AUGMENTING ACCOUNTABILITY, SECURITY AND FRAUD DETECTION IN HEALTH DATA SHARING SYSTEMS

A Thesis
Presented to
The Academic Faculty

by

Musheer Ahmed

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computer Science

Georgia Institute of Technology
May 2016

AUGMENTING ACCOUNTABILITY, SECURITY AND FRAUD DETECTION IN HEALTH DATA SHARING SYSTEMS

To my family:

my parents Zareen and Ameen,

my wife Husna,

my brother Zaheer, my sister Alia & her husband Atheeqe,

and my in-laws Farida and Rafeek.

ACKNOWLEDGEMENTS

I would like to thank Allah (the Arabic word for God) for His unconditional love, mercy and blessings and for giving me the opportunity to grow spiritually and intellectually. I pray that the work done in this dissertation leads to the benefit of humanity for which I can be rewarded perpetually according to a narration of Prophet Muhammad (peace be upon him and all other prophets such as Jesus, Moses, Abraham, Noah and Adam).

I don't think I can truly thank all the people who have helped shape me into who I am today. I hope to acknowledge at least a few of them here. I would like to start with my parents who encouraged me to pursue my doctoral studies. My wife who was extremely supportive and patient with me while I was busy with my research. My brother who was always available to help me whenever I needed him. My sister who would always help me put things into perspective and my relatives and many wonderful friends for their support, valuable time and company.

I would like to thank my advisor, Professor Mustaque Ahamad, for guiding me throughout the research process and patiently allowing me to find my own answers. I would like to also thank Dr. Mark Braunstein with whom I have worked with very extensively throughout my PhD. He has constantly shared his expertise in the healthcare domain and was available whenever I needed him. I would also like to thank the rest of my thesis committee, Professor Doug Blough, Professor Wenke Lee and Professor Ling Liu for graciously accepting to give me their valuable time and advice.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . **iv**

LIST OF TABLES . . . . **ix**

LIST OF FIGURES . . . . **x**

SUMMARY . . . . **xi**

I INTRODUCTION **1**

1.1 Healthcare Data Sharing 1

1.2 Challenges in Securing Healthcare Data 2

1.3 Thesis Statement 4

1.4 Contributions & Thesis Overview 4

II RELATED WORK **6**

2.1 Healthcare Identity and Access Management 6

2.2 Healthcare Auditing 7

2.3 Healthcare Data Provenance 8

2.4 Healthcare Fraud Detection 8

2.5 Securing Health Data on Mobile Devices 9

2.6 Conclusions 10

III HEALTHCARE ECOSYSTEM **12**

3.1 Health Information Sharing Architecture 12

3.2 U.S. Health Insurance System 15

3.2.1 Claims Lifecycle 16

3.3 Threats 21

3.3.1 Medical Identity Theft 21

3.3.2 Unauthorized Access 21

3.3.3 Phantom Clinics 22

3.3.4 Malicious Insiders 22

3.3.5 Devices 23

3.3.6 Kickbacks 24

3.4 Augmented Health Information Sharing Architecture 24

3.4.1 Enhanced Auditing & Awareness 24

3.4.2 Fraud, Waste and Abuse Detection 25

3.4.3 Securing Health Data at End Devices 25

3.5 Conclusions 26

IV ENHANCED AUDITING & AWARENESS 27

4.1 Enhanced Auditing System 29

4.1.1 Sharing Record Log 30

4.1.2 Audit Record 32

4.1.3 Component Logging Agent 33

4.1.4 CONNECT Monitoring Agent 34

4.2 Patient Data Awareness 35

4.2.1 Patient Agent 36

4.3 Implementation 38

4.3.1 Enhanced Auditing System 38

4.3.2 Patient Agent 43

4.4 Evaluation 45

4.4.1 Performance Evaluation 45

4.4.2 Security Analysis 47

4.5 Conclusions 51

V FRAUD, WASTE AND ABUSE DETECTION 53

5.1 Combating Healthcare Fraud 54

5.2 Datasets 56

5.2.1 Medicare Services and Procedures Datasets 57

5.2.2 Medicare Prescription Drug Dataset 58

5.3 Our Approach 59

5.4 FraudScope 60

5.4.1 Features 62
5.4.2 Profile Calculation Window 64
5.4.3 Generating Reference Profiles 65
5.4.4 Calculating Risk Scores and Ranks 66
5.4.5 Investigation 67
5.5 Implementation 69
5.5.1 Web Scraping 71
5.5.2 Reverse Scrape 71
5.6 Evaluation 74
5.6.1 Sampling 74
5.6.2 Detailed Analysis & Comparison with Government Technique 78
5.6.3 Comparison with Normal Subspace 86
5.6.4 Investigating Uncommon Categorical Values 87
5.6.5 Robustness 88
5.7 Conclusions 89

VI SECURING HEALTH DATA AT END DEVICES 91

6.1 Mobile Device Data Security: Motivation and Requirements 93
6.1.1 Mobile Device Data Security Policies 95
6.1.2 Requirements 96
6.2 Approach 100
6.2.1 Tagging Sensitive Data 101
6.2.2 Monitoring Tagged Data Flow 103
6.3 Implementation 104
6.3.1 Background Service 106
6.3.2 Security Policy Enforcement Engine 107
6.3.3 Controlling External Data Flow 109
6.3.4 Controlling Communication with Untrusted Applications . . 111
6.3.5 User Consent Detection 112

6.4 Evaluation 114

6.4.1 Performance Analysis 114

6.4.2 Threat Analysis 116

6.5 Conclusions 121

VII CONCLUSIONS & FUTURE WORK 122

7.1 Future Work 124

REFERENCES 126

LIST OF TABLES

1 Performance Evaluation Results 47

2 CMS Dataset Details 56

3 CMS Dataset Fields 57

4 CMS Drug Dataset Fields 58

5 Example Claims Filed by Various Dermatologists 67

6 CMS Dataset Features Selected 70

7 Known Fraudulent Clinical Labs 75

8 Known Fraudulent Chiropractors 77

9 Classification 80

10 Detailed Analysis of Results 81

11 Comparison with Normal Subspace using k-means clustering 86

12 Comparison with Normal Subspace using LOF 86

13 Comparison with Normal Subspace 87

14 Commonality values for Steven Schall 87

15 Commonality values for Shakuntala Jain 88

16 Commonality values for Gary Tanouye 88

17 Performance Evaluation Results 115

LIST OF FIGURES

1 An eHealth Exchange Node. 13

2 Claims Lifecycle 16

3 Existing eHealth Exchange Node (Left) and Node with our Enhancements (Right). 30

4 Enhacned eHealth Exchange Node with Component Logging Agents. 33

5 Enhanced eHealth Exchange Node with CLA and CMA. 34

6 Enhacned eHealth Exchange Node with CLA, CMA and PA. 36

7 Example Prospective Detection 61

8 Scree Plot for Clinical Lab 70

9 OIG Measures to Detect Fraud 85

10 A prompt generated by our framework. 107

11 Policy enforcement framework. 108

12 User consent detection mechanism. 113

SUMMARY

The U.S. government has introduced federal incentive programs to accelerate the adoption of meaningful use of electronic health records (EHR). These electronic records are expected to improve healthcare quality, reduce costs, and facilitate data sharing across different healthcare enterprises. However, electronic health data has already been subjected to various threats. Stolen identities will then be used to defraud health insurance programs by submitting fraudulent claims for reimbursement which are difficult to identify due to the large volume of claims received by them. Healthcare fraud already costs the country about $272 billion and this will increase in magnitude if we do not actively secure the health information sharing infrastructure.

We enhance the health information sharing architecture by augmenting certain components within each node. We first introduce mechanisms that augment accountability and cannot be circumvented as long as multiple independent parties that interact with one another are not compromised simultaneously. These enhancements help us ensure that we log all inter-nodal and inter-component interactions which cannot be subverted or corrupted without detection unless both interacting parties are compromised or become malicious. Our audit records are also redundantly distributed throughout the node and the sharing architecture helps in detecting deletion of records. In addition to this, we also help alleviate patient privacy concerns by providing greater awareness to patients about how their personal medical information is being shared. These enhancements enable early detection of unauthorized and malicious sharing of health data, which can help limit the resulting damage.

In case a breach or compromise occurs, sharing provenance introduced by us helps identify the medical practitioner or healthcare organization that may be a source of a leak of information or the unauthorized node that fraudulently releases or acquires a particular patient's data. We can use these records to determine all the other actions that were performed by those identities within the eHealth Exchange and which may have also been malicious. Reversing these actions would give us a starting point to rectify any patient's medical history that may have been corrupted by fraud.

We also develop a fraud, waste and abuse detection system that helps detect suspicious medical insurance claims and provides them a rank and risk score to prioritize their investigation and maximize savings. Unlike existing methods, it does not require any input in the fraud identification stage which frees up precious fraud analyst time to investigate a greater number of claims. This also enables our technique to detect newly emerging as well as previously unidentified fraud schemes. It can also adapt to natural changes in the treatment patterns or changes in the coding system and will not require any additional work by the user to include these changes.

Finally, we address security of the devices that connect to the health information sharing architecture. We achieved this by introducing the concept of a constrained application for mobile devices which can be used to safeguard sensitive data and prevent its flow to unauthorized entities. A constrained application's sensitive data can be protected even in the presence of attacks that can successfully compromise the application. Our mechanisms accommodate for varying levels of sensitivity of data within the constrained application which can be governed by different security policies. Non-sensitive data within a constrained application is unaffected by our mechanisms. Also, applications that do not deal with sensitive data are unaffected. We also developed a user consent detection mechanism which can help distinguish actual user input from scripted events that can be generated by malware. Such secure consent can enable user awareness and control over how health information is shared.

CHAPTER I

INTRODUCTION

1.1 Healthcare Data Sharing

Federal incentives by the United States Government for the Meaningful Use of certified *Electronic Health Record (EHR)* systems have accelerated the adoption of *EHR* systems across America. These systems will facilitate the sharing of *Electronic Medical Records (EMR)* across the nation and will lead to reduced costs and improved healthcare quality. The US Government has launched a certification process to assure adopters that the electronic health IT products and systems they use offer the necessary security and maintain data confidentially. However, this certification does not guarantee that sensitive healthcare information will be secure against all threats and it has already been shown to leave common code-level and design-level vulnerabilities undetected [95] which has exposed health data to various threats.

Due to this, there has been an increase in hacking and IT-related incidents to breach health data in recent years. The Washington Post declared 2015 as the year of the health-care hack [109] where we saw major breaches at healthcare institutions that affected the identities of over 112 million individuals [26]. Hacking or IT incidents have accounted for 98% of leaked health records due to breaches in 2015. In comparison, hackers compromised only 1.8 million individual identities due to health data breaches in 2014 [45]. Healthcare organizations are currently being compromised at 35 breaches per month that affect 500 individuals or more [101], and each breach costs healthcare organizations over $2.1 million on average [85]. Investigation of a breach is usually done by parsing auditing logs for malicious actions. However, research has already shown that current logging standards are lagging behind other industries [55], and are inadequate [56].

Breached healthcare data and stolen identities will eventually be used to obtain monetary benefit by filing fraudulent claims at government healthcare programs like Medicare and Medicaid or private insurance programs. Healthcare fraud costs the United States an estimated $272 billion a year and, according to several studies, billions of dollars in additional fraud, remain undetected. Due to the sheer volume of healthcare claims submitted on behalf of the millions of Americans insured under these programs, the government and private entities cannot effectively combat healthcare fraud.

1.2 Challenges in Securing Healthcare Data

Healthcare data is unique in comparison to other sensitive data as it is usually governed by the 'Break the Glass' access control policy. This allows medical personnel to view sensitive healthcare data under emergency circumstances even when such personnel do not have the necessary system access privileges. This allows health data to be easily abused and breached by employees working within a healthcare organization. As we cannot enforce a strict access control policy, we must mainly rely on a strong auditing system to detect such abuses.

Stealing sensitive health information is also very lucrative compared to financial data. It includes other sensitive information such as social security numbers, medical record numbers, date of birth, etc. and is worth more on the black market than credit-card numbers. Such stolen health data can also be used to obtain prescriptions for controlled drugs. Each stolen or lost healthcare record costs healthcare organizations $363 on average.

Criminals can also monetize stolen health data by filing claims to insurance companies and government reimbursement programs. However, fraud is difficult to detect in claims due to the complex nature of the healthcare system in the US. Fraud, waste and abuse in healthcare programs can be as high as 30%. This costs an average family of four over $200 a year above what they would pay for the honest delivery of healthcare goods and services [8]. Patients also sometimes have lifetime caps or other financial limits imposed by their health insurance policies which can be exhausted by fraud. This would prevent them from using their benefits when they legitimately need them [74]. Some criminals are switching from cocaine trafficking to prescription-drug fraud because the risk-adjusted rewards are higher: the money is still good, the work safer and the penalties lighter [100].

Usually, it takes victims of healthcare fraud longer to notice that their details have been stolen. Unlike other industries, the procedures, services and drugs provided to patients are paid for by health insurance companies. Hence, the patients are sometimes unaware of their medical identity being stolen or are not motivated to carefully examine the complicated Explanation of Benefits (EOB) that they receive. Even when patients are aware that their identity has been misused, there is little recourse available to them. Unlike the financial industry, they cannot halt all transactions or freeze their medical identity. A new credit card can easily be issued if stolen, but a compromised medical identity may require changing the patient's social security number which is a long and cumbersome process.

In addition to all this, a medical identity theft victim may receive the wrong diagnosis for his current symptoms or unexpectedly fail a physical exam for employment because a disease or condition for which he's never been diagnosed or received treatment has been unknowingly documented in his health record. Sanitizing a victim's corrupted medical history can be a grueling and stressful endeavor. The effects of this crime can plague a victim's medical and financial status for years to come [74].

1.3 Thesis Statement

We keep these unique characteristics of healthcare data in mind and make the dissertation hypothesis that *middleware in systems which exchange health information can be augmented to support better accountability and security of health data and reduce losses due to fraud.*

1.4 Contributions & Thesis Overview

As an example, we apply these techniques to the specifications, developed under the auspices of the U.S. Office of the National Coordinator for Health Information Technology (ONC), that establish secure connections across different healthcare systems called the eHealth Exchange [106]. This eHealth Exchange forms a widely distributed system that allows these connected healthcare systems to share electronic health data across the United States.

In addition to this, we also develop a healthcare billing fraud, waste and abuse detection system which can identify fraudulent claims and newly emerging fraud schemes in a timely manner. This helps prevent the monetization of stolen data by filing fraudulent claims to health plans. We also apply our end device security mechanisms to mobile devices as they are increasingly being used to access health data. We make the following contributions in this dissertation:

- We introduce mechanisms that augment accountability and cannot be circumvented as long as multiple independent parties that interact with one another are not compromised simultaneously.
- We introduce sharing provenance which helps us identify the medical practitioner or healthcare organization that may be a source of leak of information or the unauthorized node that fraudulently releases or acquires a particular patient's data.

- We enhance the eHealth Exchange architecture to support awareness over how a particular patient's data is consumed within the distributed network. Our enhancements enable early detection of unauthorized and malicious sharing of health data, which can help limit the resulting damage.

- We develop the FraudScope system which helps detect billing fraud, waste and abuse by analyzing claims data without affecting the current workflow. This helps prevent potential abuse of breached medical information.

- We improve end device security by developing a framework that enforces security policies that govern access to sensitive health data on mobile devices. We also develop a user consent detection mechanism which can help distinguish actual user input from scripted events that can be generated by malware.

A report by the JASON advisory group [50], the prestigious scientific advisory panel to the US government, mentioned these same contributions as requirements to help create a robust health data infrastructure. This serves as an independent validation that the challenges tackled by this dissertation are of great importance to securing the emerging health data infrastructure within the United States.

The rest of this dissertation is organized as follows. Chapter II contains related work within the broader area of health information security. Chapter III describes the healthcare ecosystem which includes details about the existing health information sharing architecture, the US health insurance system and weaknesses within this architecture that need to be addressed. Chapter IV describes our augmented eHealth Exchange with enhanced auditing and awareness that improves the accountability guarantees within the architecture. Chapter V describes our fraud, waste and abuse detection system that helps detect fraudulent claims and prioritizes their investigation to maximize savings. Chapter VI describes our enhancements to secure health data at end devices even if they are compromised. Finally we conclude with Chapter VII.

CHAPTER II

RELATED WORK

We discuss some of the research done in the broad area of health IT security. Past research has studied the vulnerabilities and threats of medical information leakage [51, 115] as well as certain technical countermeasures to them such as anonymization, encryption and access control mechanisms [81]. Work has also been done to create an EHR security reference model for managing security issues in healthcare clouds [122] and to consider security and privacy issues for healthcare applications in wireless sensor networks [2]. A survey of research done in the area of health IT security can also be found here [4].

2.1 Healthcare Identity and Access Management

Prior work has been done to allow a user to access patients' medical data based on attribute-based policies defined by the patient and the healthcare system [68] which allows for selective disclosure and fine-grained attribute-based access control using both static and dynamic attributes. Work has also been done to use attribute based encryption to protect patients' health data and reduces key distribution complexity by dividing the system into multiple security domains [62]. Other work has focused on developing a role based policy specification system for access control of electronic health records in large-scale distributed systems [5]. Additional work to enable patients to selectively share their data with healthcare providers can be found here [6, 40, 46, 90]. The focus of this dissertation is mainly on providing awareness in the presence of security threats.

2.2 Healthcare Auditing

Health information is usually protected by access control mechanisms that are permissive in nature so that needed information is available to medical personnel in case of an emergency. However, this makes it easier for records to be inappropriately accessed. The Health Insurance Portability and Accountability Act (HIPAA) Security Rule requires audit controls that record and examine access in systems that contain or use health information [113]. While this could help address the issue of unauthorized access, research has shown that current auditing standards are lag behind other industries [55], and are inadequate [7,54,56]. This is especially true when nodes can be compromised, and such auditing can be tampered with or disabled. As mentioned, our augmented eHealth Exchange helps address these issues by introducing secure sharing log records that can identify potentially malicious movement of data even when some nodes become compromised.

Research has also been done to offer secure logging without relying on trusted third parties or secure hardware [63] and other work introduces hardware based trust into the log producing application [9]. Different types of log systems have been created to provide strong reliability guarantees for large-scale log collection [88], protect against integrity attacks [96], construct searchable encrypted audit logs [117], save log files efficiently and, at low cost [110], distribute event correlation [39] and effective resource management in a cloud environment [60]. Log Management Systems are also used for a variety of purposes such as recovering from failures in mobile transactions [80] and detecting insider threats [72]. In contrast, our work focuses on augmenting the auditing system within a healthcare enterprise without relying on hardware based trust. We only require one of the interacting entities to not be compromised which allows us to record accurate and non-repudiable information in the logs.

2.3 Healthcare Data Provenance

Medical data usually resides at several different healthcare enterprises and is aggregated to build a comprehensive medical history for the patient or for research purposes. The data consumer requires information about the provenance of the data to assess its quality. Prior research has explored assuring provenance of health information as linked data [71] and digital watermarks [99], and on mobile devices [17,87]. Security [42,97] and privacy [53] issues, as well as tools to navigate and analyze provenance [21] information, have been studied. Mashima and Ahamad [65] use accountability tags to store provenance but rely on the fact that every legitimate consumer will verify the integrity and authenticity of data before using it. Our enhancements do not rely on this assumption. Hasan et al. intercept all applications that try to write to a document and append this application information to the provenance chain of a document; this provenance chain uses signature-based checksums to preserve the integrity of the chain [43]. However, we use append-only mutually signed logs of sharing records to track the movement of a document across nodes within the eHealth Exchange. Other provenance research work can be found in a number of survey papers [28, 93].

2.4 Healthcare Fraud Detection

Prior research has been done to study the principles for combating healthcare fraud [70] and the security assessment of health organizations [48]. We use the recommendations provided by the National Executive Committee which was convened by the American Health Information Management Association (AHIMA) to tackle fraud within the eHealth Exchange to guide our research. Most of the work on detecting fraud has been done on supervised and semi-supervised techniques that require a labeled dataset [61, 84, 120]. These include techniques that are based on neural networks [79], decision trees [118], association rules, Bayesian networks, genetic algorithms, support vector machines [57] and structure pattern mining [119]. In comparison, there has not been much research done within the area of unsupervised healthcare fraud detection techniques [82].

Past research has also explored the prevention of healthcare fraud [70]. Medical identity theft and fraud has been tackled in other work by issuing smart cards [52]. However, such a system is infeasible for use within the US Healthcare system and would cost $814 million to launch. Other work has been done to perform a visual analysis of medical fraud [83] and to tackle privacy issues [49, 76] and prescription fraud [77, 98]. Some research has also been done to assist auditors in detecting fraud [27, 33].

