PRIVACY-PRESERVING AUDIT FOR BROKER-BASED HEALTH INFORMATION EXCHANGE

BY

SE EUN OH

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Adviser:

Professor Carl A. Gunter

ABSTRACT

Health Information Technology has spurred the development of distributed systems known as Health Information Exchanges (HIEs) to enable the sharing of patient records between different health care organizations. Participants using these exchanges wish to disclose the minimum possible amount of information that is needed due to patient privacy concerns over sensitive medical information. Therefore, broker-based HIEs aim to keep limited information in exchange repositories and to ensure faster and more efficient patient care. It is essential to audit these exchanges carefully to minimize the risk of illegitimate data sharing. This thesis presents a design for auditing broker-based HIEs in a way that controls the information available in audit logs and regulates its release during audit investigations based on the requirements of applicable privacy policy. In our design, we utilized formal rules to verify access to HIE and adopted Hierarchical Identity-Based Encryption (HIBE) to support the staged release of data required for audits and a balance between automated and manual reviews. We test our methodology with a consolidated and centralized audit source that incorporates a standard for auditing HIEs called the Audit Trail and Node Authentication Profile (ATNA) protocol with supplementary audit documentation from HIE participants.

To my lord, Jesus Christ for his love and support

ACKNOWLEDGMENTS

I would like to thank my advisor Professor Carl A. Gunter, who opened my research journey in Health Information Technology security, motivated me to grapple with cutting-edge research problems, and helped me to hold onto a dream. With his advice and support, I became knowledgeable in this security area and enthusiastic in my work and study. I especially appreciate my co-workers, Ji Young Chun, Limin Jia, Deepak Garg and Professor Anupam Datta, all of whom gave me significant advice and assistance in getting this work published. My thanks also goes out to Ivan Handler and Mike Berry, who showed great interest in my work and shared crucial comments about auditing HIEs. Last but not the least, I am grateful to my husband, Ki Bum Noh, who is always at my side and gave me his full support as well as advice about how balance study and taking care of our little girl; and finally a special thanks to my parents for their devotion to me my entire life. This thesis is partially supported by NSF CNS 09-64392 (NSF EBAM) and HHS 90TR0003-01 (SHARPS).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADP	Audit Data Processor
ARIP	Access Rules Informed by Probabilities
ATNA	Audit Trail and Node Authentication
BL	Medical Billing
EHRs	Electronic Health Records
EMR	Electronic Medical Record
HCO	Health Care Organization
HHS	United States Department of Health and Human Services
HIE	Health Information Exchange
HIBE	Hierarchical Identity Based Encryption
IBE	Identity Based Encryption
ID-CRM	Identifier Cross-Reference Manager
IHE	Integrating the Healthcare Environment
ISO	International Organization for Standardization
IIHI	Individually Identifiable Health Information
KGC	Key Generation Center
LTL	Linear Temporal Logic
NIST	National Institute of Standards and Technology
ONC	U.S. Office of the National Coordinator
PACS	Picture Archiving and Communications System

PHI	Protected Health Information
PMP	Prescription Monitoring Program
PR	Patient Registration
XDS	Cross Document Sharing

CHAPTER 1

INTRODUCTION

Broker-based HIEs have been utilized to enable institutions to ensure the secure transmission of patient medical documents using well-known infrastructures such as XDS [1]. Brokers retain the index for each document as well as the repository address where each document is stored instead of using a centralized repository for documents. Broker-based exchanges provide some level of indexing, provide for the transport of data, and are equipped with security measures such as authentication and audit, often without storing key data themselves except during transmission. These systems have come into wide use in the healthcare field and the goal of HIEs is to be adopted for the international sharing of EHRs to disclose medical records in a standard format.

An additional benefit of the broker system is that the amount of medical data kept on the HIE side for sharing medical records between different organizations can be minimized to mitigate HIE consumers privacy concerns. For example, in the United States, there are many state-sponsored HIE systems where the state is reluctant to hold EMR data themselves because of citizen concerns about sharing health data with government. Broker systems can improve patient care as well as prevent unnecessary disclosure of information during the exchange.

However, it is a challenge for broker-based HIEs to protect the privacy of information in their audit logs, including any supplementary documentation, during the audit process that is used for ongoing monitoring and specific investigations of potentially illegitimate access to medical records. The more audit data the HIE holds, the better it can ensure the legitimacy of access to medical records through its audits, but the more risk there is that the audit process itself will compromise privacy. Furthermore, doing manual audits is typically not feasible since thousands of accesses to the HIE occur every day. Hence, the auditing process needs to be automatic, which will require the

HIE to keep as much standardized audit information as possible so that it is there when needed.

The goal of this thesis is to suggest the design for a privacy-preserving audit infrastructure that includes both automated auditing using a logic-based audit algorithm and staged disclosure through HIBE of information necessary to conclude an effective audit. In particular, we developed an audit subsystem in which HIBE [2] is used in coordination with a logic-based audit algorithm [3] to limit the information disclosed to the auditor to only that which is needed for the specific audit in question. Our system uses HIBE to encrypt sensitive data on the audit logs assigning appropriate levels of sensitivity. The auditor uses the audit algorithm to decide which parts of the log need to be decrypted in order to determine whether a legitimate treatment relationship exists between the patient and the accessing HIE provider. HIBE provides a convenient way to encrypt audit logs at a fine granularity (i.e., each event is encrypted using keys derived from the identifiers participating in that event) and limits auditors to decrypting the minimum data required. HIE transmission of information occurs frequently, yet audits are relatively rare, so our framework provides an effective and efficient audit procedure for such limited and infrequent decryption of data. We also propose the idea of extending the audit algorithm to provide understandable explanations about any particular access rather than only indicating whether an access is consistent with or in violation of applicable policy. We design an audit architecture that augments HIEs using the ATNA [1] profile, the current standard for HIE audits; supports HIBE encryption of audit logs; and an provides an explanation-enabled audit procedure. A key feature of the design inspired by the state HIEs in Maryland and Illinois is the ability to combine the ATNA data with external documentation that is fed into the audit algorithm. Our main contribution lies in the design and implementation of a practical system using both a novel encryption technique and audit algorithm.

The rest of this thesis is organized as follows. Chapter 2 provides background on HIEs and HIBE, and Chapter 3 discusses related work. Chapter 4 is an overview of the architecture of our design and an introduction to our sample audit scenario. Next, Chapter 5 details the hierarchical encryption algorithm. Chapter 6 introduces the audit algorithm used in our architecture, and provides an example of how our our infrastructure is used to audit access

and to produce explanations of audit results. Chapter 7 presents the implementation of our prototype and the results of our evaluation of it. Chapter 8 points out the limitations of this work. Finally, Chapter 9 incorporates a user access control mechanism into the audit architecture and discusses how this work could be extended in the future to other access control uses. Chapter 10 shows conclusion.

This thesis draws heavily on joint work described in the CODASPY paper by Oh et al. [4]. The author of this dissertation primarily worked on the topics in Chapters 4 and 6 which are discussed in [4] and on the material in Chapter 7, which is not included in [4].

CHAPTER 2

BACKGROUND

2.1 Audit Standard for HIE

A Health Information Exchange (HIE) is a system for parties to electronically transmit health-related information between various organizations. HIEs promise benefits such as reducing duplication of services and supporting the connection across statewide, regional and local services; however, many health-care providers are reluctant to use HIE due to patient concerns about privacy, since an HIE enables individual health information to be collected in a centralized repository [5]. To appease privacy concerns, the National Institute of Standards and Technology (NIST) recommends twelve services [6] to protect data from being compromised and create a secure HIE environment. One of these twelve services involves collecting and communicating audit logs that define, identify, and communicate security-related events and data that should be gathered, consistent with applicable policy. According to the Maryland Health Care Commission [7], HIE has to implement periodic audit procedures to determine whether improper access, use, or disclosure has been made, informing HIE consumers or other authorized users when an audit indicates a policy violation. These audits review, at a minimum, the following information: Identity of Protected Health Information (PHI) originator, description of the consumer's actions, identity of the users accessing the patient record, the organization the users are involved in, access date and time, source and type of PHI, date and time the PHI became available to the HIE, the user's date of registration with HIE and the user's access level during access.

The Audit Trail and Node Authentication (ATNA) [1] Profile, an IHE Integration standard, is an audit mechanism reflecting such security guidelines, with three strengths. First, it generates and keeps audit records in a

centralized audit repository controlled by a security officer; it can be readily adapted to the typical HIE environment involving Cross Document Sharing (XDS) [1] and it would be compliant with HIPAA [8]. IHE stipulates events that must be recorded in the audit trail, which are defined in XDS.b [9]. Second, ATNA is compatible with diverse types of healthcare enterprises since it generates audit logs based on existing standards, such as HL7 [10]. To make it compatible, the ATNA profile suggests using Security Audit and Access Accountability Message XML Data Definitions for Healthcare Applications (RFC-3881) [11], which define an XML schema to report security-related events described with other standards such as ASTM [12], HL7, and DICOM [13]. Third, ATNA offers certificate-based bi-directional authentication between users and nodes before transactions occur to provide data integrity and secure transmission. The ATNA logs contain information relevant to a security audit (which patient's PHI was accessed, which user accessed it, and what user or node authentication failures were reported), to identify specific evidence of policy violations with any particular access after the fact.

In short, ATNA provides infrastructure for auditing the HIE. It is still necessary to develop ways to stipulate policies and carry out the analysis of audit events on ATNA standard audit logs. Moreover, it may be necessary to work with external sources of information that are not present in ATNA logs to achieve the overall goals of auditing the HIE. For instance, if one wishes to use billing records to confirm that an access to a record has been made to provide a (billed) service, then it may be desirable to integrate billing records into the underlying analytic engine rather than trying to make billing records a part of ATNA.

2.2 Hierarchical Identity Based Encryption

HIBE [14, 2, 15, 16, 17] is a form of IBE for hierarchical structures. IBE allows a sender to encrypt messages based on a receiver's identity, such as an e-mail address. The sender can encrypt a message for the receiver using IBE before the receiver gets a secret key from a KGC. The KGC generates a secret key to a user commensurate with the level of sensitivity of the data that the user needs to access. The private key will allow access at that level, or depth, in the hierarchy and at all lower levels and will also delegate authority to the

holder of the private key to generate secret keys to users at lower levels. In other words, this is one-way access and delegation, so that a user at level k can generate a secret key for a user at any level lower than k, but lower level users can not use their secret keys to make secret keys for users at higher levels in the hierarchy. Upper level users who generate lower level keys are referred to as parents in the hierarchical structure. The HIBE system uses the following five algorithms.

$\mathsf{HIBE.Setup} : (k, \mathsf{L}) \mapsto \{mk, \mathsf{Pub}\}$ takes security parameter k and maximum depth L, and outputs a master key mk and public parameters Pub.

$\mathsf{HIBE.Extract} : (\mathsf{Pub}, mk, \mathsf{ID}) \mapsto sk_{\mathsf{ID}}$ takes the public parameters Pub, the master key mk and an identity $\mathsf{ID} = (id_1, ..., id_\ell)$ of depth $\ell (\leq \mathsf{L})$, and outputs a secret key sk_{ID} for the identity ID.

$\mathsf{HIBE.Delegate} : (\mathsf{Pub}, sk_{\mathsf{ID}'}, \mathsf{ID}) \mapsto sk_{\mathsf{ID}}$ takes the public parameters Pub, a parent's secret key $sk_{\mathsf{ID}'}$ where the parent's identity $\mathsf{ID}' = (id_1, ..., id_{\ell'})$ of depth $\ell' (< \ell \leq \mathsf{L})$, and the child's identity $\mathsf{ID} = (id_1, ..., id_\ell)$ of depth ℓ. Then the algorithm outputs a secret key sk_{ID} for the identity ID.

$\mathsf{HIBE.Encrypt}(\mathsf{Pub}, \mathsf{ID}, \mathsf{Msg}) \mapsto \mathsf{CT}$ takes the public parameters Pub, an identity ID and a message Msg, and outputs a ciphertext CT.

$\mathsf{HIBE.Decrypt} : (\mathsf{Pub}, sk_{\mathsf{ID}}, \mathsf{CT}) \mapsto \mathsf{Msg}$ takes the public parameters Pub, a secret key sk_{ID} and a ciphertext CT, and outputs a message Msg.

We will later describe how to use HIBE for a workable system to encrypt external data for later use by HIE auditors, as well as describing the security procedures that will effectively limit access to that external data in the course of a formal complaint regarding legitimacy of access. There are various HIBE systems that satisfy different properties and security goals. It is important to select an HIBE system for our system with constant and small ciphertext such as [2], since such a system will provide storage and decryption efficiency regardless of hierarchy depth. Encrypted sensitive data should rarely be decrypted for maximum patient privacy protection; however, encrypted sensitive data needs to be stored by HIE in the event of a health privacy infringement investigation. Therefore, our HIBE systems will use minimal ciphertext size to achieve maximum privacy protection with a minimum of storage space.

