

PRIVACY PRESERVING HIPAA-COMPLIANT ACCESS CONTROL MODEL FOR WEB SERVICES

Tariq Alshugran

Under the Supervision of Dr. Julius Dichter

DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE
AND ENGINEERING

THE SCHOOL OF ENGINEERING

UNIVERSITY OF BRIDGEPORT

CONNECTICUT

December, 2018

PRIVACY PRESERVING HIPAA-COMPLIANT ACCESS

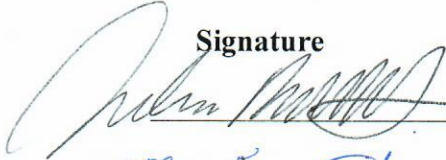
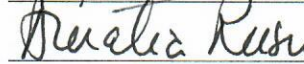
CONTROL MODEL FOR WEB SERVICES

Tariq Alshugran

Under the Supervision of Dr. Julius Dichter

Approvals

Committee Members

Name	Signature	Date
Dr. Julius Dichter		1.18.19
Dr. Miad Faezipour		01,18,2019
Dr. Navarun Gupta		1/18/2019
Dr. Prabir Patra		01/24/19
Dr. Amalia Rusu		1/20/19

Ph.D. Program Coordinator

Dr. Khaled M. Elleithy		2/7/2019
------------------------	--	----------

Chairman, Computer Science and Engineering Department

Dr. Ausif Mahmood		2/7/2019
-------------------	--	----------

Dean, School of Engineering

Dr. Tarek M. Sobh		2/7/2019
-------------------	--	----------

PRIVACY PRESERVING HIPAA-COMPLIANT ACCESS CONTROL MODEL FOR WEB SERVICES

© Copyright by Tariq Alshugran 2018

PRIVACY PRESERVING HIPAA-COMPLIANT ACCESS CONTROL MODEL FOR WEB SERVICES

ABSTRACT

Software applications are developed to help companies and organizations process and manage data that support their daily operations. However, this data might contain sensitive clients' information that should be protected to ensure the clients' privacy. Besides losing the clients' trust, neglecting to ensure the clients' data privacy may also be unlawful and inflict serious legal and financial consequences. Lately, different laws and regulations related to data privacy have been enacted specially in vital sectors such as health care, finance, and accounting. Those regulations dictate how clients' data should be disclosed and transmitted within the organization as well as with external partners. The privacy rules in these laws and regulations presented a challenge for software engineers who design and implement the software applications used in processing the clients' private data. The difficulty is linked to the complexity and length of the letter of the law and how to guarantee that the software application is maintaining the clients' data privacy in compliance with the law.

Some healthcare organization are trying to perform their own interpretation of the law privacy rules by creating custom systems. However, the problems with such

approach is that the margin of error while interpreting the letter of the law is high specially with separate efforts carried out by individual companies. According to a survey carried out to check the Healthcare Insurance Portability and Accountability Act (HIPAA) requirements interpretation created for medical and healthcare related applications, none of the frameworks were well developed to capture the relationships specified in the law. To solve this problem, a standard framework is required that will analyze the regulatory text and provide a method to extract the relevant component that can be used during software roles engineering and development. The extracted components will include all the possible arrangements of roles, purposes, permissions, temporal factors, and any carried out obligations.

In this work we propose a framework to analyze, extract, model, and enforce the privacy requirements from HIPAA regulatory text. The framework goal is to translate the law privacy rules text into more manageable components in the form of entities, roles, purposes, and obligations. Those components together can be used as building blocks to create formal privacy policies. The process concentrates on two main components; entities and their roles, and data access context. To accomplish the first part, the framework will parse the privacy sections of the regulatory text to mine all the subjects, and then categorize those subjects into roles based on their characterization in the law. To acquire the access context, the process will extract all the purposes, temporal clauses and any carried out obligations and classify them based on their permissibility.

ACKNOWLEDGEMENTS

First and foremost, my thanks are wholly devoted to God, the almighty, for blessing me and giving me the strength all the way to complete this work successfully.

I owe a great debt of gratitude to my family for their support and encouragement.

I would like to express my deep gratitude to my adviser, Dr. Julius Dichter, for guiding and supporting me throughout this entire research work and the dissertation process.

Special thanks to my dissertation committee members for their valuable comments and feedback that enriched the quality and the content of this research.

TABLE OF CONTENTS

ABSTRACT.....	IV
ACKNOWLEDGEMENTS.....	VI
TABLE OF CONTENTS.....	VII
LIST OF TABLES.....	X
LIST OF FIGURES.....	XI
NOMENCLATURE.....	XII
ACRONYMS.....	XIII
CHAPTER 1: INTRODUCTION.....	1
1.1 Overview.....	1
1.2 Research Problem and Scope.....	2
1.2.1 HIPAA Modeling Complexity.....	3
1.2.2 Context and Model parameters.....	3
1.2.3 System Implementation.....	3
1.3 Motivation behind the Research.....	3
1.4 Potential Contributions of the Proposed Research.....	4
1.5 Research Organization.....	5
CHAPTER 2: LITERATURE SURVEY.....	7
2.1 HIPAA Overview.....	7
2.2 HIPAA Formalization.....	8
2.2.1 HIPAA Formalization Complexity.....	9
2.3 Privacy Access Control Model.....	11

2.3.1 Privacy vs. Security	11
2.3.2 Access Control Models Overview	12
2.3.3 Simple Access Control Model	12
2.3.4 Role-Based Access Control Model	14
2.4 Law Formalization	15
2.4.1 Privacy Rules Extraction Methods.....	15
2.4.2 Privacy Policy Expression Languages	17
2.4.3 Privacy Policy Logics	18
2.5 Web Services and Access Control	19
CHAPTER 3: RESEARCH AND SYSTEM ARCHITECTURE.....	21
3.1 Research Plan.....	21
3.2 System Architecture.....	21
3.3 Extracting and Modeling HIPAA Privacy Requirements	22
3.3.1 Access Request Context.....	22
3.3.2 Context-Based Access Control (CBAC) Model	24
3.3.3 The Extraction Process	25
3.3.4 The Modeling process	26
3.4 Formally Expressing HIPAA Privacy Policies	28
3.4.1 Expression Languages	28
3.4.2 Extending XACML to Support CBAC	30
3.5 Privacy Access Control Decision Logic	32
3.6 Logical Specifications of HIPAA Rules	34
3.6.1 Agents, Attributes, and Knowledge States.....	34
3.6.2 Agent Roles.....	35
3.6.3 Request Purpose.....	36
3.6.4 Response and Disclosure action.....	37

3.6.5 Temporal conditions	38
3.6.6 Norms of Disclosure	38
3.6.7 Formally Expressing HIPAA rules using HPRLSL.....	39
CHAPTER 4: IMPLEMENTATION AND TEST PLAN.....	40
4.1 Implementation	40
4.2 Test Plan	41
4.2.1 Policy Formalization Test.....	41
4.2.2 Policy Generation Test.....	42
4.2.3 Policy Reasoning and Integrity Test.....	42
CHAPTER 5: CHAPTER 5: RESULTS.....	47
5.1 Test Results.....	47
5.1.1 Policy Formalization Test Results	47
5.1.2 Policy Reasoning and Integrity Test Results	48
CHAPTER 6: CHAPTER 6: CONCLUSION AND FUTURE WORK.....	50
6.1 Conclusion	50
6.2 Future Work.....	52
REFERENCES	53

LIST OF TABLES

Table 3.1	Elements classification of HIPAA rule §164.528 (a)(2)(i)	28
Table 3.2	Comparison of standard privacy policy expression languages	30
Table 4.1	Effect Size Values	46
Table 5.1	Sample results of extraction and modeling test for rule 164.512	47
Table 5.2	Sample results of Access Request Simulation showing response time	48

LIST OF FIGURES

Figure 2.1	Simple Access Control Model	13
Figure 2.2	Role-Based Access Control (RBAC) Model	15
Figure 2.3	Web service conceptual and implementation models mapping	19
Figure 3.1	Access model abstract design	22
Figure 3.2	Context-Based Access Control (CBAC) Model	23
Figure 3.3	An outline of HIPAA rule §164.528 (a)(2)(i) components	27
Figure 3.4	Sample XACML extended request: Dr. Doe requesting a medical record	31
Figure 3.5	Sample XACML extended response	32
Figure 3.6	Access Control Decision Logic	33
Figure 3.7	XACML decision engine	34
Figure 4.1	The proposed access model architecture	40

NOMENCLATURE

C	Set of all possible context for any given data access request
K	Set of knowledge states
P	Agents (set of population)
R	Set of all possible roles for any given agent in the system
S	Set of access responses
T	Set of attributes that represent data about an agent
U	Set of request purposes
φ^+	Positive norms
φ^-	Negative norms
\mathbb{P}	Privacy Policy

ACRONYMS

CBAC	Context-Based Access Control
DAC	Discretionary Access Control
EPAL	Enterprise Privacy Authorization Language
GORE	Goal-Oriented Requirements Engineering
HIPAA	Health Insurance Portability and Accountability Act
HLT	Hohfeld Legal Taxonomy
ICAO	International Civil Aviation Organization
MAC	Mandatory Access Control
P3P	Platform for Privacy Preferences Project
PAP	Policy Admin Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PHI	Protected Health Information
PIP	Policy Information Point
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

CHAPTER 1: INTRODUCTION

1.1 Overview

On August 1996, the Health Insurance Portability and Accountability Act (HIPAA) passed by the U.S. Congress, defining a set of rules to protect patients' health information and privacy. Those privacy rules in HIPAA apply to all individually identifiable information [1].

HIPAA introduced a challenge for healthcare information systems that maintain patients' information; as every operation that uses or discloses healthcare information – defined by HIPAA as Protected Health Information (PHI) – needs to comply with HIPAA privacy rules. Violation of these rules may result in significant penalties that may include substantial fines.

The lack of a common system that can be implemented by any company's information system led to the need for a platform-independent system that implements an access control model to preserve the privacy of PHI in any covered entity system. Using web services can fulfill this need as a web service is a software system designed to support interoperable application-to-application interaction over a network [2].

Recently, healthcare applications are moving toward adopting web services technologies due to its interoperability, which helps solving the issue of incompatible

systems and languages to share information. Also, web services have rich properties such as service compositions and reusable components that makes it more desirable as a communication platform [3–4].

Web services utilize a set of eXtensible Markup Language (XML) standards like Simple Object Access Protocol (SOAP), Universal Description, Discovery and Integration (UDDI), and Web Services Description Language (WSDL), which gives web services a standardize implementation sense. This helps in the creation of a standard privacy access control model that implements HIPAA privacy rules. The access control model is embedded in a web service, which then can be applied to any existing software system.

Such a model meets the set of requirements to comply with HIPAA rules as well as any standard privacy access control model.

1.2 Research Problem and Scope

The objective of this research is to create a HIPAA-compliant Access control model that uses HIPAA privacy rules as criteria source in the access decision process. Taking into consideration that such a system should be able to handle heterogonous environments and be implemented as a gateway to access existing covered entities' software systems.

The problems we are trying to solve can be divided into three categories: HIPAA modeling and formalizations complexity, designing a HIPAA-compliant access control

that takes into consideration multiple context factors, and finally the implementation of the model as an add-on on top of existing software systems.

