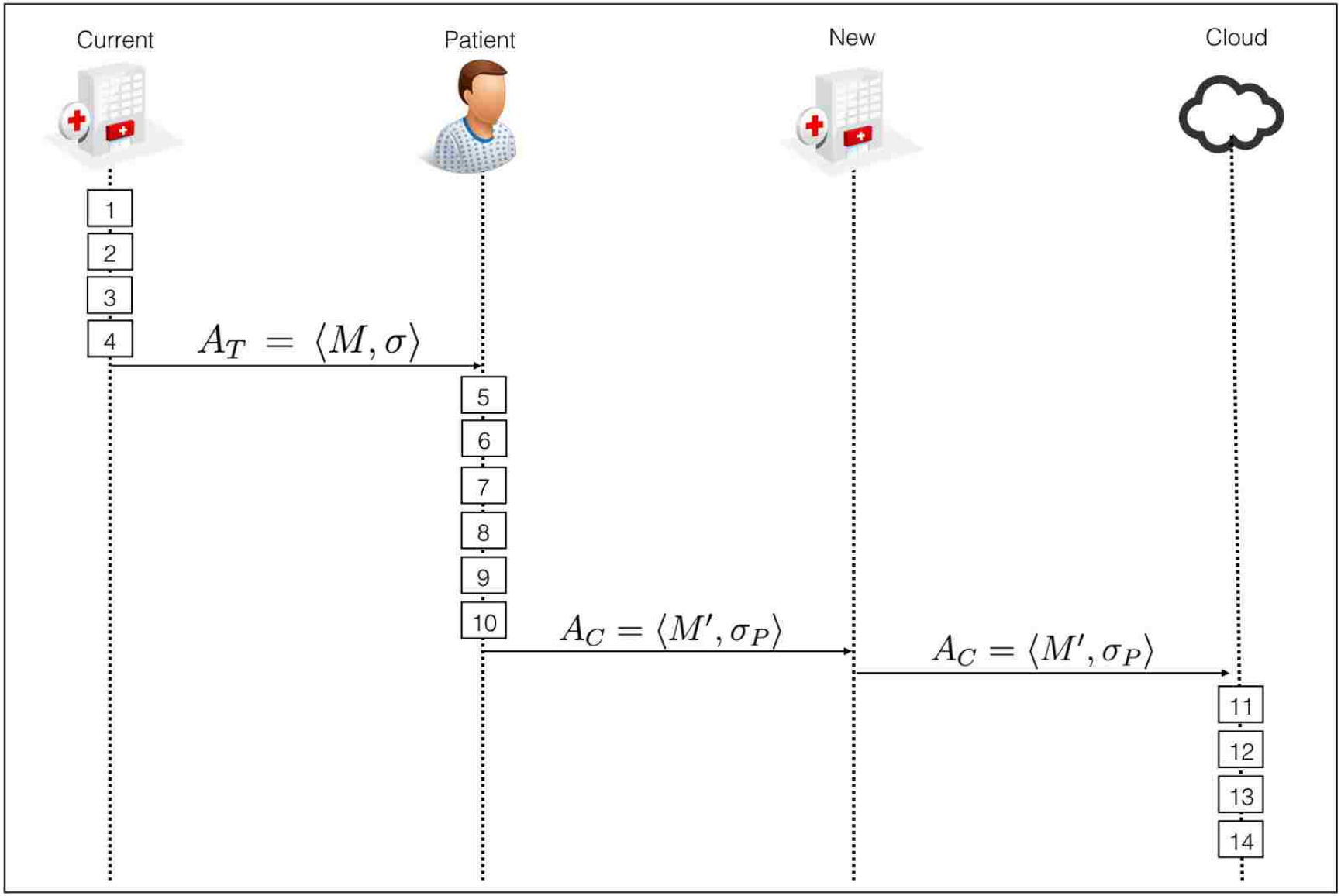

Figure 5.3: HIE Access Control Protocol Flow

follows. $\mathcal{D}$ is given a DLP instance $\langle p, q, g, Y \rangle$. $\mathcal{D}$ needs to find $y \in \mathbb{Z}_q^*$ such that $Y = g^y \mod p$. The hash function H behaves as a random oracle $\mathcal{O}_H$ that $\mathcal{D}$ simulates. This means that $\mathcal{D}$ answers any hash queries to $\mathcal{O}_H$ with a random value for each new query [161] (with identical answers if the same query is asked twice). For instance when $\mathcal{F}$ queries $\mathcal{O}_H$ with $\langle m \rangle$, where $\mathcal{D}$ returns h if $\exists h$ such that $h = H(m)$. Otherwise $\mathcal{D}$ chooses $h \in_R \mathbb{Z}_q^*$, sets $H(m) = h$ (i.e., stores h as the hash entry for $H(m)$), and returns h to $\mathcal{F}$.

$\mathcal{D}$ chooses $z \in_R \mathbb{Z}_q^*$, computes $Z = g^z \mod p$ and sets $HK = (Y, Z)$. $\mathcal{D}$ then runs an instance of forger $\mathcal{F}$ with HK as input, answering any hash queries to $\mathcal{O}_H$, until $\mathcal{F}$ produces the collision forgery $\langle m, r, m', r', HK' \rangle$, where $HK' = (Y, Z')$, and $g^{H(m||Y||Z)}(YZ)^r = g^{H(m'||Y||Z')}(YZ')^{r'} \mod q$. Let h' be the response $\mathcal{D}$ gave when $\mathcal{F}$ made the query $\langle m'||Y||Z' \rangle$ to $\mathcal{O}_H$. Using the oracle replay attack [161], $\mathcal{D}$ rewinds $\mathcal{F}$ to the point when $\mathcal{F}$ made the query, $\langle m'||Y||Z' \rangle$ to $\mathcal{O}_H$, and gives $\mathcal{F}$ a new randomly chosen value $h'' \neq h' \in_R \mathbb{Z}_q^*$. $\mathcal{D}$ continues execution of $\mathcal{F}$, until $\mathcal{F}$ produces another collision forgery of the form $\langle m, r, m', r'', HK' \rangle$, where $g^{H(m||Y||Z)}(YZ)^r = g^{H(m'||Y||Z')}(YZ')^{r"} \mod q$. Given the two collisions produced by $\mathcal{F}$, we now have $h + (y + z)r = h' + (y + z')r' \mod q$ and $h' + (y + z')r' = h'' + (y + z')r'' \mod q$, where h is the response $\mathcal{D}$ gave when $\mathcal{F}$ made the query $\langle m||Y||Z \rangle$ to $\mathcal{O}_H$, and z' is the discrete log of Z'. From $h' + (y + z')r' = h'' + (y + z')r'' \mod q$, $\mathcal{D}$ computes $tk' = (h' - h'')(r'' - r')^{-1} \mod q$, where $tk' = y + z'$. Finally, from $h + (y + z)r = h' + (y + z')r' \mod q$, $\mathcal{D}$ computes the discrete log of Y as $y = r^{-1}(h' - h + tk'r') - z \mod q$. This concludes the proof. □

Theorem 2 *The proposed trapdoor hashing scheme,* `DL-mDTH` *[cf. subsection 5.3.1], is key-exposure-resistant.*

Proof: Key exposure resistance in `DL-mDTH` implies that given a trapdoor collision tuple $\langle m, r, HK, m', r', HK' \rangle$ such that $TH_{HK}(m, r) = TH_{HK'}(m', r')$, where $HK = (Y, Z)$ and $HK' = (Y, Z')$, it is computationally infeasible to find the long-term trapdoor key, y corresponding to Y.

Assume that there exists a PPT algorithm $\mathcal{K}$ that succeeds in key exposure against `DL-mDTH` with non-negligible advantage. Given $\mathcal{K}$ we can construct a PPT algorithm $\mathcal{D}$ that breaks the subgroup DLP assumption [134] as follows. Similar to the proof of Theorem 1, $\mathcal{D}$ is given a DLP instance $\langle p, q, g, Y \rangle$ and needs to find $y \in \mathbb{Z}_q^*$ such that $Y = g^y \mod p$. Once again, the hash function H behaves as a random oracle $\mathcal{O}_H$ that $\mathcal{D}$ simulates.