CHAPTER 3

RELATED WORKS

3.1 Audit Healthcare Domain.

Due to the wealth of private patient information shared in distributed HCOs, a secure audit is indispensible to identifying specific evidence when suspicious access is detected. Previous works have envisaged a scientific and technical approach to audit the healthcare system. Gunter et al. [18] introduce ARIP to establish appropriate access rules for HCOs based on their work flows and social networks by analyzing audit logs and attributes of HCOs. Their methodology is similar to ours in analyzing audit logs to justify access to patient health information. However, their work is not feasible for infrastructure currently in place at HCOs, while our work is applicable to the real-world heterogeneous healthcare environment with IHE standard based audit infrastructure. In addition, Fabbri and LeFevre [19, 20] have proposed explanation-based auditing, which enables patients to review access to their health records with human interpretable explanations. They adopt machine learning approach to automatically generate log explanation while we use the logic-based algorithm to identify the legitimacy of access based on the privacy policy. In addition, they apply their algorithm to existing hospital logs only and do not actually produce interface with any existing front-end healthcare system. Moreover, while their explanations describe the reason for a particular user access, they cannot use the explanation to automatically show legitimacy relative to a given policy. In contrast, our work directly checks the legitimacy of access based on applicable privacy law and policy. Some prior work takes a technical approach, adopting ATNA to ensure HIPAA compliance and attempting to integrate diverse HCOs. Gregg [21] builds an audit interface utilizing ATNA for the audit logs from a PACS. Azkia et al's similar work [22] converts ATNA audit records generated by their system to

be incorporated into the Organization Based Access Control (OrBAC) policy. However, their works do not show how their policy model is specifically applicable to existing policies and, nor does it consider privacy concerns during audit while our work encodes ONC guidance into first-order logic and includes encryption for privacy-aware audit.

3.2 Audit Log Encryption.

The HIE audit log presents unique challenges because it can and should be used as a tool for detecting inappropriate access to private IIHI, and yet access to such information in the healthcare environment often involves special situations that make it hard to limit the ability of providers to access IIHI. These special situations include emergencies. We share the goal of previous encryptions of audit logs [23, 24, 25] to not only protect against malicious attackers, but also to limit exposure of private information in the log to an authorized auditor. Our approach is different from previous methods for encrypting audit logs because we supplement internal log information with external data, which we encrypt with HIBE, allowing auditors to access only the minimum necessary data. In doing so, our scheme creates an identity, a descriptive label, using terms that do not need to be encrypted, such as unidentifiable codes for patient, provider, and type and date of visit, for the audit log entry, allowing the auditor to find the log needed without the more cumbersome encrypted keyword approach previously used. In previous schemes, such as Waters et al. [25], the auditor has to match all encrypted keywords with a given trapdoor that contains those words. This makes it secure but extremely inefficient.

3.3 Misuse Detection of Bitcoin

Bitcoin is an internet currency created in 2009 that is fully decentralized; it has no association with banks or governments [26]. Bitcoin operates in blocks of transactions that form chains that show that a sequence of transactions were verified by a majority of the Bitcoin network's computing power. This majority power is presumed to be made up of honest users and so the system

now trusts people to do the right thing. However, Bitcoin is used only minimally for actual trading transactions, and misuse, such as double-spending [27, 28], has been reported, the measure for which is generally simply advertising the misuse if it is detected. If an audit mechanism were developed that could properly audit breaches of security in a privacy-preserving manner, bitcoins could become more widely accepted. Our infrastructure and HIBE key system could be useful for that purpose.

3.4 Algorithms for Policy Compliance Checking.

We build on Garg et al.s algorithm reduce for auditing policies over incomplete logs [3]. We inherit the policy language used by the reduce algorithm, which is a first-order logic that can encode first-order LTL formulas. Policies that are naturally specified in LTL can be easily translated to formulas in this logic. There has been much work on compliance checking of policies expressed in LTL [29, 30, 31, 32, 33, 34]. Most of the work focuses on runtime monitoring. In contrast, we assume that logs are recorded by audit agents, and post-hoc audit is applied to these logs. To our knowledge, reduce is the only policy-based log audit algorithm that can handle incomplete logs. We leverage reduces capability to handle incomplete logs to integrate encrypted logs. One of our contributions is to extend reduce to further generate explanations for the output of the algorithm.

CHAPTER 4

OVERVIEW

4.1 Audit Architecture

Figure 1 illustrates HIE with its ATNA repository, HIE consumers such as hospitals and the relationship of our proposed audit infrastructure to these. The audit infrastructure will be used to check the legitimacy of each access to HIE and to generate a corresponding log explanation. The audit system consists of an audit data processor (ADP), audit agent, audit algorithm and auditor viewer. The ADP collects HIE ATNA logs and external documentation which is subsequently encrypted converts them into specific DB schema of the Audit Logs. Converted audit records are stored in the Audit Logs through the audit agent. The audit algorithm informs specific encrypted information additionally needed to verify legitimacy of access after first iteration and the audit agent identifies and decrypts them. The audit algorithm ultimately verifies whether the access is legitimate with decrypted audit records. In addition, the ADP has the Key Generate Center (KGC) issuing a secret key for the decryption at the particular level of sensitivity of additionally required information that the algorithm has identified to confirm legitimacy of access. After the auditor viewer formats the result of the audit algorithm in a more user friendly format, human interpretable log explanation is presented in the auditor viewer upon request or periodic auditing.

4.1.1 Health Information Exchange (HIE)

We have simulated the HIE system based on IHE Integration profile transactions [9]. When a doctor submits a medical record to HIE for a new patient, a new patient index is created by the HIE's patient identity actor, the doc-

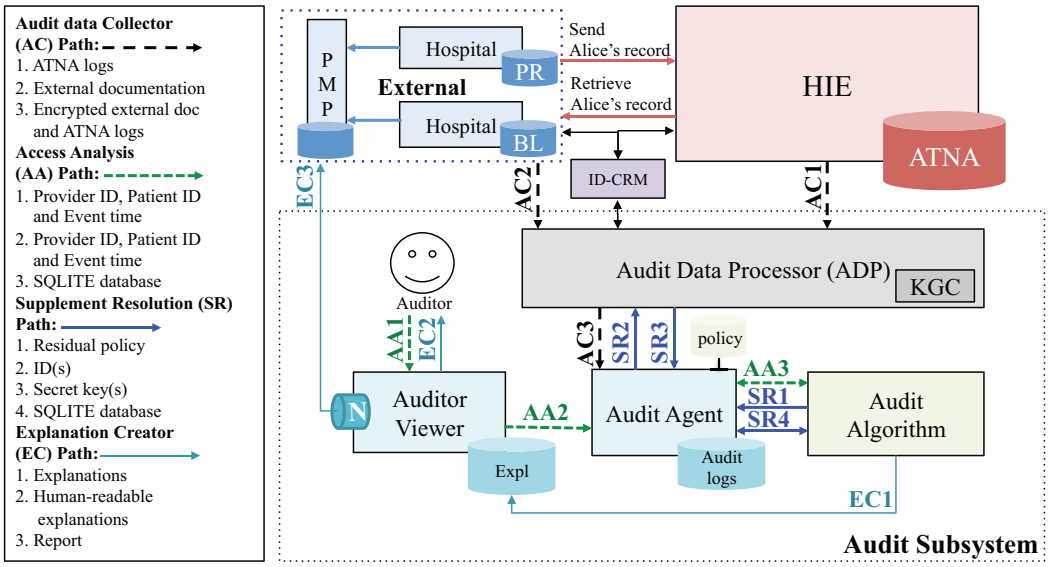

Figure 4.1: HIE Audit Infrastructure

ument goes through the HIE's document repository, and meta data for the document location is kept in the HIE's document registry. HIE provides the document viewer which is an interface for use by a requesting client to request medical documents through the HIE. When such a request is made, it goes through the document registry, which obtains the patient ID stored in the HIE's master patient index (MPI). A document repository ID and document ID are then sent back to the document viewer for the request to be transmitted to the holder of the document. When the HIE's document source receives a patient medical document from the HIE consumer such as a hospital which is an originator of a Document, it is transmitted through the document repository, which acts as a broker to send the document to the requesting client. The HIE also has the electronic referral actor which handles the referral process electronically. Apart from this, there is a master provider directory with a continually updated list of healthcare providers, consistent with the IHE Integration Profile Transaction add/update.

4.1.2 External Parties

We call the owners of supplementary external documentation the external parties assuming that the hospitals and the PMP will routinely provide doc-

umentation related to each HIE access, such as medical bills, patient registration, and prescriptions to the ADP, for use in subsequent audits. The link between the HIE and the ADP is the ID-CRM, which will link each external party's internal patient identifier and its user ID and hospital ID with the HIE's patient ID, user ID and hospital ID, the latter being described in one of IHE standards, Patient Identifier Cross-referencing (PIX). Sources of the external documentation include the PMP, the patient registration (PR), and the medical billing (BL). The PMP [35] is a centralized repository managed by the state of Illinois and prescribers and dispensers are required to report prescription details for controlled substances. The PMP reports the prescription details such as recipient's name, address and drug code of controlled substances to prove the treatment relationship with a patient whose health information is accessed via the HIE. The details about the external documentation simulation are presented in Chapter 7.

4.1.3 Audit data Collector (AC) Path

Our simulated HIE generates XML-based ATNA logs generated by IHE Integration Profile transactions as follows: Patient Identity Feed (ITI-8), assigning new patient ID to a new patient, which is subsequently stored in MPI; Provide and Register Document Set-b (ITI-41), submitting a medical document for a new patient to the document repository; Retrieve Document Set (ITI-43), retrieving a patient's medical document upon request; and Electronic Referrals. The ATNA logs are then stored in the ATNA repository. The sensitive entries in the ATNA logs and external documentation will be encrypted using Hierarchical Identity based Encryption in the ADP. Specific encryption procedures are introduced in Chapter 5. Once encrypted, all ciphertext and plaintext entries coming through the ADP will be stored in the Audit Logs as audit records and will wait for a request to analyze them.

4.1.4 Access Analysis (AA) Path

When an auditor tries to audit HIE logs, he/she has to connect to a web auditor portal, the auditor viewer. The auditor is not allowed to access to other parties in audit infrastructure, which prevents being directly disclosed

to audit records. Once an auditor makes a request to analyze a specific access shown in HIE log, an auditor's query including patient ID, provider ID and event time is sent to the audit agent. Based on this auditor-defined condition, the audit agent looks for matched information among plain text information. Before sending it to the audit algorithm called reduce, the audit agent converts it into the predicate table schema which the audit algorithm can read. In addition, HIE policy is encoded in a language that reduce understands and then encoded policy is also transmitted to reduce. reduce then starts the first iteration.

4.1.5 Supplement Resolution (SR) Path

After investigating a log entry, reduce returns results indicating what specific supplementary information is necessary to determine legitimacy of access, which is called residual. Based on all pieces of residual, the audit agent can infer patient ID, hospital ID, event time and what kinds of information it needs, such as patient visit history. That is, the audit agent can identify necessary IDs and specifically required encrypted column names based on the instances and name of returned the predicates in residual. When the audit agent sends a data ID or multiple IDs, the KGC creates a secret key or multiple secret keys and sends to the audit agent. The audit agent decrypts the requisite information which is identified based on residual and captures it in the predicate table schema. Finally, it sends additional the predicate tables to the audit algorithm for the second iteration. The decryption process is circumstantiated in Chapter 5.

4.1.6 Explanation Creator (EC) Path

After reduce checks whether the audit records conform to policy, it returns true and explanation containing satisfying sub-clause and substitutions if the policy is satisfied. If not, reduce returns false and explanation including unsatisfying sub-clause and substitutions. The details about explanation generated by the algorithm are described in Chapter 6. Based on return values, the auditor viewer translates them into human-understandable The explanations and provides them to the auditor. Serious policy violations

associated with access are sent to the HIE consumers through the notification proxy shown in Figure 1.

4.2 Audit Scenario

We illustrate an audit scenario using the infrastructure in Figure 4.1. Assume that an auditor verifies *Dr. John Kosta*'s access to *Alice*'s medical record disclosing her mental health based on the ONC privacy policy [36]. To verify access, patient medical billing documentation issued after the observation in the emergency room must be seen. The policy is encoded below:
$\varphi_{pol} =$

$$\begin{aligned}
\forall p_1, p_2, m, q, t, ty, va, tp, vl, o, p, c.(&\mathsf{send}(p_1, p_2, m, t) \wedge \\
&\mathsf{hasattrof}(m, q) \wedge \\
&\mathsf{includes}(m, ty, va)) \wedge \\
&\mathsf{patientInfo}(q, tp, vl, t) \wedge \\
&\mathsf{organization}(p_2, o, t) \wedge \\
&\mathsf{insuranceInfo}(q, p, c, t)) \\
\supset (&\exists t'.\mathsf{medical\text{-}bill}(q, visit - history, t') \wedge \\
&\mathsf{timein}(t, t', t{+}365) \wedge \\
&\mathsf{visits\text{-}in\text{-}bill}(q, p_2, vl, o, t')) \\
\vee (&\exists t'.\mathsf{medical\text{-}bill}(q, observation, t') \\
\wedge &\mathsf{timein}(t, t', t{+}365) \\
&\mathsf{observes\text{-}in\text{-}bill}(q, p_2, ty, va, o, t')) \\
\wedge &\mathsf{insurance}(q, p, c, t'))
\end{aligned}$$

In words, if the log shows that entity p_1 sends to entity p_2 a message m at time t, m describes patient q at time t, m includes ty such as observation ID and va such as order type and observation value , then patient q is provided with service vl in the hospital, such as inpatient, p_2 works in organization o, patient q has a insurance plan p of company c , consequent patient's medical bill issued within one year involves patient's insurance company c, plan p and other entries have to be matched with one of the following cases: First,

order information in medical bill includes that provider p_2 from the hospital o makes an order va for patient q at time t'. Second, visit history information in medical bill contains that patient q visits provider p_2 in the hospital o for vl service at time t'. Third, observation information in medical bill shows that provider p_2 observes patient q in the hospital o at time t' and the result indicates that observation ID is ty and observation value is va. Dr. Kosta's access generated an ATNA log whose name is the Retrieve Document Set defined by IHE standard. Every time ADP gathers external documentation from the PMP and HIE facilitated hospitals regularly, the ADP encrypts it before sending to Audit Agent's Audit Logs repository.