1.2.1 HIPAA Modeling Complexity

Several issues complicate the process of extracting the context request from HIPAA or any regulatory text in general. The most notable issues that affect HIPAA modeling are: the complexity of legal document language, the legal document structure, and cross-references and exceptions.

1.2.2 Context and Model parameters

To design the access control model, we need to define the context and factor it as parameters to the system. The parameters should enumerate all the possible combination of roles, permissions, purposes, obligations, and any temporal factors.

1.2.3 System Implementation

The system design should have a minimal overhead when implemented to an existing system. Also, it should consider the fact that those systems are heterogeneous and should be able to communicate with each other with additional translation layer or extra implementation. This requires the system to be interoperable and to use a standard policy language.

1.3 Motivation behind the Research

The lack of a common system that can be implemented by any company's information system led to the need for a platform-independent system that implements an

access control model to preserve the privacy of all Protected Health Information (PHI) in any covered entity system.

The key motive behind this research was to review the current implementations and design a model that can solve the problems highlighted in Section 1.2 by providing an access control model based on the rules and logic extracted directly from the regulatory text letter of the law.

1.4 Potential Contributions of the Proposed Research

The main contribution of this research is to create a context-based privacy access control model embedded within a platform-independent service-oriented system utilizing web services. However, in order to achieve this goal, we need to solve the research problems. The outcome of each problem is considered a contribution by itself. Similar to the research problems, those secondary contributions can also be categorized into three contributions:

1. Create a robust process that can model, extract data, and formalize HIPAA privacy rules.
2. Design privacy policy, access request, and access response format that uses the request context and implement a mechanism for enforcing the law using a context-based decision engine.
3. Embedding the policies and the decision engine in the web service layers.

1.5 Research Organization

This research consists of six chapters followed by bibliography. The work is organized as follows:

Introduction: This chapter provides an overview of the research problem and scope. Also, it highlights the motivation behind the research and the potential contributions.

Literature Survey: Chapter 2 provides background knowledge of the research topic. It includes the current access models, formalization methods, and using web services in healthcare applications. It also reviews the related work in those fields and how that can be used in this research.

Research Plan: this chapter discusses the process to formalize HIPAA rules into clear policies. Then it shows the features required in an access control model to express the formalized rules, and how to represent the policies in a service-oriented environment using web services as an access point.

Implementation and Test Plan: Chapter 4 shows initial implementation of the web service and how the privacy policies are embedded within the web services. It also explains the test plan that was carried on the system in order to validate the system design and integrity.

Results: Using the test plan from Chapter 4, this chapter demonstrates the test results and then provides a brief analysis of those results.

Conclusions: The last chapter is a summary of the research outcome. It concludes the model design and limitations. Also, it paves the road for the possible future work and enhancements that can be applied to the proposed model.

CHAPTER 2: LITERATURE SURVEY

2.1 HIPAA Overview

Initially sponsored by Ted Kennedy and Nancy Kassebaum, the Health Insurance Portability and Accountability Act (HIPAA) was passed by the U.S. congress and signed by President Bill Clinton in 1996 [5]. The U.S. Department of Health and Human Services (HHS) classifies HIPAA into five top-level titles covering different healthcare-related topics [6]. Those titles are classified based on rules main topic such as group provisioning and revenue, accessibility and portability, privacy, security, and taxation. Each title is then divided into subtitles that focus on more specific idea in the same parent topic [7].

In this work, our concentration will be on the privacy laws specified in Title 2. Preventing health care fraud and abuse (administrative simplification), which includes: transactions and code sets; identifiers; privacy; and security. The concentration of this work will be on the privacy section of the law, more specifically, subpart E of Part 164 covering §164.502 to §164.528 [1, 8]. These rules identify the security and privacy requirements applicable to any Protected Health Information (PHI) within Covered Entities. According to HHS, HIPAA covered entities are defined as health plan providers, healthcare providers, health care clearinghouses, or any other entity that maintains and

discloses PHI in any form or media regulated by HIPAA, where PHI refers to individually identifiable health information that identifies an individual and relates to past, present, or future physical and mental health disorders, illnesses, and conditions.

HIPAA privacy rules regulate, defines, and limits the purpose of PHI usage within the covered entity scope, as well as the purpose of PHI disclosure between covered entities. Based on the purpose, PHI usage or disclosure may require the PHI owner preauthorization or add an obligation to notify the owner of the action.

2.2 HIPAA Formalization

In the U.S., numerous federal laws and regulations were legislated to guarantee individuals' right to be able to access and port their private information stored and managed by service providers while protecting that information from unauthorized access. For example, the Gramm-Leach-Bliley Act (GLBA) of 1999 [9] is designed to protect individuals' financial information from being breached without proper authorization. The Health Insurance Portability and Accountability Act (HIPAA) of 1996 is another example, section 164 of HIPAA is intended explicitly to protect patients' healthcare information and medical records from unauthorized disclosure. According to HIPAA, any healthcare related information that can identify an individual and can be stored or transmitted via any media format is defined as Protected Health Information (PHI). HIPAA privacy rules control the storage, transmission, and disclosure of all Protected Health Information. Usually the PHI is collected and maintained by healthcare insurance plan, healthcare provider, healthcare clearinghouse or any other similar

organization identified by HIPAA as Covered Entity. Comparable privacy rules can also be found in other federal regulations as well.

The legal language of HIPAA is too dense and complex to be used as a guide by computer programmers and information technology professionals responsible for developing and managing software systems for healthcare companies required to comply with the law. Hence, the need to formalize the laws into a logical format that can be applied easily to automated systems and can be tested for compliance.

Recent researches show a variety of proposed solutions to the problem of formalizing privacy laws expressed in a number of logics: [10–12], and different languages: [13–15]. Those logics and languages were proposed to permit implementing the privacy laws in a simple way that can be tested for compliance easily and be able to specify if a request for data transmission or access is allowed.

Regardless of the wide variety of privacy languages and logics proposed above, most of them are either too complex to implement, or just covering a subset of the law as a Proof of Concept (POC).

2.2.1 HIPAA Formalization Complexity

The U.S. federal regulations documents are written in a complex format and technical language known as legalese. Legalese or legal English uses different vocabulary and syntax than that used in ordinary English. The complex format and legal terminologies makes legal documents hard to read and interpret [16].

Apart from the document language, the structured format makes the text prone to misinterpretations and other ambiguities like cross references and exceptions. The document is usually structured into parts (e.g. Part 164 of HIPAA). Each part is then divided into subparts, which is additionally divided into sections (e.g. Section §164.528). Some sections are also divided into subparagraphs with multiple points in the same sentence. This create some inconsistency as some privacy rules are spanning multiple points, subparagraphs, paragraphs, or event sections. For example, the subparagraph §164.528(a)(2)(ii) contains three points (A), (B), and (C) in the same sentence: “the covered entity must:(A)...;(B)...; and (C)...”, where each one of these points defines a different obligation that should be carried out by the covered entity [17].

Exceptions and Cross-references to other sections add more complexity to the modeling process as they require additional processing efforts. References usually entail priority between paragraphs and add more clarity to the privacy requirements. However, sometimes references might introduce ambiguities due to the possibility of nested and multilevel referencing in the form of cross-references [18–19]. Cross-references occur when a section or a paragraph in the law is referencing another section/paragraph that has a reference to another rule. For example, the subparagraph §164.528 (a)(2)(i) describes individuals’ suspension of rights obligation. This right is also addressed in a different paragraph as highlighted by the phrase “as provided in §164.512(d)” at the end of the paragraph establishing a reference. Nevertheless, in the subsequent paragraph §164.528(a)(2)(ii), the phrase “pursuant to paragraph (i)” is a reference to the preceding paragraph. Hence, to model the right indicated in §164.528(a)(2)(ii), we need to refer to

§164.512(d) creating an indirect relation between the two paragraphs. On the other hand, exceptions are rules that contradict or negate other rules by changing the permissibility right or by adding more conditions or obligation. So, if the first rule grants a right to access a PHI, the exception would either add more conditions to clarify that right or grant permission and vice versa. For instance, §164.512(c)(1) in HIPAA grants the right to disclose a PHI if the information is about adult victims of abuse, neglect or domestic violence, however, §164.512(c)(1)(ii) presents an exception to this right by adding victims agreement as a condition for such a disclosure.

2.3 Privacy Access Control Model

2.3.1 Privacy vs. Security

Privacy and security are two different concepts; however, it's not easy to distinguish the domain of each concept as the literature definitions are not consistent, and sometimes the domains overlap. Nevertheless, the need to have a clear definition of the privacy concept is required as the scope of this work involves only the privacy section of the access control model that represents HIPAA privacy rules.

According to the International Medical Informatics Association (IMIA), privacy is defined as “[having] control over the collection, storage, access, use, communication, manipulation and disposition of data” [19]. Other definitions of privacy involve the user consent or knowledge of data usage and storage [20–21].

In this work, we refer to security as the act of protecting information by providing mechanisms for authentication, data encryption, and other related activities, while refer to

privacy as the act of protecting information confidentiality, integrity, and availability [22]. Where confidentiality deals with PHI disclosure, integrity protect PHI against unauthorized modification and requires each PHI to be accurate and complete, and finally availability prevents withholding of PHI from authorized users when the purpose is justified by the law.

2.3.2 Access Control Models Overview

In the context of information security, Access control is used in the process of determining the permissibility of any access request to perform an operation on a specific data object made by a particular authenticated data user [20].

To describe an access control models, at least the following three terms should be used as a basis for the model: subject, object, and operations on object. Subject refers to a system-authenticated data user; it can be another system, a person, or a process. The second term is object, which is the protected data itself. Finally, operations describe any action that can be performed by the subject on the object. The permissibility to perform those operations is managed by Access Rights. Access rights are a set of rules that manages the permissions granted to users.

2.3.3 Simple Access Control Model

Simple access control model includes only the basic terms as shown in Figure 2.1 below. This model is used widely in personal computing due to implementation simplicity and effectiveness for basic requirements. The simple access control model is generally expressed in the format:

ALLOW [Subject]
TO PERFORM [Operation] ON [Object]

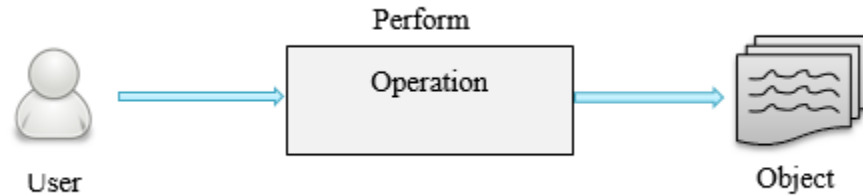


Figure 2.1: Simple Access Control Model

The simple model is divided based on the implementation method into two categories: Discretionary Access Control (DAC) and Mandatory Access Control (MAC).

Discretionary Access Control relies on direct mapping between objects and subjects. The object owner is responsible for specifying the list of subjects that will have access to the object. Permission to access an object is left to the discretion of the object owner or any subject with a full permission on that object. Most modern personal operating systems like Windows and some Linux distributions uses DAC-based models for file access control [23].

Mandatory Access Control on the other hands relies on predefined ordered privacy levels. Subjects and objects are then mapped to a specific security level. In MAC, a subject can access all objects with the same or lower privacy levels. Users including the object owner do not have the permission to assign the privacy levels of objects. Instead, a system administrator is responsible for assigning the privacy level for subjects and objects based on the organization requirements. For example, if the privacy levels where

Public, Guarded, Confidential, and Highly Confidential, where Public is the lowest privacy level and Highly Confidential is the highest privacy level. Then for a user to access a confidential document, the user must have a confidential or highly confidential privacy level. MAC is usually preferred when confidentiality is the main concern of the access control [23].