$\mathcal{D}$ chooses $z \in_R \mathbb{Z}_q^*$, computes $Z = g^z \mod p$, and sets $HK = (Y, Z)$. Next, $\mathcal{D}$ chooses $m, m' \in_R \{0,1\}^*$ and $h, h', r, r' \in_R \mathbb{Z}_q^*$. $\mathcal{D}$ computes $Z' = Y^{-1}(g^{(h-h')r'^{-1}}(YZ)^{rr'^{-1}})$ mod p and sets $H(m||Y||Z) = h$ and $H(m'||Y||Z') = h'$. It is straightforward to see that $\langle m, r, HK, m', r', HK' \rangle$, where $HK' = (Y, Z')$, is a double-trapdoor collision tuple. $\mathcal{D}$ then runs an instance of forger $\mathcal{K}$ with $\langle m, r, HK, m', r', HK' \rangle$ as input. When $\mathcal{K}$ outputs y, $\mathcal{D}$ outputs the same value y as the discrete log of Y. This concludes the proof. □

In addition to collision forgery resistance and key exposure resistance, the proposed trapdoor hashing scheme also provides semantic security [129]. A trapdoor hashing scheme is said to be semantically secure if, for all hash keys HK, and all

message pairs m and m', $m \neq m'$, the probability distributions of the hash values $\mathtt{TH}_{HK}(m, r)$ and $\mathtt{TH}_{HK}(m', r)$ are computationally indistinguishable. We omit the proof here as it closely follows the technique of Ateniese et al. [131].

Given that `DL-mDTH` exhibits properties of collision forgery resistance, key exposure resistance and semantic security, as well as the provable security guarantees provided by the well-known `DL-Schnorr` [160] signature scheme, we state the following:

Theorem 3 *The protocol employed for access and exchange of patient health information is secure against adaptive chosen message attacks in the random oracle model [162] under the well-known discrete log assumption.*

Proof: The proposed authorization protocol is based on a trapdoor hash-based proxy signature scheme, where the certificate generation phase involves H_c delegating a patient P as a proxy under warrant m_w, and subsequently, the patient generating a proxy signature σ_P on m_p, where m_p complies with the warrant m_w. The certificate verification phase involves verification of P's proxy signature (along with verification of H_c's signature on the warrant). Given this, we can now prove the security of the proposed authorization protocol by proving the security of the underlying proxy signature scheme against forgery.

The security of the trapdoor hash-based proxy signature scheme is based on the formal security model by Chandrasekhar et al. [134], which, in turn, is closely related to the model by Boldyreva et al. [143]. The model involves a multi-party setting, with several entities (in our case, both patients and HCOs) having (private, public) key pairs registered with some public authority (in our case, the HIE cloud). The

adversary has the ability to play the role of any arbitrary entity except a single honest entity. The adversary achieves this by having the ability to corrupt entities and learn their private keys or by adding new arbitrary entities and registering their public keys (for which the corresponding private key need not be known).

The adversary is given access to standard and proxy signing oracles, and can interact multiple times with the honest entity, playing the role of different entities each time. We also assume that the adversary controls all communications. Given these abilities, the adversary attempts to forge a regular signature (called a type 1 forgery) or a proxy signature of the single honest entity. In case of a forged proxy signature, the adversary can output a forgery where the honest entity is either playing the role of a proxy (called a type 2 forgery) or a delegator (called a type 3 forgery). For a detailed explanation of the formal model, the reader is referred to [134] [143], where they define the security of a proxy signature scheme against an adaptive chosen message attack.

Now, given an adversary that succeeds in producing a forged regular or proxy signature, we can construct an adversary that breaks the well known DL assumption in the random oracle model (where the hash functions G and H behave as random oracles). The proof is a straightforward adaptation of that given by Chandrasekhar et al. [134]. In short, a regular signature (or a type 1) forgery results in a forged Schnorr signature, which is known to be secure under the DL assumption. A type 2 forgery results in a trapdoor collision forgery in the `DL-mDTH` scheme, which showed to be secure under the DL assumption in Theorem 1. And finally, a type 3 forgery results in an oracle replay attack that also breaks the DL assumption. Thus, the proposed

trapdoor hash-based proxy signature scheme is secure against adaptive chosen message attacks in the random oracle model [162] under the well-known DL assumption. This, in turn, establishes the security of the proposed authorization protocol. □

5.5 Evaluation

The access control protocol is built using a trapdoor hash-based proxy signature scheme. However, we can also use other proxy signature schemes to construct the protocol. Table 5.1 shows a direct comparison of the costs and properties associated with the access control protocol using the trapdoor hash-based proxy signature scheme against using the proxy signature schemes developed by Mambo et al. (`MUO`) [141], Kim et al. (`KPW`) [144], Petersen et al. (`PH`) [163], Lee et al. (`MLKK`) [148], Huang et al. (`HSMW`) [164] and Zhang et al. (`ZNS`) [165].

The performance characteristics of the `KPW` protocol are the same as those of its provably secure variant by Boldyreva et al. [143] and its proxy non-designated variant by Lee et al. [145]. The scheme by Lee et al. [145], however, requires a secure communication channel between the delegator and the proxy. Lee et al. (`MLKK`) [148] proposed a variant of the scheme in [145] to overcome this weakness. For the sake of uniformity in comparison, we consider a security benchmark of 1024 bits — the system parameters p and q of the proposed scheme, `MUO`, `KPW`, `MLKK`, and `PH` are 1024-bit and 160-bit primes, respectively. Also, we assume the employment of the Schnorr signature for proxy signature generation in the `MUO`, `KPW`, `PH` and `MLKK` schemes.

As shown in Table 5.1, the computation overhead of the access control protocol can be divided into the cost of certificate generation and verification, where the cost of

Table 5.1: Performance Comparison with security benchmark of 1024-bits. e: modular exponentiation, s: scalar multiplications, p: pairing computation; †: subsequent verification overhead of authorization certificate with cached signed authorization template, ‡: system parameters require up to $10KB$ of additional storage [3], $*$: excluding the size of warrant and message

	MUO	PH	MLKK	ZNS	KPW	HSMW	Proposed
Certificate Generation	$4e$	$5e$	$5e$	$4s + 2p$	$4e$	$8s + 2p$	$7e$
Subsequent Certificate Generation	$1e$	$1e$	$1e$	$2s$	$1e$	$5s$	$1e$
Certificate Verification	$4e$	$3e$	$4e$	$1s + 2p$	$4e(2e\dagger)$	$5p$	$5e(2e\dagger)$
Public Key Size (bits)	2048			1532	2048	$1532^{\ddagger}$	2048
Certificate Size (bits)*	1344		1504	160	1344	480	640
Secure Channel	Y	N	N	N	N	N	N
Provably Secure	N	N	N	N	Y	Y	Y

certificate generation includes the operations necessary to generate the signed authorization template by the HCO. The subsequent certificate generation cost in Table 5.1 corresponds to the case where a patient stores the signed authorization template to save the cost of re-verifying the delegating HCO's signature on the warrant. We observe that the access control protocol using the trapdoor hash-based proxy signature scheme achieves the best overall efficiency when compared to using other well-known proxy signature schemes for subsequent certificate generation and verification. More specifically, the protocol is as efficient as the most efficient proxy signature scheme, `KPW`, and is 33% more efficient compared to the next most efficient proxy signature scheme, `PH` (which is not provably secure). Although the certificate generation process is expensive in the proposed scheme, we argue that this step would not be performed often, and the majority of computational overhead would stem from any subsequent certificate generation and verification. Moreover, the cost of certificate generation is split between the HCO and patient, where the patient only incurs 4 (of the total 7) exponentiations. The `HSMW` and `ZNS` schemes use considerably more expensive bilinear pairing operations in the delegation and proxy signature verification phases. For instance, the cost of computing a single pairing can equal approximately 11110 multiplications in $\mathbb{Z}_q$, where q is a 171-bit prime (for security benchmark of 1024-bits) [166], which is significantly higher than the cost of exponentiations with 160-bit exponents.