2.5 Securing Health Data on Mobile Devices

Mobile operating systems have traditionally been lacking in providing comprehensive data security. However, the Android operating system includes significant improvements in security over other mobile operating systems. One of these improvements is the notification of all permissions required by applications at install-time to the user. These permission labels grant the application access to phone assets. Even though such a process provides greater transparency, its success largely depends on user understanding of its implications. Issues such as these have led to the emergence of systems such as Kirin [24] - an install-time permission validation service, Saint [78] - a framework for developers to provide install-time and runtime interface policies and Apex [75] - a framework that allows users to selectively grant permissions to applications. However, none of these have focused on our goal of protecting sensitive data against the threats we consider in this dissertation. Other work has addressed how to implement SELinux enforcement policies on Android [92]. This only affects the underlying Linux kernel and such a mechanism cannot mediate inter-component communication at the application layer.

Previous work has addressed how to prevent privacy breaches of medical information from adversaries within the medical domain and outside [30]. Research done by Gardner et al. [31] shares a common goal of protecting medical records on smart phones. However, they focus on creating a cryptographic secret sharing scheme which controls access to medical information. We develop a security policy enforcement framework which helps protect medical data against a variety of threats in a mobile environment.

TaintDroid [23] is an information flow tracking system for the Android operating system. We used its tainting mechanism to specifically taint only sensitive information within an application. We then used its taint tracking mechanism to mediate sensitive data flow if it tries to leave the application by enforcing our security policies. HiStar [121] is another information flow system which provides strict information flow control and prevents private data from being leaked. However, HiStar creates a more restrictive environment for applications by denying any component which reads tainted files access to the network. In our framework, we allow any constrained application's component that reads sensitive information to send this information on the network but only to a subset of known trusted external entities allowed by the security policy.

2.6 Conclusions

In this chapter we discussed research done in the broader area of Health IT security. Work has been done in healthcare security research to address identity and access management, auditing, data provenance, fraud detection and securing data on mobile devices. This dissertation will add to the existing research by securing the emerging health information sharing architecture which is designed to share data across the nation. We will detail this architecture in the next chapter and discuss some of the threats that exist within it. We will then mitigate these threats by augmenting the architecture.

CHAPTER III

HEALTHCARE ECOSYSTEM

In this chapter, we will discuss the healthcare ecosystem in detail. This includes the health information sharing architecture, the United States health insurance process, the threats that exist and enhancements to the architecture to mitigate these threats.

3.1 Health Information Sharing Architecture

Electronic health information will eventually be shared via health information exchanges. Our dissertation helps secure such sharing of health data and can be applied to any architecture that may exist to facilitate this sharing. However, we will discuss one such emerging architecture called the eHealth Exchange and use it as an example in applying our techniques. Our techniques are in no way applicable to only this specific sharing architecture and will work with any other existing health information sharing architecture as well.

The eHealth Exchange, formerly known as the Nationwide Health Information Network (NHIN), is a set of specifications that enable secure exchange of health information over the Internet. It is a highly decentralized network that consists of participating healthcare systems that act as nodes. There is no single central data repository as patient information is retained locally by each node. Nodes share a common transport layer while interacting with other nodes. It is assumed that a public-key infrastructure (PKI) exists with trusted certificate authorities to issue certificates. Nodes will be authenticated using certificates and all messages are encrypted for confidentiality and digitally signed to provide authentication and non-repudiation.

Different types of nodes can exist within the eHealth Exchange. We broadly classify them into two categories: provider nodes and insurance nodes. Other specialized

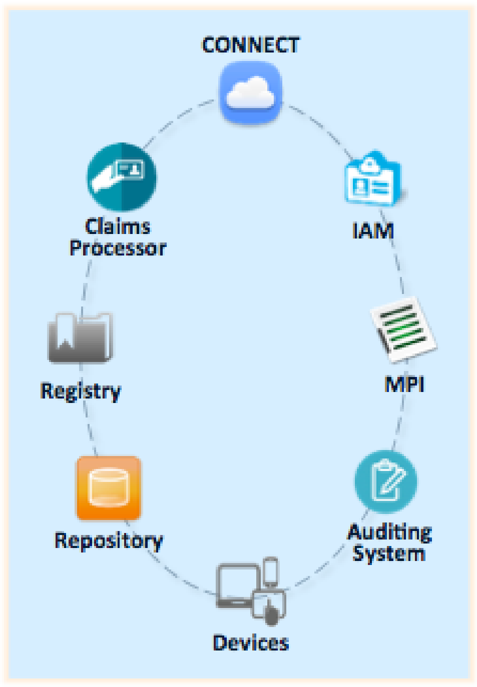


Figure 1: An eHealth Exchange Node.

nodes, such as a clearinghouse, may also exist within this architecture. Provider nodes can contain an EHR system that may be present at healthcare enterprises such as hospitals, clinics or pharmacies. Insurance nodes are run by entities that handle medical claims and billing such as Medicare or health insurance companies. Clearinghouses help route medical insurance claims from a provider node to the correct insurance node which will reimburse the provider for their services. Each node, regardless of type, has the freedom to choose any internal architecture. However, we can represent these possibly unique architectures using a general representation as shown in Figure 1. Under this general model, a Master Patient Index (MPI) contains information on all patients whose data is present at that node. A registry maintains metadata and location of all the medical documents or claims that are contained within the node's repositories. An Identity and Access Management (IAM) system usually exists which authenticates the identity of a user and keeps track of data and actions they are authorized to access and perform. Authenticated users access medical information using various types of end devices. An auditing system is also usually present that records certain actions that are performed within the node. In addition to these, each node has a Claims Processor which either generates or reimburses medical insurance claims. An additional component is also present which follows the standards needed to share medical information with other nodes on the eHealth Exchange. We call this component CONNECT which is the name given to its open-source implementation developed by the Federal Health Architecture (FHA) initiative and its federal partners. Depending on the specific function of the node, other specialized components may exist.

Interactions between different nodes are termed as transactions and the healthcare organization that initiates the transaction is called the initiating node and the responding healthcare organization is termed as the target node. Documents can refer to a patient's medical record, a medical claim filed for reimbursement or any other electronic file used in the medical care process. The initiating node can retrieve documents by sending a *Retrieve Document* request to a target node, which then evaluates the request and, if authorized, returns the documents to the initiating node. In addition, a *Document Submission* request can also be sent which will be evaluated and accepted or rejected by the target node based on a local decision. For example, a medical claim might be submitted from a provider node to the insurance node which processes the claim and makes the payment.

Every node must sign the Data Use and Reciprocal Support Agreement (DURSA) which is a multiparty trust agreement that describes the mutual responsibilities of all participating nodes. It requires them to abide by the Health Insurance Portability and Accountability Act (HIPAA) Privacy and Security Rules at a minimum. These regulatory requirements will apply to end devices used to access sensitive health data. Our framework for end devices can be used to satisfy these regulatory requirements. In addition to this, DURSA requires nodes to notify other impacted nodes of suspected breaches within one hour and confirmed breaches within 24 hours. The affected node must determine the roles of the people involved, other participants likely impacted and the number of records impacted in the breach. However, breaches are not easy to detect, and can potentially go undetected. Next we will discuss the United States health insurance process which is very vital to this architecture.

3.2 U.S. Health Insurance System

Healthcare in America is very expensive so most patients cannot pay for treatment at full cost. Due to this, individuals living in America try to obtain some form of health insurance to cover a portion of their medical expenses. Health insurance in America can be purchased directly through private health insurance companies, through employers who sponsor health insurance programs or public programs sponsored by the government, such as Medicare. Medicare is the federal health insurance program for people who are 65 or older, certain younger people with disabilities, and people with End-Stage Renal Disease [66]. Even though the claims submission process can vary based on the specific insurance type and policy, we will discuss the Medicare claims submission process in detail as it has a very large number of enrollees and we use data from Medicare for our research.

Whenever a Medicare beneficiary seeks medical treatment, the provider can submit the bill directly to Medicare. For providers that participate in the Medicare program, Medicare will reimburse a fixed amount for the service provided and the beneficiary does not need to pay any additional amount or take any further action for the provider to be reimbursed by Medicare. However, providers that do not participate in the Medicare program can charge their own price to the beneficiary beyond the amount that Medicare reimburses them. A non-participating provider usually bills the beneficiary who will pay the charges and then submit a claim for reimbursement to Medicare. In either case, an Explanation of Benefits (EOB) is automatically

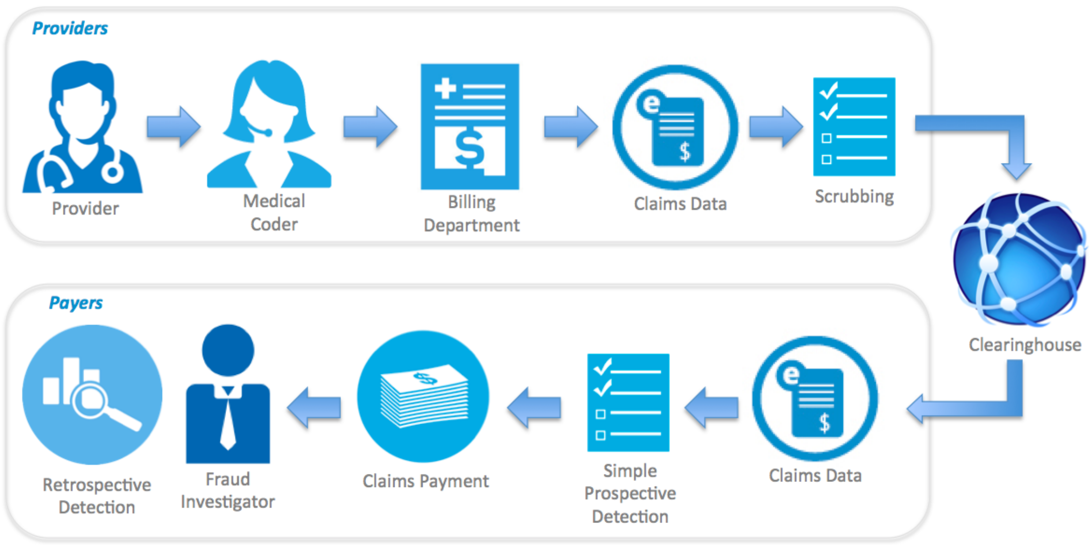


Figure 2: Claims Lifecycle

generated and sent to the beneficiary that details the services that were provided to the beneficiary by the provider and subsequently billed to Medicare.

However, this EOB is sometimes not received by the patient until a few months after the service has been provided. Most of the beneficiaries do not understand the details present within an EOB and there is no financial incentive for them to pay attention to them. Hence, these EOB are not a sufficient means to inform the patient of any malicious activity that takes place using their identity.

In the event that a claim is rejected for reimbursement by Medicare, a letter is sent to the provider giving the details of why the claim was rejected. This allows malicious parties to modify their future claims to ensure that they are not rejected for the same reason. There is no investigation performed by Medicare to discover why such a claim was submitted in the first place.

3.2.1 Claims Lifecycle

We will now discuss the medical claims lifecycle which is used for billing and reimbursement. This information was collected by performing several interviews and meetings with high-level representatives of healthcare provider systems, private and government sponsored health insurance companies as well as those companies that provide billing products to them. An overview of this lifecycle can be seen in Figure 2.

3.2.1.1 Claim Types

There are two main types of medical claims which we describe below:

- *Professional Claims*: An individual claim is submitted by providers when they care for a beneficiary.
- *Facility Claims*: The facility will submit a single claim for each date of service for all the resources used at a facility for a patient. A single claim could include services provided in multiple areas such as a lab, clinic, etc.

Fraud, waste and abuse can occur within both of these types.

3.2.1.2 Provider

When an interaction between a provider and beneficiary takes place, the provider usually makes official notes of the encounter which get forwarded to the medical coder. The medical coder then translates the provider's notes into official diagnosis and procedure codes following the appropriate rules and guidelines. These are then forwarded to the billing department that submits the claims data to the appropriate healthcare plan or payer.

The diagnosis code set is called the *International Classification of Diseases* (ICD) the international standard diagnostic tool for epidemiology, health management and clinical purposes. This code set is maintained by the World Health Organization (WHO) which is updated every 10 years and is currently in its tenth revision. The *Current Procedure Terminology* or CPT is another code set which is used to report medical, surgical, and diagnostic procedures and services to entities such as physicians, health insurance companies and accreditation organizations. This code set is maintained by the American Medical Association (AMA) which updates it annually. Another code set used for procedures is called the *Healthcare Common Procedure Coding System (HCPCS)* which is also maintained by the AMA. HCPCS is used by Medicare and Medicaid. Both CPT and HCPCS must be correspond to a diagnostic code or ICD code on the claim.

There are two main claim formats that use these codes. The first is the UB-04, also known as the CMS-1450 claim form, which is used by institutional healthcare facilities such as hospitals, etc. The other is the CMS-1500 claim form used by non-institutional healthcare facilities, such as private practices. Most claims are filed electronically but a small percentage are still filed manually through paper based forms. This mainly happens when a provider needs to file the claim with a smaller payer that has still not upgraded to an electronic system. Insurance claims processing software follows a set of standards called the HIPAA Transactions and Code Set Rule (TCS).

Before the claims are finally submitted to the healthcare plan or payer for reimbursement, they go through a scrubbing process. This process ensures that the claims follow the coding rules and contains correct information. Errors that could deny the claim or result in undercharges are also detected and removed during this stage. Usually, the healthcare provider system uses third-party software to perform the scrubbing on its claims prior to submission which helps them reduce denials and improve cash flow. Scrubbing rules can be of three types. First are those defined by the healthcare provider system. Second are those that are defined by the medical coding requirements and the final type are those that are defined by the payers themselves.

3.2.1.3 Clearinghouses

Several different claims software programs exist for both the providers and payers which, coupled with the varying insurance regulation in each state, make submission of claims from providers to payers fairly complex. To alleviate this complexity, providers submit their electronic claims to clearinghouses that function as intermediaries between providers and payers. Several clearinghouses exist within the United States which need to be individually subscribed to before a provider can send claims and a payer can receive them. Clearinghouses charge based on the number of claims submitted and larger clearinghouses with more subscriptions usually charge more than smaller clearinghouses. Enrollment into a clearinghouse can take up to 4 weeks before live claims can be submitted.

Sometimes a clearinghouse may forward a claim to another clearinghouse for multiple reasons. This includes situations where the provider and payer are not enrolled in the same clearinghouse, software compatibility issues, etc. This usually hinders the claim from reaching the payer and can delay payment. Clearinghouses also scrub the claims for potential errors before forwarding them to their destination. This helps reduce the the average error rate significantly as compared to paper claims. There is a possibility that claims can be lost or stalled within the clearinghouse which might cause the payer to never receive them. Clearinghouses accept electronic claims in a standard format called the ANSI-X12 837 or EDI 837. The response from the healthcare plan or payer to a claim that includes details on payment, co-pays, denials, etc. are sent back to the provider as an ANSI-X12 835 or EDI 835.

Large payers such as Medicare, Medicaid or BlueCross allow providers to submit claims to them directly to remove the middleman and cut down on fees. However, this can increase the complexity for the provider to deal with multiple payers directly and keeping track of claims within their systems separately. In addition, they lose the benefit of extra scrubbing provided by the clearinghouse to detect errors, a major cause for insurance claims rejection.

3.2.1.4 Healthcare Plans and Payers

Once the claim finally reaches the healthcare payer, it is checked to ensure that it does not contain any errors or violates any of the plan rules. This simple prospective detection is rule-based and checks for items such as:

- Is the beneficiary covered under a plan?
- Is the coverage valid?
- Is the provider alive?
- Is the treatment valid?

Apart from this, certain rule based prospective detection can be applied to check for known scams and red flags within the claim. However, payers are unable to detect fraud, waste and abuse schemes that span over several claims as it is difficult to analyze historical data and compare past claims from the same provider, beneficiary, facility, etc. to identify any trends without delaying the claim payment. Many states have laws that require claims payment within a specified time period after submission and payers can be financially penalized for noncompliance. This is why payers are cautious to move towards a more sophisticated prospective claims review process as it may potentially slow down the payment process. After this rule-based prospective detection is completed, the claims are paid if there are no errors or obvious signs of known fraud.

Periodically, historical claims are analyzed using supervised and semi-supervised techniques to determine if there were any fraud schemes in the claims that were adjudicated. Usually such fraud is detected 60 days after the claims have been paid.

Then the payers have to decide if the losses due to these fraudulent claims are significant enough to divert their limited recovery resources to try and recover the stolen amount. Individual claims that steal $40 or $50 may not warrant the use of these limited resources but their accumulated loss has a significant impact on payers.

3.3 Threats

There are several means by which a malicious actor can abuse the healthcare ecosystem. Even though most cybersecurity threats also apply to this ecosystem, we will primarily focus on those that are most specific to this ecosystem [25].

3.3.1 Medical Identity Theft

Medical identity theft has become one of the fastest growing crimes in America [44]. It occurs when a person steals the medical identity of another individual who is covered by some private or government sponsored health insurance program and obtains medical treatment under the stolen identity. The provided medical treatment generates an EOB which is sent to the genuine owner of the identity. However, beneficiaries do not always understand the details within an EOB and do not have any incentive to verify if the services were actually provided to them. This has made it difficult to discover medical identity theft in a timely manner [1] and patients are usually unaware that their identity has been stolen until their medical history has already been corrupted or they have incurred financial loss. Medical identity theft is estimated to cost the United States about $41.3 billion per annum [86].

3.3.2 Unauthorized Access

Healthcare enterprises have a relaxed access control mechanism that enables health workers to access anyone's medical information in case of an emergency. Such emergencies are called 'Break the Glass' where a particular individual with lesser privileges accesses a record she is unauthorized for in case of emergencies. However, sometimes users of a healthcare system may abuse this relaxed access control to view records of family members or celebrities to leak information to tabloids, cause embarrassment or simply out of curiosity [18]. Such unauthorized accesses usually remain undetected unless the perpetrators are caught using random system audits provided the auditing mechanism itself has not been tampered with.

3.3.3 Phantom Clinics

Phantom clinics are fake clinics that are set up by criminals to defraud health insurance programs [89]. They use stolen identities of doctors and patients to bill insurance companies for treatments that no doctor ever performed and no patient ever received. Such fraud is difficult to detect in a timely fashion and, in many cases, millions of dollars have already been paid by government sponsored health insurance programs such as Medicare and sometimes transferred overseas before it is detected [103]. Government sponsored health insurance programs such as Medicare and Medicaid as well as private health insurance programs (e.g., Blue Cross Blue Shield, Aetna, Cigna) are mainly affected by this threat.

3.3.4 Malicious Insiders

Healthcare professionals usually belong to a larger healthcare provider system such as a hospital, clinic or pharmacy, etc. These entities provide services to patients and file claims to health insurance programs. If a particular healthcare professional that practices within a particular system is caught engaging in fraudulent practices, all of the claims originating within the healthcare provider system could be scrutinized and this could also tarnish the reputation of the facility. Hence, large healthcare provider systems will be interested in ensuring that claims submitted by professionals affiliated with them do not appear fraudulent. In addition to this, claims that contain unusual patterns of coding also need to be identified before submission to reduce the cost and delays of reworking the claims and to ensure quick and accurate reimbursements. There are several types of fraud that can be committed by a malicious insider:

3.3.4.1 Excessive and Unnecessary Services

Excessive services are those that are superfluous to the actual needs of the patient. In this type of fraud, a legitimate provider would provide services that are either excessive or unnecessary so that they may be reimbursed for an amount that is greater than the genuine care that was provided.

3.3.4.2 Upcoding

In this type of fraud, a healthcare provider will enter a higher procedure code than that which was actually performed on the patient in the insurance claim to obtain higher reimbursement.

3.3.4.3 Unbundling

Certain services are usually provided together to beneficiaries and will be reimbursed at a cheaper bundled rate. However, in this type of fraud the provider bills procedure codes separately rather than as a bundle to obtain higher payment.

3.3.4.4 Duplicate Claims

Sometimes a provider may submit the same procedure on the same beneficiary twice for reimbursement to obtain double payment for the service that was performed.

3.3.5 Devices

Medical data is increasingly being accessed and stored by a wide variety of newer devices such as laptops, mobile phones, tablets, etc. that will connect to nodes on the widely distributed eHealth Exchange. As with any device these would also potentially introduce vulnerabilities within the healthcare enterprise infrastructure and increase the number of attack vectors for criminals. Criminals would only need to target the most vulnerable device to gain access to sensitive data within the distributed network and could potentially infect other computing devices on the network and steal the data contained within them as well. Once devices within the network are compromised, criminals could steal the existing data to make fraudulent claims, install malware such as a botnet to monitor activities on the device and on a larger scale within the network, corrupt the data to cause interruptions within the hospital workflow or prevent the genuine users of those devices from accessing the data unless they pay a ransom.

3.3.6 Kickbacks

Kickbacks in the healthcare ecosystem refer to the intentional acceptance of payments, products and services for the purposes of soliciting any healthcare program business. This is considered as fraud and the Anti-Kickback Statute specifically prohibits this type of activity. However, these types of activities take place outside of the health information sharing architecture so we do not directly address this threat in our research.