2.3.4 Role-Based Access Control Model

The simple access control model described in Subsection 2.3.3 is used as base for almost every access control model. However, other models evolved from that model to solve a specific issue that the traditional model was not able to handle properly. For example, the Role-Based Access Control model (RBAC), is an extension of the traditional model with the same three basic terms, but the subject designation is different as shown in Figure 2.2. While in the traditional model the subject is tied to the data user, RBAC defines the subject as the user role [24]. The change was made to solve the issue where users' permissions change based on their roles in the system, and hence, the access control should check the user role instead of the user identifier. A typical RBAC access right rule is generally expressed as:

ALLOW [*Role*]
TO PERFORM [*Operation*] ON [*Object*]

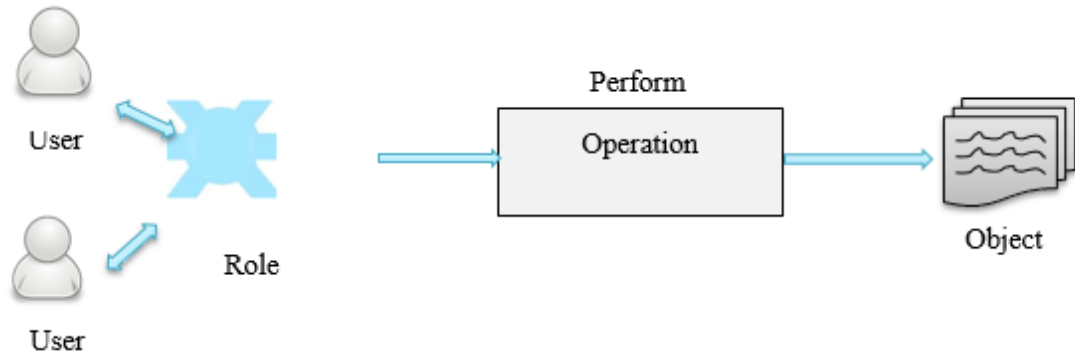


Figure 2.2: Role-Based Access Control (RBAC) Model

2.4 Law Formalization

The U.S. federal regulations documents follow a formal structured format. The legal text is dense and written in a technical language known as legalese. The problem with legalese is that it's too complex, dense, and opaque. The complex format and legal jargons prevent the document from being used as a guide by software engineers and technology professionals during the software development lifecycle. This complexity leads to the need to formalize the laws into a more computer-friendly format that can be easily applied to existing or new software systems and to simplify software law compliance tasks. Apart from the document language, the structured format allows for other ambiguities like cross references and exceptions.

2.4.1 Privacy Rules Extraction Methods

Several researches have already analyzed laws and regulations to extract rights and obligations from the text of the law. Common approaches used natural language patterns to model the law and extract the context from the legal text [15]. The process is

based on identifying patterns in any legal document, then formulate that pattern as a template. The template is used in a parser that analyzes the document to extract the required information.

In addition to natural language patterns, Hohfeld legal taxonomy can be used to elicit security and privacy requirements from the legal text as illustrated in the work of Islam et al. [25]. Islam proposed a method that uses a parser to systematically analyze the legal text and extract the core elements and dependencies. Elements such verbs, objects, goals, conditions, and scopes are used to identify dependencies and elicit the system requirements based on the legal concepts [25–26].

Siena et al. work [27] also used Hohfeld legal taxonomy for security and privacy requirements extraction from the text of the law. Their work is based on the identifying the object, subject, and nature of the right from the legal text. The approach tries to fill the gap between intentional compliance with the law and how the regulatory knowledge is represented in the system.

Jorshari et al. work [28] extended Islam [25] work by eliciting the security requirements from the law. The work enhanced the extraction process by adding a process that ensures resolving dependencies and elements extraction. The model also uses agent-oriented approach and extends Secure Tropos methodology [29] for regulatory text analysis purposes.

The Goal-Oriented Requirements Engineering (GORE) was proposed by Dorimont et al. [30] to model regulations and form goals. Those goals are then refined in

an incremental manner until all the related actors and task are discovered and added. The approach was applied in the context of the SAFEE project to model the International Civil Aviation Organization (ICAO) security regulations for civil aviation.

May et al. [31] “Privacy API” introduced an access control model and extracted the privacy requirements from the legal text. The modeling and extraction approach followed a procedure of converting laws into commands to keep the extraction tightly-coupled with the original text of the law. Each paragraph is translated into one or multiple commands depending on how many actions they allow or deny.

2.4.2 Privacy Policy Expression Languages

To express the privacy policies policy in web services, the formalized HIPAA rules should be formatted as XML. This restricts the specification language choice to XML-based privacy policy expression languages, specifically P3P, XACML, EPAL, and custom XML. In this section we will introduce each language reviewing the possibility of using it for privacy policies representation [32].

Enterprise Privacy Authorization Language (EPAL): A proprietary privacy policy formal language introduced by IBM. EPAL focuses on the privacy authorization and permission independently from the deployment details like data structure and authentication method [33].

The Platform for Privacy Preferences Project (P3P): the P3P was developed by a working group at W3C. The purpose of the language was express web sites privacy policies and practice [14].

eXtensible Access Control Markup Language (XACML): is an OASIS standard markup language used as policy expression language, and also as an access control decision request/response language [32].

Custom XML: The eXtensible Markup Language (XML) is a human/machine readable language that can be extended to support any data structure based on a unified format. Hence, for the purpose of this research we can create a custom law-compliant privacy policy expression XML format to support our work and specify the required features. However, such a format will lack two core features present in the above languages: standardization and general acceptance.

2.4.3 Privacy Policy Logics

DeYoung et al [7] proposed a complete HIPAA and GLBA formalization of all the rules and introduced a new logic called “PrivacyLFP” based on the least fixed-point logic. But the logic is not designed for web services and can’t be transformed easily to XML. However, most of the points outlined in the formalization process will be very useful in this work for developing our actual model based on the conceptual model.

Lam et al [12] work presented the pLogic for HIPAA formalization by identifying a specific fragment of stratified Datalog with limited use of negation. Their work only used the parts §164.502, §164.506, and §164.508 of HIPAA and the main concentration was on medical data messaging with no carried-out obligations.

May et al [31] work introduced a new framework called “Privacy API” with HIPAA formalization method using HRU Access control model. However, the model is incomplete as it only formalizes on HIPAA rule (§164.506).

2.5 Web Services and Access Control

To implement the privacy access model to web services, we need to examine web services architecture to find the best place to enforce the access model. The web service conceptual model consists of six layers: Discovery layer, Message layer, Service layer, Security layer, Policy layer, and Privacy layer. Each one of those layers is supported by a corresponding XML standard in the implementation model as shown in Figure 2.3.

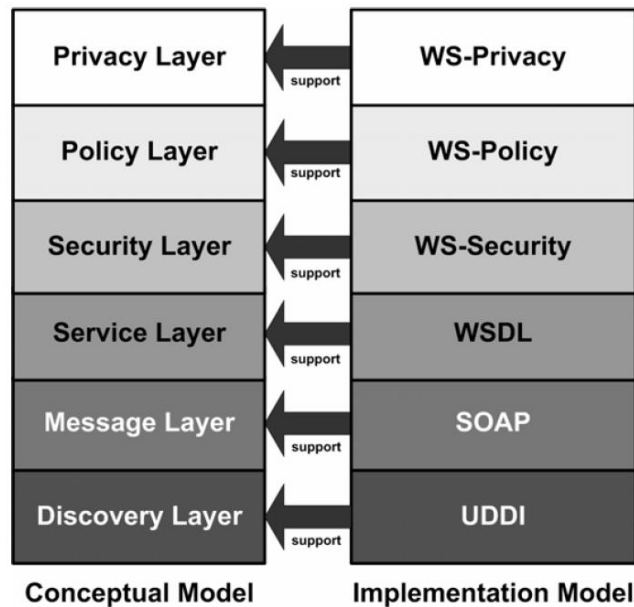


Figure 2.3: Web service conceptual and implementation models mapping

The discovery layer is used in web service finding with the means of UDDI, where the message layer is used for binding web services through SOAP implementation, and the service layer is used for publishing using WSDL, and the security layer

implements the OASIS approved WS-Security standards to safeguard the web service. This leaves only two layers as a prospect layers to implement a privacy access model: the policy layer and the privacy layer. However, both of those layers lack an approved standard implementation language. For the policy layer, the currently proposed WS-Policy does not meet the interoperable and non-propitiatory framework requirements as it was not published by a standard organization and is not adopted by any significant group of vendors [34] and there is no complete approved implementation of WS-Security for the security layer [35].

CHAPTER 3: RESEARCH AND SYSTEM ARCHITECTURE

3.1 Research Plan

The research went through multiple iterations until it reaches the current state. It started as a conceptual model and evolved into a more sophisticated system implementation. The first step was to create an abstract theoretical model for a generic HIPAA access control that utilizes web services as a gateway to control access to any covered entity existing resources in compliance with HIPAA. Then we identified the model components where each component represents a milestone in the project roadmap. The main components were treated as individual research items; it has its own literature survey, implementation, and test plan. Then we combined the components together again to form the final model as one entity. Those components are identified based on a function-oriented engineering process and are classified as: privacy rules extraction and formalization, privacy policies formal expression, access model logic, and the web service design and implementation.

3.2 System Architecture

The system introduces the idea of an access model that can be implemented as a privacy-wall between users' request and any covered entity existing system as shown in Figure 3.1. When any resource access request is created, it must go through the privacy-wall before reaching the resource. The same logic is applied regardless if the request is for a PHI internal usage or external disclosure [36].