Critiquing the access control protocol, we observe that it can also be built using sanitizable signature schemes [136] [137]. However, sanitizable signature schemes are built to achieve significantly different security properties than what is required for the

proposed authorization protocol. More specifically, sanitizable signature schemes are designed to allow a designated entity, called a sanitizer, to modify parts of a message while maintaining privacy (sanitized message does not reveal anything about the original data) and transparency (not being able to determine whether a message hash has been sanitized), among other properties [137]. Also, the property of transparency of sanitizable signatures leads to increased complexity, and a linear increase in the overhead for computing the sanitized signature. Thus, we conclude that sanitizable signatures are unsuitable for constructing the proposed authorization protocol.

For the proposed scheme, `MUO`, `KPW`, `MLKK` and `PH`, the size of the long-term public key, excluding shared components (primes p and q), equals 2048-bits. The pairing-based schemes, `HSMW` and `ZNS`, use public keys of size 1532-bits. The proposed scheme also produces the smallest proxy signatures compared to `MUO`, `KPW`, `PH` and `MLKK`. Even though the `HSMW` and `ZNS` schemes produce smaller signatures, they suffer from significantly higher computational overhead, as mentioned earlier. Thus, the proposed authorization protocol using our trapdoor hash-based proxy signature scheme achieves the best overall performance compared to other proxy signature schemes in the literature, while being provably secure.

Chapter 6 Summary and Conclusions

Patients' health records should follow them wherever and whenever needed, despite barriers that may occur due to the involvement of multiple facilities or different geographic areas. Thus, HIE governance should be done through effective collaboration between entities while considering implementation and security costs. HIE should be done with regards to privacy policies, strictly limiting the access to patients records in order to prevent any outsiders from gaining unauthorized access to the system. The information should be exchanged among entities while maintaining confidentiality and without leaking out of the system, which may occur because of the faults of the system itself or liability issues. Attackers aim to exploit the patients' records in the healthcare and medical sectors since the records contain very critical data. The value of medical records is sustained compared to other documents. When designing HIE systems, authentication, authorization, privacy, confidentiality, integrity, and auditability must be maintained.

The protocol discussed in Chapter 4 provides HCOs with mutual authentication, patient identification, patient authentication, patient consent, and symmetric key generation. The protocol mutually authenticates communication parties before preparing the patient's EHR for exchange. The interaction between communicating parties during authentication affirms the identities of the parties involved in the process. Additionally, the protocol authenticates the patient to its home HCO using the patient's assigned username and the stored hash of the patient's password. The

password supplied by the patient is used to compute the hash and is never stored by **New**.

To ensure the confidentiality of the health information during transmission, the symmetric key K_{Cons} is used for encryption. The symmetric key K_{Cons} is computed using the hash of the patient's password ($P_{current}^{H(password)}$) and **Current**'s nonce ($N_{Current}$), which is randomly generated by **Current** for each communication session. The use of the hash of the patient's password as a part of the consent key ensures that **New** will not be able to generate the correct session key without the consent of the patient implied by having the patient enter his/her password at **New**. The use of fresh nonce by **Current** ensures the generation of a new symmetric consent key for each session and avoids the reuse of older keys. The symmetric key K_{Cons} is simultaneously generated by **Current** and **New** and is never exchanged. Thus, it is impossible for an adversary to find a key (K_{Cons}) that can compromise the confidentiality of the health information exchanged.

Patients are given the ability to authorize the retrieval of only a certain category (type) of medical information. Patients' requested health information is prepared in four simple steps. Requested health information is encrypted using a symmetric key generated simultaneously at the involved healthcare organizations and reveals nothing about the content of the message. The protocol does not link patients' identity to their health information during the transmission of the exchanged health information. The integrity of the requested health information is checked to detect any possible alteration or distortion of information. The Audit System (AS) maintains adequate information for all incoming and outgoing exchange requests which allows auditing

at any time.

The privacy protocol presented in Chapter 4 is simple, secure, and does not require the usage of any special hardware or smart cards by the patient, whereas [52] requires the usage of temper-resistant and specially designed hardware to interact with high density SmartCards. The privacy protocol uses only three message exchanges for authentication and one message exchange to retrieve the health information, whereas [54] uses a total of nine message exchanges to retrieve the health information.

We conducted security analysis of the protocol which demonstrated that our protocol meets the security standards defined in the technical safeguards of the HIPAA security rule, whereas other protocol, such as [55] and [57], do not provide adequate security analysis for their proposals. We showed that our protocol maintains the transmission security, access control, integrity, and audit requirements of the HIPAA security rule. Also, the protocol prevents man-in-the-middle attacks and detect replay messages. In summary, the proposed protocol maintains authentication, authorization, privacy, confidentiality, integrity, and auditability.

In Chapter 5, a novel discrete log-based trapdoor hashing scheme that is efficient and secure against collision forgery and key-exposure was developed. Then, a novel proxy signature-based access control protocol for cloud based HIE was presented. The access control protocol allows patients to interact with their current HCOs in order to obtain an *authorization template*. The authorization template provides the patient with a standard signature on the trapdoor hash of a space descriptor (warrant). Patients can then create *authorization certificates* to allow other HCOs to access their

records from the HIE-cloud. This is done by generating a trapdoor collision between the warrant and the specific authorization parameters that specify the new HCO that needs to be granted access to patient data, the validity period of the authorization, and the types of records that can be accessed. After generating an authorization certificate A_C for an HCO H_n, the patient can store the authorization template A_T to grant authorizations to other HCOs, or modify existing authorizations for a previously authorized HCO. The patient generates subsequent (or modifies existing) authorization certificates simply by creating new authorization parameters m'_p and computing a trapdoor hash collision with m_w. The patient need not re-verify σ, which provides significant savings in computational costs.

The HIE-cloud can verify *authorization certificates* using public keys of the patient and original HCO. The access control protocol ensures that such authorizations are authentic with respect to both the original HCO and the patient granting access. In addition, the protocol allows patients to authorize the sharing of specific medical information with specific HCOs, which helps prevent any undesired or unintentional leakage of health information. Just as a patient can re-use a signed authorization template to save on computation, the HIE cloud can also save the signed authorization template and avoid the cost of re-verifying H_c's signature on the warrant. Every time the HIE cloud receives a new authorization certificate A'_C generated by the same combination of H_c and P, the HIE cloud can simply compare the t and h parameters in A'_C with those in the previously received A_C to verify the collision between trapdoor hashes of the warrant and the new authorization parameters.

The presented access control protocol achieves the best overall efficiency when

compared to other well-known proxy signature schemes for certificate generation (A_T and A_C), subsequent certificate generation (A_C only), and certificate verification. More specifically, the proposed scheme is as efficient as the most efficient proxy signature scheme, `KPW`, and is 33% more efficient than the next most efficient proxy signature scheme, `PH` (which is not provably secure). Although the certificate generation process is expensive in the proposed scheme, we argue that this step would not be performed often, and the majority of computational overhead would stem from any subsequent certificate generation and verification. Moreover, the cost of certificate generation is split between the HCO and patient, where the patient only incurs 4 (of the total 7) exponentiations. Finally, the proposed authorization protocol using our trapdoor hash-based proxy signature scheme achieves the best overall performance compared to other proxy signature schemes in the literature, while being provably secure.

Bibliography

[1] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2010.

[2] N. Yoosuf. Attribute based encryption (ABE) and its two flavors. [Online]. Available: http://mohamednabeel.blogspot.co.ke/2012/03/aattribute-based-encryption-abe-and-its.html

[3] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters, "Sequential Aggregate Signatures and Multisignatures Without Random Oracles," in *Proceedings of EUROCRYPT, the 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, ser. LNCS, S. Vaudenay, Ed., vol. 4004. Springer, 2006, pp. 465 – 485.