Once the audit records are imported in the Audit Logs, it waits for the auditor viewer's request. A few month later, the auditor makes a request to analyze Dr. Kosta's access. The auditor viewer sends Dr. Kosta 's HIE user ID (kos12), Alice' patient ID (eeb728473e1949a) and the specific access event time (2013:09:08:10:18:41) for subsequent audit agent's queries matched log entries from the Audit Logs repository. Once the audit agent locates them, it converts them into the predicate table schema , for example, send(p_1,p_2,m,t), required by reduce and encodes the policy in a first-order logic. Then, the audit agent sends them to reduce for the first iteration. After this first step, reduce returns residual inferring required specific encrypted columns and a searching key to find matched precise rows in the table of the Audit Logs. The audit agent requests secret keys to the KGC based on combination of a primary key with specific column name which will be decrypted. Especially, in this scenario, since residual informed that necessary supplementary information is Predicate1 requiring decryption up to level1 or Predicate2 requiring decryption up to level2, the KGC has three choices to issue secret keys, $d_{\mathsf{ID}_{level2}}$, $d_{\mathsf{ID}_{level1}}$ and both. Eventually, the KGC sends $d_{\mathsf{ID}_{level2}}$ to decrypt lower level. This is one of key advantages using HIBE in this paper. Having retrieved secret key, the audit agent decrypts required specific columns which are (provider-level1, visitsInBill-level2) in the Audit Logs and puts them in visits-in-bill table again. The log entries conform to the policy and reduce returns true and the explanation. Based on reduce's returned values, the auditor viewer creates the following human-understandable explanation:

Dr. Kosta's access to Alice's record is justified because Alice's medical bill shows that she visits Dr. Kosta in emergency at time t.

If the auditor viewer presents specific explanations showing serious policy violations, the viewer consequently informs the HIE consumers of the violation, which will be processed through the notification proxy. This reporting process is enforced by our audit system.

CHAPTER 5

HIERARCHICAL ENCRYPTION

In this chapter, we describe how we encrypt and decrypt external audit logs, specifically documentation, from the HCOs using HIBE for privacy protection during audits. There are various HIBE schemes that satisfy different properties and security goals. It is important to select a HIBE scheme with constant, small-size ciphertexts, e.g. [2], since such a scheme will optimize storage cost regardless of hierarchy depth. Even though privacy infringement investigations are rare in practice and encrypted sensitive data will seldom be decrypted, it must nonetheless be stored by the audit subsystem.

5.1 External Data Hierarchy and Identity

Suppose that the data in some external documentation $\mathcal{D}$ has been partitioned hierarchically into n degrees of sensitivity, resulting in data $\mathcal{D}_1, \ldots, \mathcal{D}_n$, where $\mathcal{D}_n$ is the data in the documentation that is most sensitive, and $\mathcal{D}_1$ is the data that is the least sensitive. In our system, all data in $\mathcal{D}_n$ will be encrypted using an IBE identifier $\mathsf{ID}_n = (id_1)$. We suggest this identifier be derived from non-sensitive descriptive information such as the HCO's identity, the patient's identity, degree of sensitivity, and time. Data in $\mathcal{D}_{n-1}$ will be assigned an identifier $\mathsf{ID}_{n-1} = (id_1, id_2)$ and so on. Finally, data in $\mathcal{D}_1$ will be assigned an identifier $\mathsf{ID}_1 = (id_1, ..., id_n)$. As an example, the billing table shown in Table 5.1 is hierarchically organized into 3 sensitivity

patient	HCO	date	level1	level2	level3
eeb7...	Carl...	2013...	$Enc_{\mathsf{ID}_{1,1}}(\mathsf{cs..})$	$Enc_{\mathsf{ID}_{1,2}}(\mathsf{HE...})$	$Enc_{\mathsf{ID}_{1,3}}(73.8)$
d994...	Pro...	2013...	$Enc_{\mathsf{ID}_{2,1}}(\mathsf{ra..})$	$Enc_{\mathsf{ID}_{2,2}}(\mathsf{MC...})$	$Enc_{\mathsf{ID}_{2,3}}(279)$
4221...	NW...	2013...	$Enc_{\mathsf{ID}_{3,1}}(\mathsf{pq..})$	$Enc_{\mathsf{ID}_{3,2}}(\mathsf{PL...})$	$Enc_{\mathsf{ID}_{3,3}}(11.6)$

Table 5.1: observes-in-bill table

levels. Information concerning observation result that a patient receives is level 3 sensitive (most sensitive), observation type is level 2 sensitive, and provider information is level 1 sensitive (least sensitive).

5.2 Billing Data Encryption

Table 5.1 shows encrypted patient observation information taken from illustrative medical bills. Consider data $\mathcal{D}$ that contains exactly the first row of the table. Within $\mathcal{D}$, data at the highest level of sensitivity, $\mathcal{D}_3(= \mathsf{73.8})$, will be assigned identity $\mathsf{ID}_3 = (id_1)$ which is the concatenation $\mathsf{eeb728473e1949a}$ || $\mathsf{Carle07RQ12}$ || $\mathsf{2013:09:08:10:18:41}$ || $\mathsf{level3}$ of non-sensitive identifying information. The identity of less sensitive data, $\mathcal{D}_2(= \mathsf{HE...})$, is $\mathsf{ID}_2 = (id_1, id_2)$, where $id_2 = \mathsf{level2}$. Finally, the identity of the least sensitive data $\mathcal{D}_1(= \mathsf{cs...})$ is $\mathsf{ID}_1 = (id_1, id_2, id_3)$, where $id_3 = \mathsf{level1}$. External documentation at each level of sensitivity $(\mathcal{D}_1, ..., \mathcal{D}_n)$ is encrypted using the assigned identities ID_i $(1 \leq i \leq n)$. For $i = 1$ to n, the ADP or the HCOs run $\mathsf{HIBE.Encrypt}(\mathsf{Pub}, \mathsf{ID}_i, \mathcal{D}_i)$ to get a corresponding ciphertext $Enc_{\mathsf{ID}_i}(\mathcal{D}_i)$. For example, $Enc_{\mathsf{ID}_{1,3}}(\mathsf{73.8})$ in Table 5.1 represents the encryption of the data $\mathcal{D}_3$. The corresponding HIBE encryption would be $\mathsf{HIBE.Encrypt}(\mathsf{Pub}, \mathsf{ID}_{1,3}, \mathsf{73.8})$ where $\mathsf{ID}_{1,3} = \mathsf{ID}_3$.

This example is illustrative: All external documentation in our system is organized in tables and encrypted with the HIBE. Identities used for encryption may have cell-, column- or table-granularity, depending on the nature of the documentation.

5.3 Issuance of Secret Keys and Decryption

When the audit algorithm requests encrypted data, the audit agent requests an appropriate secret key from the KGC for decryption. Assume that the audit agent needs the secret key corresponding to ID_k. To issue an appropriate secret key to the audit agent, the KGC runs $\mathsf{HIBE.Extract}(\mathsf{Pub}, mk, \mathsf{ID}_k)$ using its master key mk to get sk_{ID_k}. After receiving sk_{ID_k} from the KGC, the audit agent runs $\mathsf{HIBE.Decrypt}(\mathsf{Pub}, sk_{\mathsf{ID}_k}, Enc_{\mathsf{ID}_k}(\mathcal{D}_k))$ to obtain $\mathcal{D}_k$. If the audit agent needs additional information at any level k' that is lower

than k, the audit agent runs $\mathsf{HIBE.Delegate}(\mathsf{Pub}, sk_{\mathsf{ID}_k}, \mathsf{ID}_{k'})$ to get a secret key $sk_{\mathsf{ID}_{k'}}$, and then runs $\mathsf{HIBE.Decrypt}(\mathsf{Pub}, sk_{\mathsf{ID}_{k'}}, Enc_{\mathsf{ID}_{k'}}(\mathcal{D}_{k'}))$ to get $\mathcal{D}_{k'}$. This does not require communication with the KGC.

CHAPTER 6

AUDIT WITH EXPLANATIONS

We first review an audit algorithm, reduce [3], that our infrastructure builds on. Then we present an algorithm simplify that augments the output of reduce with an intuitive explanation of why a certain access is justified or unjustified.

6.1 An Audit Algorithm for Incomplete Logs

Garg et al. [3] develop an algorithm for finding violations of a policy on system logs. Their algorithm takes into consideration incompleteness of information in logs. Incompleteness has many practical causes. For example, if the policy carries the obligation "a notice must be sent in the next 30 days", then before the 30 day deadline is reached, the log may not contain enough information to decide whether or not this obligation is met. Similarly, non-mechanizable facts such as "some responsible party has a reasonable belief that...", on which policies often rely, cannot be represented in automatically generated system logs.

To account for incompleteness, the reduce algorithm uses a best-effort approach; it checks as much of the policy as possible given the available log and returns a residual policy that captures policy conditions which could not be verified. The residual policy can be examined by human auditors or re-checked by invoking reduce when more information is available. Formally, policies are represented in first-order logic and log incompleteness is modeled using three-valued logic, where a log is viewed as an abstract structure that maps a predicate (fact) `P` to `tt`, when `P` is true; `ff`, when `P` is false; or `uu`, when information about `P` is not available. For an atomic predicate P mapped to `tt`, `ff` or `uu` by the log, reduce returns $\top$, $\bot$ and P respectively. The algorithm recursively reduces the sub-formulas of the policy formula

and the residual formula mostly preserves the structure of the input policy formula.

A second problem addressed by Garg et al.'s work is that checking policy compliance with first-order quantification over an infinite domain is, in general, undecidable. To resolve the issue, Garg et al. restrict policies to a fragment of first-order logic through syntactic and other statically verifiable conditions. In this restricted fragment, both the universal and existential quantifiers are guarded by a condition, written c. The quantifiers have forms $\forall x.c(x) \supset \varphi$ and $\exists x.c(x) \wedge \varphi$, and it is guaranteed (statically) that the number of substitutions for x that makes c true is always finite. With these restrictions, both universal and existential quantifiers can be handled easily. Garg et al. argue that even the restricted logic is very expressive; in particular, all of the HIPAA Privacy Rule can be represented in it.

When reduce produces an actual audit decision (policy violation or not) instead of a residual policy, it helps the audit process to have an intuitive explanation of why that decision was made. For instance, it is helpful to know that a physician's access to a medical record in a hospital was allowed because the patient was referred to that hospital, or because that the patient visited that facility. In this chapter, we describe an extension to the reduce algorithm that provides such an explanation. Since we build on reduce directly, we inherit solutions to both problems mentioned above from Garg et al's work.

6.2 Explanation Generation

We begin by introducing the syntax of policies and explanations. Then, we describe an extended reduce algorithm that returns in the output additional information about the internal computation of reduce to help generate an explanation subsequently. Finally, we present the explanation generation algorithm and prove its correctness.

6.2.1 Policy syntax

Following the prior work [3], we use a first-order logic as the policy specification language. We summarize the syntax of formulas and explanations in Figure 6.1. We write α to denote formulas and φ to denote generalized

$$
\begin{array}{lrcl}
\textit{Conj clause} & C & ::= & \bigwedge_i \varphi_i \\
\textit{Disj clause} & D & ::= & \bigvee_i \varphi_i \\
\textit{Formula} & \alpha & ::= & \langle\ell\rangle P \mid \langle\ell\rangle\top \mid \langle\ell\rangle\bot \mid \langle\ell\rangle C \mid \langle\ell\rangle D \\
 & & \mid & \langle\ell\rangle\forall\vec{x}.(c \supset \varphi) \mid \langle\ell\rangle\exists\vec{x}.(c \wedge \varphi) \\
 & & \mid & \sigma \rhd \varphi \\
\textit{Generalized form.} & \varphi & ::= & \alpha \mid \texttt{expl}(\top, \gamma) \mid \texttt{expl}(\bot, \gamma) \\
\textit{Explanation} & \gamma & ::= & \ell \mid \ell \circ \gamma \mid \gamma_1 \oplus \gamma_2 \mid \sigma \rhd \gamma
\end{array}
$$

Figure 6.1: Syntax of formulas and explanations

formulas, which are either formulas or audit decisions ($\top$ = no violation, $\bot$ = violation) coupled with explanations γ. Formulas include atomic predicates, true ($\top$), false ($\bot$), conjunctions ($\wedge$) and disjunctions ($\vee$) of formulas (denoted C and D, respectively), and first-order quantifiers ($\forall$ and $\exists$). Each formula is annotated with a policy label, written ℓ. Labels have no semantic meaning except to establish a syntactic link between an explanation and the original formula from which the explanation was derived. The formula $\sigma \rhd \varphi$ means that the substitution for free variables in φ is σ. This formula itself is not used for policy specification. It can appear in residual formulas output by our extended reduce algorithm. c denotes a restricted class of formulas borrowed from [3]; readers may ignore the distinction between c and α for the purpose of understanding this chapter.