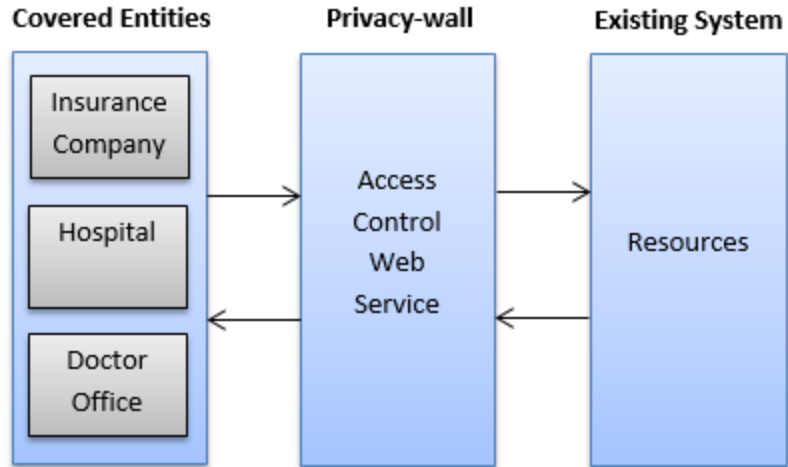


Figure 3.1: Access model abstract design

3.3 Extracting and Modeling HIPAA Privacy Requirements

3.3.1 Access Request Context

Privacy Access Control is an access control where the access right rules are expressed based on set of privacy rules. It is a typical access control; however, it relies and aims to protect the object privacy. In this type of access control, the format of access right rules is different, as it depends on the user attributes, roles, request purpose, and other conditions as shown in Figure 3.1. It also carries out obligations. Powers et al. [10] suggested a new format to express privacy access right rules. We added modified the access right rule to work [37] with our model and the new syntax is represented as:

```

ALLOW [Active Role]
TO PERFORM [Operation]
ON [Data Type]
  
```

RELATED TO [*Data Owner Type*]

FOR [*Purpose*]

PROVIDED [*Condition*]

CARRY OUT [*Obligation*]

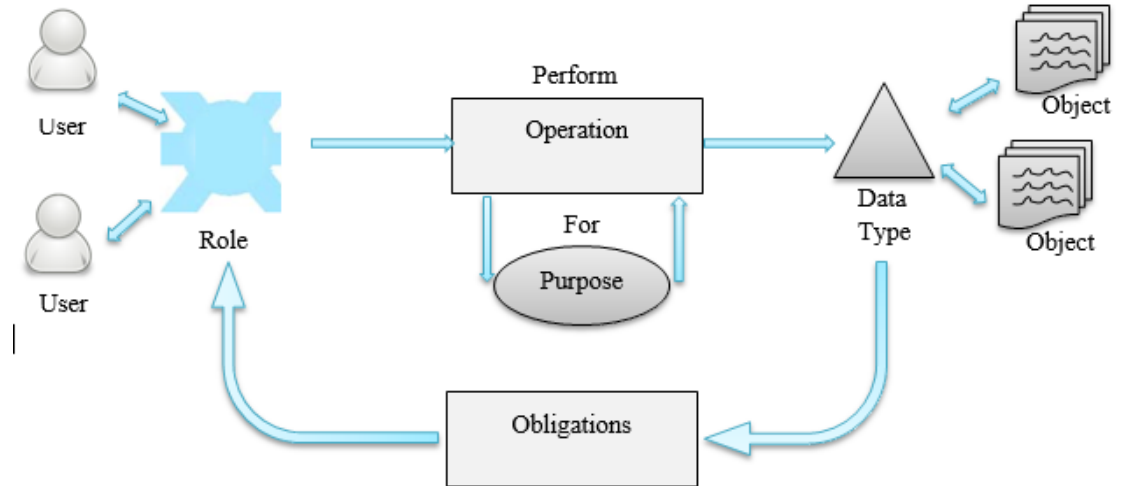


Figure 3.2: Context-Based Access Control (CBAC) Model

The privacy access rule consists of the following components:

- **Subject**: is the data user, however, it's not tied to a specific user, but rather, to a user type or role similar to RBAC roles.
- **Operations**: any set of actions that can be invoked on an object, similar to the traditional model.
- **Object type**: in this type of rules is not tied directly to a specific object, but rather to a specific type of object, it can be defined at high-level (i.e. Treatment related files), or at a lower-level (i.e. Medical chart files, CT scan images ... etc.).

- Purpose: This represents how the data user will use the object, or the reason behind the request. A purpose can be something like treatment, accounting, legal case...etc. [38].
- Condition: Any additional rules that needs to be qualified.
- Obligations: Any additional steps that the system must take after a certain access is allowed, this might include logging, object deletion, archiving...etc.

3.3.2 Context-Based Access Control (CBAC) Model

The privacy access control model described in the previous section is an extension to the Role-Based Access Control (RBAC) with the following distinctions:

1. Instead of relying solely on the requester role, the entire Access Request Context (ARC) is used. This makes the access control Context-Based Access Control Model (CBAC).
2. While RBAC is used mainly for security access control, CBAC is used usually for privacy access control.
3. In RBAC, no obligations are carried out by default after granting access. On the other hand, CBAC usually carries out obligations.

To allow access to a specific PHI, the access request context will be analyzed and should meet HIPAA privacy requirements. However, to be able to check the permissibility of a request against HIPAA, the requirements should be modeled to extract

the access criteria. The modeling process will extract all the possible combination of roles, permissions, purposes, obligations, and any temporal factors [39–40].

3.3.3 The Extraction Process

The process is broken into two activities: Model and Analyze HIPAA and Identify and Extract the Privacy Requirements. Each activity consists of several steps. Figure 1 illustrates the extraction process with the two main activities and shows each activity inputs and outputs.

The purpose of this activity is to overcome the difficulties and issues that complicate HIPAA modeling as explained in section II.B. Raw HIPAA rules are used as an input to the activity, and then the following steps are performed:

Step 1: Identify scopes and definitions: the first step is to define the extraction scope. In this work the focus will be only on the rules related to privacy preserving and PHI disclosure. After analyzing HIPAA rules and eliminating all the abstract, non-operational requirements and regulations, the extraction will take care of the privacy requirements from §164.502, §164.506, §164.510, §164.512, §164.514, and §164.524 of HIPAA.

Step 2: Identify and replace rules dependencies, cross-references, and exceptions: the purpose of this step is to formalize the subset of rules selected in step 1 and to add more clarity to the legal text. This is achieved by replacing cross-references, self-references, and dependencies with the exact definition from the reference or by using a similar meaning to improve the rule text less readability. Alternatively, if the reference is

identified as a condition it might stay in-place to be processed in a later stage of the modeling process [41].

The output of activity one is a reference-free subset of HIPAA rules that is focused only on privacy preserving. This output will then be used as one of the inputs of the next activity.

3.3.4 The Modeling process

Once all the rules are elaborated and modeled in a clear way in activity one, any additional organization or business-related privacy rules are then analyzed. Combined with the rules from activity one, all the rules are used as an input to activity two.

The purpose of this activity is to perform the extraction process. Like the previous activity, this activity consists of multiple steps where the final step is the actual context elements extraction. Before that, any business-related privacy rules are analyzed. Then combined with the subset of HIPAA rules, all ambiguities are eliminated. Next, the text is parsed to highlight phrases that might constitute a possible element. Figure 3.3 outlines the components of the rule §164.528 (a)(2)(i) of HIPAA.

Step 1: Analyzing business-related privacy rules: In addition to HIPAA rules, most covered entities have their own privacy policies and rules that their system users must agree upon in order to use the covered entity software application. During this step the covered entity privacy policy can be analyzed and prepared for the modeling process. However, this step is optional as such a privacy policy can be formalized separately and then the two privacy policies can be combined using a combining algorithm [42].

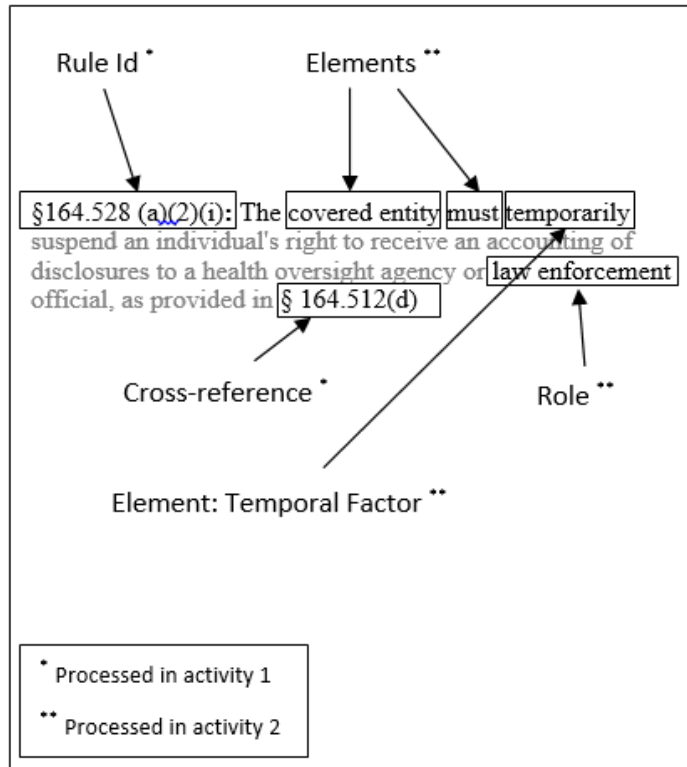


Figure 3.3: An outline of HIPAA rule §164.528 (a)(2)(i) components

Step 2: Clarifying the ambiguities: The legal text of the law might contain domain-specific wording that will require a law professional to elaborate the definition of the text. Also, some words might have multiple meanings which will cause an ambiguity. At this stage, all the ambiguous words and domain specific words will be elaborated [43].

Step 3: Text parsing and elements extraction: in this step all the words and phrases in the text of the rule are tested for possibility of being a context element.

Step 4: Context elements classification: Various methods were proposed for elements classifications [44–47]. However, each method solves a specific problem or introduces a special notation. For example, Hohfeld classification introduces the notion of rights and responsibilities. Hohfeld theory embodies the relationship between actors

based on their responsibilities and rights within the legal text context [17]. However, this work is more focused on the notion of Context-Based Access Control (CBAC) where the main concern is to identify roles, permissions, purposes, obligations, and temporal factors. Hence, a Goal-Driven approach can be applied where the goal is to extract the CBAC components. Table 3.1 shows an example of elements classification based on the data from Figure 2 of §164.528 (a)(2)(i).

Table 3.1: Elements classification of HIPAA rule §164.528 (a)(2)(i).

Element	Element Classification
R164_528_a_2_i	Rule Id
covered entity	Role
must	Operator
temporarily	Temporal factor
suspend rights to receive an accounting of disclosure	Action
health oversight agency	Role
law enforcement official	Role
(R164_512_d) OR (R164_512_d)	Condition

3.4 Formally Expressing HIPAA Privacy Policies

3.4.1 Expression Languages

In our access control model, we proposed the use of web services as an entry point for all access request. The web service will act as a privacy-wall that filters out all unauthorized access requests. The access control model will be encapsulated inside the web service. The web service policy layer stores the formalized privacy policies, whereas the privacy layer will include the logic to enforce those policies [36].

In order to store the privacy policies in the web service policy layer, the formalized privacy rules extracted from the previously explained process should be expressed as XML formatted policies. This condition restricts the specification language choice to be an XML-based like EPAL, P3P, XACML, and custom XML. In this section we will review those languages and assess their adequacy to express and/or enforce the formalized law privacy rules.

To compare the XML languages, we need to specify comparison criteria. We will rely on the work of Anderson [33]. Anderson outlined 8 criteria to compare policy expression languages. Those criteria are based on the language features and ability to express privacy policies. We extended the criteria to include a standard recognition and the ability to use the language without obtaining a prior agreement between the model users. Also, we will extend the comparison to include P3P and custom XML in addition to the EPAL and XACML. Table 3.2 shows a comparison between the four remaining language.

Based on the comparison, we chose to use XACML due to its standardized nature and extensibility, also as it will allow expressing the privacy policies as well as the access request. While custom XML was very close to XACML, however, it was ruled out as an option due to its inability to express directly enforceable policies, and the fact that it will require prior approval from its users [48].

Table 3.2: Comparison of standard privacy policy expression languages

	XACML	EPAL	P3P	Custom XML
Developed by	OASIS	IBM	W3C	N/A
Standard status recognition	Yes	Yes	Yes	No
Proprietary	No	Yes	No	No
Can express privacy policies	Yes	Yes	Yes	Yes
Can express directly-enforceable policies	Yes	Yes	No	Yes
Can express access request / response	Yes	No	No	Yes
Support constraints on Subject	Yes	Yes	Yes	Yes
Platform independent	Yes	Yes	Yes	Yes
Can be extended	Yes	No	No	Yes

3.4.2 Extending XACML to Support CBAC

Based on the Context-Based Access Control Model (CBAC) proposed previously, XACML access request should be extended to contain all the extracted context attributes. The XML fragment shown in Figure 3 include a sample XACML request [36].