3.4 Augmented Health Information Sharing Architecture

We will now discuss enhancements to the eHealth Exchange architecture and components to counter some of the threats mentioned in the previous section. In some cases we will augment the architecture with additional components while in other cases we will improve the security within the component itself.

3.4.1 Enhanced Auditing & Awareness

Unauthorized access to sensitive healthcare data and medical identity theft can be detected if we have non-repudiable auditing and awareness of patient data consumption within the eHealth Exchange. If we do not enhance the security of the auditing system then a malicious user may delete the audit logs that record unauthorized access of sensitive healthcare information. In addition to this, if the patient is informed on how their identity and data is used within the eHealth Exchange in a timely manner then they could potentially alert the authorities if some fraudulent activity is taking place. As mentioned earlier, the EOB is sent to the patient several months after the service was performed which does not allow the patient to detect misuse of their identity until extensive damage has been done to their medical record and insurance benefits.

3.4.2 Fraud, Waste and Abuse Detection

To tackle the threat of phantom clinics and malicious insiders that submit fraudulent claims to health insurance programs, we need to enhance the claims processing component to accurately detect suspicious claims in a timely manner. Such fraud is difficult to detect mainly due to the large volume of claims submitted by health care providers and received by healthcare plans. We must also prioritize these claims so a fraud analyst at any of these nodes can investigate those claims which will help maximize savings.

3.4.3 Securing Health Data at End Devices

To reduce the risk of sensitive healthcare data breach from end devices that connect to each eHealth exchange node, we must enhance the existing security framework to guarantee that third-party healthcare applications will not leak sensitive medical information even when they become infected by malware. We must also capture genuine user consent from such devices and differentiate it from malicious scripted actions so that sensitive health data only leaves the device based on the device security policy or explicit user consent.

3.5 Conclusions

In this chapter we discussed the healthcare ecosystem in detail. This included the health information sharing architecture and a general representation of the nodes that are present within it. We then discussed the US health insurance system along with the lifecycle of the claims that are used for billing. This helps us understand how the sharing architecture would be used to submit claims as well as share medical records. We also discussed various threats that affect this architecture and our approach to mitigating these threats by augmenting components within it. We primarily focus on securing the auditing system, claims processor and end devices that connect to nodes within this architecture. Now that we have determined what to augment within the health information sharing architecture, we will discuss each one of these enhancements in detail in each subsequent chapter of this dissertation.

CHAPTER IV

ENHANCED AUDITING & AWARENESS

Health information is usually protected by access control mechanisms that are permissive in nature so that needed information is available to medical personnel in case of an emergency. However, this makes it easier for records to be inappropriately accessed. The HIPAA Security Rule requires audit controls that record and examine access in systems that contain or use health information [113]. This could help address the issue of unauthorized access. Unfortunately, research has shown that current auditing standards in health systems lag behind other industries [55], and are inadequate [7, 54, 56]. This is especially true when nodes can be compromised, and such auditing can be tampered with or disabled. We address these issues in this dissertation by enhancing the auditing system present within the eHealth Exchange.

Audit specifications within the eHealth Exchange exist to establish accountability but nodes are currently not required to follow those specifications to participate within the eHealth Exchange. This specification uses syslog to transport messages between different parties which include communicating nodes or components within a node [47]. Various versions of this protocol try to improve the security by adding TLS and signatures. However, the syslog protocol assumes both of the interacting parties to be trusted and the auditing mechanism could be subverted or corrupted with inaccurate information if any single interacting party was compromised.

We improve the existing accountability guarantees within the eHealth Exchange because the guarantees can be met when at least one of the interacting parties is not compromised. All types of nodes within our enhanced eHealth Exchange generate mutually signed audit records that cannot be subverted or corrupted without detection and are stored with redundancy. This makes it ideal to investigate any abuses because, even if one of the interacting nodes or components was compromised, we would still have the ability to accurately determine the identities that were involved in the fraud. We can also use these records to determine all the other actions that were performed by those identities within the eHealth Exchange which may have also been malicious. Reversing these actions would give us a starting point to rectify any patient's medical history that may have been corrupted by fraud.

We also introduce patient awareness within the eHealth Exchange beyond the limited information that is conveyed by an Explanation of Benefits (EOB) that is delivered to the patient often months after treatment. Our enhancements help inform the patient of how their data is used in a timely manner. Patients are informed of any movement of their data across nodes as well as claims that are filed using their identity. Patients can also be notified of certain actions performed on their data at provider nodes based on their personal preferences. This helps alleviate patient privacy concerns [94] and also helps in early detection of fraudulent actions that may take place within the eHealth Exchange. Once we have our enhancements in place, we can do the following things within the eHealth Exchange:

- Our enhanced auditing mechanism ensures that we log all inter-nodal and inter-component interactions which cannot be subverted or corrupted without detection unless both interacting parties are compromised or malicious.

- Our audit records are redundantly distributed throughout the node and eHealth Exchange which helps in detecting deletion of records.

- We introduce sharing provenance which helps us identify the medical practitioner or healthcare organization that may be the source of a leak of information or the unauthorized node that fraudulently releases or acquires a particular patient's data.

- We can use these records to determine all the other actions that were performed by those identities within the eHealth Exchange which may have also been malicious. Reversing these actions would give us a starting point to rectify any patient's medical history that may have been corrupted by fraud.

- We help alleviate patient privacy concerns by providing greater awareness to patients about how their personal medical information is being shared which can enable early detection of unauthorized and malicious sharing of health data.

4.1 Enhanced Auditing System

We enhance the eHealth Exchange with new architectural components and mechanisms that help prevent certain attacks that can lead to fraud. Our mechanisms help in withholding payments for fraudulent claims which would potentially help reduce financial losses due to fraud. In the event that such a claim does get reimbursed, our mechanisms help in the investigation of the fraud to accurately determine the identities that were involved in perpetrating the crime.

We also augment the eHealth Exchange to provide non-repudiable auditing of interactions that take place across different nodes as well as within each node. Our enhancements ensure that we log all such interactions and detect tampering even if one of the interacting parties is compromised. This auditing mechanism cannot be subverted unless multiple interacting nodes or components within a node are simultaneously compromised, which makes it a valuable tool to investigate malicious actions and the identities that were involved in perpetrating those actions. Once a compromise has been detected, our auditing mechanism can be used to roll-back any

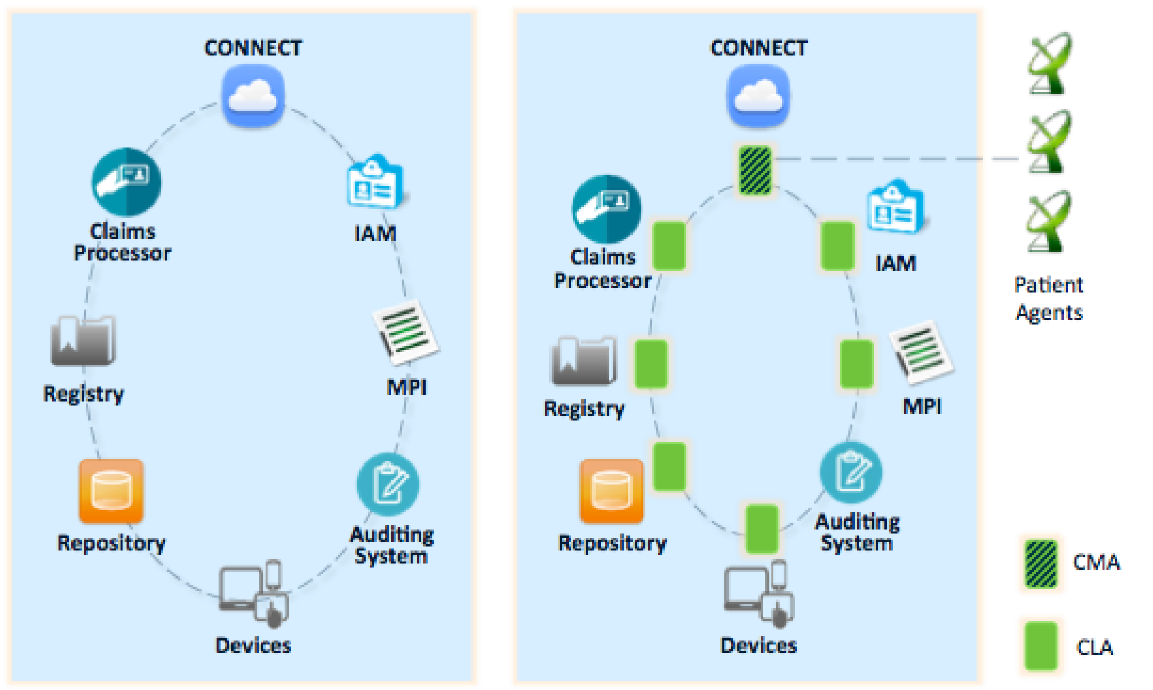


Figure 3: Existing eHealth Exchange Node (Left) and Node with our Enhancements (Right).

corruption in the medical history that may have taken place and also determine all the nodes that may have received the patient's corrupted medical history.

To achieve our goals, we introduce two new types of records called the *Sharing Record Log (SRL)* and *Audit Record (AR)*. These are created, propagated, signed and stored by two additional architectural components, namely the *CONNECT Monitoring Agent (CMA)* and the *Component Logging Agent (CLA)*. Figure 3 gives an overview of our enhancements to the eHealth Exchange Node.

4.1.1 Sharing Record Log

We enhance the eHealth Exchange by introducing what we call sharing provenance which securely records the action that results in the sharing of medical data between different nodes or submission of medical claims by a provider node to an insurance node. Every document and claim within the eHealth Exchange has an accompanying

SRL which contains this sharing provenance information. A *SRL* helps in establishing the medical practitioners and nodes that were involved in the sharing or submission of the associated medical document or claim respectively. In particular, it helps us identify the medical practitioner or healthcare organization that may be a source of leak of information or the unauthorized node that fraudulently releases or acquires a particular patient's data or submits a claim. A *SRL* for a particular medical document or claim is always present at the *CMA* of the node that possesses the document or claim.

The beginning of each *SRL* consists of Metadata (*MD*) and the remainder of the *SRL* consists of a sequence of sharing log entries. The *MD* includes the patient identity information, information about the medical practitioner and node that created the data and how and where the information in the document was collected. The following entries describe the chain of nodes along which it was shared. Each of these entries contains an *AR* and associated signatures that capture the identities and action that resulted in the sharing of the document. A claim is usually only submitted once to the insurance node and will generally contain only one sharing log entry. An *SRL* does not exist when a node first creates a document or receives it from an entity external to the eHealth Exchange. In these cases, the first eHealth Exchange node is responsible for creating the *SRL*, which will only include the *MD*.

A node can share a document with multiple other nodes. We need to handle such cases carefully because a target node does not need to know information about other nodes with which the document has been shared in the past. Thus, whenever a sharing transaction is executed, only a prefix of the *SRL* should be forwarded to the target node. This log prefix contains all entries up to the entry that corresponds to the transfer of the document to the node holding this document. Entries within the *SRL* that describe further sharing that may have occurred will not be forwarded to protect the privacy of those transfers.

Each of these shared document copies that reside in different nodes can be further shared with other nodes which will lead to unique versions of each corresponding *SRL*. These document copies may be merged by a common target node at some point in the future which would require us to merge the *SRL* as well. This would change the linear structure of the *SRL* into a directed graph structure where each vertice represents a different version of the *SRL*. This graph retains information on how the unique versions of the document were shared until they were eventually merged later. Sometimes the *SRL* may need to be truncated as well if they cross a certain threshold size. In such cases, a prefix of the *SRL* excluding the metadata will be truncated leaving sufficient redundancy within the eHealth Exchange to recover the truncated portion if desired.

4.1.2 Audit Record

The AR helps the augmented eHealth Exchange meet the accountability guarantees even when one of the interacting nodes or components is compromised. The general AR structure for all interactions within the eHealth Exchange is described as:

$$< I, U, T, Q, t | H(D), MDorH(SRL_{i,t}) >$$

where I , U and T are the certificates of the initiating party, user that initiated the request and target party respectively. Q is the query that prompted the interaction to occur and t is the timestamp. The remaining values are optional and are only included in certain interactions. Among these optional values are *H(D)* which is a hashed value of document D involved in the interaction and *MD* which is the metadata of that document. *H(D)* and *MD* are included in all intra-nodal interactions that deal with a particular document. An additional optional value is the hashed value of the current *SRL* at node i at time t. *H(D)* and $H(SRL_{i,t})$ are included in all inter-nodal transactions that involve sharing of medical documents. These allow us to verify the document and log copies that existed at the time of sharing. AR that are created

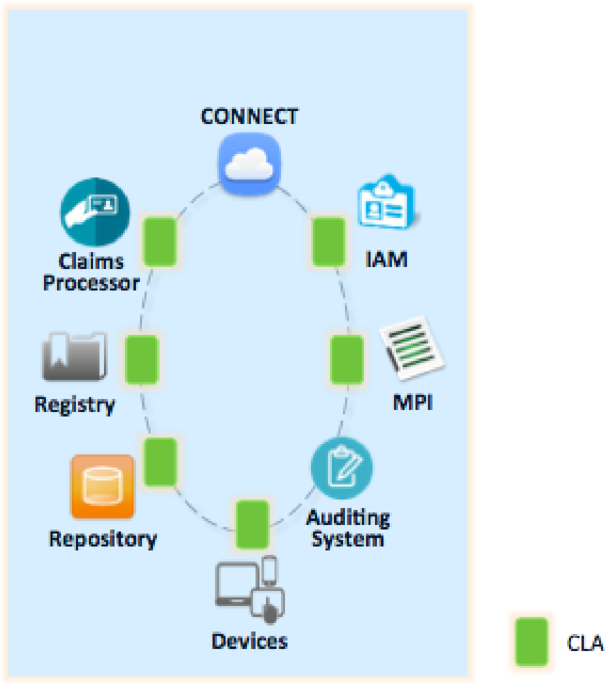


Figure 4: Enhacned eHealth Exchange Node with Component Logging Agents.

for interactions that deal with documents are always generated by the party that possesses the document.

4.1.3 Component Logging Agent

The *CLA* enhances the auditing mechanism within a particular node and does not affect the auditing mechanism for interactions across different nodes. Each component within the node, except for the CONNECT component and existing auditing system, has a *CLA* that mediates all of its interactions with other types of components as shown in Figure 4. Depending on the size of the node, if a component type is distributed, then several *CLA*s may exist for that component type. For every inter-component interaction, the mediating *CLA*s securely generate mutually signed *AR* that describe the interaction. These *AR* are stored securely separately from the auditing mechanism that already exists within the node to prevent a corruption in case the existing auditing system is compromised. The *CLA* does not prevent a

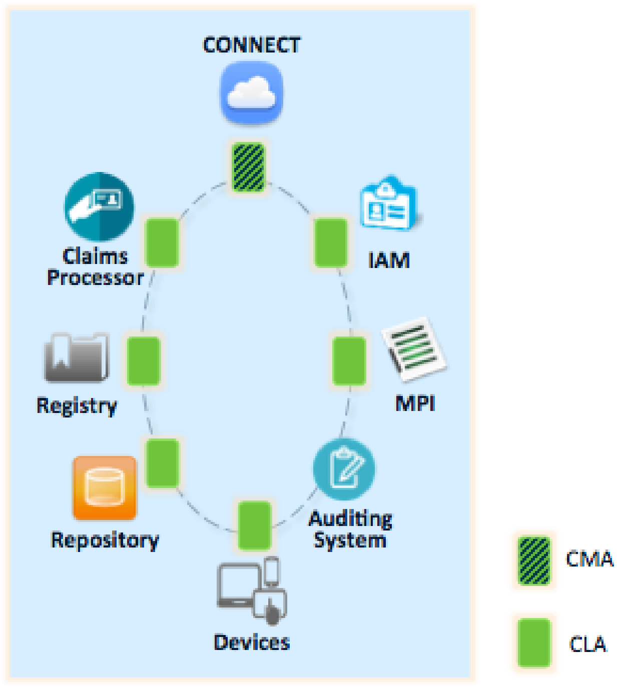


Figure 5: Enhanced eHealth Exchange Node with CLA and CMA.

transaction from proceeding if a valid AR or signed AR is not accompanied with the interaction as its primary purpose is to log such activities. However, such a transaction will generate a notification to further investigate the matter. It is desirable that such a CLA run on a separate virtual machine (VM) having its own certificate, which would make it less likely to be compromised in case the remainder of the node gets compromised. It is also much smaller in size which allows for easier analysis to detect and remove vulnerabilities.

4.1.4 CONNECT Monitoring Agent

As shown in Figure 5, the CMA is present at the CONNECT component of each node within the eHealth Exchange and is involved in enhancing both our inter-nodal as well as intra-nodal auditing mechanism. At the inter-nodal level, the CMA mediates all interactions to securely generate mutually signed AR which describe them and are used to create new sharing log entries for interactions that contain medical documents. The resulting *AR* and *SRL* are then stored at both the interacting *CMA*s. The *CMA* can prevent a transaction from proceeding if a valid *SRL* or signed *SRL* is not accompanied with interactions that contain medical documents. For other interactions that do not contain a valid *AR* or signed *AR*, a notification will be generated to further investigate the matter. At the intra-nodal level, the *CMA* performs the same actions as the *CLA*. Similar to the *CLA*, it is desirable that the *CMA* run on a separate VM having its own certificate and its smaller size allows for easier analysis.

4.2 Patient Data Awareness

Medical identity theft has become one of the fastest growing crimes in America [44], and it is not easy to detect in a timely manner [1]. Patients are usually unaware that their identity has been stolen until their medical history has already been corrupted or they have incurred financial loss. As data is shared within the eHealth Exchange, it could reside in multiple nodes and research has shown that patients want greater control [124] and awareness as to know who has accessed their data [22] at all these locations. Such increased awareness could also help alleviate privacy concerns that exist among patients [94].

We further augment the eHealth Exchange to provide greater awareness to patients about how their personal medical information is being accessed and shared. Our enhancements enable early detection of unauthorized or potentially malicious sharing and accesses of data or submission of claims within the eHealth Exchange as long as at least one of the interacting nodes is not compromised or malicious. This early detection can potentially help contain the resulting damage that may be caused. If unwanted sharing does occur because of a compromised node or due to stolen credentials, we can identify the source of a document leak and the path along which it is shared through the document's *SRL*. This can help in accurately determining

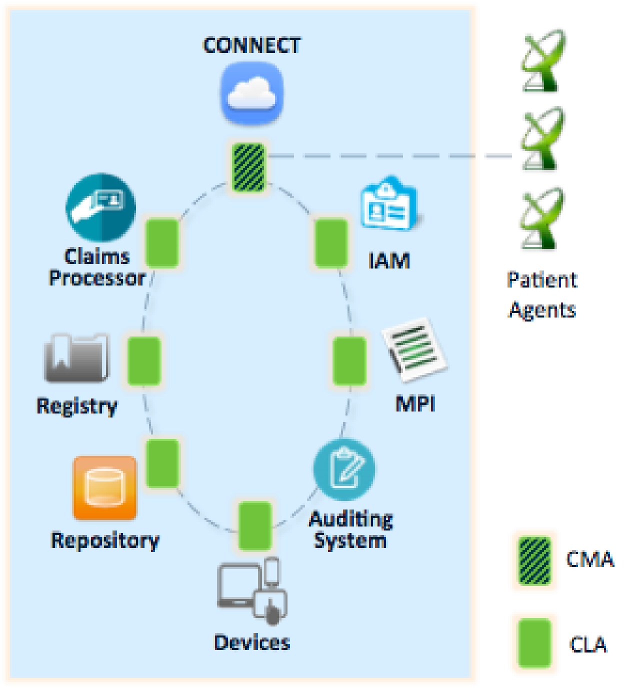


Figure 6: Enhacned eHealth Exchange Node with CLA, CMA and PA.

all the legitimate nodes that have been affected and could serve as a starting point in fixing a patient's corrupted medical history. A corrupted record could include incorrect information about the patient's health that can lead to a life threatening situation. To achieve this goal we introduce a new entity called the *Patient Agent (PA)* within the eHealth Exchange architecture.

4.2.1 Patient Agent

The *PA* is not a part of any single node but several of them exist within the larger eHealth Exchange architecture. Figure 6 shows the updated node architecture. Each patient chooses one particular *PA* to observe the movement of their data within it. This chosen *PA* maintains a list of nodes and medical personnel that have been authorized by the patient to access their data and a profile, which includes information about the events the patient would like to be notified. Notifications can be sent through text messages or email based on the user's preference. When seeking treatment at a new healthcare organization, the patient updates their list of authorized nodes and medical personnel and notifies the healthcare organization of the chosen *PA*, similar to how the patient's insurance information is currently shared. For emergency situations and cases where information about the patient's chosen online agent is not on file, it could be located using demographic information specific to the patient. This is similar to the patient discovery specification of the eHealth Exchange where the patient's demographic information is used to determine the nodes where a given patient's healthcare records exist.