An explanation γ corresponds to a sub-tree of labels of a formula's abstract syntax tree. An explanation is only meaningful relative to a formula. Explanations can be a single label, which points to a leaf position of the formula. A concatenated explanation $\ell \circ \gamma$ is an explanation for a formula labeled by ℓ at the root, where γ is, recursively, the explanation of the root's children. An explanation can also combine explanations from branches of a conjunction or disjunction (denoted $\gamma_1 \oplus \gamma_2$). Finally, an explanation can be guarded by a substitution σ (syntax: $\sigma \rhd \gamma$).

6.2.2 Extended Reduce Algorithm

The reduce algorithm, as presented in [3], takes as argument a policy formula α and an audit log $\mathcal{L}$ and returns a residual policy α', which may be $\top$ (no violation), $\bot$ (violation) or another formula called the residual formula (meaning that critical information is absent from the log; α' must be checked

$$
\begin{array}{lcl}
\mathsf{reduce}(\mathcal{L}, \langle\ell\rangle P, \sigma) & = & \begin{cases} \langle\ell\rangle\top & \text{if } \mathcal{L}(P\sigma) = \mathtt{tt} \\ \langle\ell\rangle\bot & \text{if } \mathcal{L}(P\sigma) = \mathtt{ff} \\ \langle\ell\rangle P & \text{if } \mathcal{L}(P\sigma) = \mathtt{uu} \end{cases} \\
\mathsf{reduce}(\mathcal{L}, \langle\ell\rangle \bigwedge_{i=1}^{n} \varphi_i, \sigma) & = & \langle\ell\rangle \bigwedge_{i=1}^{n} \mathsf{reduce}(\mathcal{L}, \varphi_i, \sigma) \\
\mathsf{reduce}(\mathcal{L}, \langle\ell\rangle\forall\vec{x}.(c \supset \varphi), \sigma) & = & \\
\quad \text{let } \{\sigma_1, \ldots, \sigma_n\} \leftarrow \widehat{\mathtt{sat}}(\mathcal{L}, c \cdot \sigma) & & \\
\quad\quad \{\vec{t_i} \leftarrow \sigma_i(\vec{x})\}_{i=1}^{n} & & \\
\quad\quad S \leftarrow \{\vec{t_1}, \ldots, \vec{t_n}\} & & \\
\quad\quad \{\psi_i \leftarrow \mathsf{reduce}(\mathcal{L}, \varphi, \sigma \cdot \sigma_1)\}_{i=1}^{n} & & \\
\quad\quad \psi' \leftarrow \forall\vec{x}.((c \wedge \vec{x} \notin S) \supset \varphi) & & \\
\quad \text{return } \langle\ell\rangle(\sigma_1 \rhd \psi_1 \wedge \ldots \wedge \sigma_n \rhd \psi_n \wedge \psi') & & \\
\mathsf{reduce}(\mathcal{L}, \sigma' \rhd \varphi, \sigma) & = & \sigma' \rhd \mathsf{reduce}(\mathcal{L}, \varphi, \sigma \cdot \sigma') \\
\mathsf{reduce}(\mathcal{L}, \mathtt{expl}(\top(\bot), \gamma), \sigma) & = & \mathtt{expl}(\top(\bot), \gamma)
\end{array}
$$

Figure 6.2: selected rules for reduce

when more information is available). To extend reduce to generate explanations, we change it to take as input a policy represented as a generalized formula φ, a log $\mathcal{L}$ and, additionally, a substitution σ for free variables of φ. The output of our extended reduce algorithm is also a generalized formula. If the output is α, it means that some information necessary for audit is missing from the log, and α is the residual policy to be checked when that information becomes available. The output $\mathtt{expl}(\top, \gamma)$ means that there is no policy violation and γ explains why that is the case. Similarly, the output $\mathtt{expl}(\bot, \gamma)$ signals a policy violation justified by explanation γ.

Selected rules for the extended reduce algorithm are presented in Figure 6.2. When the formula is atomic $\langle\ell\rangle P$, reduce returns $\langle\ell\rangle\top$, $\langle\ell\rangle\bot$ and $\langle\ell\rangle P\sigma$, when $\mathcal{L}(P\sigma)$ is tt, ff and uu, respectively. For a universally quantified formula $\langle\ell\rangle\forall\vec{x}.(c \supset \varphi)$, reduce first finds the set S of substitutions for $\vec{x}$ that make c true, using a special function called $\widehat{\mathtt{sat}}$ in [3]. The details of $\widehat{\mathtt{sat}}$ are unimportant here; the only important point is that S is always finite. The output of reduce is the conjunction of all the formulas obtained by reducing $\varphi(\vec{x})\sigma\sigma_i$, for each σ_i in S, and ψ', which is the same universally quantified formula as the input formula, except that the guard of the quantifiers includes an additional constraint that the substitutions S need not be considered. The substitution σ_i marks ψ_i to help explanation generation. The last clause ψ' is necessary in the residual formula because we do not assume a priori that the log $\mathcal{L}$ determines all possible substitutions for $\vec{x}$ that will satisfy c in future.

$$\mathsf{sc}(\textstyle\bigwedge_{i=1}^{n}\varphi_i) \;=\; \begin{cases} \mathsf{simplify}(\varphi_i) & n = 1 \\ \mathtt{expl}(\bot,\gamma) & \text{if } \mathsf{simplify}(\varphi_1) = \mathtt{expl}(\bot,\gamma) \\ & \text{ or } \mathsf{sc}(\bigwedge_{i=2}^{n}\varphi_i) = \mathtt{expl}(\bot,\gamma) \\ \mathtt{expl}(\top,\gamma_1 \oplus \gamma_2) & \text{if } \mathsf{simplify}(\varphi_1) = \mathtt{expl}(\top,\gamma_1) \\ & \text{ and } \mathsf{sc}(\bigwedge_{i=2}^{n}\varphi_i) = \mathtt{expl}(\top,\gamma_2) \\ \varphi_1' \wedge \varphi_2' & \text{if } \mathsf{simplify}(\varphi_1) = \varphi_1' \\ & \text{ and } \mathsf{sc}(\bigwedge_{i=2}^{n}\varphi_i) = \varphi_2' \text{ and } \varphi_1 \text{ or } \varphi_2 = \alpha \end{cases}$$

Figure 6.3: Simplify for conjunctive formulas

$$\begin{aligned} &\mathsf{simplify}(\langle\ell\rangle\top) \;=\; \mathtt{expl}(\top,\ell) \\ &\mathsf{simplify}(\langle\ell\rangle\textstyle\bigwedge_{i=1}^{n}\varphi_i) \;= \\ &\qquad \begin{cases} \mathtt{expl}(\bot,\ell\circ\gamma) & \text{if } \mathsf{sc}(\bigwedge_{i=1}^{n}\varphi_i) = \mathtt{expl}(\bot,\gamma) \\ \mathtt{expl}(\top,\ell\circ\gamma) & \text{if } \mathsf{sc}(\bigwedge_{i=1}^{n}\varphi_i) = \mathtt{expl}(\top,\gamma) \\ \langle\ell\rangle C & \text{if } \mathsf{sc}(\bigwedge_{i=1}^{n}\varphi_i) = C \end{cases} \\ &\mathsf{simplify}(\sigma \rhd \varphi) \;= \\ &\qquad \begin{cases} \mathtt{expl}(\bot,\sigma\rhd\gamma) & \text{if } \mathsf{simplify}(\varphi) = \mathtt{expl}(\bot,\gamma) \\ \mathtt{expl}(\top,\sigma\rhd\gamma) & \text{if } \mathsf{simplify}(\varphi) = \mathtt{expl}(\top,\gamma) \\ \sigma \rhd \alpha & \text{if } \mathsf{simplify}(\varphi) = \alpha \end{cases} \end{aligned}$$

Figure 6.4: Selected rules of simplify

When the input formula is guarded by a substitution σ' (case $\sigma' \rhd \varphi$), reduce checks the formula with a composed substitution $\sigma\sigma'$ because the free variables in φ can be in the union of the domains of σ and σ'. The residual formula is still guarded by σ'. When the input is a pair of $\top$ ($\bot$) and an explanation, reduce simply returns the input formula, since the formula has already been reduced (perhaps in a prior invocation of reduce).

6.2.3 Explanations

Next, we define a function simplify that takes a generalized formula and returns another generalized formula, in simpler form. The function simplify serves a dual purpose. First, it rewrites the original formula using basic rules of logic, e.g., it replaces $\varphi \wedge \top$ with φ. Second, and more importantly, if the input formula is equivalent to either $\top$ or $\bot$, it produces a succinct explanation of why that is the case by combining and selectively retaining explanations from the original formula. So, the output of simplify is either a residual policy formula α, or a binary answer ($\top$ or $\bot$) paired with an explanation γ.

We present a few key rules for simplify in Figure 6.4. If the input is $\langle \ell \rangle \top$, then the input formula ($\top$) is trivially satisfied and the output is simply $\top$ coupled with the explanation, which in this case is simply the label ℓ. Hence, the output is $\texttt{expl}(\top, \ell)$. For input conjunctions $\langle \ell \rangle \bigwedge_{i=1}^{n} \varphi_i$, we use a sub-routine sc, which returns the result of simplifying the conjunctive clause (explained later). When sc returns an explanation, simplify returns the same explanation extended with the top-level label of the entire conjunction: $\ell \circ \gamma$. When sc returns a formula α, simplify wraps α with the label ℓ. Disjunction is similarly handled, though not shown. When the input is $\sigma \rhd \varphi$, simplify is invoked on the formula φ. If $\mathsf{simplify}(\varphi)$ returns $\texttt{expl}(\top, \gamma)$ or $\texttt{expl}(\bot, \gamma)$, then the final explanation guards γ with substitution σ. If $\mathsf{simplify}(\varphi)$ returns a formula α, the output is the formula $\sigma \rhd \alpha$. Although not shown, simplify returns the input as is if the input begins with a quantifier.

The sub-routine sc for conjunctive formulas is listed in Figure 6.3. sc is defined inductively on the number of conjuncts. In the base case, there is only one formula, so sc calls simplify. In the inductive case, if the simplification of the first formula φ_1 or the simplification of the rest of the conjunction ($\bigwedge_{i=2}^{n} \varphi_i$) is logical falsity ($\texttt{expl}(\bot, \gamma)$), then the entire conjunction can be rewritten to false and the explanation of why this conjunction is false is the same as that of the branch that makes it false (i.e., γ). On the other hand, if both branches can be simplified to ($\texttt{expl}(\top, \gamma_i)$), then the entire conjunction is true and the explanation combines the explanations of both branches (written $\gamma_1 \oplus \gamma_2$). When at least one of the branches can only be simplified to a formula, the entire conjunctive formula is neither true nor false, and therefore we cannot generate an explanation. Instead, the simplified formula is a conjunction. There is a similar simplification sub-routine for disjunctions, where the roles of truth and falsity are reversed. We omit the straightforward details.

Level 1 (least)	Level 2	Level 3 (most)
provider-id (p_2)	service-type (vl)	OBS-value(va)
INS-company (p)	INS-plan (c)	
	OBS-type (ty)	

Table 6.1: Sensitivity levels in the audit scenario

6.3 Example Scenario

In this chapter, we present a simple audit scenario that illustrates reduce, simplify and HIBE.

6.3.1 Policy

Suppose a HIE's policy for data sharing is that a provider p_1 can send detailed information about a patient q to another provider p_2 if, within a year of such sharing, p_2 bills the insurance company for services provided to q at p_2. The encoding of a HIE's policy for data sharing is shown below. A provider p_1 can send a patient document m to a provider p_2 at time t ($\mathsf{send}(p_1, p_2, m, t)$), where (1) m describes patient q ($\mathsf{hasattrof}(m, q)$), (2) m includes detailed information ty and va about the patient, e.g., ty is the type of observation q is under and va is the result of the observation ($\mathsf{includes}(m, ty, va, t)$), (3) q is classified as type tp and provided with service vl at t ($\mathsf{patientInfo}(q, tp, vl, t)$), (4) p_2 works in organization o ($\mathsf{organization}(p_2, o, t)$), and (5) organization o records that patient q has an insurance plan p from company c at time t ($\mathsf{insuranceInfo}(q, p, c, t)$); then there should be a consequent patient medical bill of type b at time t' ($\mathsf{medical\text{-}bill}(q, b, t')$), t' should be within 365 days of the data sharing, the organization o should note that q has an insurance plan c with company p ($\mathsf{insurance}(q, p, c, o, t')$), and either the bill is from q's visit to p_2 or for an observation carried out by p_2 on q. Predicates $\mathsf{visits\text{-}in\text{-}bill}(q, p_2, vl, o, t')$ and $\mathsf{observes\text{-}in\text{-}bill}(q, p_2, ty, va, o, t')$ represent p_2's records of medical bills of the two specific types.

$$
\begin{aligned}
&\varphi_{pol} = \langle \texttt{DISC} \rangle \\
&\forall p_1, p_2, m, q, t, ty, va, tp, vl, o, p, c \\
&\quad \mathsf{send}(p_1, p_2, m, t) \wedge \mathsf{hasattrof}(m, q) \wedge \\
&\quad \mathsf{includes}(m, ty, va, t) \wedge \mathsf{patientInfo}(q, tp, vl, t) \wedge \\
&\quad \mathsf{organization}(p_2, o, t) \wedge \mathsf{insuranceInfo}(q, p, c, t) \\
&\quad \supset \langle \texttt{AC} \rangle \exists t', b. \mathsf{medical\text{-}bill}(q, b, t') \wedge \\
&\qquad \langle \texttt{BLL} \rangle (\langle \texttt{time} \rangle \mathsf{timein}(t, t', t + 365) \\
&\qquad\quad \wedge \langle \texttt{INS} \rangle \mathsf{insurance}(q, p, c, o, t') \\
&\qquad\quad \wedge \langle \texttt{DJ} \rangle (\langle \texttt{VST} \rangle (\langle \texttt{B} \rangle b = \textit{visit-history} \wedge \\
&\qquad\qquad\quad \langle \texttt{visit} \rangle \mathsf{visits\text{-}in\text{-}bill}(q, p_2, vl, o, t')))
\end{aligned}
$$

$$\vee(\langle \texttt{OBS} \rangle(\langle \texttt{B} \rangle b = \mathit{observation} \wedge \langle \texttt{obsv} \rangle \textsf{observes-in-bill}(q, p_2, ty, va, o, t'))))$$

6.3.2 Audit Logs

The internal log contains information about all the predicates to the left of the implication in φ_{pol} as well as the predicate medical-bill. Predicates visits-in-bill and observes-in-bill record detailed information about patients' hospital visits, which are considered external documentations that belong to the hospital. Both external and internal logs are represented as database tables. Tables visits-in-bill and observes-in-bill are HIBE-encrypted according to levels shown in Table 6.1.