As illustrated in the access request shown in Figure 3.4, the user id (Dr. Doe) and role (Doctor) are included in the access request nested under the subject attributes. The action consists of two parts, the action type (read) and the request purpose (Treatment). While finally the resource specifies a private document (patient medical record). The

extended XACML request might also include additional conditions and temporal factors as well.

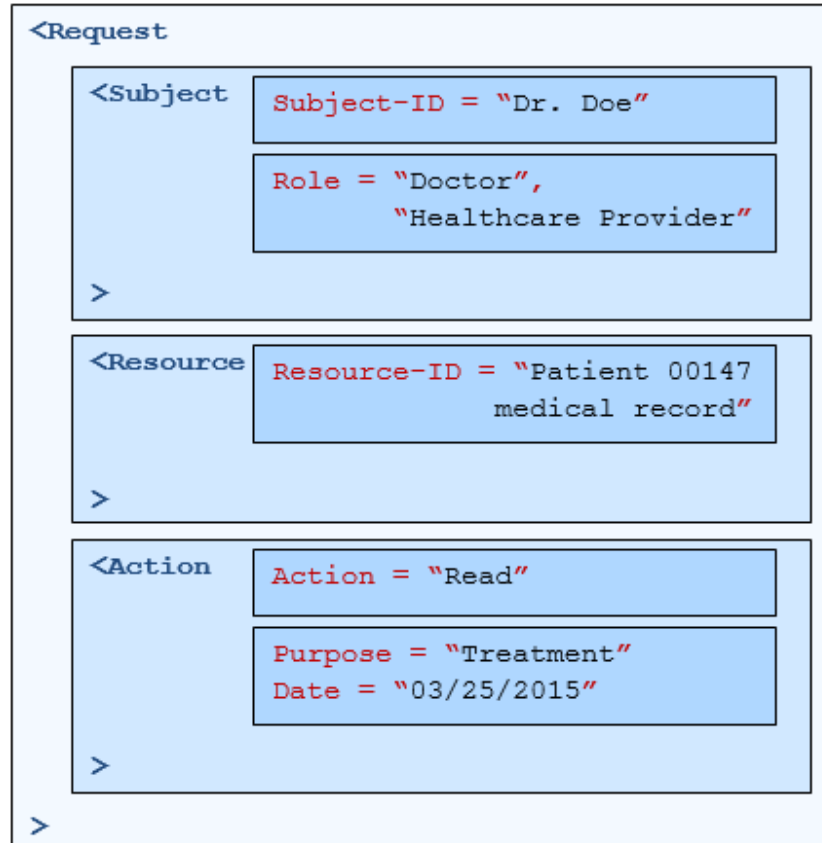


Figure 3.4: Sample XACML extended request: Dr. Doe requesting a medical record

Out of the box, XACML access response will include two values: the request status and the request decision. The status will return the request status code where OK is equivalent to the HTTP code 200. The decision will specify the permissibility of the request. XACML allows the use of one of four values: Permit, Deny, Indeterminate, or Not Applicable [14]. XACML response should be extended also to comply with the law privacy rules. In addition to the request decision answer, it should also include any carried-out obligations. Where obligations define the steps that the requester must carry

out after access to the resource is granted. An example of carried out obligation would be notifying the patient about the transaction or just informing the system to log and audit the transaction [49]. Figure shows a sample XACML response with no carried-out obligations.

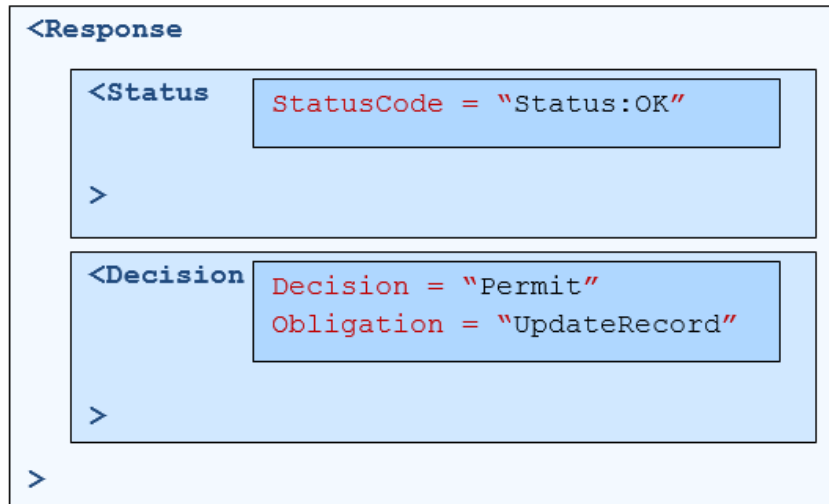


Figure 3.5: Sample XACML extended response

3.5 Privacy Access Control Decision Logic

The access control logic for the model is inserted in the web service. Figure 3.6 shows the flow of a user request to access a specific resource in the system. When the user conducts an access request, the decision engine collects the request context from the system resources. The context will include the user identity, roles, temporal factors, and other user related data based on the user current role. Then it creates an internal XACML formatted access request and sends the new request to XACML decision engine. If the request is authorized, then the system passes the resource request to the covered entity system for regular handling, otherwise, an access denied response is sent back to the user.

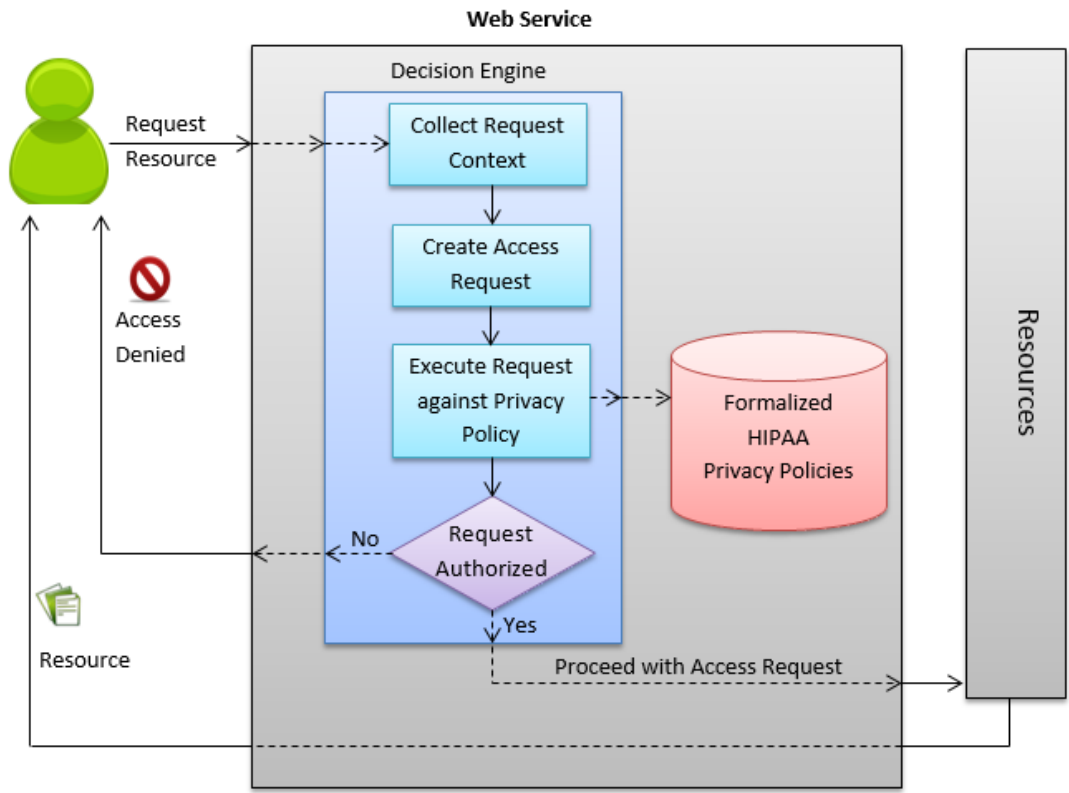


Figure 3.6: Access Control Decision Logic

More fine-grained logic is applied internally inside XACML decision engine. Figure 3.7 shows the way we handle the internal XACML access request created by the web service. When the internal access request formatted in XACML is sent, the Policy Enforcement Point (PEP) receives the request. PEP sends a request to the Policy Information Point (PIP) to request attributes. PEP also sends the request to the Policy Decision Point (PDP) to analyze the request elements. The PDP will receive the request from PEP, the attributes from PIP, and the privacy policies from the Policy Admin Point (PAP). It will then use all those to check if the request is permissible and send the

response back to PEP. PEP will then format the response in XACML and send the internal response to web service.

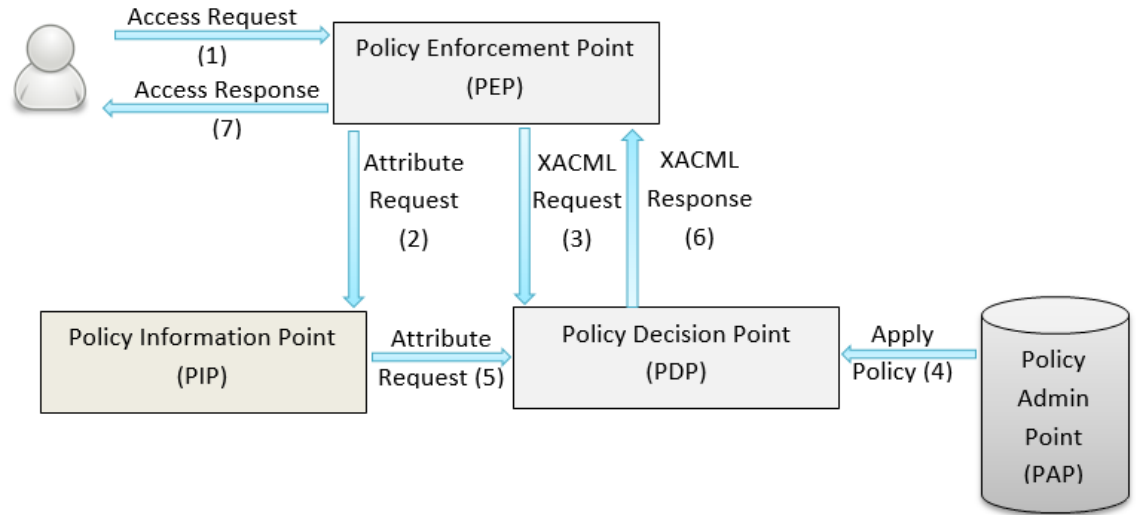


Figure 3.7: XACML decision engine

3.6 Logical Specifications of HIPAA Rules

The logical specifications will follow a similar path. We will identify the systems agents, attributes, the roles, the purpose, and the relationships between them all.