The *PA* notifies the patient of when and who is accessing or sharing their data or submitting claims using their identity. This information is forwarded to the *PA* by the respective *CMA* that possesses the *SRL* or *AR* that was generated after a relevant transaction. If the *CMA* cannot communicate with the *PA*, the *SRL* and *AR* will be forwarded at a later time as soon as communication is restored. This allows the document or claim to be submitted even when communication with the *PA* is disrupted. The resulting patient awareness helps alleviate some privacy concerns with the eHealth Exchange and provides a mechanism for the patient to detect potential malicious access or sharing of their data before significant damage is done to their medical history. It allows patients to play a role in earlier detection of medical fraud. The Centers for Medicare and Medicaid Services already encourages patients to help detect medical identity theft by reviewing their medical bills and Explanation of Benefits to identify services that were not received by them. However, these are sometimes not received by the patient until considerable time has passed after the medical identity theft has taken place. We provide a faster mechanism for patients to determine which healthcare providers and users are viewing their medical information or submitting claims using their identity so they can potentially detect medical identity theft sooner. Such an early detection by a vigilant patient will also eventually lead to financial savings. A *PA* can also be run on behalf of an enterprise where professionals responsible for detecting and handling data abuse will benefit from the notifications that will be generated.

4.3 Implementation

This section describes the details of our enhancements to the auditing system and awareness within the distributed eHealth Exchange. Our decisions have been informed by participating in discussions with key stakeholders that are involved in CONNECT. We have given presentations and attended audio conferences with representatives of various healthcare enterprises, federal and state agencies that are currently using CONNECT to participate in the eHealth Exchange.

4.3.1 Enhanced Auditing System

We implemented our auditing enhancements on the Open Medical Record System (OpenMRS) version 1.9.6. This is as a open source EMR system platform which is in use in at least 23 countries and has become a national standard in several of them. We also implemented our enhancements within CONNECT 3.3, which is an open source software implementation of eHealth Exchange standards. CONNECT was initially developed by federal agencies but is now available to all organizations. Since OpenMRS does not yet implement the eHealth Exchange standards, we configured the Vangent HIEOS (Health Information Exchange Open Source) Document Registry and Repository version 1.2 to work with CONNECT. HIEOS is the chosen document registry and repository component for enterprise-type installations of CONNECT. It provides a repository and registry to facilitate sharing of information across nodes but is not a full-fledged EMR system.

We launched OpenMRS along with four such instances of CONNECT nodes with HIEOS on VM running Debian 6.0.4. These VM were configured to communicate with one another by assigning unique identifiers to each instance. Each of these instances acts as a node within the eHealth Exchange, and two of them implement our enhancements. The VM hosting the initiating node and target node were hosted in different segments of a university campus network. We created a CA using OpenSSL with private keys of length 2048 bits. We then used the Bouncy Castle Java cryptography API to generate certificates along with their public and private key pairs of length 2048 bits for each node within the eHealth Exchange, our introduced components and identities of users within them.

The *CLA* was written in Java and is multi-threaded so that it can simultaneously handle multiple transactions from other *CLA*s. The OpenMRS source code was modified so that all interactions with a component are mediated by the *CLA*. The *CMA* is also written in Java and multi-threaded. The CONNECT source code was modified so that all Retrieve Document and Document Submission transactions are mediated by the *CMA*. The *PA* was created using Java and is also multi-threaded to handle simultaneous connections with *CMA*s and uses its own certificate and keys. All communication between our introduced components are performed securely with SSL. We assume the *PA* and one of the interacting *CMA* or *CLA* to be trusted. We also assume that the keys and certificates have not been compromised and the signing process is authenticated and secure.

4.3.1.1 Audit Record

The *AR* helps the augmented eHealth Exchange meet the accountability guarantees even when one of the interacting nodes or components is compromised. *AR* copies are maintained by the *CMA* and *CLA* that sign the records, and the *PA* chosen by the patient for inter-nodal transactions as well as those intra-nodal transactions that the patient would like to be notified. The general *AR* structure for all interactions within the eHealth Exchange is:

$$< I, U, T, Q, t | H(D), MDorH(SRL_{i,t}) >$$

where I , U and T are the certificates of the initiating node, user that initiated the request and target node respectively. Q is the query that prompted the interaction to occur and t is the timestamp. The remaining values are optional and are only included in certain interactions. Among these optional values are *H(D)* which is a hashed value of document D involved in the interaction and *MD* which is the metadata of that document. *MD* includes the patient identity information, information about the medical practitioner that created the data and how and where the information in the document was collected. *H(D)* and *MD* are included in all intra-nodal interactions that deal with a particular document. An additional optional value is the hashed value of the current *SRL* at the node i at time t. *H(D)* and $H(SRL_{i,t})$ are included in all inter-nodal transactions that involve sharing of medical documents. These allow us to verify the document and log copies that existed at the time of sharing. AR that are created for interactions that deal with documents are always generated by the party that possesses the document.

4.3.1.2 Sharing Record Log

The *SRL* only applies to transactions that share medical documents across different nodes. It consists of a time-ordered linear sequence of log entries. Each log entry consists of an AR and signatures of AR that are generated by nodes that participate in a document sharing transaction. *SRL* copies are maintained at the *CMA* that signs the AR stored in them and the *PA* chosen by the patient who owns the document being shared.

When a 'Document Submission' transaction is executed across two different nodes, the record AR representing the transaction is always generated by the *CMA* of the initiating node, as it possesses the $SRL_{I,t}$ of the document which is needed to generate a new *AR*. A signature for AR is generated by both the monitoring agents at the interacting nodes to make it verifiable. The actions executed by the *CMA* at the two nodes are as follows:

$$CMA_I \rightarrow CMA_T : SRL_{I,t}, AR, [AR]_{kI}$$

$$CMA_T \rightarrow CMA_I : [[AR]_{kI}]_{kT}$$

where CMA_I and CMA_T are the *CMA* at the initiating node and target node respectively, and kI and kT are the private keys of CMA_I and CMA_T, respectively. At the completion of the transaction, the two nodes update the *SRL* maintained by each. A new sharing log entry consisting of AR and the signature generated by both nodes will be appended to the *SRL* of the initiating node. The target node will append the same record and signature to the *SRL* copy it received from CMA_I, which becomes its *SRL* for the newly received document. In a 'Retrieve Document' transaction from one node to another, where CMA_T possesses the document and its *SRL*, the *CMA* at the initiating node I, CMA_I, cannot complete the sharing record AR. Thus, it must make a request to the target node, and the steps executed after such a request are:

$$CMA_T \rightarrow CMA_I : SRL_{T,t}, AR, [AR]_{kT}$$

$$CMA_I \rightarrow CMA_T : [[AR]_{kT}]_{kI}$$

The *SRL* at the two nodes are updated appropriately with the new log entry that contains the record AR and its signature. For any document at an eHealth Exchange node, its *SRL* contains the sharing provenance information for the document. The first sharing log entry in an *SRL* for a particular document provides information about where the document was created, and following entries describe the chain of nodes along which it was shared (each log entry, which contains a verifiable sharing record, has this information).

4.3.1.3 Accountable Data Access

We will now show an example of accountable data access within our augmented eHealth Exchange. We will use the example of a retrieve document request sent from Node A to Node B. We assume that these requests are for medical records that belong to patient P. We use the notation $[AR]_{ki}$ to represent the signature generated for AR with a private key k_i of the CMA at node i (CMA_i) or the CLA at component i (CLA_i).

1. When a physician named John launches a request from his hospital's local EHR system at node A to retrieve a document at node B, the Viewer component at node A must communicate this to the CONNECT component at same node. The CLA at the Viewer, CLA_V, intercepts this interaction and creates AR_1 which includes the certificate of CLA_V, the certificate of the user that initiated the request, the actual query, the certificate of CMA_A and the timestamp t_1. The CLA_V signs AR_1 and forwards this to CMA_A.

$$AR_1 = [CLA_V, Physician\_John, "RetrieveDocument...",$$

$$CMA_A, "2013-04-26\ 05:25"]$$

$$CLA_V \rightarrow CMA_A : AR_1, [AR_1]k_V$$

2. CMA_A verifies that the identities and other information included in AR_1 it receives correspond to the actual request that was received from the Viewer. CMA_A then signs over the signature it receives and returns it to CLA_V. This mutually signed audit record is then stored at both CLA_V and CMA_A for future auditing purposes.

$$CMA_A \rightarrow CLA_V : [[AR_1]k_V]k_A$$

3. The CONNECT component then forwards this request to the CONNECT component at node B. This retrieve document request is then intercepted by CMA_B which creates creates AR_2 and digitally signs it. It sends the signed AR_2 along with AR_2 and the SRL_{B,t_2} to CMA_A.

$AR_2 = [Node\_A, Physician\_John,' RetrieveDocument',$

$$Node\_B, 2013-04-26\ 05:30, H(Document),$$

$$H(SRL_{B,t_2})]$$

$$CMA_B \rightarrow CMA_A : SRL_{B,t_2}, AR_2, [AR_2]k_B$$

4. CMA_A verifies that the identities and other information included in the AR it receives correspond to the actual request that was sent by it. CMA_A then signs over the signature it receives and returns it to CMA_B. A new sharing log entry consisting of AR_2 and the signature generated by both *CMA* will be appended to the *SRL* for the document at both *CMA*.

$$CMA_A \rightarrow CMA_B : [[AR_2]k_B]k_A$$

5. This mutually generated *SRL* entry, SRL_1, is then forwarded to patient P's *PA* by the *CMA* of both nodes A and B and a copy is also stored locally.

The *PA* receives the same *SRL* entry from both the initiating and target nodes for each of the two transactions described above. Upon receiving the *SRL* entry, it will check the list of patient authorized sources of information to ensure that this medical document originated from and was sent to an authorized node. If either of the interacting nodes is not authorized, then a notification is sent to the patient and an investigation could be launched to determine if any malicious activity has taken place.

4.3.2 Patient Agent

Whenever a document needs to be viewed by an internal user within the same node, the viewer component must interact with the repository to request such information. This is captured by the *CLA* at both the repository and viewer and will generate a mutually signed *AR* that represents the interaction. This mutually signed *AR* is then forwarded to the *CMA*. External users that belong to a different node but would like to download a particular patient's data must initiate a document transfer across both nodes to obtain that document. Such document transfers will result in the *CMA* at their respective nodes generating an *SRL* entry that contains a mutual signature. These mutually signed *AR* and *SRL* entries will be periodically forwarded to the respective patient's *PA*. This *PA* will inform the patient of all the users that have accessed their medical document and also allow the patient to determine that their data transfer has taken place between two patient-authorized nodes.

The *PA* does not need to receive the same *AR* or *SRL* entry from both the interacting parties and even if one of the parties was compromised or malicious, it would still be able to generate a user notification based on a single mutually signed *AR* or *SRL* entry it receives from an uncompromised party. In cases where new data has been generated or has been sent to a non-authorized node, the patient is notified of this transaction and can investigate it in more detail.

We assume that more cautious patients will choose to be notified of every movement of their data within the eHealth Exchange and in this case a notification is generated every time the *PA* receives a *SRL* entry for the patient. Other patients may want to receive a notification only when data is shared with a previously unknown node. This is defined by a patient when they sets their profile at the *PA*. As we mentioned earlier, we do not want to impede the movement of health data even if communication between a *CMA* and *PA* is not possible. If both the interacting nodes in a transaction are unable to communicate with the *PA*, we would want subsequent transactions to inform us of any such previously unreported transactions. Hence, the update containing the *SRL* entry can also include information about *SRL* records for all earlier inter-nodal transactions of the current document that are known to the *CMA*. In such situations, the *PA* can also verify that all previous unreported interactions took place between authorized nodes. We can use a variety of existing techniques to ensure that entries in a *SRL* known to both an *CMA* and *PA* are not sent again by the *CMA* in the future. A *PA* can also be run on behalf of an enterprise where professionals responsible for detecting and handling data abuse will benefit from the notifications that will be generated.

4.4 Evaluation

In this section we will evaluate the performance of our enhancements to the eHealth Exchange and also perform a security analysis of them.

4.4.1 Performance Evaluation

We used our implementation of the augmented OpenMRS and eHealth Exchange to quantify additional processing and communication overheads that may arise mainly because of the addition of the components that we discussed and the communication between them. Our study focuses on several micro-benchmarks, as we do not have access to workloads that would allow the evaluation of a fully deployed system at a healthcare organization.

4.4.1.1 Component Logging Agent

When an inter-component transaction occurs, the *CLA* of the initiating component is responsible for contacting the *CLA* at the target component and executing the accountable data access procedure. We evaluated the overhead generated by the *CLA* by measuring the additional time it takes to perform an inter-component transaction. This overhead includes the time to generate the new *AR* at the initiating *CLA* and signing it, communication overhead to send this *AR* to the target *CLA*, verifying that the contents and signatures of the *AR* and mutually signing the *AR* and sending it back to the initiating *CLA*. The *AR* used within our experiment contained only the required values and was 3.18 KB in size. With the inclusion of the optional *AR* elements the size of the *AR* will increase. The signatures that were generated were 256 bytes. The average overhead time that was recorded during inter-component transactions was 367.66 ms. This additional time does not significantly alter end-user experience.

4.4.1.2 CONNECT Monitoring Agent

When an inter-nodal transaction occurs, the *CMA* of the initiating node is responsible for contacting the *CMA* at the target node and executing the accountable data access procedure. To evaluate the overhead generated by the *CMA* within our augmented eHealth Exchange, we measured the time taken to submit and retrieve a *Continuity of Care Document* (CCD), which is a standard electronic document for exchanging patient summary information. Our experiment was conducted using a 265 KB CCD with and without the presence of our enhancements to the eHealth Exchange. We ensured that the environment was consistent for all measurements. In Table 1, we summarize our findings for the average times taken to complete a Retrieve Document and Document Submission transaction between two nodes. We ensured that the existing *SRL* for both types of transactions only contained the document metadata and did not contain any additional *SRL* entries. Our experimental results show that our enhancements add a 0.35 second overhead for Retrieve Document transactions and a 0.37 second overhead for Document Submission transactions in the environment that was described earlier. In a healthcare work flow, we believe this overhead is acceptable. This overhead includes the time to generate the new *SRL* entry at the *CMA* that possesses the document, communication overhead to send this *SRL* entry along with the existing *SRL* to the target *CMA*, verifying that the contents and signatures of the *SRL* entry match with the actual transaction that initiated the accountable data access procedure, and mutually signing the digital signature and sending it back to the initiating *CMA*.

Table 1: Performance Evaluation Results

	eHealth Exchange	**Augmented eHealth Exchange**
Retrieve Document	901.07 ms	1251.80 ms
Document Submission	676.50 ms	1049.96 ms

4.4.2 Security Analysis

We explore the security provided by our augmented eHealth Exchange by analyzing its response to different threat scenarios. Each scenario describes a situation where a potentially malicious action may have taken place. We examine the effect these potentially malicious actions have on our system components and generated logs across different nodes of the eHealth Exchange. We do this by following these scenarios and observing their results on our system.

4.4.2.1 Phantom Medical Clinics

Our first scenario deals with a fake medical clinic, similar to one that was set up by criminals to defraud Medicare of considerable amount of money [19]. These phantom clinics use stolen identities of doctors and patients to bill Medicare for treatments that no doctor ever performed and no patient ever received. Such fraud is difficult to detect in a timely fashion and in many cases, millions of dollars have already been paid by Medicare and sometimes transferred overseas before it is detected. Once the eHealth Exchange infrastructure comes into existence, phantom clinics would need to set up their own eHealth Exchange node and *CMA*, which contains the stolen identities of physicians and patients. These identities are used to file Medicare claims for bogus procedures that never take place by following the 'Document Submission' specification. Since these nodes and their monitoring agents are maliciously set up, they can try to ignore the secure accountability protocol and avoid communicating with the *CMA* of the target node. However, the target node will not allow any transaction to proceed unless the *CMA* of the initiating node sends the *SRL* entry that contains the digital signature. To allow this transaction to proceed, the phantom clinic will be required to follow the secure accountability protocol and include the digital signature within the *SRL* entry that is verified by the target *CMA* to correspond to the requested transaction. The malicious *CMA* of this clinic can then avoid sending the *SRL* entry to the *PA*, as it happens after document transfer. However, if the target node is a genuine node, it will forward a copy of the *SRL* entry that contains the identities of both interacting nodes and the patient. The *PA* will then check if the nodes that participated in the transaction have been authorized by the patient in question, which will reveal the fact that the phantom clinic is unauthorized. This will generate a notification to the patient, which includes the name of the phantom medical clinic. The patient (or person acting on the patient's behalf) can subsequently launch an investigation and notify the target node that the information it received was bogus.

4.4.2.2 Compromised Nodes

As with every other complex software system, the implementation of the eHealth Exchange specifications may contain vulnerabilities that could be exploited to compromise a node. The *CMA* would be more difficult to compromise than the rest of the node software, as it can run in a separate virtual machine and is much smaller in size, which allows for easier analysis to detect and remove vulnerabilities. Apart from this, sometimes a malicious insider can use the eHealth Exchange inappropriately. Such compromised nodes and malicious users may attempt to issue send and receive requests for medical information for patients that were not authorized. Any such request needs to be processed by the *CMA* of the initiating node, and the secure accountability protocol is followed with the *CMA* of the target node. In cases where the *CMA* of the compromised node has also been compromised, it will still be required to include a signature within the *SRL* entry that represents the transaction corresponding to the request that was issued by its node. If the *SRL* entry contents do not match the actual transaction or the secure accountability protocol is not followed by the initiating *CMA*, the transaction will be declined by the target node. This will prevent any loss of privacy and corruption of the respective patient's medical history within the eHealth Exchange. Subsequently, the compromised *CMA* of the initiating node will avoid sending the mutually signed *SRL* entry to the *PA*. However, the *CMA* of the target node will send the *SRL* entry to the *PA* anyway, so long as it is not malicious or compromised itself; the *SRL* entry contains the identities of both interacting nodes. If either of the interacting nodes has not been authorized by the patient to handle their data, a notification for the patient will be generated by the *PA*. Such notifications will be similar to what banks send their customers when malicious transactions are suspected. Thus, we believe that both healthcare organizations and their customers will benefit from such notifications. A patient who receives a notification can launch an investigation, which will lead to an early breach detection at the initiating node and help in fulfilling the breach notification requirements of DURSA. Similar investigations may be launched by other patients whose data may have been affected. This can help determine the role of the person involved in the breach, other participants likely affected, and the number of medical records that may have been breached, which are all requirements of DURSA. We must also note that transactions between authorized nodes will not generate notifications to the patient unless the patient specifically requests notification for every transaction. If the compromised node, malicious insider, or external identity thief initiate transactions between authorized nodes, these transactions may not generate an immediate notification to the patient. However, by requiring that the *PA* periodically notify its owner of all activity known to it from *SRL*, we can ensure that such malicious activity does not go unnoticed for very long.

4.4.2.3 Medical Identity Theft

Medical identity theft usually involves a criminal acquiring medical services in another individual's name. This introduces inaccuracies in the victim's medical history, which could lead to incorrect treatment that can be life threatening. The problem is further compounded by the fact that it is not easy to determine all the parties that received incorrect health information, which makes restoring the victim's correct medical history even more difficult. Whenever a corruption of a particular patient's medical history is detected, the patient can review all the transactions involving their data that have previously taken place within our augmented eHealth Exchange. This information is stored at the patient's *PA*, which is aware of all inter-nodal data exchange transactions that involve the patient's medical data, the users who initiated the transactions and the actual documents that were involved in those transactions. This helps the patients determine the documents that were affected, which is the first step to rectifying their medical history. However, medical documents that contain inaccuracies, which were not shared with other nodes on the eHealth exchange, will not appear on the *PA*'s list. The patient will have to rely on the local audit logs at the nodes to make that determination.

4.4.2.4 Unauthorized Access

Healthcare enterprises have a relaxed access control mechanism that enables health workers to access anyone's medical information in case of an emergency. Such emergencies are called 'Break the Glass' where a particular individual with lesser privileges accesses a record she is unauthorized for in case of emergencies. However, sometimes users of a healthcare system may abuse this relaxed access control to view records of family members or celebrities to leak information to tabloids, cause embarrassment or simply out of curiosity. Such unauthorized accesses usually remain undetected unless the perpetrators are caught using random system audits provided the auditing mechanism itself has not been tampered with.

When a user accesses data within the node's repository, the *CLA* at the device that the user logs into initiates the accountable data access procedure to create an *AR* with the *CLA* at the repository that contains the data. This *AR* contains the certificates of the user that requested the data. This redundantly stored *AR* helps determine the identity of the user and action that was performed even if the entries within the auditing system were removed or one of the *CLA* was compromised. If a 'Break the Glass' situation has occurred with an action that was performed by the user, we flag that particular action for a manual audit to determine if any unauthorized access to data has taken place. Thus, catching any abuse of the relaxed auditing system and not depending on random audits to catch the abuse.