[4] C. Williams, F. Mostashari, K. Mertz, E. Hogin, and P. Atwal, "From the office of the national coordinator: The strategy for advancing the exchange of health information," *Health Affairs*, vol. 31, no. 3, pp. 527–536, 2012. [Online]. Available: http://content.healthaffairs.org/content/31/3/527.abstract

[5] L. W. Clark, "Health Information Technology, Patient Data and Health Care Reform: Rewards and Risks in the New Ecosystem," *The Computer & Internet Lawyer*, vol. 32, no. 2, February 2015.

[6] D. McGraw, J. X. Dempsey, L. Harris, and J. Goldman, "Privacy as an enabler, not an impediment: Building trust into health information exchange," *Health Affairs*, vol. 28, no. 2, pp. 416–427, 2009. [Online]. Available: http://content.healthaffairs.org/content/28/2/416.abstract

[7] J. D. Szerejko, "Reading Between the Lines of Electronic Health Records: The Health Information Technology for Economic and Clinical Health Act and Its Implications for Health Care Fraud and Information Security," *Connecticut Law Review*, vol. 47, no. 4, May 2015.

[8] A. T. Strauss, D. A. Martinez, A. Garcia-Arce, S. Taylor, C. Mateja, P. J. Fabri, and J. L. Zayas-Castro, "A user needs assessment to inform health information exchange design and implementation," *BMC Medical Informatics and Decision Making*, vol. 15, no. 1, pp. 1–11, 2015. [Online]. Available: http://dx.doi.org/10.1186/s12911-015-0207-x

[9] C. P. Saiz, I. C. Markina, A. A. Yarza, M. R. López, and L. E. Eizaguirre, "Disclosure of Health Information: a challenge of trust between the various sectors involved," *Revista Latina de Comunicación Social*, vol. 69, pp. 125 – 151, 2014.

[10] U.S. Department of Health and Human Services, "Summary of the HIPAA Privacy Rule." [Online]. Available: http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/index.html

[11] U.S. Department of Health and Human Services, "Summary of the HIPAA Security Rule." [Online]. Available: http://www.hhs.gov/ocr/privacy/hipaa/understanding/srsummary.html

[12] A. Appari and M. E. Johnson, "Information security and privacy in healthcare: current state of research," *International journal of Internet and enterprise management*, vol. 6, no. 4, pp. 279–314, 2010.

[13] Center for Medicare and Medicaid Services, "The Medicare and Medicaid Electronic Health Records (EHR) Incentive Programs," 2013. [Online]. Available: http://www.cms.gov/Regulations-and-Guidance/Legislation/EHRIncentivePrograms/index.html

[14] J. L. Fernández-Alemán, I. C. Señor, P. Á. O. Lozoya, and A. Toval, "Security and privacy in electronic health records: A systematic literature review," *Journal of Biomedical Informatics*, vol. 46, no. 3, pp. 541 – 562, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1532046412001864

[15] F. H. Cate, "Information security breaches and the threat to consumers," *Published by the Center for Information Policy Leadership*, 2005.

[16] F. H. Cate, "Information security breaches: Looking back & thinking ahead," *Published by the Centre for Information Policy Leadership*, 2008.

[17] A. Gawlik, L. Köster, H. Mahmoodi, and M. Winandy, "Requirements for integrating end-to-end security into large-scale ehr systems," in *University of Amsterdam, Amsterdam Privacy Conference*, 2012.

[18] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating security and systems engineering*. John Wiley & Sons, 2013.

[19] F. Rezaeibagha, K. T. Win, W. Susilo *et al.*, "A systematic literature review on security and privacy of electronic health record systems: technical perspectives," *Health Information Management Journal*, vol. 44, no. 3, p. 23, 2015.

[20] Health Level Seven International. HL7. [Online]. Available: http://www.hl7.org

[21] HL7 Modeling and Methodology Committee, "Message Development Framework," 1999. [Online]. Available: http://www.hl7.org/documentcenter/public_temp_113347F9-1C23-BA17-0C597A03919110A0/wg/mnm/Mdf99.pdf

[22] B. Smith and W. Ceusters, "HL7 RIM: An Incoherent Standard," pp. 124: 133–138, August 2006.

[23] openEHR Foundation. openEHR: An open domain-driven platform for developing flexible e-health systems. [Online]. Available: http://www.openehr.org

[24] D. Gritzalis and C. Lambrinoudakis, "A security architecture for interconnecting health information systems," *International Journal of Medical Informatics*, vol. 73, no. 3, pp. 305–309, 2004.

[25] D. J. Brailer, "Interoperability: the key to the future health care system," *Health affairs*, vol. 24, p. W5, 2005.

[26] M. M. Goldstein, A. L. Rein, P. P. Hughes, J. J. K. Lappas, S. A. Weinstein, and B. Williams, "CONSUMER CONSENT OPTIONS FOR ELECTRONIC HEALTH INFORMATION EXCHANGE: POLICY CONSIDERATIONS AND ANALYSIS," White Paper, March 2010. [Online]. Available: https://www.healthit.gov/sites/default/files/privacy-security/cover-page-and-executive-summary-032610.pdf

[27] L. Agha, "The effects of health information technology on the costs and quality of medical care," *Journal of Health Economics*, vol. 34, pp. 19 – 30, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167629613001720

[28] iSheriff, "The New Heathcare Crisis: Cybercrime, Patient Records and Information Security," White Paper, 2015.

[29] D. Mohammed, R. Mariani, and S. Mohammed, "Cybersecurity challenges and compliance issues within the us healthcare sector," *International Journal of Business and Social Research*, vol. 5, no. 2, pp. 55–66, 2015.

[30] S. Pettypiece, "Rising Cyber Attacks Costing Health System $6 Billion Annually." [Online]. Available: http://www.bloomberg.com/news/articles/2015-05-07/rising-cyber-attacks-costing-health-system-6-billion-annually

[31] D. Gleeson and S. Friel, "Emerging threats to public health from regional trade agreements," *The Lancet*, vol. 381, no. 9876, pp. 1507–1509, 2013.

[32] M. Kiely. St. joseph health system breach leaves thousands of records vulnerable. [Online]. Available: http://www.theeagle.com/news/local/st-joseph-health-system-breach-leaves-thousands-of-records-vulnerable/article_541d3f86-8a43-5913-af16-d7cd0b847c0a.html

[33] C. Falls. St. joseph health system confirms data security incident. [Online]. Available: http://www.kbtx.com/home/headlines/St-Joseph-Health-System-Confirms-Data-Security-Incident-243558231.html

[34] Joseph Conn. Patient data held for ransom at rural illinois hospital. [Online]. Available: http://www.modernhealthcare.com/article/20141217/NEWS/312179948

[35] Jessica Bartlett. Partners healthcare reports data breach. [Online]. Available: http://www.bizjournals.com/boston/blog/health-care/2015/04/partners-healthcare-reports-potential-data-breach.html

[36] D. Wu and S. Shan, "Meta-analysis of network information security and web data mining techniques," in *First International Conference on Information Sciences, Machinery, Materials and Energy*. Atlantis Press, 2015.

[37] H. JOURNAL. Ohiohealth reports loss of flash drive containing 1,006 protected health records. [Online]. Available: OHIOHEALTHREPORTSLOSSOFFLASHDRIVECONTAINING1,006PROTECTEDHEALTHRECORDS

[38] S. Heath. Ohiohealth missing flash drive leads to health data breach. [Online]. Available: http://healthitsecurity.com/news/ohiohealth-missing-flash-drive-leads-to-health-data-breach

[39] A. Jayanthi. University hospitals fires employee for inappropriately accessing medical records. [Online]. Available: http://www.beckershospitalreview.com/healthcare-information-technology/university-hospitals-fires-employee-for-inappropriately-accessing-medical-records.html

[40] B. Dicken. Elyria hospital patients' records improperly accessed by worker.