3.6.1 Agents, Attributes, and Knowledge States

Agents is a set of population (**P**), this set includes individuals (patients) and covered entities as well. For each agent in the set, there are attributes associated with that agent. Attributes can be Protected Health Information (PHI), or other attributes that can identify an agent. A knowledge state is a set of triples that indicate an agent p knows the value of an attribute t of the agent q . We can formally express the above as:

Let **P** be a set of agents (e.g. p, q),

T be a set of attributes (e.g. t_1, t_2) about agents

K be a set of knowledge states

Then $\mathbf{K} \subseteq \mathbf{P} \times \mathbf{P} \times \mathbf{T}$

If an agent p knows the value of an attribute t of agent q then: $(p, q, t) \in \mathbf{K}$

3.6.2 Agent Roles

Agent role is an important factor in the authorization process. Our system uses an extended RBAC model that still uses the agent role. However, the access request consists of other factors as relying solely on roles does not satisfy HIPAA rules. To formally express the roles, then assume that **R** is the set of roles of all system agents. For example, if Bob is a doctor in our system, then:

Bob belongs to the agents set: $p_1 \in \mathbf{P}$,

and doctor belongs to the roles set: $r_1 \in \mathbf{R}$

The relation can be formally expressed as a tuple with the verb *inRole*. For the previous example we can express the relation as

$inRole(p_1, r_1)$ where $p_1 \in \mathbf{P}, r_1 \in \mathbf{R}$.

By analyzing HIPAA rules, we can see that most roles are generic or at an abstract level, hence, we need to provide a structural order in the logic. For example, a pediatric role is a specialization of the doctor role. The relation $\leq \mathbf{R}$ is used to define roles specialization/generalization. If $r_1 \leq \mathbf{R} r_2$, then the role r_1 is a specialization of r_2 . We can

formally express the previous example as, pediatric $\leq \mathbf{R}$ doctor. For the purpose of this study, access request should be normalized to the highest role in the structure unless if the specialization role is mentioned directly in the privacy policy.

Due to the fact that an agent can have multiple roles in the system, a role state (ρ) is required. Role state is the set of all roles for agents as a tuple of agent and role where:

$$\rho \subseteq \mathbf{P} \times \mathbf{R}, \text{ where } (p_1, r_1) \in \rho$$

However, for any specific access request, we need to capture the current user role from the context of the request. For example, if Alice is a doctor and a patient in our system, and Alice is requesting to access her own medical file, then we use the patient role in the access request. The context of the request can be formalized as c_1 , where c_1 belongs to the set of all contexts \mathbf{C} . The relation between the agent and the context can be formalized as the verb *inContext* on the tuple (p_1, c_1) as:

$$inContext(p_1, c_1), \text{ where } p_1 \in \mathbf{P}, c_1 \in \mathbf{C}$$

To make this formalization easier to read, syntactic sugar can be used to create the verb *inActiveRole* which can be formalized as:

$$inActiveRole(p_1, r_1) \triangleq inRole(p_1, r_1) \wedge inContext(p_1, c_1), \text{ where } p_1 \in \mathbf{P}, r_1 \in \mathbf{R}, \\ (p_1, r_1) \in \rho, c_1 \in \mathbf{C}$$

3.6.3 Request Purpose

In addition to the role, we also check the purpose of the request. For example, if Alice is a doctor who is trying to access a patient file, we are required by HIPAA to

check the purpose of accessing the file. Being in the role doctor by itself does not guarantee proper authorization to access the patient PHI. If the purpose is to treat the patient, then we check the corresponding HIPAA rule before granting access to the file. To formally express the purpose, we assume that U is the list of all purposes that can be used in any given access request.

3.6.4 Response and Disclosure action

When an agent request access to a PHI, the decision engine will check the request, and if the request is authorized, the response will include the requested PHI as well. According to HIPAA rules, this kind of response where the PHI is sent to the requester is considered information disclosure. To formally express response, we assume that each response (s_1) is drawn from a set of Responses (\mathbf{S}). Each response s contains a set of attributes, if the request is denied, then the set is empty. Because the content of the message is about an agent (the patient) and it includes information (attribute) associated with that agent, the content of the message can be formally expressed as:

$$content(s_1) \subseteq \mathbf{P} \times \mathbf{T}$$

The action of sending the response is considered as “disclosure action”. The relation between a response s_1 , agent q_1 , and the disclosed attribute t_1 can be defined as:

$$contains(s_1, q_1, t_1), \text{ where } s_1 \in \mathbf{S}, q_1 \in \mathbf{P}, t_1 \in \mathbf{T}, \text{ and } t_1 \in \mathbf{phi}$$

where **phi** is the set of protected health information attributes.

The relation between a purpose and a disclosure action is formalized as a tuple (s_1, u_1) . We can say that the purpose u_1 was in the request that produced the response s_1 if $forThePurpose(s_1, u_1)$ is true where $s_1 \in \mathbf{S}$, $u_1 \in \mathbf{U}$.

3.6.5 Temporal conditions

Time plays an important role in the authorization decision. Because roles are dynamic, time can be used to check the current agent role and make sure that the agent is still in the request active role. Also, temporal factors affect the state of an attribute for future disclosure. For example, if the value of an attribute was disclosed to an agent in confidentiality, then in the future, this attribute cannot be disclosed unless PHI disclosure is allowed. Also, past operations can be used to capture any agent pre-authorization to disclose PHI within the limits of HIPAA. Numerous temporal logics can be used for capturing temporal logic for law formalization. First-Order Linear Temporal Logic (LTL) is appropriate and can be used for this purpose [7, 11]. The grammar to generate formulas is:

$$\begin{aligned} \varphi ::= & \text{disclose}(p_1, p_2, s_1) \mid \text{contains}(s_1, q, t_1) \mid \text{inActiveRole}(p_1, r_1) \mid \varphi \wedge \varphi \mid \neg\varphi \mid \\ & \varphi \mathcal{U} \varphi \mid \varphi \mathcal{S} \varphi \mid \exists x : \mathbf{T}. \varphi \end{aligned}$$

3.6.6 Norms of Disclosure

For the purpose of this study, we follow the popular scheme of dividing the norms to positive norms and negative norms. Positive norms are HIPAA rules that allows PHI disclosure hence granting access when the request fulfil the rule requirements. On the other hand, negative norms are those rules that restrict or deny PHI disclosure. In order

for any given request to be granted access, it should satisfy at least one positive norm, and all negative norms. This means that at least one HIPAA positive rule allows the disclosure, and no negative rules deny the request. This can be formally expressed as:

Positive norms ϕ^+ :

$$inActiveRole(p_1, r_1) \wedge inActiveRole(p_1, r_2) \wedge inActiveRole(q, r_3) \wedge (t_1 \in \mathbf{T}) \wedge (t_1 \in \mathbf{phi}) \wedge \theta \wedge \psi$$

Negative norms ϕ^- :

$$inActiveRole(p_1, r_1) \wedge inActiveRole(p_1, r_2) \wedge inActiveRole(q, r_3) \wedge (t_1 \in \mathbf{T}) \wedge (t_1 \in \mathbf{phi}) \wedge \theta \rightarrow \psi$$

Where θ is an agent constraint, and ψ is a temporal condition.

3.6.7 Formally Expressing HIPAA rules using HPRLSL

For any given HIPAA rule, we can formally express the rule using the logic specified above. The same can be applied to any given request, and the decision engine then can use the formalized rules to issue a response.

Based on the above logical formalization, we can say that any given privacy policy can be formally expressed as:

$$\mathbb{P} ::= \square (\forall p_1, p_2, q \mid \mathbf{P}. \forall s_1 : \mathbf{S}. \forall t_1 : \mathbf{T}. \forall u_1 : \mathbf{U}.$$

$$disclose(p_1, p_2, s_1) \wedge contains(s_1, q, t_1) \wedge forThePurpose(s_1, u_1) \rightarrow ((\forall \phi^+) \wedge (\wedge \phi^-))$$

CHAPTER 4: IMPLEMENTATION AND TEST PLAN

4.1 Implementation

The implementation consists mainly of a web service that embeds the XACML privacy policies internally in the policy layer and has the logic embedded in the policy layer. When invoking any web service method, it internally uses XACML decision engine to determine the request permissibility as shown in Figure 4.1 below.

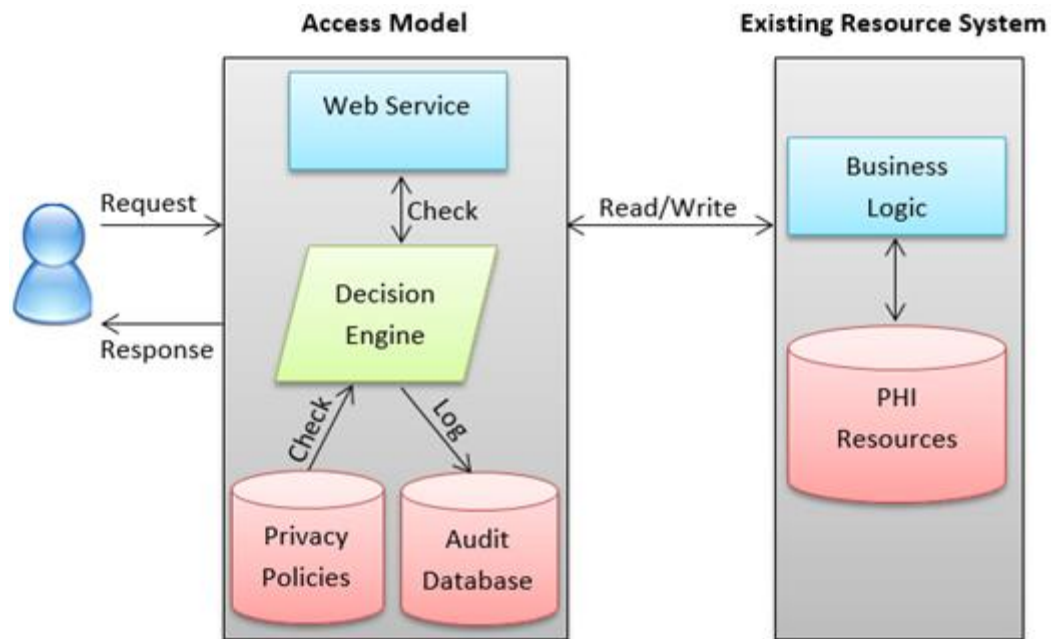


Figure 4.1: The proposed access model architecture

This architecture assumes that any request to the system will pass the web services. This way the web service will provide the access control mechanism and act as an ethical-wall that prevents an unauthorized access to the system.

In order to apply the access control mechanism to an existing system, all paths to access resources will have to be re-routed through the web service to ensure compliance. For example, if the existing system is a web site that is provided by a covered entity to access resources on the covered entity system, then one suggested approach is to add an http handler that will examine all requests and then re-route all those requests to the web services. Once the web service approves the request, then it will log the request to an audit logger and re-route the request back to the original path. If the request is rejected, then the rejection response is issued back to the original system (the web site in our example) to formalize and handle the response back to the client (e.g. response with 401 Unauthorized request).

4.2 Test Plan

To evaluate the proposed access control model, three test plans were developed. Each test plan will assess a specific area of the model. The test plans are: policy formalization test, policy generation test, and policy reasoning and integrity test.

4.2.1 Policy Formalization Test

The policy formalization test is designed to evaluate the extraction and modeling processes described in the Subsections 3.3.3 and 3.3.4. The extraction and modeling process takes HIPAA legal text as an input, applies the process steps, and then return the main components in a semi-formalized form as shown in Figure 3.3 and Table 3.1. The evaluation is based on the work of Kiyavitskaya et al [50–51] on assessing the automation of extracting rights and obligation from regulatory text. The procedure

quantifies the outputs of the modeling process by counting the number of extracted actors, actions, policies, and constraints. Then the results can be compared with a human extracted data to confirm the accuracy of the formalized rules. The human extracted data used in the test comparison were reported by Breaux et al [17] which was collected through an extensive survey on a set of students.