4.5 Conclusions

In this chapter, we augmented the eHealth Exchange to enhance accountability and patient awareness at each node. We introduced two new types of records called the *Sharing Record Log (SRL)* and *Audit Record (AR)*. These are created, propagated, signed and stored by two additional architectural components, namely the *CONNECT Monitoring Agent (CMA)* and the *Component Logging Agent (CLA)*. In addition to this another architectural component called the *Patient Agent (PA)* helps inform the patients of how their data is being shared within the eHealth Exchange.

These enhancements help us ensure that we log all inter-nodal and inter-component interactions which cannot be subverted or corrupted without detection unless both interacting parties are compromised or malicious. Our audit records are also redundantly distributed throughout the node and eHealth Exchange which helps in detecting deletion of records. In addition to this, we also help alleviate patient privacy concerns by providing greater awareness to patients about how their personal medical information is being shared. These enhancements enable early detection of unauthorized and malicious sharing of health data, which can help limit the resulting damage.

In case a breach or compromise occurs, our introduced concept of sharing provenance helps us identify the medical practitioner or healthcare organization that may be a source of leak of information or the unauthorized node that fraudulently releases or acquires a particular patient's data. We can use these records to determine all the other actions that were performed by those identities within the eHealth Exchange which may have also been malicious. Reversing these actions would give us a starting point to rectify any patient's medical history that may have been corrupted by fraud.

Now that we have this enhanced auditing system and patient awareness added to the eHealth Exchange that allows us to detect suspicious activity after it takes place, we would like to limit the damage caused by malicious actions by proactively detecting suspicious activity that may affect the nodes. A major concern of such activity is the submission of claims that contain fraud, waste and abuse within them. Nodes are usually unable to detect such activity in a timely manner due to the large volume of claims that they process and laws that require their quick reimbursement. Hence, in the following chapter we will take proactive steps to detect such activity in a timely and accurate manner. This will help reduce losses due to suspicious claims that may contain fraud, waste and abuse.

CHAPTER V

FRAUD, WASTE AND ABUSE DETECTION

The US government has defined healthcare fraud as knowingly and willfully executing a scheme or artifice to defraud any healthcare benefit program or to obtain any of the money or property owned by any health care benefit program. Sometimes fraud is further divided into subcategories called fraud, waste and abuse. However, we use the term fraud to represent the general abuse of health data for monetary benefit. Perpetrators of fraud could be physicians, hospitals, clinical laboratories, employees of any providers, billing services, beneficiaries, health plan employees, or any person in a position to file a claim for benefits.

Fraud can be committed by individuals acting alone or could be perpetrated by groups of individuals or institutions that use sophisticated schemes to lure customers. Most fraud schemes defraud several private and public sector victims simultaneously rather than target one insurer or a particular sector exclusively [3]. We have already discussed several threats in Chapter 3. However, these threats can be materialized in two different ways. First, short term fraud attempts to steal large amounts of money quickly and stops before anyone can detect the fraud. The second is a longer term fraud that attempts to steal small amounts over a large period of time to avoid detection. Usually legitimate providers that submit claims that can be viewed as fraud follow the second approach while malicious actors that have stolen identities follow the first approach.

In this chapter, we will discuss the FraudScope system and the technique used by it to identify fraud, waste and abuse in the healthcare system. We make the following contributions in this chapter:

- Our fraud, waste and abuse detection system, FraudScope, helps detect suspicious claims and provides a rank and risk score to the most suspicious claims, providers, beneficiaries, etc. to prioritize the investigation of those that are most suspicious.

- FraudScope can be run prospectively to detect fraud before claims are adjudicated. It can also be run retrospectively to detect the most suspicious claims after adjudication.

- Existing methods usually require input in the fraud identification stage, however, our technique does not require any input in the fraud identification stage. This allows our technique to detect newly emerging as well as previously unidentified fraud schemes.

- It can also adapt to natural changes in the treatment patterns or in the coding system and will not require any additional work by the user to include these changes.

5.1 Combating Healthcare Fraud

The Office of the National Coordinator for Health Information Technology (ONC) contracted with the Foundation of Research and Education (FORE) of the American Health Information Management Association (AHIMA) to study how the use of health IT could enhance and expand fraud management. To this effect, FORE convened a cross-industry The National Executive Committee, consisting of a multi-stakeholder group of experts with significant experience and insight about the U.S. healthcare system and fraud, made the following recommendations to tackle fraud within the eHealth Exchange:

- The eHealth Exchange policies, procedures, and standards must proactively prevent, detect, and support prosecution of healthcare fraud rather than be neutral to it.
- Fully integrate and implement fraud management programs and advanced analytics software in interoperable EHRs and the eHealth Exchange to achieve all of the estimated potential economic benefits.
- Data required from the eHealth Exchange for monitoring fraud and abuse must be derived from its operations and not require additional data transactions.
- Fraud management features should not disrupt the provider workflow or interfere with patient care.

In this chapter, we use these recommendations as key goals to secure the eHealth Exchange. The ONC Health Care Anti-Fraud Project defines three approaches to manage fraud. They term the first approach as *Preventive* where ongoing pattern monitoring and analysis is used to prevent fraud. In this approach, an attack on health data is prevented from materializing. The second approach is termed as *Prospective* where fraudulent claims are detected prior to payment at the insurance node. In this case, a successful attack has already taken place on health data and we are able to identify and deny payment for any fraudulent claims that are generated using the compromised data. The final approach is termed as *Retrospective* which identifies fraudulent claims after they have been adjudicated. Here a successful attack has taken place and monetary benefit has also been derived by using it in a fraudulent claim. Such an approach uses historical claims information to determine if any fraud has taken place to prosecute the perpetrators and try to recover the claim payment.

About 80% of the current fraud and abuse detection happens after payments have been made. However, such a detection is still valuable because they identify and store provider patterns and discrepancies. A demonstrable return on investment from the

Table 2: CMS Dataset Details

Details	Dataset 1	Dataset 2	Dataset 3
Data Type	Services and Procedure Data	Services and Procedure Data	Drug Data
Years	2012	2013	2013
Size	1.7 GB	1.7 GB	2.7 GB
Provider Types	89	90	202
Number of Providers	880,644	909,605	808,020
Number of Procedures/Drugs	5949	5983	2737
Data Aggregation	1 year	1 year	1 year
Number of Records	9,153,272	9,287,876 year	23,650,520

use of retrospective fraud and abuse detection is reported by payers [3]. Even though a prospective approach would help stop payment of fraudulent claims, payers are cautious to move towards a prospective claims review process as it may potentially slow down the payment process. Many states have laws that require claims payment within a specified time period after submission and payers are financially penalized for noncompliance.

However, with the Patient Protection and Affordable Care Act which was signed into law in 2010, the Health and Human Services (HHS) Secretary can suspend payments to a particular provider pending an investigation of a credible allegation of fraud. Auditors and investigators will have more opportunities to detect attempts at fraud and prevent financial loss if claims are not rushed through the payment process. This is because the retrospective approach does not provide any large incentives to expend large amounts of money trying to recover an individual claim of \$40 or \$50 [73]. Hence, we must use real-time data analysis techniques to detect fraudulent claims before payment so that fraudulent providers can be quickly identified and expelled from networks [73].

5.2 Datasets

We use a data driven approach to detect suspicious claims within the data which will help us validate our medical insurance fraud claims research. We were able to acquire three datasets for our insurance fraud claims research to validate our approach. All of these health insurance claims datasets were obtained from Centers for Medicare &

Table 3: CMS Dataset Fields

Field	Description
npi	National Provider Identifier (NPI)
nppes_provider_last_org_name	Last Name/Organization Name
nppes_provider_first_name	First Name
nppes_provider_mi	Middle Initial
nppes_credentials	Credentials
nppes_provider_gender	Gender
nppes_entity_code	Entity Code: 'I' for individuals and 'O' for organizations
nppes_provider_street1	Street Address 1
nppes_provider_street2	Street Address 2
nppes_provider_city	City
nppes_provider_zip	Zip Code
nppes_provider_state	State Code
nppes_provider_country	Country Code
provider_type	Provider Type
medicare_participation_indicator	Medicare Participation Indicator
place_of_service	Place of Service
hcpcs_code	HCPCS Code
hcpcs_description	HCPCS Description
line_srvc_cnt	Number of Services
bene_unique_cnt	Number of Medicare Beneficiaries
bene_day_srvc_cnt	Number of Medicare Beneficiary/Day Services
average_Medicare_allowed_amt	Average Medicare Allowed Amount
stdev_Medicare_allowed_amt	Standard Deviation Medicare Allowed Amount
average_submitted_chrg_amt	Average Submitted Charge
stdev_submitted_chrg_amt	Standard Deviation Submitted Charge Amount
average_Medicare_payment_amt	Average Medicare Payment Amount
stdev_Medicare_payment_amt	Standard Deviation Medicare Payment Amount

Medicaid Services (CMS) to evaluate FraudScope. Table 2 gives an overview of the data contained within these datasets.

5.2.1 Medicare Services and Procedures Datasets

The first two datasets are called the 'Medicare Provider Utilization and Payment Data: Physician and Other Supplier' Dataset [16]. They contain aggregated information on services and procedures provided to Medicare beneficiaries by physicians, non-physician practitioners, laboratories, imaging, ambulances, etc. in the year 2012 and 2013 respectively. This amounts to about 9 million records each with around 90 different provider types. Table 3 describes the fields contained within both datasets.

The datasets do not contain individual claim information but are aggregated to the NPI of the performing provider, the Healthcare Common Procedure Coding System (HCPCS) code, and the place of service (either facility or non-facility). For each unique healthcare provider identifier called the National Provider Identifier (NPI),

Table 4: CMS Drug Dataset Fields

Field	Description
npi	National Provider Identifier (NPI)
nppes_provider_last_org_name	Last Name/Organization Name
nppes_provider_first_name	First Name
nppes_provider_city	City
nppes_provider_state	State
specialty_description	Provider Specialty Type
description_flag	Source of Provider Specialty
drug_name	Brand Name
generic_name	USAN Generic Name - Short Version
bene_count	Number of Medicare Beneficiaries
total_claim_count	Number of Medicare Part D claims, including refills.
total_day_supply	Number of days supply for all claims.
total_drug_cost	Aggregate cost paid for all claims.
bene_count_ge65	Number of medicare beneficiaries, aged 65 or older
bene_count_ge65_redact_flag	Flag detailing reason for redaction of bene_count_ge65 field
total_claim_count_ge65	Number of claims, including refills, where the beneficiary was 65 or older.
ge65_redact_flag	Flag detailing reason for redaction of ge65 fields.
total_day_supply_ge65	Number of days supply for all claims, where the beneficiary is 65 or older.
total_drug_cost_ge65	Aggregate cost paid for all claims, where the beneficiary is 65 or older.

there are several records within the database based on the number of distinct HCPCS codes that were billed and the place of service. Any aggregated records derived from 10 or fewer beneficiaries are excluded from the dataset to protect their privacy [14].

5.2.2 Medicare Prescription Drug Dataset

The third dataset is called the 'Medicare Provider Utilization and Payment Data: Part D Prescriber' Dataset [15]. It contains aggregated information on prescription drug events (PDEs) incurred by Medicare beneficiaries with a Part D prescription drug plan in the year 2013. It contains about 23 million records and 202 provider types. Table 4 describes the fields contained within the drug dataset.

The dataset does not contain individual claim information but is aggregated to the NPI of the performing provider and drug name and generic name. For each NPI, there are several records within the database based on the number of distinct drugs that were filled. Any aggregated records derived from 10 or fewer beneficiaries are excluded from the dataset to protect their privacy [13].

5.3 Our Approach

Several machine learning techniques can be considered to achieve our goal of detecting suspicious claims. However, as mentioned earlier we use a data driven approach to determine the most appropriate technique in this section.

Machine learning techniques are usually divided into two categories, supervised and unsupervised. Supervised learning methods require a training set that consists of both fraudulent as well as genuine data which is labeled accurately by a domain expert. This labeling is called the ground truth. Several supervised methods have been used in healthcare fraud detection such as neural networks, decision trees and Bayesian networks. However, obtaining labeled data is extremely difficult and we do not have access to such data. The datasets currently available to us contain unlabeled data that includes a mix of genuine medical insurance claims as well as potentially suspicious claims. Hence, we are unable to use supervised techniques to detect fraud within the previously described datasets.

On the other hand, unsupervised learning methods do not require training data and can be used to analyze data and find outliers within them. We use the assumption that claims filed for each provider type will be similar and providers that abuse medical insurance claims will show deviations from them. We use this data model and explore proximity based machine learning techniques, which is best suited for this data model, to detect abuse within our claims datasets.

The most common ways of defining proximity for outlier detection are cluster-based, distance-based and density-based. We used several variations of these techniques on a single procedure code for each provider type with interesting results. However, when we consider all procedure codes for each provider type, these techniques do not necessarily work well with the resultant high dimensionality data to produce the desired results [123]. Even though there has been a lot of work done recently to perform outlier detection in high dimensionality data, this is still an emerging field where implementations and libraries are not always readily available. We chose to reduce the dimensions of our dataset to detect abuse within it that we describe in the next section. We specifically aim to achieve the following goals with our appraoch:

- Our technique should be able to run prospectively to detect fraud before claims are adjudicated. It should have the capability to run retrospectively to detect the most suspicious claims after adjudication.

- Existing methods usually require input in the fraud identification stage, however, our technique should not require any input in the fraud identification stage. This allows the technique to detect newly emerging as well as previously unidentified fraud schemes.

- Our technique should also provide a rank and risk score to the most suspicious claims, providers, beneficiaries, etc. to prioritize the investigation of those that are most suspicious.

5.4 FraudScope

We developed a technique to detect fraud in medical insurance claims data. However, our technique is in no way restricted to the healthcare domain and can be used generally to detect anomalies in any data. Our technique can be used in conjunction with other prospective and retrospective claims processing systems and techniques. The technique itself can be used to detect suspicious claims, providers, beneficiaries at any granularity. The user of the system could be any individual tasked with reviewing the claims. We will call the user a fraud analyst for the remainder of this document. However, the system can be completely automated and may require minimal user involvement.

Features that can help detect fraud are extracted from claims to create a candidate profile which is compared with a reference profile. The level of conformity between the

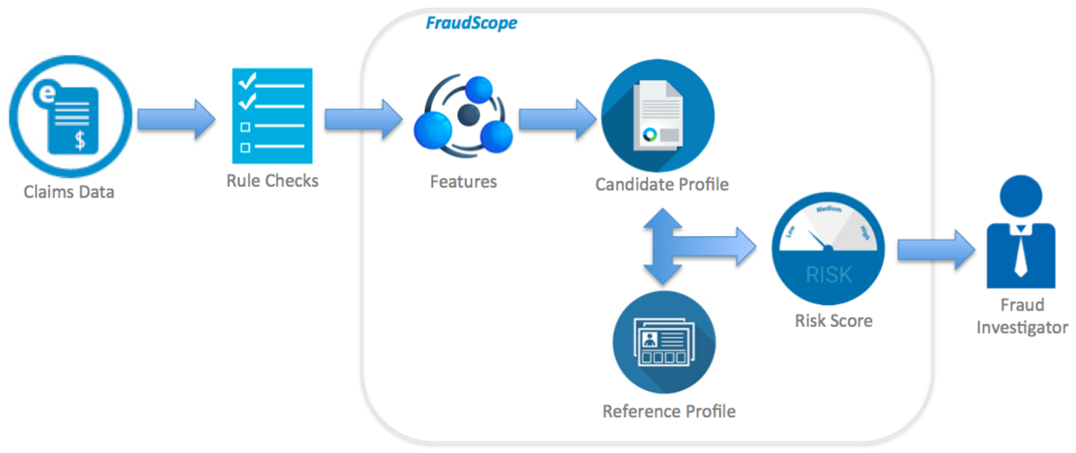


Figure 7: Example Prospective Detection

candidate profile and reference profile will result in risk score that will be presented to the user of the system. This risk score can also be used prospectively to accept or deny the claim automatically without the need for a user to make a decision. An example configuration of our technique in prospective detection is shown in Figure 7.

We will describe a retrospective application of our technique as the datasets available to us are adjudicated claims without any timestamps. Hence, we cannot simulate a live stream from the aggregated datasets available to us. The datasets are aggregated at the provider level so we will apply our technique to detect suspicious providers even though this can be used to detect suspicious individual claims, treatments, beneficiaries, etc. We create a separate reference profile for each provider type based on the medical insurance claims filed by all providers of that type. We then provide a risk score to each of these providers based on the extent of deviation of their generated candidate profiles from their respective provider type reference profile. These risk scores allow a fraud analyst to focus on the most suspicious providers and improve the chances of detecting malicious providers.

Profiles that are created by our fraud detection system are not static. They change as the pattern of claims being filed by a particular provider type changes over time. Our reference profiles are created using an unlabeled dataset so we make an assumption that majority of the providers are legitimate and those that deviate from the majority are likely suspicious in nature. In the presence of a labeled dataset, we can use feedback from the output to fine tune our detection technique.

5.4.1 Features

Features are measurable properties of claims that will enable us to determine the underlying patterns which providers of a particular type naturally follow while performing their duties. Our system will work with both aggregated medical insurance claims data as well as detailed information on every claim that was filed. We use the notation f_i to represent feature i used in calculation of patterns and profiles. The features chosen by our system will depend on the data available to it. For the specific datasets available to us we use the following features:

5.4.1.1 Procedure Codes

The first feature we use are the procedure codes that were filed by each provider within the dataset. A large Medicare and Medicaid fraud case carried out by an organized crime group operating in the US was able to commit fraud to the tune of \$163 million. Over 50 people involved in this scam were able to use stolen identities of doctors and submit fake claims to these government programs for services that were never provided. Some of these claims showed eye doctors doing bladder tests; ear, nose and throat specialists performing pregnancy ultrasounds; obstetricians testing for skin allergies; and dermatologists billing for heart exams [12]. Due to the large volume of claims received by the government, such blatant provider type and procedure mismatch was undetected. We include the procedure code of the claim as a categorical feature in our fraud detection system to automate the detection of such mismatch in the future.

5.4.1.2 Drug Names

The next feature used by our fraud detection system is the drug names prescribed by providers. This is also used as a categorical value within FraudScope. Similar to our previous feature, we would like to ensure that the provider speciality type that prescribes a certain medication is not unusual. This would help a fraud analyst become aware of any suspicious prescriptions.

5.4.1.3 Beneficiaries

Another feature used by our fraud detection system is the number of beneficiaries treated by a particular provider. This feature will be compared to other features such as claim counts, etc. for other providers and the detection system will determine if the provider looks suspicious. For example, a fake clinic that has stolen identities of beneficiaries may bill several claims for a few stolen identities or may have a very large beneficiary base in comparison to other providers. This would mark the particular provider as suspicious within our system.

5.4.1.4 Claim Counts

This feature takes into consideration the number of claims a particular provider files for each procedure and drug code determines if there is anything suspicious based on the number of beneficiaries that received these services.

5.4.1.5 Unique Services

This metric helps identify whether a unique service was performed by the provider on a single beneficiary on a single day. In other words, this helps us determine if a beneficiary receives multiple services of the same type in one day (e.g. single vs multiple cardiac stents on a single day).

5.4.1.6 Drug Supply

Finally, we take into account the total supply for which a particular drug was dispensed. This will help the fraud detector determine if a particular provider is generous in his supply of drugs to patients in comparison to other providers.

In addition to the above mentioned features, several other fields were available to us within the dataset but were not chosen. These include cost based features that do not necessarily indicate whether a provider has engaged in fraud. Medicare allowed amounts and payments for procedures performed on beneficiaries vary based on location as most services amounts are adjusted based on the location's cost of living. An increase or decrease in the intensity of the service performed, deviation from the procedure definition, other services performed during the same visit and a few other factors can also affect the Medicare allowed amounts and payments. Apart from this, if a claim amount seems to be out of the expected ranges the claim is rejected and a letter is sent to the provider detailing the exact reason why it was rejected. This allows malicious providers to conveniently adjust their claims to the Medicare expected amounts and get future claims reimbursed.

Additional fields that contained physician identifying information such as name, address, etc. and other fields that were populated only for a small subset of the data were ignored.

5.4.2 Profile Calculation Window

The calculation window for our profiles define the time period over which claims for a particular provider type are analyzed to create a reference profile and detect deviations from it. Reference profiles may need to be recalculated over certain time periods to account for a natural change in the claims filed by any provider type. The calculation window for all features within a profile are usually the same. The reference profiles could exhibit variations based on the chosen calculation window.

A calculation window that spans a larger time period can help create reference profiles that are not severely affected by any temporary changes in filing pattern during the calculation window. However, certain types of threats can be better detected within a shorter calculation window such as that of a phantom clinic. Phantom clinics usually only operate for a few months and will be outliers if their deviations are measured for the period that they were active. However, that same phantom clinic will blend in with the remainder providers if the calculation window spans a much larger period than when they were actually active.