For illustration, we assume that the following substitution σ is the only one that satisfies the condition of the outermost universal quantification on the example log (here, terms like $P1$ starting with uppercase letters are constants):

$$\sigma = p_1 \mapsto P1, p_2 \mapsto P2, m \mapsto M1, q \mapsto Q1, t \mapsto T1, ty \mapsto TY1, va \mapsto VA1, tp \mapsto TP1, vl \mapsto VL1, o \mapsto O1, p \mapsto PI, c \mapsto C1$$

We further assume that timein($T1, T2, T1 + 365$) and timein($T1, T3, T1 + 365$) are true. Below are the (only) log entries about predicates to the right of $\supset$ in φ_{pol}.

medical-bill($Q1$, *visit-history*, $O1$, $T2$)
medical-bill($Q1$, *observation*, $O1$, $T3$)
visits-in-bill($Q1, P2, VL1, O1, T2$)
observes-in-bill($Q1, P2, TY2, VA2, O2, T3$)
insurance($Q1, PI, C1, O1, T2$)

6.3.3 Reduce on Encrypted Data

In the initial phase of audit, the auditor does not possess decryption keys for external data. This poses no problem because reduce can handle log incompleteness; reduce treats the log incomplete in predicates like visits-in-bill, and simply returns such predicates in the residual output. The output of running reduce on φ_{pol} and the example log is shown in the next page.

$$
\begin{aligned}
\varphi_{r1} = & \langle \texttt{DISC} \rangle \sigma \rhd \\
& \langle \texttt{AC} \rangle \sigma_1 \rhd \langle \texttt{BLL} \rangle \\
& (\langle \texttt{time} \rangle \top \wedge \langle \texttt{INS} \rangle \mathsf{insurance}(q, p, c, o, t') \wedge \\
& \langle \texttt{DJ} \rangle (\langle \texttt{VST} \rangle (\langle \texttt{B} \rangle \top \wedge \langle \texttt{visit} \rangle \mathsf{visits\text{-}in\text{-}bill}(q, p_2, vl, o, t')) \\
& \qquad \vee \langle \texttt{OBS} \rangle (\langle \texttt{B} \rangle \bot \wedge \\
& \qquad\qquad \langle \texttt{obsv} \rangle \mathsf{observes\text{-}in\text{-}bill}(q, p_2, ty, va, o, t'))) \\
& \vee \sigma_2 \rhd \langle \texttt{BLL} \rangle \\
& (\langle \texttt{time} \rangle \top \wedge \langle \texttt{INS} \rangle \mathsf{insurance}(q, p, c, o, t') \wedge \\
& \langle \texttt{DJ} \rangle (\langle \texttt{VST} \rangle (\langle \texttt{B} \rangle \bot \wedge \langle \texttt{visit} \rangle \mathsf{visits\text{-}in\text{-}bill}(q, p_2, vl, o, t')) \\
& \qquad \vee \langle \texttt{OBS} \rangle (\langle \texttt{B} \rangle \top \wedge \\
& \qquad\qquad \langle \texttt{obsv} \rangle \mathsf{observes\text{-}in\text{-}bill}(q, p_2, ty, va, o, t')))
\end{aligned}
$$

Here, $\sigma_1 = t' \mapsto T2,\ b \mapsto \textit{visit-history}$
$\sigma_2 = t' \mapsto T3,\ b \mapsto \textit{observation}$

The existentially quantified variables t' and b in φ_{pol} have two possible substitutions, corresponding to the two entries in the medical-bill table. The residual formula hence contains a disjunction over these two possibilities.

6.3.4 Simplification

Next, we call simplify on φ_{r1} to condense as many explanations as possible. This yields:

$$
\begin{aligned}
\varphi_{s1} = & \langle \texttt{DISC} \rangle \sigma \rhd \\
& \langle \texttt{AC} \rangle \sigma_1 \rhd \langle \texttt{BLL} \rangle \\
& (\texttt{expl}(\top, \texttt{time}) \wedge \langle \texttt{INS} \rangle \mathsf{insurance}(q, p, c, o, t') \wedge \\
& \langle \texttt{DJ} \rangle (\langle \texttt{VST} \rangle (\texttt{expl}(\top, \texttt{B}) \wedge \\
& \qquad \langle \texttt{visit} \rangle \mathsf{visits\text{-}in\text{-}bill}(q, p_2, vl, o, t')) \\
& \qquad \vee \texttt{expl}(\bot, \texttt{OBS} \circ \texttt{B}))) \\
& \vee \sigma_2 \rhd \langle \texttt{BLL} \rangle \\
& (\texttt{expl}(\bot, \texttt{time}) \wedge \langle \texttt{INS} \rangle \mathsf{insurance}(q, p, c, o, t') \wedge \\
& \langle \texttt{DJ} \rangle (\texttt{expl}(\bot, \texttt{VST} \circ \texttt{B}) \\
& \qquad \vee \langle \texttt{OBS} \rangle (\texttt{expl}(\top, \texttt{B}) \wedge \\
& \qquad\qquad \langle \texttt{obsv} \rangle \mathsf{observes\text{-}in\text{-}bill}(q, p_2, ty, va, o, t'))))
\end{aligned}
$$

6.3.5 Requesting Decryption Keys

To proceed further, we must decrypt the insurance table and either the observes-in-bill table or the visits-in-bill table. Since the maximum sensitivity level of entries in visits-in-bill is lower than that in observes-in-bill, the audit agent asks the KGC for keys at level 2 (the maximum level of visits-in-bill and insurance).[1] The KGC generates keys based on its master key mk and gives them to the audit agent. Both the KGC and the audit agent may log why the keys were generated (by recording the residual formula φ_{s1}) to aid a subsequent audit of this audit process. The audit agent then decrypts entries in those two tables and provides the formula φ_{s1} with the decrypted tables (added to the original log) to reduce. The output of reduce is the following formula φ_{d1}. Note that the clause guarded by σ_2 remains the same because even the extended log contains no information to reduce it.

$$
\begin{aligned}
&\varphi_{d1} = \\
&\ \langle \mathtt{DISC} \rangle \sigma \rhd \\
&\ \langle \mathtt{AC} \rangle \sigma_1 \rhd \langle \mathtt{BLL} \rangle \\
&\ (\mathtt{expl}(\top, \mathtt{time}) \wedge\ \mathtt{expl}(\top, \mathtt{INS}) \wedge \\
&\ \ \langle \mathtt{DJ} \rangle (\langle \mathtt{VST} \rangle (\mathtt{expl}(\top, \mathtt{B}) \wedge \mathtt{expl}(\top, \mathtt{visit})) \vee \mathtt{expl}(\bot, \mathtt{OBS} \circ \mathtt{B}))) \\
&\ \ \vee \sigma_2 \rhd \langle \mathtt{BLL} \rangle \\
&\ (\mathtt{expl}(\bot, \mathtt{time}) \wedge\ \langle \mathtt{INS} \rangle \mathsf{insurance}(q, p, c, o, t') \wedge \\
&\ \ \langle \mathtt{DJ} \rangle (\mathtt{expl}(\bot, \mathtt{VST} \circ \mathtt{B}) \\
&\qquad \vee \langle \mathtt{OBS} \rangle (\mathtt{expl}(\top, \mathtt{B}) \wedge \\
&\qquad\qquad \langle \mathtt{obsv} \rangle \mathsf{observes\text{-}in\text{-}bill}(q, p_2, ty, va, o, t'))))
\end{aligned}
$$

6.3.6 Simplification and Explanation

Finally, simplify is run on φ_{d1} to obtain the following output:

$$
\begin{aligned}
\varphi_{s2} = \mathtt{expl}(\top,\ & \mathtt{DISC} \circ \sigma \rhd \mathtt{AC} \circ \sigma_1 \rhd \langle \mathtt{BLL} \rangle \circ \\
& (\mathtt{time} \oplus \mathtt{INS} \oplus (\mathtt{DJ} \circ \mathtt{VST} \circ (\mathtt{B} \oplus \mathtt{visit})))
\end{aligned}
$$

The result indicates that the log satisfies the policy. The reason is the following: (1) σ is the only substitution that makes the conditions associated

[1] For simplicity, we assume here that the audit agent decrypts entire tables atomically. In practice, it could decrypt only specific rows of interest.

with the action send true and (2) the conditions required for such a send (AC) under σ are true. The explanation of (2) is that there exists a substitution σ_1 that matches an entry in medical-bill and makes BLL true. More concretely, the time of the bill, the insurance information, and hospital's billing record of the patient all satisfy the policy constraints. In particular, the hospital's record shows that the patient visited the hospital (VST).

CHAPTER 7

IMPLEMENTATION AND EVALUATION

To validate our proposal, we implement the HIE (Figure 4.1) based on IHE Profile XDS.b [9]. It supports the sharing of patient clinical documents based on the HIE's document registry which keeps a patient document index and location where the documents are stored in. Based on these, the web-based document viewer looks for patient documents based on patient information. We implement a Java API to create ATNA-based XML logs [11] on top of HIE. We report our evaluation of HIBE key generation, decryption and the reduce algorithm based on a policy encoding the guidelines [36] of the ONC for Health Information Technology and a synthetic audit log. All experiments are performed on a machine with an Intel Core i7 2.3GHz processor and 1GB of memory, running Ubuntu 12.04. We use the Charm library [37] to implement the HIBE module. In particular, we use a symmetric curve with a 512-bit base field to initiate a group in the elliptic curve with bilinear pairings. For illustration purposes, we assume a maximum depth of three sensitivity levels in the hierarchy. To encrypt arbitrary messages with HIBE, we use a hybrid encryption scheme: we extract a session key after hashing [14] a random element from the message space of HIBE, encrypt messages with the session key via AES (CBC mode) symmetric encryption and encrypt the random element using HIBE [2].

7.1 Implementation

7.1.1 HIE Simulation

We chose to use NIST's open source package, IheOS [38] adopting IHE Profile XDS.b, described in Chapter 4. We used both Apache Tomcat 5.5.23 as a web application server and Apache 2.2.23 as a web server for our patient

document sharing server, built a web service for a patient document viewer, and used Postgresql 8.4.17 to create the Document Registry. To adopt ATNA as the audit mechanism in our HIE system, OpenSSL 1.0.1 14 was used for the user and node authentication and RFC-3881-based XML logs were generated by a Java API ATNA generator (Appendix A) for the audit trail.

7.1.2 HIBE Module

We used the Charms library [37] to implement our cryptographic module HIBE-BBG05 (Appendix D) and prototype an HIBE scheme [2]. As a platform to implement HIBE, we used Python2.7, Pyparsing 2.0.1, GMP 5.1.2 and PBC 5.14. To initiate a group in the elliptic curve with billinear pairings, we used a symmetric curve with a 512-bit base field. We decided on a maximum of three depths of sensitivity in the hierarchy for the evaluation shown in Chapter 7.4. To encrypt arbitrary messages with the HIBE scheme, we used key encapsulation. In other words, we extracted a session key after hashing the group GT elements with a Waters hash technique, encrypted a message with a session key via AES (CBC mode) symmetric encryption and encrypted a session key using HIBE.

7.2 Policy

According to the ONC, providers requesting a patient's IIHI by electronic means for treatment must verify a treatment relationship with a patient by attestation or artifacts such as patient registration, prescriptions, consults, and referrals. The top-level encoding of the policy is shown below and consists of a disjunction of six sub clauses, and at least one must be satisfied for each access. The details of these clauses are presented in Appendix C and φ_{pol}, shown in Chapter 6.3, is a simplified encoding of $\varphi_{Billing}$.

$$
\begin{aligned}
\varphi_{ONC} = \forall p_1, p_2, m, q, t, ty, va, tp, vl, o, p, c \\
&\mathsf{send}(p_1, p_2, m, t) \wedge \ \mathsf{hasattrof}(m, q) \wedge \\
&\mathsf{includes}(m, ty, va, t) \wedge \ \mathsf{patientInfo}(q, tp, vl, t) \wedge \\
&\mathsf{organization}(p_2, o, t) \wedge \ \mathsf{insuranceInfo}(q, p, c, t) \\
&\supset \varphi_{Exception} \vee \varphi_{Billing} \vee \varphi_{Registration} \vee \varphi_{Prescription} \\
&\quad \vee \varphi_{Referral} \vee \varphi_{Consult}
\end{aligned}
$$

7.3 Audit Logs

We generate synthetic data representing both external audit logs and internal ATNA logs (Appendix B). The generated ATNA log has a size of 5.7 MB, which represents 9,644 accesses to HIE over 4 months (this realistic number is based on Johnson *et al.*'s data [39]). The external audit data has a size of 12 MB, and includes roughly 9,644 entries about patient registration, billing and referral. The external logs are encrypted using HIBE with three pre-defined sensitivity levels.