4.2.2 Policy Generation Test

As explained in section 3.4, the formalized HIPAA rules will be finally expressed as an extended-XACML policies. In order to convert the semi-formalized policy components extracted by the extraction and modeling process, a policy generator was used. The policy generator job is to apply the first-order temporal logic and convert the legal text components into a well-formed corresponding XACML policies.

The policy generation test is planned as a measurement to quantify the number of norms, current states, and generation time for any given policy, then we compare that results with the original HIPAA legal text as well as with the generation time from similar models.

4.2.3 Policy Reasoning and Integrity Test

The policy reasoning and integrity test aims to evaluate the system logic and reasoning time and to ensure policies integrity. This test is based on a simulation application that uses the extracted context from the extraction process to create random access requests. Every request invokes the web service and logs the response a long with

the processing time. All the results are then checked to calculate the response reasoning and logic time and number of policies checked.

Each generated request will consist of: the requesting agent identity, role, purpose of request, and the desired action formatted in XACML. The request components domain is a subset of the extracted HIPAA components, where:

- Identity: The agent identity is randomly generated. The identity property is added to the request to differentiate between requests. In real-life applications the identity is very important as it's tightly-coupled with the agent relation to the covered entity and also its role in system. However, for this simulation test, we assume that the agent current role is the same as the role exclusively encoded in the request.
- Role: The agent role r_1 belongs to the HIPAA identified roles set \mathbf{R} ($r_1 \in \mathbf{R}$). We apply role abstraction method to generalize the role to highest possible level, for example a pediatric role is generalized to a doctor role. However, the original role is still maintained in request as it might have a significance in the law, for example, a pediatric is not allowed to review psychiatric notes.
- Purpose: as explained previously, the request purpose is used as one of main criteria in determining the request permissibility. Similar to roles, the purpose p_1 belongs to the general set of HIPAA identified purposes \mathbf{P} ($p_1 \in \mathbf{P}$).

Based on the analysis above, the test will contain a set of requests \mathbf{Q} , each request q_i can be formally expressed as tuple $q_i (p_i, r_i, p_j)$ where $q_i \in \mathbf{Q}$, p_i is a randomly generate identity, $r_i \in \mathbf{R}$, $p_j \in \mathbf{P}$.

As an initial hypothesis, we expect that any request reasoning time should be equal to or less than 0.1 seconds. To generate a statically acceptable sample request, power analysis is used to calculate the sample size for the simulation application. Power analysis or statistical power is the probability that a null hypothesis H_0 will be rejected if an alternative hypothesis H_A is confirmed to be true [52]. To calculate the sample size, we use the initial hypothesis as a starting point that reasoning time should be less than or equal to 0.1. We also assume that the required level of confidence is within 2 standard deviations around the mean, which is statistically (95.45%) or simply just 95%.

The statistical hypothesis for this work is as follows:

$$H_0 \rightarrow \mu_s > 0.1 \dots\dots\dots (0.1)$$

$$H_A \rightarrow \mu_s \leq 0.1 \dots\dots\dots (0.2)$$

Where μ_s is the expected average reasoning time for any given request q_i , H_0 is the null hypothesis that assumes for any given request the reasoning time is greater than 0.1 seconds, which is exactly the opposite of our initial hypothesis H_A .

We apply the numbers into the statistical power formula to calculate the sample size n as:

$$n = (N*S*(1-S)) / (((N-1)* (\alpha^2 / Z\alpha/2^2)) + S*(1-S))\dots\dots\dots (0.3)$$

Where N is the general population size of all possible requests, S is the effect size, α is the complement of the confidence, and Z is the t-value of $\alpha/2$.

To compute N , the general population size, we need to find the number of elements in the set of all possible requests Q .

$$Q = \{R \times P\} \dots\dots\dots (0.4)$$

By checking HIPAA extracted components, we find that the size of Q is 900.

In order to find $Z\alpha/2$, first we need to calculate α which is defined as:

$$\alpha = 1 - \beta \dots\dots\dots (0.5)$$

Where β is the required level of confidence, which we assumed is 95% so:

$$\alpha = 1 - 0.95 = 0.05$$

$$\alpha / 2 = 0.05 / 2 = 0.025$$

$Z\alpha/2$ is $Z0.025$ by looking up the value for $Z0.025$ in the t-table we find that the value is:

$$Z0.025 = 1.96$$

To calculate the value of the effective size S , we relied on Thompson [53] justification of using the t-test on means for large test size. Based on that and by checking Cohen's [52] table for effect size values (Table 4.1 below), we can assume that:

$$S = 0.8$$

Table 4.1: Effect Size Values

	Effect size Index	Small	Medium	Large
t-test on Means	<i>D</i>	0.20	0.50	0.80
t-test on Correlations	<i>R</i>	0.10	0.30	0.50
F-test ANOVA	<i>F</i>	0.10	0.25	0.40
F-test regression	<i>f</i> ²	0.02	0.15	0.35
Chi-Square test	<i>w</i>	0.10	0.30	0.50

By applying the numbers to the equation 4.3, the sample size is:

$$n = (900 * 0.8 * (1 - 0.8)) / (((900 - 1) * (0.05^2 / 1.96^2)) + 0.8 * (1 - 0.8)) = 193.2775153$$

The result is rounded up to get a whole number

$$n = 194 \text{ sample size}$$

CHAPTER 5: RESULTS

5.1 Test Results

As explained in the test plan in the previous chapter, three tests were designed to assess different aspects of the proposed model. The first two are designed to test the formalization and generation processes, which will test the in-term procedures followed to create the policies. Where the third test will perform a simulation to evaluate how the access model will behave to response for access requests.

5.1.1 Policy Formalization Test Results

Following the test plan described in 4.2.1, the extraction process was applied to a selected set of HIPAA privacy rules to model and extract the context components. The test results are then compared to a human extracted context reported by Breaux et al [17] to evaluate the formalization process. Table 5.1 shows a sample of the test results showing the rule id, the count of agents, roles, temporal factors, and purposes.

Table 5.1: Sample results of extraction and modeling test for rule 164.512

Extraction method	Agents	Specialized Roles	Generalized Roles	Temporal Factors	Purpose
Automated	4	5	3	1	Treatment
Human	4	4	2	1	Treatment

5.1.2 Policy Reasoning and Integrity Test Results

A simulation application was created to produce a total of 194 access requests, each request contained a unique tuple of agent identity, agent role, and the purpose of the request. The values are randomly selected from lists containing the entire extracted population from HIPAA rules. The simulation application logs all the results including the reasoning / response time, the number of policies checked, and the response text to a file. Table 5.2 shows a sample of the simulation application output.

Table 5.2: Sample results of Access Request Simulation showing response time

Request Id	Response / Reasoning Time	Number of Policies checked	Response results
59	0.0917	95	Permit
60	0.0858	93	Deny
61	0.0963	96	Deny
62	0.1047	97	Permit
63	0.1031	96	Permit
64	0.0904	94	Permit
65	0.0991	96	Deny

To check if the hypothesis that was developed in the test plan is true, we need to prove that the null Hypothesis H_0 is false, or the alternative hypothesis H_1 is true: $\{H_0 \text{ is false} \mid H_1 \text{ is true}\}$. Because the null hypothesis $H_0 \rightarrow \mu_s > 0.1$ uses the greater than operator, we need to perform a right-tailed test. So, in order to reject H_0 and prove its falseness, t-statistic t^* should be less than the critical value t for H_0 where:

$$t\text{-statistic } t^* = (\bar{x} - \mu_s) / (\sigma / \sqrt{n})$$

$$\text{Critical } t = \text{TINV}(\alpha, \text{df})$$

Where \bar{x} is the statistical mean, σ is the variance, n is the sample size, df is the degree of freedom. The following statistics were calculated from the test results:

$$\text{Mean } (\bar{x}) = \Sigma x / n = 17.856 / 194 = 0.091569231$$

$$\text{Standard Deviation } (\sigma) = \Sigma(x - \bar{x}) / n = 0.006893176$$

$$\text{Degree of freedom } (\text{df}) = n - 1 = 194 - 1 = 193$$

$$\text{Significance } (\alpha) = 1 - \beta = 0.05$$

$$\text{Standard Error } (S_x) = \sigma / \sqrt{n} = 0.000493631$$

$$\text{Critical } t = \text{TINV}(\alpha, \text{df}) = 1.972267533$$

$$\text{Statistic-t } t^* = (\bar{x} - \mu_s) / (\sigma / \sqrt{n}) = (0.091569231 - 0.1) / 0.000493631 = -17.07910654$$

Based on the statistics values above, H_0 is rejected because t^* is less than H_0 critical t as $17.07910654 < 1.972267533$.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this research, we proposed and implemented a HIPAA-complaint privacy access control model for web services. We identified the model components and pointed out each component requirements, design, and role in the model. Furthermore, we evaluated the previous work and extended it to support our model.

The research started initially with a theoretical model for a generic HIPAA access control that uses web services as a unified entry point. The idea is to regulate access to the system resources based on HIPAA privacy rules to ensure compliance with the law. The initial conceptual design was then broken into four smaller components: the extraction and formalization process, the expression process, the access model logic, and the web service.

To extract and formalize HIPAA privacy rules from HIPAA regulatory text, we proposed and implemented a two-step process. The process is based on Natural Language Processing (NLP) and followed a goal-oriented role engineering method. The process was then applied to HIPAA privacy rules to generate a set of classified elements representing the context of the access control model. The elements represent all the applicable roles, permissions, purposes, obligations, and any temporal factors.

In order to express the extracted context in a formal language that can be used in the model, we evaluated the current privacy expression languages and compared each

language adequacy to represent the privacy policies as well as the access request and response. We also considered the use of a custom language format based on XML. However, we found that XACML can be extended to express the formalized rules. We extended XACML access request, access response, and policy structure to represent HIPAA privacy rules. We also utilized XACML ability to implement an external logic point to implement our access control logic.

Then, we implemented the formalized HIPAA privacy policies and the access control model within the web service layers to create a complete access control model. The web service will store the formalized HIPAA privacy policies in the policy layer, while the access control logic was implemented within the privacy layer.

Finally, three test plans were created to evaluate the system compliance and to ensure policies integrity. The tests were designed to assess the context extraction process, the policy generation process, and the request reasoning and response time. Also, a simulation application was created to mimic real-life access request with a context extracted from HIPAA text. The simulation application used the extracted context from the extraction process to create random access requests. Every request invoked the web service and logged the response along with the processing time. The preliminary results showed a reasonable generation and response time with a 95% confidence level in compliance with HIPAA.

6.2 Future Work

Based on the current design and the test results, we identified two possible enhancements for the system: automating XACML policy generation and better policy integrity check.

Our current procedure of converting the letter of the law into XACML policies passes through multiple processes. Where the first step is to create the keyword dictionary, then apply the two-step extraction process, and finally convert the elements into policies. Manual human intervention is required in almost each part of the process. However, we should automate the process of converting the context from extracted elements to XAML policies.