[41] M. K. McGee. Clinic breach involved authorized user. [Online]. Available: http://www.databreachtoday.com/clinic-breach-involved-authorized-user-a-8677

[42] H. Landi. Children's medical clinics of east texas reports data breach of 16k pediatric patient records. [Online]. Available: http://www.healthcare-informatics.com/news-item/children-s-medical-clinics-east-texas-reports-data-breach-16k-children

[43] Richard Winton. Hollywood hospital pays $17,000 in bitcoin to hackers; FBI investigating. [Online]. Available: http://www.latimes.com/business/technology/la-me-ln-hollywood-hospital-bitcoin-20160217-story.html

[44] S. Wong, "Pay up or your medical records will be toast." *New Scientist*, vol. 229, no. 3062, p. 26, 2016. [Online]. Available: http://ezproxy.uky.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=113270306&site=ehost-live&scope=site

[45] E. Accreditation, "Making wireless," 2016.

[46] M. D. Huesch, "Privacy Threats When Seeking Online Health Information," *JAMA Internal Medicine*, vol. 173, no. 19, pp. 1838–1840, 2013. [Online]. Available: +http://dx.doi.org/10.1001/jamainternmed.2013.7795

[47] FBI Cyber Division, "Health Care Systems and Medical Devices at Risk for Increased Cyber Intrusions for Financial Gain," April 2014.

[48] R. Luna, M. Myhra, E. Rhine, R. Sullivan, and C. Kruse, "Cyber threats to health information systems: A systematic review." *Technology and health care: official journal of the European Society for Engineering and Medicine*, 2015.

[49] J. S. McCullough, S. Parente, and R. Town, "Health information technology and patient outcomes: The role of organizational and informational complementarities," National Bureau of Economic Research, Tech. Rep., 2013.

[50] D. C. Streeter, "The effect of human error on modern security breaches," *Strategic Informer: Student Publication of the Strategic Intelligence Society*, vol. 1, no. 3, p. 2, 2015.

[51] K. T. Win, "A review of security of electronic health records," *Health Information Management*, vol. 34, no. 1, pp. 13–18, 2005.

[52] W. Yu and M. Chekhanovskiy, "An Electronic Health Record Content Protection System Using SmartCard and PMR," in *The 9th International Conference on e-Health Networking, Application and Services*, June 2007, pp. 11–18.

[53] K. E. Mayes and K. Markantonakis, "On the Potential of High Density Smart Cards," *Information Security Technical Report*, vol. 11, no. 3, pp. 147 – 153, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S136341270600032X

[54] J. Hu, H.-H. Chen, and T.-W. Hou, "A Hybrid Public Key Infrastructure Solution (HPKI) for HIPAA Privacy/Security Regulations," *Computer Standards and Interfaces*, vol. 32, no. 5–6, pp. 274 – 280, 2010, information and communications security, privacy and trust: Standards and Regulations. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0920548909000324

[55] S. Haas, S. Wohlgemuth, I. Echizen, N. Sonehara, and G. Müller, "Aspects of privacy for electronic health records," *International Journal of Medical Informatics*, vol. 80, no. 2, pp. e26 – e31, 2011, special Issue: Security in Health Information Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1386505610001723

[56] Wohlgemuth, Sven and Müller, Günter, "Privacy with Delegation of Rights by Identity Management," in *Emerging Trends in Information and Communication Security*, ser. Lecture Notes in Computer Science, G. Müller, Ed. Springer Berlin Heidelberg, 2006, vol. 3995, pp. 175–190.

[57] J. Vergara and T. Johnson, "Chains of Trust for On-Demand Requests of Electronic Health Records," in *Proceedings of the Wireless Communications and Networking Conference (WCNC)*, March 2015, pp. 2227–2232.

[58] C. Wang, X. Liu, and W. Li, "Implementing a personal health record cloud platform using ciphertext-policy attribute-based encryption," in *Intelligent Networking and Collaborative Systems (INCoS), 2012 4th International Conference on*, Sept 2012, pp. 8–14.

[59] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 1, pp. 131–143, Jan 2013.

[60] Y. Tong, J. Sun, S. S. M. Chow, and P. Li, "Cloud-assisted mobile-access of health data with privacy and auditability," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 2, pp. 419–429, March 2014.

[61] B. Fabian, T. Ermakova, and P. Junghanns, "Collaborative and secure sharing of healthcare data in multi-clouds," *Information Systems*, vol. 48, pp. 132 – 150, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S030643791400088X

[62] J. L. F. Alemán, I. C. Señor, P. Á. O. Lozoya, and A. Toval, "Security and Privacy in Electronic Health Records: A Systematic Literature Review," *Journal of Biomedical Informatics*, vol. 46, no. 3, pp. 541 – 562, 2013.

[63] L. Guo, C. Zhang, J. Sun, and Y. Fang, "Paas: A privacy-preserving attribute-based authentication system for ehealth networks," in *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on*, June 2012, pp. 224–233.

[64] W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, November 1976.

[65] V. Silva-Garcia, R. Flores-Carapia, C. Renteria-Márquez, and B. Luna-Benoso, "The Triple-DES-96 Cryptographic System," *International Journal of Contemporary Mathematical Sciences*, vol. 8, no. 19, pp. 925–934, 2013.

[66] K. Sindhuja and P. Devi, "A Symmetric Key Encryption Technique Using Genetic Algorithm," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 1, pp. 414–6, 2014.

[67] N. J. AlFardan, D. J. Bernstein, K. G. Paterson, B. Poettering, and J. C. Schuldt, "On the security of rc4 in tls." in *Usenix security*, vol. 2013. Washington DC, USA, 2013.

[68] H. Kodera, M. Yanagisawa, and N. Togawa, "Scan-based attack against des and triple des cryptosystems using scan signatures," *Information and Media Technologies*, vol. 8, no. 3, pp. 867–874, 2013.

[69] E. Biham and A. Shamir, *Differential Cryptanalysis of the Data Encryption Standard*. London, UK, UK: Springer-Verlag, 1993.

[70] M. Agrawal and P. Mishra, "A comparative survey on symmetric key encryption techniques," *International Journal on Computer Science and Engineering*, vol. 4, no. 5, p. 877, 2012.

[71] J. Daemen and V. Rijmen, *The Design of Rijndael*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2002.

[72] T. Hoang and V. L. Nguyen, "An efficient fpga implementation of the advanced encryption standard algorithm," in *2012 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*. IEEE, 2012, pp. 1–4.

[73] H. Qiu and G. Memmi, "Fast selective encryption methods for bitmap images," *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, vol. 6, no. 3, pp. 51–69, 2015.

[74] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, 1978.

[75] T. Kleinjung, J. W. Bos, A. K. Lenstra, D. A. Osvik, K. Aoki, S. Contini, J. Franke, E. Thomé, P. Jermini, M. Thiémard, P. Leyland, P. L. Montgomery, A. Timofeev, and H. Stockinger, "A heterogeneous computing environment to solve the 768-bit rsa challenge," *Cluster Computing*, vol. 15, no. 1, pp. 53–68, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10586-010-0149-0

[76] R. Sinha, H. K. Srivastava, and S. Gupta, "Performance based comparison study of rsa and elliptic curve cryptography," *International Journal of Scientific & Engineering Research*, vol. 4, no. 5, pp. 720–725, 2013.

[77] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed. John Wiley & Sons, Inc., 1996.

[78] K. Hickman and T. Elgamal, "The ssl protocol," *Netscape Communications Corp*, vol. 501, 1995.

[79] B. Möller, T. Duong, and K. Kotowicz, "This poodle bites: exploiting the ssl 3.0 fallback," *Google, Sep*, 2014.

[80] R. Hat. POODLE: SSLv3 vulnerability. [Online]. Available: https://access.redhat.com/articles/1232123

[81] S. Turner and T. Polk, "Prohibiting secure sockets layer (ssl) version 2.0," *Internet Engineering Task Force (IETF)*, vol. RFC 6176, 2011.