7.3.1 External documentation Simulation

To simulate the information about patient registration, billing, and prescription, which we relied upon for external documentation in this paper, we utilized HL7 [10] messages. For patient registration, we simulated data based on "Register a patient", which is an HL7 ADT A04 meesage, since this message is used in updating patient clinical data to change address or add next of kin. This clinical data will then be sent upon request to different places such as laboratories. For patient billing, we simulated data based on "Order", which is an HL7 ORM 001 message, "Add patient account", which is an HL7 BAR P01 message, and "Unsolicited transmission of an observation", which is an HL7 ORU R01 message. The ORM message is used in laboratories when they receive orders from referring providers via an HL7 interface, which indicates that some of entries in the message are needed to complete a medical bill. The BAR and the ORU messages are in fact used in imaging centers when referring providers request patient billing information and final coding. For prescriptions, we simulated the PMP, as described in Chapter 4, based on the Illinois Compiled Statutes (ILCS) [40]. All entries of simulated external documentation are stored in three tables, the Patient Registration, the PMP and the Billing, in the Audit Logs repository, which is an SQLite database.

7.3.2 ATNA log

Even though the ATNA generator (Appendix A) creates ATNA log entries for log-in and log-out in HIE, assigning a new patient ID to a new patient, submitting a new patient document(s) and retrieving a patient document(s),

	Key Gen. (ms)	Session Key Dec. (ms)	Msg Dec. (ms)	Reduce (ms)	Single access (ms)	Day (s)	Month (m)
`Level3`	17.73	20.76	0.06	42.6	81.15	6.57	3.26
`Level2`	11.73	13.73	0.04	41.3	66.80	5.41	2.68

Table 7.1: Consumption time for HIBE and reduce

we only use the logs that are generated by retrieval of a patient document(s) for audit analysis since such retrieval accompanies the same key entries as in other transactions. All entries in this log are captured in tables whose names are Predicates in SQLite. Especially sensitive information, such as a diagnosis code, is already decoded in the audit logs prior to audit analysis as specified in the ATNA Profile. Therefore, we do not include such a decoding process in our audit system.

7.4 Evaluation Results

We evaluate the efficiency and scalability of both HIBE and the reduce algorithm. We use the audit scenario shown in Chapter 6.3 and break it into three phases. In the first phase, the auditor does not have any keys, and we measure the time reduce takes to generate a residual policy; in the second phase, we measure the time it takes the KGC to generate a decryption key given an ID, and the time it takes to decrypt relevant log data using the key (because our encryption is hybrid, the latter further splits into the time taken to decrypt the symmetric key, and the time take to decrypt the data using the symmetric key); in the third phase, we run reduce again and measure the time reduce takes to check the residual policy on the decrypted data. We run the audit scenario on two accesses, one requires a decryption key of level 2, and the other requires a decryption key of level 3. The size of messages encrypted up to level 2, shown in Table 7.1, is 416B including the session key and the size of messages encrypted up to level 3 is 580B including the session key. Table 7.1 summarizes our results. All numbers are averages of 20 trials (all have negligible standard deviations). The first column shows the time taken to generate HIBE decryption keys, the second column indicates the

time needed to decrypt a session key with the HIBE key, the third column shows the time to decrypt a message using AES and the fourth column shows the total time consumed by reduce, which is derived by adding up the time for each iteration of reduce(before and after decryption). As can be seen, the total time is split almost evenly between reduce and the cryptographic operations for data at level 3 and is dominated by reduce for data at level 2. Johnson et al [39] report approximately 81 accesses per day in a typical HIE. Based on this number, we calculate the total time for auditing all accesses in a day and in a month to be 6.57/5.41 seconds (level 3/level 2) and 3.26/2.68 minutes, respectively. Since audit is an offline process, we consider these numbers practical.

CHAPTER 8

DISCUSSION

8.1 Practical Issues in Deployment

Integrating our audit architecture into an actual distributed healthcare environment will be challenging since it will require the HCOs not only to use the HIE system but to accept the audit infrastructure as sufficiently secure. In addition, the details of the process for accepting diverse formats of external documentation from HCOs and converting them into the ATNA based audit record, as would be needed for use with our infrastructure, will need to be worked out. In addition, even though HIBE supports privacy preserving audit, we still need to monitor audit system during collection and analysis of audit records if the audit system is compromised by malicious auditors, which subsequently discloses sensitive information in the audit data. Furthermore, our current work did not involve access control, however, we need to utilize our explanations about the legitimacy of access to establish access control for HIE in the future since audit analysis can only notify of the violations after the fact. Establishing fine-grained access control for HIE is cumbersome by exceptional cases such as emergency that might result in false positives or false negatives in the application of access control, and which can significantly affect the effective running of HIE. For example, Bob accesses Alice's clinical record through HIE in an emergency: his allowed access level is low since he is not her PCP. In this case, HIE needs to give him a higher level of access to provide him with sufficient health information for Alice's treatment. We need an access control model that can automatically process these exceptions, which can be feasible with our audit infrastructure. Details concerning converting file format of documentation, access control, and follow-up monitoring of our infrastructure are left for future work.

8.2 Limitation of Audit Algorithm

In the examples shown in Chapter 6.3, the encrypted tables are not used to generate substitutions for quantified variables; i.e., they do not appear in the guards of quantified formulas. It could be cases where the guards refer to external data. For instance, ty and va can be encrypted in $\mathsf{includes}(m, ty, va, t)$. The current method for handling encrypted data does not work for such scenario because reduce requires that no predicates in the guards be subjective. We can augment reduce to allow encrypted data to appear in the guard position as follows. When searching for substitutions for quantified variables based on a guard c, we mark the substitutions that depend on encrypted data to indicate that such substitution exists, but the concrete values require decryption. In the simplification phase, we decide whether decryption is needed. It could be the case that such decryption is not needed. For instance, if `VST` branch is true, then `OBS` is irrelevant, so we do not need to decrypt ty or va. In more complicated scenarios where there are multiple encrypted predicates in the guard, and identifying the substitutions requires examining the values of encrypted data, we need additional mechanisms to integrate decryption key requests with reduce. We leave them as future work.

CHAPTER 9

CASE STUDY

9.1 Auditor Privilege

According to ISO specifications [41], the EHR audit log and EHR requestors and provider must all conform to the ISO data flow protocol as well as to the access control policy, which consists of access control factors being collected in the data flow. ISO has spelled out sensitivity levels for EHR records, functional roles [5] of EHR recipients, and mapping instructions applicable to various recipient functional roles and record sensitivity levels. We will

[1]Class of identity, pseudo identity, endpoint address, organization, etc.

[2]Provider Directory.

[3]Hospital, IDN, Provider Org, Provider, HIE, Connector, etc.

[4]for individuals, NIST has levels of proofing, for organizations, the individual representing the org is proofed, and then the org identity is established through records search. Not sure how apps (services) are proofed, or if that is even relevant, although it maybe should be.

[5]Functional role is a role at the hospital such as consultant pediatrician at a hospital or head of child screening for the region. The principals (person, agent, etc) can hold one or more functional roles

Identity Axis	**Policy Axis**	**Contract Axis**
Identity (name)	stores a copy?(Y/N)	Reciprocal obligations
Identity Details [1]	PD[2] policy	Notification of breach
Identity Type[3]	Patient disambiguation	Explicit AGMT
Proofing level[4]	MPI MGMT	Explicit practices
Certified?(Y/N)	Consent REQ	Suspending REQ
There is chain(s)?(Y/N)	Privacy policies	Termination REQ
Accreditation	Security policies	Update REQ
Accrediting entity	Audit review policy	Inspection AGMT
User Authentication level	Standards supported	AGMT Version
User authorization type	Profiles supported	
Authorization content	Permitted purposes	
Contact person		

Table 9.1: User Trust Attributes

utilize these specifications with two strategies in mind. The first one is to build future access control based on analysis of audit logs that conform to these specifications. The pieces of information in the audit log explanations generated by our audit log subsystem will be utilized to construct an access policy model and we will request specific additional information that has not already been gathered through the audit log from external third parties based on ISO-specified standards. Therefore, we will establish a plausible access policy for HIE by analyzing the audit log. Our second strategy is to incorporate the access policy into the auditor viewer. If we need to allow patients or medical providers as well as auditors to investigate the audit log, we will comply with ISO specifications to limit all three of them to accessing audit information only at the appropriate level of sensitivity.

9.2 Access Policy for HIE

HHS has provided a document entitled Trust Framework for HIE [42], which suggests that when a provider requests an EHR, the request has to include trust attribute profile, which explains the conditions for a trusted exchange. This profile is established and verified through means such as self-attestation, certification, or accreditation. If a requesters profile meets local policy requirements, the requested records will be sent to the requester. HHS recommends grouping attributes into three categories, namely identity, policy, and contract to describe the initial trust framework, since EHR communications may vary depending on local policy. These groupings are helpful to determine whether to share information between unaffiliated entities. In addition, HHS proposes that the automation of the evaluation of trust attributes (Table 9.1) can make the process of assessing differences in trust elements more quick and efficient. Accordingly, we will extend our work, a logic-based reasoning with formal policy, to effectively automate a system to verify whether the trust attributes of principals satisfy the policy or not.

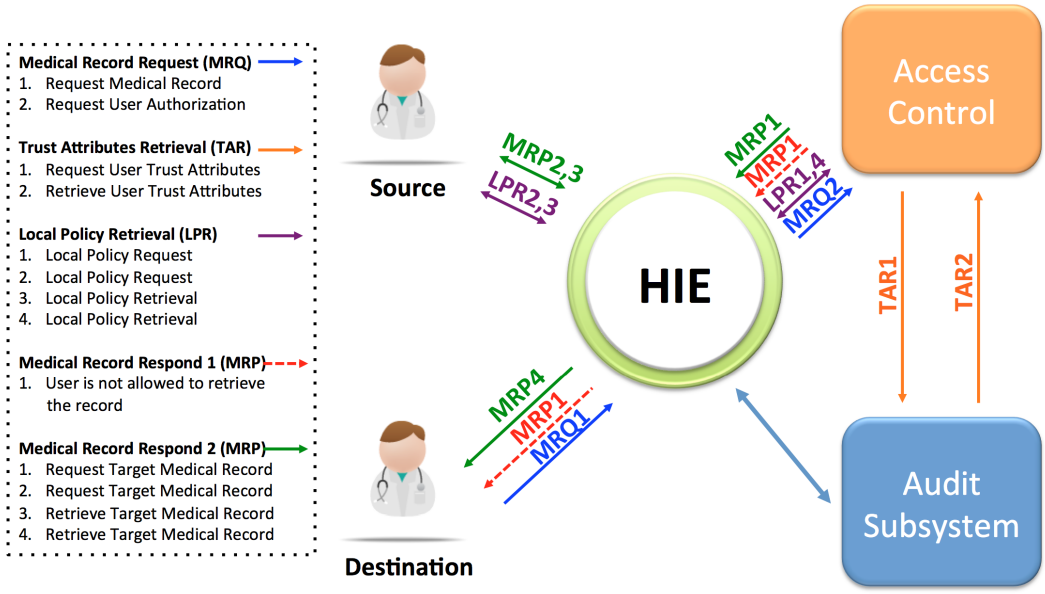

Figure 9.1: User Access control via Trust Attributes in Audits

9.3 User Authorization Using Trust Attributes

Figure 9.1 illustrates the overview of access control mechanism for HIEs. The audit log is regularly accumulated through the audit subsystem and trust attributes (Table 9.1) are extracted per a user such as medical provider, on a daily basis. When a provider requests the medical record via the HIE (path MRQ1), the HIE needs to get the user access permission to the target record from the access control (path MRQ2). Based on requester information, the access control retrieves user trust attributes from the audit subsystem (path TAR1 and TAR2). If attributes don't exist, the request will be manually reviewed by the source provider. The access control encodes user trust attributes into the first-order logic and requests applicable local policy from the source provider (path LPR1 and LPR2). If the policy is successfully retrieved (path LPR3 and LPR4), it is converted into the first-order logic as well. Since the access control includes the logic-based algorithm similar to our audit algorithm, it can successfully check if user trust attributes conform to the local policy. If the HIE decides to admit user request based on the result (true or false) from access control module, the target medical record is sent to the requester from the source provider (path MRP3 and MRP4). If not, the user request is denied (path MRP1).

CHAPTER 10

CONCLUSION

We have proposed an audit infrastructure for broker-based HIE systems that limits the information shared through HIE. The audit logs complying with ATNA are stored in a centralized audit repository to make it easy to effectively audit HIE, with a bare minimum of them being decrypted only as needed to justify each access using HIBE. Our extended logic-based audit algorithm provides further evidence of the auditors behavior, and thus increases the trustworthiness of the system. Initial performance evaluation of a prototype implementation shows that our proposed infrastructure is practical and scalable. As future work, we plan to implement the extended audit algorithm and investigate the possibility of combining audit and access control mechanisms.