Hence, we leave the calculation window parameter as an adjustable option to the fraud analyst to detect specific kinds of fraud they may be interested in. The datasets discussed earlier are snapshots of claims filed over an entire year without any associated timestamps. Hence, the profiles created in the experiments discussed will be calculated over claims filed over a period of one year. We use w to describe the calculation window used to determine the reference profile for each provider type t. For example, $[f_1, f_2]^t_w$ means data from features f_1 and f_2 for provider type t over a calculation window of w was used.

5.4.3 Generating Reference Profiles

Now that we have our feature data and profile calculation window chosen, we can generate reference profiles for each provider type within the chosen dataset. We use a statistical technique called Principal Component Analysis (PCA) to generate reference profiles for each provider type. PCA helps reduce high dimensional data by projecting it to new axes on a low dimensional subspace. These axes are called the Principal Components (PC). The number of PCs is generally less than the original number of variables. The first PC captures the largest variance possible on a single axis and the subsequent PCs capture the largest possible variance along the remaining orthogonal directions. This gives us an ordered sequence of PCs with decreasing amount of variance along each orthogonal axis.

To generate our provider type reference profiles we use data from the selected features of the provider type aggregated over the calculation window and run the PCA procedure to determine the resultant n PCs. We use the notation $PC_i^t(w)$ to represent principal component i for provider type t calculated over a window w.

$$PCA([f_1, f_2, ..., f_i]_w^t) \rightarrow PC_1^t(w), PC_2^t(w), ..., PC_n^t(w)$$

Once we obtain the resultant PCs from PCA performed on the data of a particular provider type, we determine the amount of variance captured by each PC. Usually, a small subset of initial PCs capture a majority of the variation of the data which allows the data to have a low effective dimension. These initial k PCs that have a significantly larger variance than the remainder PCs form the reference profile for provider type t in calculation window w which is represented as $p^t(w)$.

$$p^t(w) = PC_1^t(w), PC_2^t(w), ..., PC_k^t(w)$$

These initial k PCs form the normal subspace which explains the predominant normal behavior for t in w and the remainder PCs form the residual subspace $\overline{p^t(w)}$.

$$\overline{p^t(w)} = PC_{(k+1)}^t(w), PC_{(k+2)}^t(w), ..., PC_n^t(w)$$

5.4.4 Calculating Risk Scores and Ranks

Now that we have generated the reference profile $p^t(w)$, we will determine deviations from normal behavior by providers in the dataset. The majority of the providers' behavior can be defined by the normal subspace i.e. $p^t(w)$. However, if we detect a large component that cannot be described in terms of most providers' behavior, then it is potentially suspicious. In other words if a particular provider has a large component in $\overline{p^t(w)}$ then that would indicate that their candidate profile has a deviation from the reference profile that was generated [59,116]. We determine the deviations of each provider candidate profile within $\overline{p^t(w)}$ which helps generate their risk score. Risk

Table 5: Example Claims Filed by Various Dermatologists

Rank	**Claims**	**Field 1 (Categorical)**	**Field 2 (Numerical)**	**Field 4 (Categorical)**
1	Claim A	A1	A2	A3
2	Claim B	B1	B2	B3
3	Claim C	C1	C2	C3
4	Claim D	D1	D2	D3

scores range between 0 and 1000 based on each provider's extent of deviation from the reference profile. These risk scores are then used to rank providers which helps prioritize their investigation. This significantly reduces the time required by a fraud analyst to detect suspicious activity within the data.

5.4.5 Investigation

Once we have obtained risk scores, a fraud analyst would like to have some information on why a high risk score was provided to the claim, provider, beneficiary, etc. We will use an example of ranked claims shown in Table 5 that were filed by various dermatologists to describe this technique. However, this technique can be applied to the extracted features, individual claim field values or aggregated tables that are generated using the raw data. We provide this information to the fraud analyst by checking for any uncommon values that were entered within the claims filed by a particular provider that may have caused them to have a high risk score and rank.

Unusual numerical values within the claims can be detected by using many statistical techniques. However, it is sometimes more challenging to detect uncommon categorical values that exist within the claims. In an effort to provide detailed information on the presence of uncommon categorical values within a specific claim, we examine the presence of every categorical field value individually as well as the combinations of pairs of categorical values within the claim and inform the fraud analyst which categorical values and combinations of them are unusual within the claim. This technique can also be extended to check unusual combinations of categorical values across sequential claims as well. We determine if a particular categorical variable commonly appears within the claim of that provider type by evaluating the commonality function with the following input:

$$c(< provider\_type >, < categorical\_field\_value >)$$

We determine if the presence of two categorical values within the same claim is common by evaluating the following:

$$c(< first\_categorical\_field\_value >, < second\_categorical\_field\_value >)$$

This can be used to determine if the specific procedure is usually provided for a specific diagnosis, etc. and several other combinations of categorical values within the claim. An example application of the commonality function for Claim A in Table 5 is shown below:

$$c(dermatologist, A1)$$
$$c(dermatologist, A3)$$
$$c(A1, A3)$$

To calculate the commonality function we first calculate the joint probability of both the input values. However, since several categorical fields within a claim usually have a high arity where many values could be rare. These rare values can make any combination with other fields to look anomalous. To correct this, we normalize the joint probability of these attributes with the marginal probability of both attributes [20] as shown below:

$$c(a_t, b_t) = \frac{P(a_t,b_t)}{P(a_t)P(b_t)}$$

where a_t is either the provider type or any other categorical value in the claim and b_t is another categorical value in the claim. In cases where a_t is the provider type, then we are checking if it is unusual for that provider type to file a claim that includes b_t. If a_t is also a categorical value, then we are determining if it is unusual for a_t and b_t to occur together. A smaller resultant value of c signifies that a_t and b_t do not usually co-occur naturally and their combination in a particular claim can be used to explain a potential anomaly.

5.5 Implementation

We developed the FraudScope system using the R programming language and created an interactive web application using the Shiny package which helps a fraud analyst to use the system. There were no libraries to execute the FraudScope technique directly so we had to write all the functions to perform the analysis and display the results of our technique. The FraudScope system can be installed at an insurance node such as a private health insurance company or a government sponsored health insurance program such as Medicare. We can also use these techniques at a large healthcare provider to detect suspicious claims before they are submitted for reimbursement.

As mentioned earlier, we use a data driven approach and since the data contains a snapshot of adjudicated claims at a payer, we simulated a health insurance node and evaluated FraudScope in retrospective mode. However, we can also run the system prospectively to maximize savings by detecting fraudulent claims before payment. To simulate this node, we installed our system on a Debian Jessie server and preloaded these databases into the node for use by the fraud analyst. A fraud analyst can connect to our system at any time to generate profiles for every provider type and view suspicious providers.

We will use an example interaction of the fraud analyst with the system to explain our implementation. The fraud analyst decides to check the existing data for suspicious Clinical Laboratories within Dataset 1. The system sets t to 'Clinical Laboratory' and automatically selects the appropriate features for that dataset. Table 6 lists the appropriate features that are used for each dataset for calculating the profiles. We used the PCA algorithm implemented in the R `stats` library to reduce this matrix into several PCs. Apart from this our system does not use any other library for our approach.

We then used the scree plot [11] to determine the effective dimension which is represented by the 'knee' in the PCA curve plotted in Figure 8. The greatest variance

Table 6: CMS Dataset Features Selected

Details	**Dataset 1**	**Dataset 2**	**Dataset 3**
Procedure Codes	Yes	Yes	No
Drug Names	No	No	Yes
Beneficiaries	Yes	Yes	Yes
Claim Counts	Yes	Yes	Yes
Unique Services	Yes	Yes	No
Drug Supply	No	No	Yes

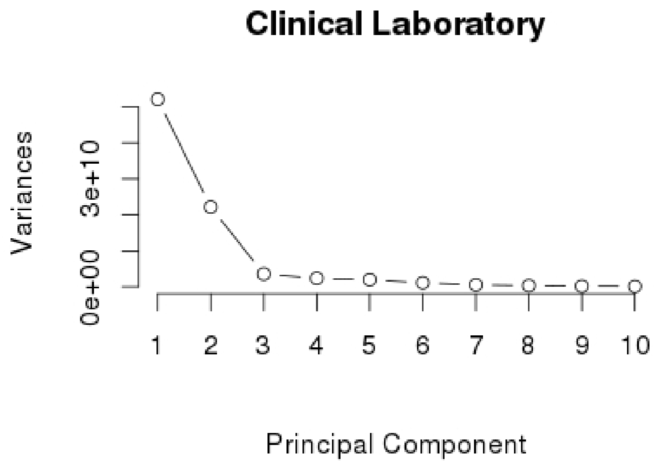


Figure 8: Scree Plot for Clinical Lab

is captured by the PC on the steep slope of the line while the remainder variance is captured by the PC where the slope is flat. The figure shows us that the first 3 PCs capture the greatest variance and hence are used to divide the subspace into p_w^t and $\overline{p_w^t}$. The deviations for each provider were then calculated in R using $\overline{p_w^t}$ that helps us rank the providers based on how much they deviate from the reference profiles. Ultimately, the fraud analyst is shown a sorted list of providers based on their extent of deviation from their established profile.

To evaluate the risk scores and ranks provided by the FraudScope system, we need to compare the results with all known cases of healthcare fraud that have been perpetrated by providers. Unfortunately, such a list does not readily exist for us to directly compare our results. Hence, we had to compile such a list before we could evaluate our output. There are two main approaches we could take to compile this list as discussed below.

5.5.1 Web Scraping

Web scraping is the process of collecting information from the web using software that automates the process. We initially decided to create a web scraper that can compile the list of all providers that are known to have engaged in fraud. We investigated several frameworks that could be used for this purpose. Our findings show that even though web scrapers are great to collect formatted data such as phone numbers, email addresses, etc., data such as names are harder to scrape as they do not follow any particular format. Usually scrapers target consistency of the placement or unique styling of unformatted data for extraction. However, the websites we would like to scrape provider names from do not have any consistent placement or unique styling for their names. This makes it difficult for us to use a web scraper to create a list of all providers known to have engaged in healthcare fraud.

5.5.2 Reverse Scrape

Since scraping the web to create a list of all known fraudulent providers was not feasible, we decided to use the reverse approach - to determine if the specific providers contained within our dataset have engaged in healthcare fraud. We do this by extracting the names of providers contained within our dataset for a given provider type and searching for their involvement in any healthcare fraud settlements or ongoing fraud investigations.

There are two main sources which can help us determine if a particular provider is involved in any ongoing health fraud investigation or settlements. The first is through the United States Department of Justice news [112] and the second is through articles in newspapers. We found the best way to extract information from these sources to be through a search engine as these provide more accurate results compared to most search functions at these sources.

The provider names contained within the datasets include the company suffixes that describe the type of business entity. These suffixes include 'INC', 'LLC', 'LTD', etc. Such suffixes are sometimes not used in the sources mentioned when discussing the same provider. To capture all fraud committed by providers in the dataset, we remove such suffixes from the names of providers before performing a search at these sources.

Google used to provide a Web Search API which would allow us to search Google using a program and retrieve results [36]. Unfortunately, this API was officially deprecated in 2010 and replaced with Google Custom Search [35]. This Custom Search API only allows 100 search queries per day for free and it could take several weeks just to perform queries to search one source for healthcare fraud involvement for each of the clinical labs in the dataset. We discuss our experiments to overcome this limitation in the following subsections:

5.5.2.1 Web Search Results Parser

Our first approach was to search these sources by directly querying Google web search from a program and retrieving the results as a web page. We performed the following query on Google web search:

[site:www.justice.gov "health" "fraud" "First or Organization Name" "Last Name (if any)"]

In the above example query, *site:* is a search operator that helps narrow the results to a specific web source which is the Department of Justice website in this example. The words inside quotes require those terms to be present in the results returned by the search query. In the example shown above, the results must contain the term 'health', 'fraud' as well as the first and last names of the provider or organization name.

We created a program that takes provider names from the chosen dataset and performs the above query on Google. The result is received as an HTML document from which the program extracts the search results and determines if there was any involvement of the particular provider in any healthcare fraud. We were able to obtain results for a few providers until Google blocked our program from making any additional requests. In an attempt to prevent malware from making automated search requests, Google blocks any program from making multiple queries to their website. Unfortunately, this meant we could not use this technique to detect providers who have engaged in healthcare fraud.

5.5.2.2 Bing Search API

Our next approach was to investigate other search engines to overcome the search limitations we experienced. Our best option was Microsoft Bing Search whose results we found to be very similar to that of Google. Bing also provided a search API [67] which allowed for 5,000 free queries per month with no restrictions on the number of queries per day. This seemed like a very feasible solution that removed any prior limitations.

We created an account with Bing and applied for an account key that allowed us to access the search results through the Bing API. We developed a second program which uses a similar query to the one shown previously to perform a Bing search to determine if providers in the chosen dataset have committed any healthcare fraud. The results were returned in an XML format which was parsed by our program and stored for each provider.

While we were collecting results using this program, we realized that the results returned by the Bing Search API were different from the ones returned by Bing web search which was more accurate. The API response contained several results that were unrelated to the search query in comparison to the results returned by the web search directly. In addition to this, not all pages within the Department of Justice website were indexed by Bing and some true positive results were not available through both the API or web search.

5.5.2.3 Google Custom Search API

Our final approach was to use the paid Google Custom Search API [35] to overcome the discussed limitations and determine if a particular provider is involved in any ongoing health fraud investigation or settlements. We created an account and developed a program that performs the query shown earlier to the API which returns the results in JSON format. The program parses the result and stores them for each provider. Since we have to pay a fee based on the number of queries performed, we decided to use only one source. The Department of Justice website is a central source with all health fraud case information contained within it and is the logical source of choice. We use this technique to determine if specific providers are involved in any fraudulent activity.

5.6 Evaluation

We will evaluate our enhancements to the claims processor at the augmented health information sharing node in this section. We will use the datasets discussed previously to validate our fraud, waste and abuse detection system.

5.6.1 Sampling

To evaluate the results of our system we will evaluate its output for a few provider types. As mentioned earlier, there is a cost associated with determining if each provider has engaged in fraudulent activity in the dataset year. In addition to this,

Table 7: Known Fraudulent Clinical Labs

Rank	Risk Score	Amount	Lab Name	Year	Whistleblower
1	999.999	$256 million	Millennium Laboratories Of California	2015	Whistleblower
2	831.971	$71 million	Natural Molecular Testing Corporation	2013	
4	680.802	$47 million	Health Diagnostic Laboratory	2015	Whistleblower
13	146.807	$1.8 million	Quest Diagnostics Incorporated	2015	Whistleblower
31	94.221	$17.5 million	Kan-di-ki, LLC	2013	Whistleblower
41	81.346	Not Reported	Berkeley Heartlab Inc	2015	
60	61.933	$1.5 million	Singulex, Inc	2015	Whistleblower
74	50.840	$4.7 million	Calloway Laboratories Inc	2014	

the results returned by the reverse scraper require detailed manual analysis to validate the output. This involves verifying the following:

- Ensure that the fraudulent provider has engaged in fraud during the dataset year. This involves verifying that the fraud had not ended before the dataset year or started after it.
- Sometimes multiple providers may have the same name. We need to verify fraudulent providers are in the same location as the provider being evaluated within the datasets.
- Verifying that the fraud in question is actually healthcare fraud. The Department of Justice mentions several other types of fraud as well such as tax fraud.
- Reading scanned pdf files of several pages to verify the previous conditions are met.

Due to this intensive manual process and associated cost factor, we were not able to verify all the providers for a specific type but took a sample from the providers that have been assigned the highest risk scores and rank, those with medium risk scores and those with low risk scores. More specifically, we manually examined top 100 ranks, mid 100 ranks and last 100 ranks for various provider types.

5.6.1.1 Clinical Laboratories

We first used the 2012 procedures and services dataset to determine which clinical labs have behaved in fraudulent activity. There are 2,743 independent clinical labs in this dataset. Table 7 shows the clinical labs that are known to be fraudulent within the top 100 ranks along with additional details such as the amount they defrauded Medicare, the year in which they were caught and whether they were caught due to a whistleblower. We will discuss the top three labs mentioned in the Table in detail.

The clinical lab with the highest risk score of 999.999 within this dataset is 'Millenium Laboratories of California, Inc'. Recent news articles from 2015 show that this lab has reached a $250 million settlement for allegations it performed unnecessary tests on Medicare patients for a wide range of drugs that resulted in inflated bills [108]. Even though Medicare performs several levels of checking through various contractors, this lab was never caught by any of their fraud, waste and abuse detection systems. This fraudulent activity was able to continue until a whistleblower contacted authorities in 2015. If the FraudScope system was used by CMS to detect suspicious claims we could have detected this activity in a timely manner and reduced the millions of dollars in losses due to fraudulent claims submitted by this lab.

The second lab on this list is 'Natural Molecular Testing Corporation' which performed genetic testing used to determine response to medications called pharmacogenomic testing. After being paid tens of millions of dollars by Medicare it detected potential fraud such as billing patients for tests to determine a genetic sensitivity to warfarin, a blood thinner, when the patients were not on the drug. Natural Molecular filed for bankruptcy in October 2013 after the justice department began a criminal investigation [102].

The third lab 'Health Diagnostic Laboratory' recently agreed to pay $47 million dollars for violating the False Claims Act by billing Medicare for medically unnecessary testing. This fraud was uncovered thanks to whistleblowers who notified the appropriate authorities. However, our technique can detect their suspicious behavior purely based on analytics without the need for whistleblowers [107].

Table 8: Known Fraudulent Chiropractors

Rank	**Risk Score**	**Lab Name**
1	999.999	Sophia Lin
39	341.559	Mace Richter

There were no known fraudulent labs present within the mid 100 ranks and bottom 100 ranks provided by our fraud, waste and abuse detection system. This particular scenario which starts from the preparation of data, performing the dimensionality reduction, generating profile and detecting deviations takes 5 minutes and 47 seconds on a multi core server. It would take about 9 minutes and 11 seconds if performed in a linear manner. These timings are acceptable for the magnitude of the data used to calculate the rankings.

5.6.1.2 Chiropractor

For our second example, we used the 2012 procedures and services dataset to determine which chiropractors have engaged in fraudulent activity. There are 36,399 chiropractors within this dataset. Table 8 shows the chiropractors that are known to be fraudulent within the top 100 ranks. Sophia Lin who had the highest risk score according to FraudScope was caught along with three others for having committed $4 million Medicare fraud in 2015. Mace Richter who was also assigned a high rank was caught that same year and faces multiple insurance fraud charges for allegedly billing insurance companies for procedures and visits that never happened. Our system is able to detect their fraudulent activity using the 2012 dataset. There were no known fraudulent chiropractors present within the mid 100 ranks and bottom 100 ranks provided by our fraud, waste and abuse detection system.

5.6.1.3 Neurology

For our third example, we used the 2013 drug dataset to determine which neurologists have engaged in fraudulent activity. We intend to show that our technique works with both procedure as well as drug datasets. There are 12,194 neurologists within this dataset. There is only one neurologist known to be fraudulent within the top 100 ranks of this dataset. This is Gavin Awerbuch who has been provided the highest risk score of 999.999 and ranked number 1. According to the U.S. Attorney's Office, Dr. Awerbuch has been accused of defrauding Medicare of $7 million and prescribing an excess of the cancer painkiller Subsys. There were no known fraudulent neurologists present within the mid 100 ranks and bottom 100 ranks provided by our fraud, waste and abuse detection system. Similar results can be discussed for any other provider type within the three datasets available to us.

5.6.2 Detailed Analysis & Comparison with Government Technique

Since there is a cost factor associated with the number of providers we check for healthcare fraud, we will only be able to determine the detailed analysis of FraudScope results for one dataset. We choose the 2012 dataset as most fraud detection happens retrospectively [38] and this dataset from an earlier year would allow the Office of Inspector General (OIG), which is responsible for identifying and investigating fraud in government healthcare programs, greater time to identify and investigate fraud cases since that year. This will make it easier for us to identify true positives in the dataset.

The dataset we have chosen contains 89 different provider types where each provider type has a varying number of providers. For example, Internal Medicine has 91,525 providers while Family Practice has 77,792 providers. As mentioned earlier, there is a cost associated with the number of providers we check for healthcare fraud. In addition to this, we also want to manually inspect the results for any false negatives.

Due to these restrictions, we choose the Clinical Laboratory provider type which contains 2743 members which is a modest number to evaluate the results provided by FraudScope. FraudScope analyzes the aggregated medical insurance claims data for all these labs and generates a reference profile that represents the normal claims filing pattern by loading the appropriate features and performing the necessary calculations. It then generates a risk score for each lab based on their extent of deviation from the generated profile for clinical labs. This risk score is then used to rank all the clinical labs contained in the dataset. We then used the reverse scraper to determine which of these labs are known to have engaged in healthcare fraud.