[82] A. Langley, A. Pironti, R. Barnes, and M. Thomson, "Deprecating secure sockets layer version 3.0," *Internet Engineering Task Force (IETF)*, vol. RFC 7568, 2015.

[83] M. Marlinspike, "sslstrip Tool," 2009. [Online]. Available: http://www.thoughtcrime.org/software/sslstrip/

[84] M. Marlinspike, "sslsniff Tool," 2009. [Online]. Available: http://www.thoughtcrime.org/software/sslsniff/

[85] *Black Hat DC*, February 16-19 2009. [Online]. Available: https://www.blackhat.com/html/bh-dc-09/bh-dc-09-main.html

[86] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 89–98. [Online]. Available: http://doi.acm.org/10.1145/1180405.1180418

[87] A. Sahai and B. Waters, "Fuzzy Identity-Based Encryption," in *Advances in Cryptology – EUROCRYPT 2005*, ser. Lecture Notes in Computer Science, R. Cramer, Ed. Springer Berlin Heidelberg, 2005, vol. 3494, pp. 457–473.

[88] S. G. Weber, "A Hybrid Attribute-Based Encryption Technique Supporting Expressive Policies and Dynamic Attributes," *Information Security Journal: A Global Perspective*, vol. 21, no. 6, pp. 297–305, Jan. 2012. [Online]. Available: http://dx.doi.org/10.1080/19393555.2012.738374

[89] N. Attrapadung, B. Libert, and E. de Panafieu, *Public Key Cryptography – PKC 2011: 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, ch. Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts, pp. 90–108. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-19379-8_6

[90] N. Attrapadung, J. Herranz, F. Laguillaumie, B. Libert, E. de Panafieu, and C. Ràfols, "Attribute-based encryption schemes with constant-size ciphertexts," *Theoretical Computer Science*, vol. 422, pp. 15 – 38, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397511009649

[91] C. Wang and J. Luo, "An efficient key-policy attribute-based encryption scheme with constant ciphertext length," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[92] V. Božović, D. Socek, R. Steinwandt, and V. I. Villányi, "Multi-authority attribute-based encryption with honest-but-curious central authority," *International Journal of Computer Mathematics*, vol. 89, no. 3, pp. 268–283, 2012.

[93] S. Gorbunov, V. Vaikuntanathan, and H. Wee, "Attribute-based encryption for circuits," *J. ACM*, vol. 62, no. 6, pp. 45:1–45:33, Dec. 2015. [Online]. Available: http://doi.acm.org/10.1145/2824233

[94] D. Koo, J. Hur, and H. Yoon, "Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage," *Computers & Electrical Engineering*, vol. 39, no. 1, pp. 34–46, Jan. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.compeleceng.2012.11.002

[95] H. Lin, Z. Cao, X. Liang, and J. Shao, "Secure threshold multi authority attribute based encryption without a central authority," *Inf. Sci.*, vol. 180, no. 13, pp. 2618–2632, Jul. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2010.03.004

[96] H. Deng, Q. Wu, B. Qin, J. Domingo-Ferrer, L. Zhang, J. Liu, and W. Shi, "Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts," *Information Sciences*, vol. 275, pp. 370–384, August 2014.

[97] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, *Information Security Applications: 10th International Workshop, WISA 2009, Busan, Korea, August 25-27, 2009, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ch. Mediated Ciphertext-Policy Attribute-Based Encryption and Its Application, pp. 309–323. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10838-9_23

[98] G. Wang, Q. Liu, J. Wu, and M. Guo, "Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers," *Computers & Security*, vol. 30, no. 5, pp. 320–331, Jul. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2011.05.006

[99] X. Wang and H. Yu, "How to Break MD5 and Other Hash Functions," in *Advances in Cryptology – EUROCRYPT*, ser. Lecture Notes in Computer Science, R. Cramer, Ed. Springer Berlin Heidelberg, 2005, vol. 3494, pp. 19–35.

[100] M. Stevens, A. Lenstra, and B. de Weger, "Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities," in *Advances in Cryptology - EUROCRYPT*, ser. Lecture Notes in Computer Science, M. Naor, Ed. Springer Berlin Heidelberg, 2007, vol. 4515, pp. 1–22.

[101] R. Housley, W. Polk, W. Ford, and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," Tech. Rep., 2002.

[102] M. Stevens, A. Sotirov, J. Appelbaum, A. Lenstra, D. Molnar, D. Osvik, and B. de Weger, "Short chosen-prefix collisions for md5 and the creation of a rogue ca certificate," in *Advances in Cryptology - CRYPTO 2009*, ser. Lecture Notes in Computer Science, S. Halevi, Ed. Springer Berlin Heidelberg, 2009, vol. 5677, pp. 55–69.

[103] sKyWIper Analysis Team, "sKyWIper: A Complex Malware for Targeted Attacks," Budapest University of Technology and Economics, Tech. Rep., May 2012. [Online]. Available: http://www.crysys.hu/skywiper/skywiper.pdf

[104] Symantec, "Flamer: Highly Sophisticated and Discreet Threat Targets the Middle East," May 2012. [Online]. Available: http://www.symantec.com/connect/blogs/flamer-highly-sophisticated-and-discreet-threat-targets-middle-east

[105] D. McElroy and C. Williams. (2012, May) Flame: World's Most Complex Computer Virus Exposed. [Online]. Available: http://www.telegraph.co.uk/news/worldnews/middleeast/iran/9295938/Flame-worlds-most-complex-computer-virus-exposed.html

[106] N. McHugh, "How I Made Two PHP Files with the Same MD5 Hash," October 2014. [Online]. Available: http://natmchugh.blogspot.co.uk/2014/10/how-i-made-two-php-files-with-same-md5.html

[107] N. McHugh, "How I Created Two Images with the Same MD5 Hash," October 2014. [Online]. Available: http://natmchugh.blogspot.co.uk/2014/10/how-i-created-two-images-with-same-md5.html

[108] N. McHugh, "Three Way MD5 Collision," November 2014. [Online]. Available: http://natmchugh.blogspot.co.uk/2014/11/three-way-md5-collision.html

[109] F. Chabaud and A. Joux, *Advances in Cryptology — CRYPTO '98: 18th Annual International Cryptology Conference Santa Barbara, California, USA August 23–27, 1998 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, ch. Differential collisions in SHA-0, pp. 56–71. [Online]. Available: http://dx.doi.org/10.1007/BFb0055720

[110] X. Wang, H. Yu, and Y. L. Yin, "Efficient collision search attacks on sha-0," in *Advances in Cryptology–CRYPTO 2005*. Springer, 2005, pp. 1–16.

[111] A. Mohamed and A. Nadjia, "Sha-2 hardware core for virtex-5 fpga," in *12th International Multi-Conference on Systems, Signals Devices (SSD)*, March 2015, pp. 1–5.

[112] S. Manuel and T. Peyrin, *Fast Software Encryption: 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, ch. Collisions on SHA-0 in One Hour, pp. 16–35. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-71039-4_2

[113] P. Morawiecki and M. Srebrny, "A sat-based preimage analysis of reduced keccak hash functions," *Information Processing Letters*, vol. 113, no. 10, pp. 392–397, 2013.

[114] V. Rijmen and E. Oswald, *Topics in Cryptology – CT-RSA 2005: The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, ch. Update on SHA-1, pp. 58–71. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30574-3_6

[115] X. Wang, Y. L. Yin, and H. Yu, *Advances in Cryptology – CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, ch. Finding Collisions in the Full SHA-1, pp. 17–36. [Online]. Available: http://dx.doi.org/10.1007/11535218_2

[116] Amerk. Sha1 deprecation policy. [Online]. Available: https://web.archive.org/web/20131113032317/http://blogs.technet.com/b/pki/archive/2013/11/12/sha1-deprecation-policy.aspx

[117] K. SUSENA and K. K. KUMAR, "Implementation of sha-1 hmac-sha-1 proof of ownership protocol on a mixed-breed cloud resemble for unharmed licensed deduplication," *International Journal of Scientific Engineering and Technology Research*, vol. 4, no. 28, pp. 5329–5332, July 2015.