APPENDIX A

ATNA LOG GENERATOR

```
// WriteXml_ret.java
package xml.create;

import java.io.*;
import java.util.*;
import java.net.*;
import java.text.*;
import java.lang.*;
import org.w3c.dom.*;
import javax.xml.transform.*;
import javax.xml.transform.stream.*;
import javax.xml.transform.dom.*;
import javax.xml.parsers.*;
import db.UserInfo;

public class WriteXml_ret
{
       public void writeXml(int outcome,int filenum,String
          alterid,String pid, String[] partid, String[] repid,
          String[] hcid, String[] consult, String uid, String[]
          sen, String gua,String com,String plan,String[]
          author,String[] pclass,String[] type,String[]
          value,String[] order) throws FileNotFoundException,
          UnknownHostException
       {
              try
              {
                     DocumentBuilderFactory docFactory =
                        DocumentBuilderFactory.newInstance();
```

```
DocumentBuilder docBuilder =
    docFactory.newDocumentBuilder();
/*ATNA Elements*/
//Time
        Calendar rightnow =
         Calendar.getInstance();
        String timenow = "";
        timenow =
        Integer.toString(rightnow.get(Calendar.YEA...
        ...
        ..
        //Client IP address(Local)
        InetAddress ip=InetAddress.getLocalHost();
//File name
String fname = "retb_"+timenow;
//User name retrieval with user id
        UserInfo u = new UserInfo();
        String uid =
            (String)session.getAttribute("uid");
        String username = u.getName(uid);

//ATNA repository(Local)
        File fileXml =
         new File("/usr/local/tomcat1/webapps/
          atna_repository/"+fname+".xml");
        Document document = null;
        Element rootElement = null;
        Element eventElement = null;
        Element eventidElement = null;
        Element eventcodeElement = null;
        Element activeElement = null;
        ...
        ..
        Document doc = docBuilder.newDocument();
        rootElement =
         (Element)doc.createElement("AuditMessage");
        rootElement.setAttribute("xmlns:xsi",
            "http://www.w3.org/2001/XMLSchema-instance");
```

```
rootElement.setAttribute
("xsi:noNamespaceSchemaLocation",
 "healthcare-security-audit.xsd");
doc.appendChild(rootElement);
/* Event Identification */
eventElement =
    (Element)doc.createElement("EventIdentification");
eventElement.setAttribute("EventActionCode",
    "R");
...
..
/* Active Participant: Document Repository */
activeElement =
    (Element)doc.createElement("ActiveParticipant");
activeElement.setAttribute("UserID",
    "http://localhost:9080/tf6/services/xdsrepositoryb");
activeElement.setAttribute("UserIsRequestor",
    "false");
...
..
/* Active Participant: Client */
activeElement2 =
    (Element)doc.createElement("ActiveParticipant");
activeElement2.setAttribute("UserID", uid);
activeElement2.setAttribute("UserName",
    username);
...
..
/* Audit Source Identification */
auditElement = (Element)doc.createElement
("AuditSourceIdentification");
auditElement.setAttribute("AuditSourceID",
 "http://localhost:9080/tf6/services/xdsrepositoryb");
..
/* Participant Object Identification: Patient
    */
partElement = (Element)doc.createElement
("ParticipantObjectIdentification");
```

```
                partElement.setAttribute("ParticipantObjectID",
                    pid);
                ...
                ..
                /* Participant Object Identification: Patient
                    Medical Document Details */
                for(int i=0;i<filenum;i++)
                {
                        partElement2=(Element)doc.createElement
                        ("ParticipantObjectIdentification");
                                partElement2.setAttribute
                                ("ParticipantObjectID",
                                    partid[i]);

                                    partElement2.setAttribute
                                ("ParticipantObjectTypeCode",
                                    "2");
                                    partElement2.setAttribute
                                ("ParticipantObjectTypeCodeRole",
                                    "3");
                                    partElement2.setAttribute
                                ("ParticipantObjectSensitivity",
                                    sen[i]);
                        ...
                        ..
                }
        /* Write and export XML */
        TransformerFactory transformerFactory =
            TransformerFactory.newInstance();
        Transformer transformer =
            transformerFactory.newTransformer();
            transformer.setOutputProperty(OutputKeys.ENCODING,
            "UTF-8");
        ...
        ..
```

APPENDIX B

ATNA LOG

Figure B.1: Retrieve Patient Documents


```
−<AuditMessage xsi:noNamespaceSchemaLocation="healthcare-security-audit.xsd">
  −<EventIdentification EventActionCode="R" EventDateTime="2013.10.27.4.54.59" EventOutcomeIndicator="0">
     <EventID code="110106" codeSystemName="DCM" displayName="Export"/>
     <EventTypeCode code="ITI-43" codeSystemName="IHE Transactions" displayName="Retrieve Document Set"/>
   </EventIdentification>
  −<ActiveParticipant NetworkAccessPointID="127.0.0.1" NetworkAccessPointTypeCode="2" UserID="http://localhost:9080
   /tf6/services/xdsrepositoryb" UserIsRequestor="false">
     <RoleIDCode code="110153" codeSystemName="DCM" displayName="Source"/>
   </ActiveParticipant>
  −<ActiveParticipant NetworkAccessPointID="127.0.1.1" NetworkAccessPointTypeCode="2" UserID="cs1010"
   UserIsRequestor="true" UserName="Drake^Joe">
     <RoleIDCode code="110152" codeSystemName="DCM" displayName="Destination"/>
   </ActiveParticipant>
   <AuditSourceIdentification AuditSourceID="http://localhost:9080/tf6/services/xdsrepositoryb"/>
  −<ParticipantObjectIdentification ParticipantObjectID="f0728f659e9f4d8" ParticipantObjectName="Harry^Hie^Porter"
   ParticipantObjectTypeCode="1" ParticipantObjectTypeCodeRole="1">
     <ParticipantObjectIDTypeCode code="2" codeSystemName="RFC-3881" displayName="Patient Number"/>
     <ParticipantObjectDetail type="Guarantor" value="Clara"/>
     <ParticipantObjectDetail type="InsuranceCom" value="WA02"/>
     <ParticipantObjectDetail type="InsurancePlan" value="PPO"/>
   </ParticipantObjectIdentification>
  −<ParticipantObjectIdentification ParticipantObjectID="1.2.3.4.5.2352" ParticipantObjectSensitivity="sensitive"
   ParticipantObjectTypeCode="2" ParticipantObjectTypeCodeRole="3">
     <ParticipantObjectIDTypeCode code="9" codeSystemName="RFC-3881" displayName="Report Number"/>
     <ParticipantObjectDetail type="Repository Unique Id" value="1.19.6.24.109.42.1.5"/>
     <ParticipantObjectDetail type="ihe:homeCommunityID" value=""/>
     <ParticipantObjectDetail type="Author" value="ra1010"/>
     <ParticipantObjectDetail type="Classification" value="Physical"/>
     <ParticipantObjectDetail type="Patient-class" value="I"/>
     <ParticipantObjectDetail type="Observation-1082^REVIEW OF SYSTEMS, CENTRAL NERVOUS SYSTEM^DCT"
     value="207^DEPRESSION^DCT"/>
     <ParticipantObjectDetail type="Order-type" value="NW"/>
   </ParticipantObjectIdentification>
  −<ParticipantObjectIdentification ParticipantObjectID="1.2.3.4.5.2353" ParticipantObjectSensitivity="normal"
   ParticipantObjectTypeCode="2" ParticipantObjectTypeCodeRole="3">
     <ParticipantObjectIDTypeCode code="9" codeSystemName="RFC-3881" displayName="Report Number"/>
     <ParticipantObjectDetail type="Repository Unique Id" value="1.19.6.24.109.42.1.8"/>
     <ParticipantObjectDetail type="ihe:homeCommunityID" value=""/>
     <ParticipantObjectDetail type="Author" value="pq1010"/>
     <ParticipantObjectDetail type="Classification" value="Consult"/>
     <ParticipantObjectDetail type="Patient-class" value="O"/>
     <ParticipantObjectDetail type="Prescription-DrugCode" value="0777-3105-02"/>
     <ParticipantObjectDetail type="Order-type" value="NW"/>
   </ParticipantObjectIdentification>
 </AuditMessage>
```


Figure B.2: Referring


```
−<AuditMessage xsi:noNamespaceSchemaLocation="healthcare-security-audit.xsd">
  −<EventIdentification EventActionCode="R" EventDateTime="2013.7.20.4.16.51" EventOutcomeIndicator="0">
     <EventID code="110106" codeSystemName="DCM" displayName="Referring"/>
   </EventIdentification>
  −<ActiveParticipant NetworkAccessPointID="127.0.0.1" NetworkAccessPointTypeCode="2" UserID="cs1010" UserIsRequestor="false"
   UserName="DRAKE^JOE">
     <RoleIDCode code="110153" codeSystemName="DCM" displayName="Source"/>
   </ActiveParticipant>
  −<ActiveParticipant NetworkAccessPointID="127.0.1.1" NetworkAccessPointTypeCode="2" UserID="hi1010" UserIsRequestor="true"
   UserName="JONE^RANDALL">
     <RoleIDCode code="110152" codeSystemName="DCM" displayName="Destination"/>
   </ActiveParticipant>
   <AuditSourceIdentification AuditSourceID="http://localhost:9080/tf6/services/xdsrepositoryb"/>
  −<ParticipantObjectIdentification ParticipantObjectID="981139165" ParticipantObjectTypeCode="1" ParticipantObjectTypeCodeRole="1">
     <ParticipantObjectIDTypeCode code="2" codeSystemName="RFC-3881" displayName="Patient Number"/>
   </ParticipantObjectIdentification>
  −<ParticipantObjectIdentification ParticipantObjectID="REF1010" ParticipantObjectTypeCode="2" ParticipantObjectTypeCodeRole="3">
     <ParticipantObjectIDTypeCode code="9" codeSystemName="RFC-3881" displayName="Report Number"/>
     <ParticipantObjectDetail type="Patient ID" value="981139165"/>
     <ParticipantObjectDetail type="ihe:homeCommunityID" value="aa"/>
   </ParticipantObjectIdentification>
 </AuditMessage>
```

Figure B.3: Billing

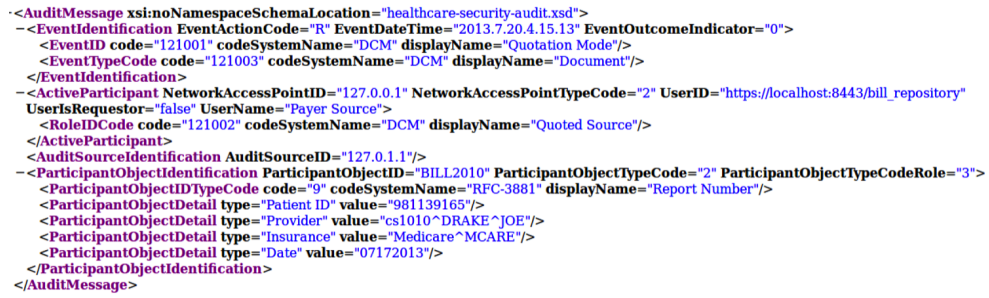

Figure B.4: Prescription

APPENDIX C

FORMAL PRIVACY POLICY

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Policy Title : Privacy and Security Framework Requirements and Guidance for the State Health Information Exchange
%%                Cooperation Agreement Program (March 22 2012, ONC-HIE-PIN-003, page 8)
%% Policy  components.
%% p1: source(or referring) provider id
%% p2: recipient(or referred) provider id
%% m: patient document ID
%% q: patient id
%% t, t',tt:  event date
%% ty, tp: type
%% va, vl: value
%% o: organization
%% p: insurance plan
%% c: insurance company
%% g: guarantor
%% v: observation value
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
<POL/DISCLOSE>
    all [p1][p2][m][q][t][ty][va][tp][vl][o][p][c]
        (and

                (send p1 p2 m t)
                (hasattrof m q)
                (includes m ty va)
                (patientinfo q tp vl t)
                (organization p2 o t)
                (insuranceinfo q p c t)

        )
        (or
                (<POL/Exception>

                                (or
                                        (and ex[t']

                                                (workin p2 public-health-authority t')
                                                (time_in (plus t ~30) t' t)
                                        )
                                        (and ex[t']

                                                (permission p2 q t')
                                                (time_in (plus t ~30) t' t)
                                        )
                                )
                )
                (<POL/Billing>



                                (or

                                        (and
                                                (ex[t']
                                                        (medical_bill q order o t')
                                                        (time_in  t t' (plus t 365))

                                                )
```

```
                                        (order_in_bill q p2 va o t) %% va:order-type (level2), p2:provider ID (level1)
                                        (insurance q p c t)
                                        (violations11 p1 p2 m q o va p c order t)

                                )
                                (and
                                        (ex[t']
                                                (medical_bill q visit-history o t')
                                                (time_in  t t' (plus t 365))
                                                 %% vl:patient_class (level1), p2:provider ID (level1)

                                        )
                                        (visits_in_bill q p2 vl o t)
                                        (insurance q p c t)
                                        (violations11 p1 p2 m q o va p c visit-history t)

                                )
                                (and
                                        (ex[t']
                                                (medical_bill q observation o t')
                                                (time_in  t t' (plus t 365))
                                                %%p2:provider ID (level1), observation_id (level2), observation_value (level3)