We need to keep in mind that not all healthcare fraud has been uncovered or detected [37] so it would not be unusual to have providers who have no currently known fraudulent activity to be provided a higher rank. However, all providers with known fraudulent activity should be provided higher ranks. In other words, no known cases of fraud should be provided a low rank by our system.

Our testing program performed 2743 queries using Google Custom Search API to determine if any of the clinical labs within the dataset were involved in healthcare fraud. The results obtained from the Department of Justice website were then stored for each provider. These results were also manually inspected for any false negatives.

During our testing, we noticed that some large clinical laboratories operated under several NPIs. Initially we considered combining all such clinical labs into one organization for testing purposes. However, a large clinic may use a different NPI for each branch or location and not every location may be engaged in fraudulent activities. By keeping them separated, we can actually determine which specific locations may be engaged in fraudulent activity.

The news articles from our sources do not provide the specific NPI or address involved in healthcare fraud and hence our testing program cannot determine which specific location was engaged in fraud. Therefore, even if one location was engaged in fraud, the reverse scraper will mark all NPIs of the same organization as having committed fraud.

Table 9: Classification

Output	**Known Fraudulent Providers**	**Other Providers**
High Rank	True Positive	False Positive
Low Rank	False Negative	True Negative

To account for this in our testing, we consolidate all such records within the results into a single record with the highest rank that was provided for the clinical laboratory. For example if an organization had used three NPIs which were provided ranks 30, 60 and 120, our result will list that organization as having rank 30. In an actual installation of FraudScope, a fraud investigator can investigate every location of the organization independently as the claims will detail which NPI had filed them even if their names were identical. Since our technique outputs a ranked risk score rather than classifying fraud or legitimate transactions, we clarify our terminology in Table 9. We will now analyze these results to evaluate the output of FraudScope.

5.6.2.1 False Positives and True Negatives

Not all fraud committed by healthcare providers is currently detected or reported [37]. Due to this, we do not posses the ground truth (i.e. an actual indicator of whether a particular provider has engaged in fraud). We are only aware of a small number of fraudulent providers that have been exposed, mainly by whistleblowers. Hence, the other providers that are not currently known to have committed fraud could still be potentially engaged in fraudulent activity. Due to these reasons we cannot make any definitive statements about any false positives or true negatives in the results. We naturally expect to see several false positives which could be exposed in the future for having committed fraud.

5.6.2.2 False Negatives

The presence of false negatives in the FraudScope results will show that the results are not accurate. In other words if we find that known fraudulent providers are provided low ranks, this will show that the FraudScope provides inaccurate results. Table 10

Table 10: Detailed Analysis of Results

Rank	Provider Name	Risk Score	Note
1	Millennium Laboratories Of California, Inc.	999.999	**FRAUD**
2	Natural Molecular Testing Corporation	831.971	**FRAUD**
4	Health Diagnostic Laboratory, Incorporated	680.802	**FRAUD**
12	Laboratory Corporation Of America Holdings	146.807	Old Case - Fraud in 1997
13	Quest Diagnostics Incorporated	142.594	**FRAUD**
31	Kan-di-ki, LLC	94.221	**FRAUD**
41	Berkeley Heartlab Inc	81.346	**FRAUD**
60	Singulex, Inc	61.933	**FRAUD**
74	Calloway Laboratories Inc	50.840	**FRAUD**
85	Spectra Laboratories Inc	46.632	Old Case - Fraud in 1996
91	Dianon Systems Inc	42.986	Old Case - Fraud in 2007
96	Bostwick Laboratories, Inc.	41.590	Unrelated - Illegal Payments
140	Biodiagnostic Laboratory Services, LLC.	28.859	Unrelated - Bribes
187	University Of Miami	19.998	Unrelated - Medicaid fraud in 2005
196	Metwest Inc	19.494	Old Case - Fraud in 1997
386	Yale University	8.591	Old Case - Fraud in 1998
468	Coastal Medical Inc	6.576	Name Mismatch
490	Quality Home Health Care	6.078	Name Mismatch
494	State Of Connecticut	6.010	Unrelated - Wire fraud
538	Johns Hopkins University	5.323	Unrelated - Use of tainted syringes
715	The Trustees Of Columbia University	3.413	Unrelated - Federal grant fraud
789	Omni Healthcare PA	3.001	Old Case - Fraud in 2006
821	Clinical Laboratory, Inc.	2.845	Name Mismatch
866	Empire Clinical Laboratory, Inc.	2.610	**FRAUD**
980	Highland Medical Center	2.244	Unrelated - Tax Fraud
1017	Acculab Inc.	2.144	Old Case - Fraud in 2008
1065	Polaris Allergy Labs, Inc.	2.014	Unrelated - Fabricating test results
1110	Southern California Permanente Medical	1.925	Old Case - Fraud in 2002
1163	Connecticut Oncology & Hematology LLP	1.860	Name Mismatch
1446	University Of Alabama At Birmingham	1.666	Old Case - Fraud in 2005
1471	University Medical Center Inc	1.661	Old Case - Fraud ended in 2010
1591	The Medical College Of Wisconsin, Inc.	1.652	Unrelated
2258	Community Health & Emergency Services, Inc	1.638	Name Mismatch
2304	Royal Inc	1.635	Name Mismatch
2408	Guardian Angel Home Health Care, Inc	1.622	Unrelated - Kickback
2430	State Of New Mexico	1.617	Unrelated - Medicaid Fraud
2452	Trinity Health	1.613	Name Mismatch
2502	Emory University	1.601	Old Case - Fraud ended in 2010
2536	Nova Southeastern University Inc.	1.587	Old Case - Fraud in 1999
2586	University of Louisville	1.567	Old Case - Fraud ended in 2010

shows the providers that are involved in healthcare fraud based on the output of our reverse scraper. We also included the rank and risk scores provided by FraudScope and ordered it from high risk score to low risk score.

We manually inspected these results and noticed that several of these providers had engaged in fraud that had ended prior to the dataset year which is 2012. Such providers who had stopped their fraudulent activities prior to 2012 will not be detected by FraudScope. In addition to this, several results from our testing program were for labs that were named similar to the provider in question. We also identified such labs and mentioned it within the notes. There were also some unrelated cases that cannot be detected from the data presented to FraudScope. For example, fraud committed by filing fraudulent claims to Medicaid, tax fraud, use of tainted syringes to spread disease, etc.

The lowest ranked fraudulent case found within the results is 'Empire Clinical Laboratory, Inc.' which is ranked at 866 out of 2743 clinical labs by FraudScope. Unlike other fraud reports, there are very little details available about this particular case as it was recently reported. However, from the little details that have been made available, there could be a few reasons why this particular lab did not deviate much from the generated profile and was not provided a high risk score and rank by FraudScope:

- This fraud scheme involved five locations which filed fraudulent claims to both Medicare and Medicaid programs. It's possible that this particular clinic location participated mainly in Medicaid fraud which is why FraudScope did not find anything suspicious in the Medicare data that was provided to it.

- The fraud scheme began in 2006 but the online reports do not mention which year their fraudulent practices ended. It is possible that this scheme ended prior to the year in which the claims data was aggregated for the dataset similar to a few other fraud cases.

- As per the Department of Justice report, this scheme involved kickbacks and medically unnecessary tests. Such fraud cannot be directly detected from the data provided to FraudScope as the aggregated dataset does not contain any diagnostic codes to determine if the tests were necessary or not. If FraudScope were provided detailed claims data, then such unnecessary tests would be detected.

All the other known fraud cases are ranked 74 or higher which forms the top 2.7% of the most suspicious labs in the dataset according to FraudScope.

5.6.2.3 True Positives

There are 880,644 providers in the chosen dataset while the selected provider type contains only 2743 providers. Out of these, nine providers are known to have engaged in healthcare fraud during the same time period over which the data was aggregated. All of these providers with the exception of one are at the top 2.7% of the high risk scores provided by FraudScope which would prioritize their investigation by a fraud analyst.

5.6.2.4 Government Technique - Office of Inspector General

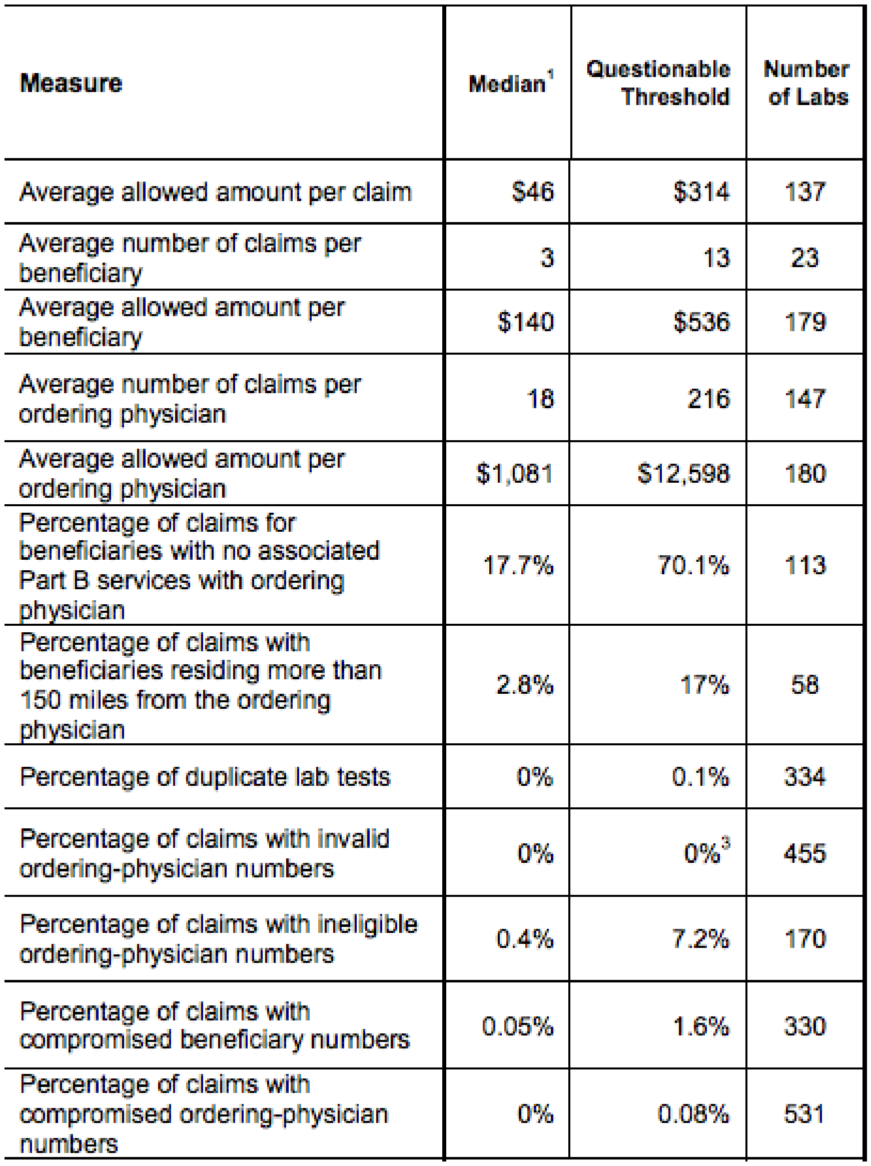
Now that we have obtained detailed results from our technique we can compare this output with that of the government technique. Public health insurance programs such as Medicare and Medicaid are protected by the Office of Inspector General (OIG) which has the responsibility to identify and investigate fraud, waste and abuse within its entrusted programs. The OIG has published a few reports publicly that allow us to determine some of the techniques used by them to detect fraud [105]. They perform fraud detection within medical insurance claims separately for each provider type. Several measures are developed for each of these provider types that help identify several different types of possible abuse. These measures are designed to identify providers with questionable billing who are outliers when compared to their peers. The measures are developed after consultation with officials from CMS, interviews with experts in the field being examined and the OIG's own analysis.

The chosen measures for each provider type could include general information such as average number of services provided per day or per beneficiary per visit, etc. They could also be very specific, such as scrutinizing claims for procedure codes that result in a higher payment which are usually targeted for upcoding and possibly other abuses, providing treatment more frequently than the recommended dosing guidelines, unusually high billing for complex procedures and using specific modifiers on the claim that will result in a higher payout, etc. Construction of these measures requires knowledge on how billing is performed by the provider type and how such billing can be abused. Also, such measures can only capture abuse which is already anticipated by the OIG and newer avenues of abuse will not readily be detected unless a chosen measure was designed to capture it.

Once the measures have been chosen, the OIG uses the Tukey method to determine the outliers for each measure separately [111]. Tukey's method helps identify outliers in data by dividing the data into quartiles. A quartile is a division of the observed values in the dataset into four intervals each of which contains 25% of values in the data. The values demarcating these quartiles are referred to as Q_1, Q_2 and Q_3 where Q_2 is the median of the data. The Interquartile Range (IQR) is then calculated to be (Q_3 - Q_1). The outliers in the data are then calculated to be $(Q_3 + (IQR * 1.5))$ and the extreme outliers are calculated to be $(Q_3 + (IQR * 3))$. The OIG uses either the outlier or extreme outlier defined by Tukey on the values for a particular measure to determine if the provider in question has engaged in questionable billing.

The outlier determination using these techniques does not provide conclusive evidence that the providers have engaged in questionable or improper billing. Some providers may be highly specialized or offer complex care that may be a legitimate reason for exceeding the Tukey thresholds on the measures that were defined. This technique only identifies providers who warrant further scrutiny.

Figure 9 shows the measures used by OIG to detect suspicious clinical laboratories [104]. We can see from this that some measures can have as high as 531 labs to be investigated in an unranked manner. In comparison, FraudScope is able to identify all known fraudulent labs in the top 74 highest risk scores and ranks which successfully prioritizes the most suspicious labs while reducing the search space. We can provide a similar description for each provider type but due to the cost and time associated with the evaluation process we have only detailed one provider type.



Measure	Median[1]	Questionable Threshold	Number of Labs
Average allowed amount per claim	$46	$314	137
Average number of claims per beneficiary	3	13	23
Average allowed amount per beneficiary	$140	$536	179
Average number of claims per ordering physician	18	216	147
Average allowed amount per ordering physician	$1,081	$12,598	180
Percentage of claims for beneficiaries with no associated Part B services with ordering physician	17.7%	70.1%	113
Percentage of claims with beneficiaries residing more than 150 miles from the ordering physician	2.8%	17%	58
Percentage of duplicate lab tests	0%	0.1%	334
Percentage of claims with invalid ordering-physician numbers	0%	0%[3]	455
Percentage of claims with ineligible ordering-physician numbers	0.4%	7.2%	170
Percentage of claims with compromised beneficiary numbers	0.05%	1.6%	330
Percentage of claims with compromised ordering-physician numbers	0%	0.08%	531

Figure 9: OIG Measures to Detect Fraud

Table 11: Comparison with Normal Subspace using k-means clustering

Residual Rank	**Normal Subspace k-Means**	**Lab Name**
1	1782	Millennium Laboratories Of California, Inc.
2	2065	Natural Molecular Testing Corporation
4	1687	Health Diagnostic Laboratory, Incorporated
13	1526	Quest Diagnostics Incorporated
31	1235	Kan-di-ki, LLC
41	106	Berkeley Heartlab Inc
60	1727	Singulex, Inc
74	1795	Calloway Laboratories Inc
866	51	Empire Clinical Laboratory, Inc.

Table 12: Comparison with Normal Subspace using LOF

Residual Rank	**Normal Subspace LOF**	**Lab Name**
1	74	Millennium Laboratories Of California, Inc.
2	56	Natural Molecular Testing Corporation
4	82	Health Diagnostic Laboratory, Incorporated
13	1366	Quest Diagnostics Incorporated
31	543	Kan-di-ki, LLC
41	318	Berkeley Heartlab Inc
60	57	Singulex, Inc
74	172	Calloway Laboratories Inc
866	664	Empire Clinical Laboratory, Inc.

5.6.3 Comparison with Normal Subspace

We performed several experiments to compare the results achieved by our residual subspace detection technique with techniques that use the normal subspace. We provide the results for a few of them in this subsection. In Table 11 we compare the residual subspace results with those generated by using k-means clustering on the normal subspace. In this experiment, we try to create a cluster that represents the reference profile for the independent clinical lab provider type using the k-means algorithm. We then determine the distance of each of the labs from the centroid which is used to determine their deviation from the reference profile and risk score. From the table, we see that the fraudulent providers are scattered throughout the ranks and do not comprise of the highest risk scores which shows that this technique does not work well in the normal subspace to detect fraud, waste and abuse in medical insurance claims.

Similarly, we compared FraudScope's results with those generated by using the Local Outlier Factor (LOF) algorithm on the normal subspace. The results for the

Table 13: Comparison with Normal Subspace

Residual Rank	Normal Rank	Lab Name
1	238	Millennium Laboratories Of California, Inc.
2	259	Natural Molecular Testing Corporation
4	129	Health Diagnostic Laboratory, Incorporated
13	14	Quest Diagnostics Incorporated
31	1	Kan-di-ki, LLC
41	2588	Berkeley Heartlab Inc
60	2507	Singulex, Inc
74	2302	Calloway Laboratories Inc
866	2465	Empire Clinical Laboratory, Inc.

Table 14: Commonality values for Steven Schall

Commonality Value	Provider Type	Code	Description
0.00021	Diagnostic Radiology	92014	Eye exam & treatment
0.00026	Diagnostic Radiology	92004	Eye exam new patient
0.00033	Diagnostic Radiology	92012	Eye exam established pat
0.00039	Diagnostic Radiology	92133	Cmptr ophth img optic nerve
0.00048	Diagnostic Radiology	92134	Cptr ophth dx img post segmt
0.01631	Diagnostic Radiology	99203	Office/outpatient visit new

LOF algorithm in a given locality that comprised of 10 nearest neighbors are shown in Table 12. This table also shows that the fraudulent providers are scattered throughout the ranks and do not comprise of the highest risk scores which shows that this technique does not work well in the normal subspace to detect fraud, waste and abuse in medical insurance claims. We obtained similar results for a different locality sizes provided to the LOF algorithm.

Finally, Table 13 shows the ranks that are obtained by determining a large component for each provider in both the normal and residual subspace. As expected, this technique doesn't provide any meaningful results for the normal subspace.

5.6.4 Investigating Uncommon Categorical Values

We finally evaluate our technique to detect unusual combinations of categorical values within the claims that are filed by a particular provider. To demonstrate this, we will use a few examples to discuss some of the lowest commonality values within the 2012 procedures and services dataset. To calculate this value we use the following example application:

$$c(provider\_type, procedure\_code)$$

Table 15: Commonality values for Shakuntala Jain

Commonality Value	Provider Type	Code	Description
0.00062	Cardiology	90801	Psy dx interview

Table 16: Commonality values for Gary Tanouye

Commonality Value	Provider Type	Code	Description
0.00017	Diagnostic Radiology	90732	Pneumococcal vaccine
0.00019	Diagnostic Radiology	G0009	Admin pneumococcal vaccine
0.00050	Diagnostic Radiology	81000	Urinalysis nonauto w/scope
0.00065	Diagnostic Radiology	69210	Remove impacted ear wax
0.00347	Diagnostic Radiology	99232	Subsequent hospital care

The commonality values within the dataset for the above application range from 0.00017 to 26454.54. As a reminder the lower this value, the more unusual it is for the combination to occur within the dataset. Table 14 shows the commonality values for one specific provider, Steven Schall. As can be seen, all claims submitted from this provider have unusually low c values which shows that their combinations do not naturally co-occur. A quick search on Google reveals that this provider's speciality type has been wrongly entered within the dataset. Steven Schall is actually an Opthalmologist which explains the codes he filed and the low c values.

In another example shown in Table 15, Shakuntala Jain only filed for one type of code which has a very low c value. Just like the previous case, it turns out that her speciality has been wrongly entered into this dataset. She is actually a Psychiatrist which explains the code she filed claims for with Medicare.

Our final example for Gary Tanouye, shows all filed codes to have a low c values. A subset of claims filed have been shown in Table 16. Again a quick search for this provider shows that their specialty is actually Internal Medicine and this has also been wrongly entered into this dataset. This helps us validate our technique to detect unusual combinations of claims data field values.

5.6.5 Robustness

Once the FraudScope system is deployed at health insurance companies to prevent losses due to fraud, waste and abuse, malicious actors will try to circumvent the detection system. Since every health insurance company has limited resources for investigation of suspicious claims and providers, the malicious actor will only need to lower their risk score and rank to an extent that will not be investigated by the fraud analyst. In the process of obtaining a low risk score, the malicious actor will need to blend their candidate profile with the reference profile of the provider type which will reduce the amount for which they are able to defraud the health insurance program. This means that egregious fraud that is currently undetected by existing analytics tools will still be easily detected and only fraudsters that try to defraud for comparatively smaller amounts will be successful in evading our detecting technique. Hence, this will ultimately reduce losses for the health insurance company and prioritize the detection of the most egregious fraud which helps us achieve our goal in creating FraudScope.