[118] M. Stevens, P. Karpman, and T. Peyrin, "Freestart collision for full sha-1," Cryptology ePrint Archive, Report 2015/967, 2015, http://eprint.iacr.org/.

[119] N. Sklavos and O. Koufopavlou, "On the hardware implementations of the sha-2 (256, 384, 512) hash functions," in *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on*, vol. 5, May 2003, pp. V–153–V–156 vol.5.

[120] K. Aoki, J. Guo, K. Matusiewicz, Y. Sasaki, and L. Wang, *Advances in Cryptology – ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ch. Preimages for Step-Reduced SHA-2, pp. 578–597. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10366-7_34

[121] S. Indesteege, F. Mendel, B. Preneel, and C. Rechberger, *Selected Areas in Cryptography: 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ch. Collisions and Other Non-random Properties for Step-Reduced SHA-256, pp. 276–293. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04159-4_18

[122] S. K. Sanadhya and P. Sarkar, *Progress in Cryptology - INDOCRYPT 2008: 9th International Conference on Cryptology in India, Kharagpur, India, December 14-17, 2008. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, ch. New Collision Attacks against Up to 24-Step SHA-2, pp. 91–103. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-89754-5_8

[123] C. Dobraunig, M. Eichlseder, and F. Mendel, *Advances in Cryptology – ASIACRYPT 2015: 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 – December 3, 2015, Proceedings, Part II.*

Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, ch. Analysis of SHA-512/224 and SHA-512/256, pp. 612–630. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-48800-3_25

[124] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, *Advances in Cryptology – EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. Keccak, pp. 313–314. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38348-9_19

[125] N. I. of Standards and Technology. Nist releases sha-3 cryptographic hash standard. [Online]. Available: http://www.nist.gov/itl/csd/201508_sha3.cfm

[126] H. Krawczyk and T. Rabin, "Chameleon Signatures," in *Proceedings of NDSS, Network and Distributed System Security Symposium.* The Internet Society, 2000.

[127] G. Brassard, D. Chaum, and C. Crépeau, "Minimum disclosure proofs of knowledge." *Journal of Computer and System Sciences*, vol. 37, no. 2, pp. 156–189, 1988.

[128] S. Chandrasekhar and M. Singhal, "Efficient and Scalable Query Authentication for Cloud-based Storage Systems with Multiple Data Sources," *to appear in IEEE Transactions on Services Computing*, 2016.

[129] G. Ateniese and B. de Medeiros, "Identity-Based Chameleon Hash and Applications," in *Proceedings of FC, the 8th International Conference on Financial Cryptography*, ser. LNCS, A. Juels, Ed., vol. 3110. Springer, 2004, pp. 164 – 180.

[130] X. Chen, F. Zhang, and K. Kim, "Chameleon hashing without key exposure," in *Proceedings of ISC, Information Security, 7th International Conference, Palo Alto, CA, USA, September 27-29*, ser. LNCS, K. Zhang and Y. Zheng, Eds., vol. 3225. Springer, 2004, pp. 87–98.

[131] G. Ateniese and B. de Medeiros, "On the Key Exposure Problem in Chameleon Hashes," in *Proceedings of SCN, the 4th International Conference on Security in Communication Networks*, ser. LNCS, C. Blundo and S. Cimato, Eds., vol. 3352. Springer, 2004, pp. 165 – 179.

[132] X. Chen, F. Zhang, H. Tian, B. Wei, W. Susilo, Y. Mu, H. Lee, and K. Kim, "Efficient Generic On-line/Off-line (threshold) Signatures Without Key Exposure," *Information Sciences*, vol. 178, no. 21, pp. 4192 – 4203, 2008.

[133] A. Shamir and Y. Tauman, "Improved Online/Offline Signature Schemes," in *Proceedings of CRYPTO, Advances in Cryptology, 21st Annual International Cryptology Conference*, ser. LNCS, J. Kilian, Ed., vol. 2139. Springer, 2001, pp. 355 – 367.

[134] S. Chandrasekhar, S. Chakrabarti, M. Singhal, and K. L. Calvert, "Efficient Proxy Signatures Based on Trapdoor Hash Functions," *IET Information Security, Special Issue: Selected papers on multi-agent and distributed information security*, vol. 4, no. 4, pp. 322 – 332, December 2010.

[135] M. Mehta and L. Harn, "Efficient One-time Proxy Signatures," *IEE Proceedings Communications*, vol. 152, no. 2, pp. 129 – 133, 2005.

[136] G. Ateniese, D. H. Chou, B. de Medeiros, and G. Tsudik, "Sanitizable Signatures," in *Proceedings of ESORICS, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14*, ser. LNCS, S. D. C. di Vimercati, P. F. Syverson, and D. Gollmann, Eds., vol. 3679. Springer, 2005, pp. 159 – 177.

[137] C. Brzuska, M. Fischlin, T. Freudenreich, A. Lehmann, M. Page, J. Schelbert, D. Schröder, and F. Volk, "Security of Sanitizable Signatures Revisited," in *In Proceedings of PKC, the 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20*, ser. LNCS, vol. 5443. Springer, 2009, pp. 317 – 336.

[138] S. Chandrasekhar, S. Chakrabarti, and M. Singhal, "A Trapdoor Hash-Based Mechanism for Stream Authentication," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 5, pp. 699 – 713, 2012.

[139] S. Chandrasekhar and M. Singhal, "Multi-trapdoor Hash Funtions and Their Applications in Network Security," in *Proceedings of IEEE-CNS, Second IEEE Conference on Communications and Network Security, San Francisco, California, San Francisco, California, USA, October 29-31.* IEEE, 2014, pp. 472 – 480.

[140] S. Chandrasekhar and M. Singhal, "Efficient and Scalable Aggregate Signcryption Scheme Based on Multi-trapdoor Hash Functions," in *Proceedings of CNS, IEEE Conference on Communications and Network Security, Florence, Italy, September 28-30.* IEEE, 2015, pp. 610 – 618.

[141] M. Mambo, K. Usuda, and E. Okamoto, "Proxy Signatures for Delegating Signing Operation," in *Proceedings of CCS, Third ACM Conference on Computer and Communications Security.* ACM Press, 1996, pp. 48 – 57.

[142] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure proxy signature schemes for delegation of signing rights," Cryptology ePrint Archive, Report 2003/096, 2008, http://eprint.iacr.org/2003/096.

[143] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure Proxy Signature Schemes for Delegation of Signing Rights," *Journal of Cryptology*, vol. 25, no. 1, pp. 57 – 115, 2012.

[144] S. Kim, S. Park, and D. Won, "Proxy Signatures, Revisited," in *Proceedings of ICICS, First International Conference on Information and Communication Security*, ser. LNCS, Y. Han, T. Okamoto, and S. Qing, Eds., vol. 1334. Springer, 1997, pp. 223 – 232.

[145] B. Lee, H. Kim, and K. Kim, "Strong Proxy Signature and its Application," in *Proceedings of SCIS, Symposium on Cryptography and Information Security Osio, Japan*, vol. 2/2, 2001, pp. 603 – 608.

[146] A. K. Awasthi and S. Lal, "Proxy Blind Signature Scheme," *Transactions on Cryptology*, vol. 2, no. 1, pp. 5 – 11, 2005.

[147] S. Lal and A. K. Awasthi, "A Scheme for Obtaining a Warrant Message from the Digital Proxy Signatures," Cryptology ePrint Archive, Report 2003/073, 2003, http://eprint.iacr.org/2003/073.

[148] J.-Y. Lee, J. H. Cheon, and S. Kim, "An Analysis of Proxy Signatures: Is a Secure Channel Necessary?" in *Proceedings of CT-RSA, Cryptographers' Track at the RSA Conference*, ser. LNCS, M. Joye, Ed., vol. 2612. Springer, 2003, pp. 68 – 79.