                                        )
                                        (observes_in_bill q p2 ty va o t)
                                        (insurance q p c t)
                                        (violations11 p1 p2 m q o va p c observation t)

                                )

                        )



                )
                (<POL/ONC>
                        (or
                        (<POL/ONC/Registration>

                                (or
                                        (and
                                                (isadoctor p2 t)
                                                (ex[t']
                                                        (inrelation q p2 PCP t')
                                                        (time_in (plus t ~40) t' t)
                                                )



                                        )
                                        (and
                                                (registration q visit-history o t)
                                                (visits q p2 vl o t) %% vl:patient_class (level2), p2:providerID (level1)

                                        )
                                        (and
                                                (registration q observation o t)
                                                (observes q p2 t)

                                        )
                                        (and
                                                (registration q procedures o t)
                                                (procedure q p2 vl o t) %% p2:provider ID (level1), vl:procedure details (level3)
                                                %%vl:ICD9 code (level2), p2:providerID (level1)

                                        )
                                )
                        )
                        (<POL/ONC/Prescription>

                                (and
                                        (ex[t']
                                                (prescription q PMP t')
                                                (time_in (plus t ~7) t' t)
                                        ) %% va:drug_code (level2), p2:providerID (level1)
                                        (pmp p2 q va o t)

                                )

                        )
                        (<POL/ONC/Referral>

                                (and
                                        (ex[t']
                                                (refers p1 p2 q transition-of-care t')
                                                (time_in (plus t ~365) t' t)
                                        )
                                )
                        )
                        (<POL/ONC/Consult>

                                (and
                                        (ex[t']
                                                (refers_consult p1 p2 q consult t')
                                                (time_in (plus t ~365) t' t)
                                        )
                                )
                        )

                )
        )
)
```

APPENDIX D

HIBE MODULE

```
\\hibenc_bbg05.py

'''
Created on Jun 26, 2013

@author: Se Eun Oh (seeunoh2@illinois.edu)
'''
#from charm.toolbox.pairinggroup import
    PairingGroup,ZR,G1,G2,GT,pair

from charm.toolbox.pairinggroup import
    PairingGroup,ZR,G1,G2,GT,pair
from charm.toolbox.ecgroup import ECGroup
from charm.core.math.pairing import hashPair as sha1
from charm.toolbox.symcrypto import AuthenticatedCryptoAbstraction
from charm.toolbox.iterate import dotprod2
from charm.toolbox.hash_module import Waters
import sys
import time
debug = False
class HIBE_BBG05_KeyEnc:
    """
    >>> from charm.toolbox.pairinggroup import PairingGroup, GT
    >>> group = PairingGroup('SS512')
    >>> hibe = HIBE_BBG05(group)
    >>> (master_public_key, master_key) = hibe.setup()
    >>> ID = "bob@mail.com"
    >>> (public_key, secret_key) = hibe.extract(3,
        master_public_key, master_key, ID)
```

```
>>> msg = group.random(GT)
>>> cipher_text = hibe.encrypt(master_public_key, public_key,
    msg)
>>> decrypted_msg = hibe.decrypt(public_key, secret_key,
    cipher_text)
>>> decrypted_msg == msg
"""
def __init__(self, groupObj):
    global group, ibenc
    group = groupObj

def setup(self, l=3, z=32):
    """ k represents maximum depth of HIBE system,
        z represents the bit size of each integer_j of identity.
    """
    assert l > 0, "invalid number of levels (need more than 0)"
    alpha = group.random(ZR)
    beta = group.random(ZR)
    """paring group vs ecgroup"""
    g = group.random(G2)
    g1 = g ** alpha
    g2 = group.random(G2)
    g3 = group.random(G2)
    h = {}
    for i in range(l) :
        h[i] = group.random(G2)
    g2alpha = g2 ** alpha
    mpk = { 'g': g, 'g1':g1, 'g2':g2, 'g3':g3, 'h':h, 'z':z,
        'l':l }
    mk = { 'g2alpha':g2alpha }
    return (mpk, mk)

def keygen(self,level,mpk,mk,ID):
    k = level
    #assert k >= 1 and k <= mpk['l'], "invalid level: 1 - %d" %
        mpk['l']
    print "z====",mpk['z']
    I = Waters(group, k, mpk['z']).hash(ID)
```

```
    r = group.random(ZR)
    hi = {}
    for i in range(k) :
        hi[i] = mpk['h'][i] ** I[i]
    mhi = 1
    for t in range(k):
        mhi *= hi[t]
    hg3r = (mhi * mpk['g3']) ** r
    hashID = mk['g2alpha'] * hg3r
    gr = mpk['g'] ** r
    hr = {}
    for i in range(k+1,mpk['l']) :
        hr[i] = mpk['h'][i] ** r

    return { 'ID':ID, 'k':k },{ 'd0':hashID,'d1':gr, 'dn':hr}

"""
## Encrypt the message using key encapsulation
"""
def encrypt(self,mpk,pk,M):

    if type(M) != str: raise "message not right type!"
    key = group.random(GT)
    s = group.random(ZR)
    print "s==",s
    I = Waters(group, pk['k'], mpk['z']).hash(pk['ID'])
    """
    1. Key Encryption
    """
    A_s = pair(mpk['g1'],mpk['g2']) ** s
    A = A_s * key
    B = mpk['g'] ** s
    h_i = {}
    for i in range(pk['k']):
        h_i[i] = mpk['h'][i] ** I[i]
    mh_i = 1
    for t in range(pk['k']):
        mh_i *= h_i[t]
```

```
        C = (mh_i * mpk['g3']) ** s
        """
        2. Message Encryption using key
        """
        cipher = AuthenticatedCryptoAbstraction(sha1(key))
        c2 = cipher.encrypt(M)

        """
        3. Return : Ciphertext_Key = {A,B,C}, Ciphertext_Message =
            c2
        """
        return {'A':A,'B':B,'C':C, 'c2':c2} #A,B,C : Encrypted key,
            c2 : Encrypted Message

    def decrypt(self,pk,sk,ct):
        """
        1. Key Decryption
        """
        start_time = time.time()
        num = ct['A'] * pair(sk['d1'],ct['C'])
        den = pair(ct['B'],sk['d0'])
        key = num/den
        encM_time = time.time();
        """
        2. Message Decryption using key & Return decrypted message
        """
        start_time2 = time.time()
        cipher = AuthenticatedCryptoAbstraction(sha1(key))
        encM_time2 = time.time();
        return cipher.decrypt(ct['c2'])
```

REFERENCES

[1] "Integrating the healthcare enterprise volume 1 integration profiles," 2007, aCC, HIMSS and RSNA Integrating the Healthcare Enterprise.

[2] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology–EUROCRYPT 2005*. Springer, 2005, pp. 440–456.

[3] D. Garg, L. Jia, and A. Datta, "Policy auditing over incomplete logs: theory, implementation and applications," in *Proc. of CCS*, 2011.

[4] S. E. Oh, J. Y. Chun, L. Jia, D. Garg, C. A. Gunter, and A. Datta, "Privacy-preserving audit for broker-based health information exchange," in *Proceedings of the 4th ACM conference on Data and application security and privacy - CODASPY '14*. New York, New York, USA: ACM Press, Mar. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2557547.2557576 pp. 313–320.

[5] J. R. Vest and L. D. Gamm, "Health information exchange: persistent challenges and new strategies." *Journal of the American Medical Informatics Association*, vol. 17, no. 3, pp. 288–94, Jan. 2010. [Online]. Available: http://jamia.bmj.com/content/17/3/288.full

[6] M. Scholl, K. Stine, K. Lin, and D. Steinberg, *Security Architecture Design Process for Health Information Exchanges (HIEs)*, 2010. [Online]. Available: http://66.77.252.5/content/files/Code 175 NISt Security Architecture Design Process for HIE_Scholl et al.pdf

[7] "Health Information Exchange Approved Policies and Resolutions," 2011, the MARYLAND HEALTH CARE COMMISSION.

[8] D. of Health and H. S. USA, "HIPAA Security Series, 4 Security Standards: Technical Safeguards," 2007. [Online]. Available: http://www.hhs.gov/ocr/privacy/hipaa/administrative/securityrule/techsafeguards.pdf

[9] "IHE IT Infrastructure Technical Framework Supplement 2007-2008 Cross-Enterprise Document Sharing-b (XDS.b)," 2008, aCC, HIMSS and RSNA Integrating the Healthcare Enterprise.

[10] "Health Level Seven International - Homepage." [Online]. Available: https://www.hl7.org/

[11] G. Marshall, "Security Audit and Access Accountability Message XML Data Definitions for Healthcare Applications," no. September, 2004.

[12] "ASTM International." [Online]. Available: http://webstore.ansi.org/

[13] "DICOM Homepage." [Online]. Available: http://dicom.nema.org/

[14] D. Boneh and X. Boyen, "Efficient selective identity-based encryption without random oracles," *Journal of Cryptology*, vol. 24, no. 4, pp. 659–693, 2011.

[15] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *Advances in Cryptology-CRYPTO 2006*. Springer, 2006, pp. 290–307.

[16] J. H. Seo, T. Kobayashi, M. Ohkubo, and K. Suzuki, "Anonymous hierarchical identity-based encryption with constant size ciphertexts," in *Public Key Cryptography–PKC 2009*. Springer, 2009, pp. 215–234.

[17] A. Lewko and B. Waters, "Unbounded hibe and attribute-based encryption," in *Advances in Cryptology–EUROCRYPT 2011*. Springer, 2011, pp. 547–567.

[18] C. Gunter, D. Liebovitz, and B. Malin, "Experience-based access management: A life-cycle framework for identity and access management systems," *Proc. of IEEE Security & Privacy*, vol. 9, no. 5, 2011. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3199376/

[19] D. Fabbri and K. LeFevre, "Explanation-based auditing," *Proc. VLDB Endowment*, vol. 5, no. 1, 2011.

[20] D. Fabbri and K. LeFevre, "Explaining accesses to electronic medical records using diagnosis information," *Journal of the American Medical Informatics Association*, vol. 20, no. 1, 2013.

[21] B. Gregg, H. D'Agostino, and E. Toledo, "Creating an IHE ATNA-based audit repository," *Journal of Digital Imaging*, 2006.

[22] H. Azkia, N. Boulahia, F. Cuppens, and G. Coatrieux, "Reconciling IHE-ATNA profile with a posteriori contextual access and usage control policy in healthcare environment," in *Proc. of IAS*, 2010.

[23] D. Davis, F. Monrose, and M. K. Reiter, "Time-scoped searching of encrypted audit logs," in *Information and Communications Security*. Springer, 2004, pp. 532–545.

[24] B. Schneier and J. Kelsey, "Secure audit logs to support computer forensics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 2, no. 2, pp. 159–176, 1999.

[25] B. R. Waters, D. Balfanz, G. Durfee, and D. K. Smetters, "Building an encrypted and searchable audit log." in *Proc. of NDSS*, 2004.

[26] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *IEEE Symposium on Security and Privacy*, 2013.

[27] S. Barber, X. Boyen, E. Shi, and E. Uzun, "Bitter to better?how to make bitcoin a better currency," in *Financial Cryptography and Data Security*. Springer, 2012, pp. 399–414.

[28] G. O. Karame, E. Androulaki, and S. Capkun, "Double-spending fast payments in bitcoin," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 906–917.

[29] P. Thati and G. Roşu, "Monitoring algorithms for metric temporal logic specifications," *Electronic Notes in Theoretical Computer Science*, vol. 113, pp. 145–162, 2005.

[30] F. Baader, A. Bauer, and M. Lippmann, "Runtime verification using a temporal description logic," in *Proc. of FroCos*, 2009.

[31] G. Roşu and K. Havelund, "Rewriting-based techniques for runtime verification," *Automated Software Engineering*, vol. 12, pp. 151–197, 2005.

[32] D. A. Basin, F. Klaedtke, and S. Müller, "Policy monitoring in first-order temporal logic," in *Proc. of CAV*, 2010.

[33] H. Barringer, A. Goldberg, K. Havelund, and K. Sen, "Rule-based runtime verification," in *Proc. of VMCAI*, 2004.

[34] M. Roger and J. Goubault-Larrecq, "Log auditing through model-checking," in *Proc. of CSF*, 2001.

[35] "Illinois Prescription Monitoring Program," https://www.ilpmp.org/.

[36] "Privacy and Security Framework Requirements and Guidance for the State Health Information Exchange Cooperative Agreement Program," pp. 1–11, 2012, office of the National Coordinator for Health Information Technology.

[37] "Charm: A tool for rapid cryptographic prototyping." [Online]. Available: http://www.charm-crypto.com

[38] "IHE open source — Free software downloads at SourceForge.net." [Online]. Available: http://sourceforge.net/projects/iheos/

[39] K. B. Johnson, K. M. Unertl, Q. Chen, N. M. Lorenzi, H. Nian, J. Bailey, and M. Frisse, "Health information exchange usage in emergency departments and clinics: the who, what, and why." *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 690–7, 2011.

[40] "Illinois General Assembly - Illinois Compiled Statutes." [Online]. Available: http://www.ilga.gov/legislation/ilcs/ilcs.asp

[41] "Health informatics Electronic health record communication Part 4:Security," 2009, international Organization for Standardization.

[42] "Trust Framework for Health Information Exchange," 2013, u.S. Department of Health and Human Services.