The second possible enhancement is to develop a more robust test method for policy integrity check. Our current test methodology is based on simulation application that creates a random-access request using random context and then we assess the results to ensure compliance, integrity, and response time. However, we need to take more factors into considerations including context that will check the interaction between multiple policies especially if one policy is referencing or represent an exception to another policy.

REFERENCES

- [1] S. J. Dwyer III, A. C. Weaver, and K. K. Hughes, “Health insurance portability and accountability act,” *Security Issues in the Digital Medical Enterprise*, vol. 72, no. 2, pp. 9–18, 2004.
- [2] World Wide Web Consortium (W3C), “Web services glossary,” <http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211>, 2004, [Online; accessed 21-May-2015].
- [3] J. Zhang and W. Xu, “Web Service-based Healthcare Information System (WSHIS): A case study for system interoperability concern in healthcare field,” in *the Proc. International Conference on Biomedical and Pharmaceutical Engineering*, pp. 588–594, 2006.
- [4] S. Chatterjee, “Developing enterprise web services and applications: Opportunities and best practices for the healthcare industry,” in *Proc. 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry*, pp. 159–207, 2003.
- [5] B. K. Atchinson and D. M. Fox, “From the field: The politics of the health insurance portability and accountability act,” *Health Affairs*, vol. 16, no. 3, pp. 146–150, 1997.
- [6] U.S. Department of Health and Human Services, “Understanding health information privacy,” <http://www.hhs.gov/ocr/privacy/hipaa/understanding/>, 2003, [Online; accessed 21-May-2015].

- [7] “Health Insurance Portability and Accountability Act of 1996. Public Law 104-191.” *United States Statutes at Large*, vol. 110, pp. 1936–2103, 1996.
- [8] Datta, J. Blocki, N. Christin, H. DeYoung, D. Garg, L. Jia, D. Kaynar, and A. Sinha, “Understanding and protecting privacy: Formal semantics and principled audit mechanisms,” in *Proc. International Conference on Information Systems Security*, pp. 1–27, 2011.
- [9] “Gramm-Leach-Bliley Act of 1999. Public Law 106-102.” *United States Statutes at Large*, vol. 113, pp. 1338–1481, 1999.
- [10] H. DeYoung, D. Garg, L. Jia, D. Kaynar, and A. Datta, “Experiences in the logical specification of the HIPAA and GLBA privacy laws,” in *Proc. 9th annual ACM workshop on Privacy in the electronic society*, pp. 73–82, 2010.
- [11] Barth, A. Datta, J. C. Mitchell, and H. Nissenbaum, “Privacy and contextual integrity: Framework and applications,” in *Proc. 2006 IEEE Symposium on Security and Privacy*, pp. 15–pp, 2006.
- [12] P. E. Lam, J. C. Mitchell, and S. Sundaram, “A formalization of HIPAA for a medical messaging system,” in *Proc. International Conference on Trust, Privacy and Security in Digital Business*, pp. 73–85, 2009.
- [13] M. Backes, B. Pfitzmann, and M. Schunter, “A toolkit for managing enterprise privacy policies,” in *Proc. European Symposium on Research in Computer Security*, pp. 162–180, 2003.
- [14] L. Cranor, “Web privacy with P3P,” *O’Reilly Media, Inc.*, pp. 5–11, 2002.

- [15] H. Lockhart, B. Parducci, and R. Levinson, "OASIS eXtensible Access Control Markup Language (XACML)," <https://www.oasisopen.org/committees/xacml/>, 2004, [Online; accessed 21-May-2015].
- [16] I. Anton, J. B. Earp, M. W. Vail, N. Jain, C. M. Gheen, and J. M. Frink, "HIPAA's effect on web site privacy policies," *IEEE Security & Privacy*, vol. 5, no. 1, pp. 45–52, 2007.
- [17] T. D. Breaux, M. W. Vail, and A. I. Anton, "Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations," in *Proc. 14th IEEE International Conference on Requirements Engineering*, pp. 49–58, 2006.
- [18] T. Breaux and A. Anton, "Analyzing Regulatory Rules for Privacy and Security Requirements," *IEEE transactions on software engineering*, vol. 34, no. 1, pp. 5–20, 2008.
- [19] International Medical Informatics Association (IMIA), "IMIA code of ethics for health information professionals," <https://imia-medinfo.org/wp/imia-code-of-ethics/>, 2003, [Online; accessed 21-May-2015].
- [20] C. S. Powers, P. Ashley, and M. Schunter, "Privacy promises, access control, and privacy management. Enforcing privacy throughout an enterprise by extending access control," in *Proc. Third International Symposium on Electronic Commerce*, pp. 13–21, 2002.
- [21] R. J. Anderson, "A security policy model for clinical information systems," in *Proc. IEEE Symposium on Security and Privacy*, pp. 30–43, 1996.
- [22] D. Ferraiolo and R. Kuhn, "Role-Based Access Controls," in *Proc. 15th National Computer Security Conference*, pp. 554–563, 1992.

- [23] M. Bishop, “Computer Security: Art and Science,” *Addison-Wesley Professional*, 2003.
- [24] J. Williams, “Role-based access control models for E-healthcare systems,” *Florida A&M University*, Department of Computer and Information Sciences, 2007.
- [25] S. Islam, H. Mouratidis, and S. Wagner, “Towards a framework to elicit and manage security and privacy requirements from laws and regulations,” in *Proc. International Working Conference on Requirements Engineering: Foundation for Software Quality*, pp. 255–261, 2010.
- [26] S. H. Houmb, S. Islam, E. Knauss, J. Jürjens, and K. Schneider, “Eliciting security requirements and tracing them to design: An integration of common criteria, heuristics, and UMLsec,” *Requirements Engineering*, vol. 15, no. 1, pp. 63–93, 2010.
- [27] Siena, J. Mylopoulos, A. Perini, and A. Susi, “From laws to requirements,” in *Proc. Requirements Engineering and Law. RELAW’08.*, pp. 6–10, 2008.
- [28] F. Z. Jorshari, H. Mouratidis, and S. Islam, “Extracting security requirements from relevant laws and regulations,” in *Proc. Sixth International Conference on Research Challenges in Information Science (RCIS)*, pp. 1–9, 2012.
- [29] H. Mouratidis and P. Giorgini, “Secure tropos: a security-oriented extension of the tropos methodology,” *International Journal of Software Engineering and Knowledge Engineering.*, vol. 17, no. 02, pp. 285 – 309, 2007.
- [30] R. Darimont, M. Lemoine, and B. Cediti, “Goal-oriented analysis of regulations,” in *Proc. International Workshop on Regulations Modelling and their Verification & Validation*, pp. 838–844, 2006.

- [31] M. J. May, C. A. Gunter, and I. Lee, "Privacy APIs: Access control techniques to analyze and verify legal privacy policies," in *Proc. IEEE Computer Security Foundations Workshop*, pp. 85–97, 2006.
- [32] OASIS, "A brief introduction to XACML." https://www.oasis-open.org/committees/download.php/2713/Brief_Introduction_to_XACML.html, 2003, [Online; accessed 21-May-2015].
- [33] A. Anderson, "A comparison of two privacy policy languages: EPAL and XACML," in *Proc. 3rd ACM workshop on Secure web services*, pp. 53–60, 2006.
- [34] World Wide Web Consortium (W3C), "Publishing web service policies," *W3C Constraints and Capabilities Workshop.*, vol. 9, no. 3, pp. 1–6, 2004.
- [35] Y. Peng and Q. Wu, "Secure communication and access control for web services container," in *Proc. International Conference on Grid and Cooperative Computing*, pp. 412–415, 2006.
- [36] T. Alshugran and J. Dichter, "Toward a privacy preserving HIPAA-compliant access control model for web services," in *Proc. IEEE International Conference on Electro/Information Technology (EIT)*, pp. 163–167, 2014.
- [37] T. Alshugran and J. Dichter, "A framework for extracting and modeling HIPAA privacy rules for healthcare applications," *Health Informatics - An International Journal*, vol. 5, no. 1, pp. 1–10, 2016.
- [38] M. C. Tschantz, A. Datta, and J. M. Wing, "Purpose restrictions on information use," in *Proc. European Symposium on Research in Computer Security*, pp. 610–627, 2013.

- [39] R. Alur and T. A. Henzinger, “A really temporal logic,” *Journal of the ACM (JACM)*, vol. 41, no. 1, pp. 181–203, 1994.
- [40] T. Alshugran and J. Dichter, “Extracting and modeling the privacy requirements from HIPAA for healthcare applications,” in *Proc. Long Island Systems, Applications and Technology Conference (LISAT)*, pp. 1–5, 2014.
- [41] M. A. Mizani and N. Baykal, “A software platform to analyse the ethical issues of electronic patient privacy policy: the S3P example,” *Journal of Medical Ethics*, vol. 33, no. 12, pp. 695–698, 2007.
- [42] P. Mazzoleni, B. Crispo, S. Sivasubramanian, and E. Bertino, “XACML policy integration algorithms,” *ACM Transaction on Information System Security*, vol. 11, no. 1, pp. 1–29, 2008.
- [43] P. N. Otto and A. I. Anton, “Addressing legal requirements in requirements Engineering,” in *Proc. Requirements Engineering Conference*, pp. 5–15, 2007.
- [44] R. Crook, W. Hall, and M. Keynes, “Towards an analytical role modelling framework for security requirements,” in *Proc. 8th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ’02)*, pp. 1–14, 2002.
- [45] G. Neumann and M. Strembeck, “A Scenario-driven role engineering process for functional RBAC roles,” in *Proc. 7th ACM Symposium on Access Control Models and Technologies (SACMAT’02)*, pp. 33–42, 2002.
- [46] P. Epstein and R. Sandhu, “Towards A UML-based approach to role engineering,” in *Proc. 4th ACM Workshop on Role-Based Access Control (RBAC’99)*, pp. 135–143, 1999.

- [47] W. N. Hohfeld, “Fundamental legal conceptions as applied in judicial reasoning,” *Yale Law Journal*, vol. 26, no. 8, pp. 710–770, 1917.
- [48] T. Alshugran, J. Dichter, and M. Faezipour, “Formally expressing HIPAA privacy policies for web services,” in *Proc. IEEE International Conference on Electro/Information Technology (EIT)*, pp. 295–299, 2015.
- [49] T. Alshugran, J. Dichter, and A. Rusu, “Extending XACML to express and enforce laws and regulations privacy policies,” in *Proc. Long Island Systems, Applications and Technology Conference (LISAT)*, pp. 1–5, 2015.
- [50] N. Kiyavitskaya, N. Zeni, T. D. Breaux, A. I. Ant, J. R. Cordy, L. Mich, and J. Mylopoulos, “Automating the extraction of rights and obligations for regulatory compliance,” in *Proc. International Conference on Conceptual Modeling*, pp. 154–168, 2008.
- [51] N. Zeni, N. Kiyavitskaya, L. Mich, J. R. Cordy, and J. Mylopoulos, “GaiusT: Supporting the extraction of rights and obligations for regulatory compliance,” *Requirements Engineering*, vol. 20, no. 1, pp. 1–22, 2015.
- [52] J. Cohen, “Statistical power analysis for the behavioral sciences,” *Lawrence Erlbaum Associates, Inc*, 1977.
- [53] S. K. Thompson, “Adaptive cluster sampling,” *Journal of the American Statistical Association*, vol. 85, no. 412, pp. 1050–1059, 1990.