[149] T. Okamoto, M. Tada, and E. Okamoto, "Extended Proxy Signatures for Smart Cards," in *Proceedings of ISW, Second International Workshop on Information Security*, ser. LNCS, M. Mambo and Y. Zheng, Eds., vol. 1729. Springer, 1999, pp. 247 – 258.

[150] H.-M. Sun, "An Efficient Nonrepudiable Threshold Proxy Signature Scheme with Known Signers," *Computer Communications*, vol. 22, no. 8, pp. 717 – 722, 1999.

[151] H.-M. Sun and B.-T. Hsieh, "Remarks on Two Nonrepudiable Proxy Signature Schemes," in *Proceedings of the 9^{th} National Conference on Information Security*, 1999, pp. 241 – 246.

[152] T. Malkin, S. Obana, and M. Yung, "The Hierarchy of Key Evolving Signatures and a Characterization of Proxy Signatures," in *Proceedings of Eurocrypt, International Conference on the Theory and Applications of Cryptographic Techniques*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Springer, 2004, pp. 306 – 322.

[153] J. C. N. Schuldt, K. Matsuura, and K. G. Paterson, "Proxy Signatures Secure Against Proxy Key Exposure," in *Proceedings of PKC, the 11^{th} International Workshop on Practice and Theory in Public-Key Cryptography*, ser. LNCS, R. Cramer, Ed., vol. 4939. Springer, 2008, pp. 141 – 161.

[154] S. Santessona, M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure, Online Certificate Status Protocol - OCSP (RFC 6960)," 2013. [Online]. Available: https://tools.ietf.org/html/rfc6960

[155] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (RFC 5280)," 2008. [Online]. Available: https://tools.ietf.org/html/rfc5280

[156] P. Zheng, "Tradeoffs in Certificate Revocation Schemes," *SIGCOMM Computer Communications Review*, vol. 33, no. 2, pp. 103–112, Apr. 2003. [Online]. Available: http://doi.acm.org/10.1145/956981.956991

[157] A. Abbas and S. U. Khan, "A Review on the State-of-the-Art Privacy-Preserving Approaches in the e-Health Clouds," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 4, pp. 1431 – 1441, 2014.

[158] N. Li, J. Byun, and E. Bertino, "A Critique of the ANSI Standard on Role-Based Access Control," *IEEE Security & Privacy*, vol. 5, no. 6, pp. 41 – 49, 2007.

[159] R. Wu, G. Ahn, and H. Hu, "Secure Sharing of Electronic Health Records in Clouds," in *Proceedings of CollaborateCom, the 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing, Pittsburgh, PA, USA, October 14-17*. ICST / IEEE, 2012, pp. 711 – 718.

[160] C.-P. Schnorr, "Efficient Signature Generation by Smart Cards," *Journal of Cryptology*, vol. 4, no. 3, pp. 161 – 174, 1991.

[161] D. Pointcheval and J. Stern, "Security Proofs for Signature Schemes," in *Proceedings of EUROCRYPT, Advances in Cryptology, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996*, ser. LNCS, U. M. Maurer, Ed., vol. 1070. Springer, 1996, pp. 387 – 398.

[162] M. Bellare and P. Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols," in *Proceedings of CCS: First ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA*. ACM Press, 1993, pp. 62 – 73.

[163] H. Petersen and P. Horster, "Self-certified Keys – Concepts and Applications," in *Proceedings of CMS, the 3rd International Conference on Communications and Multimedia Security*. Chapman & Hall, 1997, pp. 102 – 116.

[164] X. Huang, W. Susilo, Y. Mu, and W. Wu, "Proxy Signature Without Random Oracles," in *Proceedings of MSN, Second International Conference on Mobile Ad-hoc and Sensor Networks*, ser. LNCS, J. Cao, I. Stojmenovic, X. Jia, and S. K. Das, Eds., vol. 4325. Springer, 2006, pp. 473 – 484.

[165] F. Zhang, R. Safavi-Naini, and W. Susilo, "An Efficient Signature Scheme from Bilinear Pairings and Its Applications," in *Proceedings of PKC, the 7th International Workshop on Theory and Practice in Public Key Cryptography*, ser.

LNCS, F. Bao, R. H. Deng, and J. Zhou, Eds., vol. 2947. Springer, 2004, pp. 277 – 290.

[166] P. S. L. M. Barreto, B. Lynn, and M. Scott, "On the Selection of Pairing-Friendly Groups," in *Proceedings of SAC: Tenth Annual International Workshop on Selected Areas in Cryptography, Revised Papers*, ser. LNCS, M. Matsui and R. J. Zuccherato, Eds., vol. 3006. Springer, 2003, pp. 17 – 25.

Vita

Personal Information:
Name: Ahmed Fouad Shedeed Ibrahim

Education:

- M.Sc. in Computer Science, University of Kentucky, 2014.
- Certificate in College Teaching and Learning, University of Kentucky, 2014.
- M.Sc. in Computer Science, Arab Academy for Science, Technology, and Maritime Transport - Egypt, 2009.
- B.Sc. in Computer Science, October 6 University - Egypt, 2004.

Employment:

- Spring 2016, 1 semester, Teaching Assistant, University of Kentucky.
- Spring 2011 - Spring 2014, 3.5 years, Teaching Assistant, University of Kentucky.
- Fall 2009 - Spring 2010, 1 year, Teaching Assistant, Arab Academy for Science, Technology, and Maritime Transport - Egypt.
- Fall 2008 - Spring 2009, 1 year, Teaching Assistant, Modern University for Technology and Information - Egypt.
- Fall 2004 - Spring 2007, 3 years, Teaching Assistant, October 6 University - Egypt.

Publications:

- A. Ibrahim, B. Mahmood, and M. Singhal, "A Secure Framework For Medical Information Exchange (MI-X) Between Healthcare Providers," in *Proceedings of the 2016 IEEE 4^{th} International Conference on Healthcare Informatics (ICHI)*, 10-pages, October 2016. (Accepted)
- B. Mahmood, A. Ibrahim, and D. Manivannan, "Hybrid On-demand Greedy Routing Protocol with Backtracking for Mobile Ad-Hoc Network," in *Proceedings of the 2016 IEEE 9^{th} IFIP Wireless and Mobile Networking Conference (WMNC)*, 7-pages, July 2016.

- A. Ibrahim and M. Singhal, "A Simultaneous Key Generation Technique For Health Information Exchange (HIE) Based On Existing Patients' Credentials," in *Proceedings of the 2016 IEEE 1th Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, 3-pages, June 2016.

- A. Ibrahim, B. Mahmood, and M. Singhal, "A Secure Framework for Sharing Electronic Health Records over Clouds," in *Proceedings of the 2016 IEEE 4th International Conference on Serious Games and Applications for Health (SeGAH)*, 8-pages, May 2016.

- A. Ibrahim and M. Singhal, "A New Authentication Protocol for an Authentication-as-a-Service (AaaS) Cloud using Pedersen Commitment Scheme," in *Proceedings of the 2016 IEEE International Conference on Industrial Informatics and Computer Systems (CIICS)*, 6-pages, March 2016.

- A. Ibrahim and M. Singhal, "An Abstract Architecture Design For Medical Information Exchange," in *Proceedings of the 2016 IEEE International Conference on Industrial Informatics and Computer Systems (CIICS)*, 6-pages, March 2016.

- M. Aborizka and A. Shedeed, "Secure E-voting Scheme Through Polling Stations," in *Proceedings of the 11th International Business Information Management Association (IBIMA) Conference*, ser. Innovation and Knowledge Management in Twin Track Economies, K. S. Soliman, Ed., pp. 250–256, January 2009, ISBN: 987-0-9821489-0-7.

- M. Aborizka, A. Shedeed, and S. Saad, "E-voting Scheme Over the Internet," in *Proceedings of the 6th International Business Information Management Association (IBIMA) Conference*, ser. Managing Information in the Digital Economy: Issues and Solutions, K. S. Soliman, Ed., pp. 503–507, June 2006, ISBN: 0-9753393-5-4.