5.7 Conclusions

In this chapter, we described FraudScope, our fraud, waste and abuse detection system that helps detect suspicious claims. FraudScope also provides a rank and risk score to the most suspicious claims, providers, beneficiaries, etc. to prioritize their investigation to maximize savings. It can be run prospectively to detect fraud before claims are adjudicated to prevent losses due to fraud. It can also be run retrospectively to detect the most suspicious claims after adjudication.

Unlike existing methods, it does not require any input in the fraud identification stage which frees up precious fraud analyst time to investigate a greater number of claims. This also enables our technique to detect newly emerging as well as previously unidentified fraud schemes. It can also adapt to natural changes in the treatment patterns or change in the coding system and will not require any additional work by the user to include these changes.

Now that we have secured the health information sharing infrastructure by enhancing the auditing system as well as improved its fraud detection capabilities, we will now focus on securing the devices that actually connect to this infrastructure. Such devices as well as the applications that execute on them could possibly be compromised which could lead to the breach of sensitive health data. Hence, we discuss how to secure these devices in the next chapter.

CHAPTER VI

SECURING HEALTH DATA AT END DEVICES

Every eHealth Exchange node can use a variety of devices to interact with healthcare data. Mobile devices are increasingly being used at healthcare systems so we will use these specific type of end devices, such as smart phones and tablets, for our research. They provide convenient access to health information to health professionals and patients. Also, patients use these devices to transmit health information captured by sensing devices in settings like the home to remote repositories. Such timely and convenient access could improve healthcare quality and reduce costs but it introduces the problem of protection of health data on mobile devices. Since health data is highly sensitive, it must be secured to build user trust and meet regulatory requirements.

Unfortunately, mobile devices are known to be vulnerable to a wide variety of security threats [29] and they are becoming increasing targets of malware authors [10]. Studies have shown that medical data disclosure is already the second highest breach category [41]. Thus, if such sensitive data is accessed on mobile devices, we need to protect it against attacks that would exploit mobile device vulnerabilities.

There are several challenges that must be addressed for secure health information access on mobile devices. Because such devices are popular platforms for running a variety of applications, we must ensure that sensitive data does not flow to untrusted applications. Also, such data must not be allowed to flow outside of the device to untrusted hosts. Explicit user consent can be useful when it is not clear if certain data sharing should be permitted.

In this chapter, we explore security mechanisms that provide a framework for enforcing security policies that are inspired by health information security and privacy principles, including use limitation, security safeguards, and patient awareness [64]. Our security framework can help protect sensitive medical data against unsafe and unintended uses on mobile devices. Our security enhanced framework helps prevent third-party healthcare applications from leaking sensitive medical information even when they become infected by malware. These third-party healthcare applications do not require any modifications to work with our framework.

Explicit user consent plays an important role in how medical information is shared. This is motivated by healthcare regulation that gives patients and healthcare providers the ability to decide if sharing of medical information needs to occur. However, the user consent notification and user response cannot be captured within a simple dialog prompt as sophisticated malware can script events to fool healthcare applications into believing that user consent has been obtained.

To counter such attacks, we also explore a secure consent detection mechanism for mobile platforms which helps distinguish between user initiated actions and scripted malware actions. The policy enforcement mechanisms use user consent detection whenever input from the user is required regarding a particular action. Once activated, the framework displays a prompt to the user detailing the action which caused it, and requests the user to accept or deny the communication. This mechanism could be highly effective in preventing unintended disclosure of medical information. Since it is only activated when necessary, it does not consume significant amount of resources. Our policy enforcement and user consent detection mechanisms can support security policies that will be deployed to meet the requirements for a high-quality mobile health (mHealth) system [58]. mHealth is a term used for the practice of medicine and public health supported by mobile devices.

Although the mechanisms proposed by us can help protect sensitive data for a variety of applications, we make certain assumptions that make them particularly useful for health applications. Our approach requires tagging of sensitive data which is easier when it is accessed for a small number of trusted repositories. We also rely on user consent which is well accepted in the healthcare domain. Finally, user consent based override is well suited for health applications where 'Break the Glass' scenarios allow exceptions to security policies in case of emergencies. We make the following contributions using our added mechanisms to end devices within the eHealth Exchange:

- We introduce the concept of a constrained application for mobile devices which can be used to safeguard sensitive data and prevent its flow to unauthorized entities. A constrained application's sensitive data can be protected even in the presence of attacks that can successfully compromise the application.
- Our mechanisms accomodate for varying levels of sensitivity of data within the constrained application which can be governed by different security policies. Non-sensitive data within a constrained application is unaffected by our mechanisms. Also, applications that do not deal with sensitive data are unaffected.
- We develop a user consent detection mechanism which can help distinguish actual user input from scripted events that can be generated by malware. Such secure consent can enable user awareness and control over how health information is shared.

6.1 Mobile Device Data Security: Motivation and Requirements

The motivation for securing health data accessed on mobile devices comes from two observations. First, mobile devices offer a convenient way for users to access information. Second, since such devices can run a variety of applications which can communicate with each other and external entities, patients and healthcare providers are naturally concerned about unauthorized disclosure of health data. Such disclosure could have serious consequences on patients, ranging from medical identity theft to blackmail or discrimination. If a health application captures and stores new data into medical records maintained in a remote repository, unauthorized updates could corrupt the medical history of a patient which could have serious consequences for future diagnosis or treatment. Because of these reasons, healthcare regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and Health Information Technology for Economic and Clinical Health (HITECH) Act address privacy and security of health information when it is accessed by entities engaged in healthcare related activities. These outline rules that apply to doctors, pharmacists, medical insurance and billing agents and others who fall in the category of covered entities. We briefly discuss two relevant rules which help us understand the need for protecting health data when it is accessed on mobile devices.

The HIPAA privacy rule protects all individually identifiable health information held or transmitted by a covered entity. Disclosures can only be made for specific purposes or situations such as the treatment, payment or other healthcare related operation. The privacy rule requires covered entities to maintain reasonable technical data safeguards to prevent intentional or unintentional use or disclosure of protected health information [114]. The HIPAA security rule requires a covered entity to ensure the confidentiality, integrity, and availability of health information that it creates, receives, maintains, or transmits. The covered entities must also protect against reasonably anticipated threats, hazards and disclosures that are not permitted by the privacy rule [113]. As healthcare professionals access sensitive patient medical data on mobile devices, regulatory requirements will apply to these devices as well. To secure data as mandated for covered entities or desired by patients, it becomes vital for us to understand the threats faced by electronic health information on a mobile device.

Unintended disclosures of protected health information could happen on mobile devices due to malware infections. Malicious software is known to install itself on computing devices through some vulnerability in an application or by using social engineering techniques to trick the user. Such malware, once installed on the device, can obtain sensitive information stored on the device or from other applications and can send it to untrusted applications or remote malicious entities.

Another threat comes from application developers who do not take appropriate measures to ensure data security. This could leave the data vulnerable to violations of confidentiality and integrity. Already, there is evidence that many applications share data with external entities without explicit consent of the owners of devices where such applications run [23]. Although the operating system could be the target of attacks, we focus on more common attacks that target applications.

Finally, devices could be stolen or used when left unattended which would eventually lead to disclosure of sensitive medical information. Other work has explored techniques like data encryption and remote monitoring of access to data to counter this threat and we will not address it in this work [32].

6.1.1 Mobile Device Data Security Policies

Based on the security and privacy requirements of electronic health information and the relevant threats in a mobile environment, we can now outline key features of security policies that can be used to ensure proper use of health data on mobile devices. Mobile devices are commonly used by a single user and operate under user control. Therefore, their security policies are different from other platforms. The security policy on these devices typically does not rely on identity credentials but deals with information sharing or exchange decisions based on the context of an action. More specifically, the policy must prevent disclosure of sensitive information by mediating its movement outside the authorized application's boundary. Security policies for mobile devices can be divided into two categories. For mobile devices that are used by healthcare professionals such as doctors and nurses, the policy may be specified by the healthcare enterprise. If the enterprise is a covered entity, such a policy may capture regulatory and compliance requirements. Although the device is in the control of a user, the policy may be 'locked down' and the enterprise may enforce that no changes are made to it by the user. A user may also access her or his health data on a mobile device and may choose to share it with other entities. In this case, the policy is defined by the user of the device. Our goal is to develop mechanisms that can support a variety of such policies. Physicians and patients using our mechanisms can install third-party healthcare applications with the guarantee that sensitive medical information will not be sent without their knowledge even when these applications are compromised. To motivate these mechanisms, we first start by outlining key requirements of such policies.

6.1.2 Requirements

Mobile device security policies primarily focus on sharing of health data between applications on the device or exchange of data with remote entities. We will now describe framework mechanisms that are necessary to enforce a variety of such policies to meet the needs of healthcare applications. We will use a sample third-party mobile healthcare application called Sana Mobile [91] to illustrate key features of our security framework. Sana is an open-source remote telemedicine platform which collects patient data from procedures performed by health workers and uploads this information for a doctor to review. Doctors can then send their diagnosis to health workers through the mobile application.

Our mobile device data security framework must monitor and prevent disclosure of sensitive health information to unauthorized parties that include untrusted applications running on the device and remote services. It should also stop transfer of sensitive health data to insecure storage devices. We also want a user to be able to allow or deny certain health data sharing requests. We discuss each of these in detail.

6.1.2.1 Controlling Remote Communication

Healthcare applications like Sana must be able to communicate with external entities over several channels such as the Internet and Bluetooth. These channels allow transfer of data and hence need to be secured. Sana could connect to a Blood Pressure Monitor via Bluetooth and read the user's blood pressure and pulse reading. This could then be uploaded to a repository in a doctor's office via the Internet. However, all information accessed by an application like Sana may not be sensitive. In particular, information downloaded from a public health advice portal need not be protected. Thus, we need to ensure fine-grain control over the sharing of application state by tracking access to sensitive health data by Sana. The security policy enforcement engine must also mediate requests for sending data over network channels by Sana. It must detect data flow on both incoming and outgoing channels and the source of incoming data itself could be useful in deciding if the data is sensitive (e.g., data is received from an electronic medical record repository).

For the Internet channel, the framework should be aware of all connections made by the application and monitor those specifically rather than all open connections on the device. The framework should allow communication with trusted external entities and prevent it when it involves unauthorized entities. One can take an approach that keeps a list of trusted entities which could include the user's PHR and EHR repositories or offices of healthcare providers. The policy enforcement engine allows flow of health data between Sana and these trusted entities and data received from them is marked as sensitive. It is possible to seek user input before communication is disallowed and it can be allowed when explicitly permitted by the user. Obviously, one challenge is to determine what remote entities are trusted. We believe that since health data only needs to be shared with a limited number of external entities which do not change frequently, such lists can be built with user input without imposing excessive burden on users.

6.1.2.2 Preventing Data Sharing with other Applications

As with every other mobile application, Sana could share information with other applications. However, sharing between a healthcare application and other untrusted applications could lead to disclosure of sensitive health information. Thus, we need to monitor inter-process communication channels and place safeguards to prevent such disclosures. The policy enforcement framework must detect flow of sensitive data between Sana and other applications and can respond in one of two ways. It could disallow the data sharing and notify the user. This is meaningful because a healthcare application must avoid interacting with other potentially unsafe applications and should typically include all required functionality within itself. Another response would be to notify the user and allow the user to decide if such sharing should be allowed. In this case, interactions between the healthcare application and other applications must be explicitly consented for by the user of the device. The policy enforcement does not monitor communication of untrusted applications and any data that is allowed to be shared with them must be viewed as non-sensitive.

6.1.2.3 Controlling Insecure Data Storage

Apart from sharing health information with other applications or sending it to remote entities, Sana can also store data permanently on the mobile device's main memory or an external memory card. Usually, mobile operating systems provide data protection and separation between applications on the device's main memory. However, sometimes an application could be allowed to share data on the file system with other applications which could lead to a disclosure of sensitive information.

Similar to data sharing, the policy enforcement framework must detect operations performed on files containing sensitive information and respond in multiple ways. The framework could prevent untrusted applications from reading files that are owned by healthcare applications. This is because untrusted applications should not need access to sensitive information. The framework could also prevent storage of sensitive information on the file system and require the application to connect to a remote repository for storing sensitive information. This provides a higher degree of security but is more restrictive for the application. Another response would be to notify the user and allow the user to decide if the particular file operation should be allowed. In this case, explicit consent of the user of the device must be required for all operations performed on files containing sensitive information. The policy enforcement does not monitor files of untrusted applications and any files that are allowed to be accessed with such applications must be viewed as non-sensitive.

Another concern is that external memory such as an SD card can be inserted into other devices which can read all stored data and this could also lead to disclosure of health information. Hence, our policy enforcement framework must mediate all external storage requests of sensitive data made by Sana. It can either disallow requests that want to store information on the external memory or it can notify the user when such an attempt is made. Such a notification would allow the user to decide if the action in question is safe and would not lead to disclosure of health data. Again, once the user allows storage of some information that comes from Sana, it should be viewed as non-sensitive.

Finally, some mobile devices can be connected to other computing devices via USB and data from certain memory locations of the mobile device can be accessed by the computing device. This could also lead to a loss of sensitive information. Hence, the policy enforcement framework must prevent applications from storing sensitive information to such memory locations or explicit user consent should be obtained to sanitize this information before it can be stored at such a location.

6.1.2.4 User Consent Detection

We take an approach where user consent is sought in case the policy enforcement engine suspects that potentially sensitive data may flow to untrusted applications, remote entities or even to a storage device. This is motivated by healthcare regulation that give patients and healthcare providers the ability to decide if sharing of medical information needs to occur. In the previous sections, we discussed mechanisms which notify the user of a potential disclosure of healthcare information and require him to decide if the particular action should be allowed or blocked. However, this notification and user response cannot be captured within a simple dialog prompt due to the sophistication of current malware. Such malware is capable of executing scripted actions on the device as though the user generated a response. Hence, we require a secure mechanism which will help differentiate automated malicious activity from genuine user activity. This is termed as user consent detection. The policy enforcement framework must use user consent detection whenever input from the user is required regarding a particular action. Once activated, the framework should display a prompt to the user detailing the action which caused it, and request the user to accept or deny the communication.

6.2 Approach

We will now discuss the design of a framework which will enforce the security goals outlined in the previous section. We rely on a set of assumptions to build a trusted mobile device platform for secure handling of medical data. We assume the operating system on the mobile device is trusted. This includes the kernel and layers below applications. We also assume a third party mobile healthcare application runs on the device and accesses electronic health information. This application downloads and facilitates meaningful sharing of sensitive health data when valid credentials are provided by the user. We do not assume that the health application is trusted. It may be compromised by a malware infection and it may coexist and share non-sensitive information with other untrusted applications.

Since security of sensitive data relies on proper enforcement of a security policy, the enforcement engine must be part of the trusted computing base. Also, we assume that the operating system is trusted and add mechanisms to it to implement the enforcement engine functionality. We also advocate that applications that handle sensitive health information must be treated differently by our framework compared to other untrusted applications that do not deal with sensitive information. We term the applications which deal with sensitive information as *constrained applications* because their behavior is constrained in order to achieve security goals. Policy enforcement must only apply to constrained applications and only to data that is deemed sensitive. No restrictions are placed on other applications unless they try to interact with constrained applications. Applications can be declared as constrained by the application developer in the manifest file or by the user at the application installation screen, which could appear as a checkbox along with other permissions that the application requires. We also assume that certain system applications that implement limited functionality can also be trusted. We use one such application, the background service, that is used to support user awareness and control of how data is accessed on the device.

6.2.1 Tagging Sensitive Data

We do not assume that all data in a constrained application is sensitive. A healthcare application may contain publicly available information such as symptoms of a disease and these may not require our additional data safeguards. If such non-sensitive data is not allowed to be shared, it may interfere with the normal functioning of an application. Thus, to support secure access only for data that is sensitive in a constrained application, we must be able to distinguish it from other non-sensitive information within the application. This can be done in multiple ways, one is to maintain a list of all specific sensitive data locations within the application. This list would need to be updated every time sensitive information enters the application and the list would need to be traversed every time data tries to leave the application. Such a design choice may not be scalable and would expand the list size greatly if we track information flows.

Our other option is to tag memory locations which contain sensitive information. This does not require us to maintain a master list of all sensitive information locations within the application and anytime data tries to leave the application, we will only need to check its memory location tag. Hence, this design choice is more scalable. Another point to consider is that information within a constrained application may be of different levels of sensitivity which may need to be accommodated by the policy being enforced.

We must tag all incoming data with a label describing its level of sensitivity or mark it as non-sensitive information. We also need to maintain tags properly as sensitive information flows across memory locations. Data tagging can be done in multiple ways. All incoming data could be classified by the source itself and include the nature and sensitivity level of the information as metadata. Another method would be to classify data on the device based on the sensitivity level of the source repository from which the information comes. The device could be pre-populated with a list of trusted private repositories which contain sensitive information and a list of public repositories which contain publicly accessible non-sensitive information. A private repository could be classified by the highest level of sensitivity of the data it contains. All incoming data from a repository connected via a communication channel such as the Internet will be tagged with its specific sensitivity level. In case the application connects to a previously unknown repository, the user can be prompted to define its sensitivity level.

Apart from external sources, data can also enter the constrained application through other channels such as the file system and the external memory card. Data read from files containing sensitive information should be tagged with its level of sensitivity. Whenever new data on a memory card is accessed by a constrained application, the user could be prompted to choose whether the entire memory card or only certain folders and files contain sensitive information. The level of granularity can be left at the convenience of the user. All sensitive information that is loaded from the memory card is tagged by our framework.

Finally, data input by the user into an application could also be sensitive. Our framework must provide two modes of data entry into the constrained application. A sensitive data entry mode, which can be selected from our service application, tags all user input as sensitive. The regular data entry mode is used to input non-sensitive information. One could argue that sensitive data can also come from other applications, but such applications should also be marked as constrained in the first place. We will discuss how tracking of tagged data can be used in this case. Collaborating constrained applications could be allowed to share sensitive information but explicit consent must be obtained from the user first when tagged data crosses application boundaries.

6.2.2 Monitoring Tagged Data Flow

Once information has been tagged, we must allow it to move freely within the constrained application. As tagged information flows, we need to track it. We achieve this by using TaintDroid [23]. TaintDroid is an information flow tracking system that taints data within the Android operating system. For example, if a tagged data item is copied into another part of memory, TaintDroid tags the memory locations where the data is copied. We chose TaintDroid because it can provide an efficient solution for tracking the flow of sensitive health information in an application. If any data from tagged memory tries to leave the constrained application, we must ensure that such transfer does not violate the security policy of the application. If tagged information is being sent to an external repository via a communication channel, the policy enforcement engine must check if the external repository is at the same or higher sensitivity level as the information itself. Only when this is the case, the information is allowed to leave the device. If the external repository contains only public information and does not contain any sensitive information, then a default data security policy can be enforced. The user will be prompted to provide the sensitivity level of any newly connected external repositories. If the user chooses not to provide the level or when it is lower than the data, communication is disallowed.

Tagged information can also be stored on the file system. Files stored on the mobile device's main memory must be tagged with the highest level of sensitivity of the information contained along with the constrained application that created the file. Applications can also store tagged information on a memory card. The first time an application attempts to store tagged information on a memory card, the user is alerted about the risk of sensitive data disclosure if the memory card is removed from the mobile device and used elsewhere. The user can then choose to allow storage request or deny requests made by the application. Similarly, tagged information that is detected at the inter-process communication between two applications will be subjected to the constrained application's security policy. In the next Section, we explore a concrete implementation of mechanisms that can support these policies.

6.3 Implementation

We chose the Android mobile operating system to explore the implementation of constrained applications and security policies that can be used to govern access to sensitive health data. We chose this system because it is open source and an emerging healthcare platform for which several healthcare applications have already been developed. Similar implementations can be done on other mobile operating systems. Before we discuss our implementation, we briefly describe some relevant details of the Android operating system. At the base of the Android operating system lies the Linux kernel which provides core system functionality. Above this layer are the libraries, application framework, dalvik virtual machine and application layers.

All Android applications require an `ApplicationManifest.xml` file. This file contains information about the application which is required by the Android operating system before it can run the application's code. This includes the permissions required by the application to access protected parts of the API, interactions with other applications and permissions required by other applications to interact with its components. These permissions are declared statically at install time. Hence, even though these permissions could provide some degree of protection against data breaches, they cannot be tailored at run-time. In addition, the permission model which exists in Android is coarse-grain and cannot fully enforce the fine-grain tagged data policies we explore. Permissions such as those required to access the Internet grant the application unchecked access and do not allow us to restrict the application to communicate with a subset of known safe or user consented repositories.

Finally, while certain features, such as internal data storage within the device's memory, may seem private to the application, exceptional cases exist which can allow sharing of this data with another application or the data can be made public. Hence, the security features provided by Android are insufficient to enforce the security policies required to protect sensitive healthcare information. However, once we add our mechanisms to the Android operating system, such policies can be enforced.

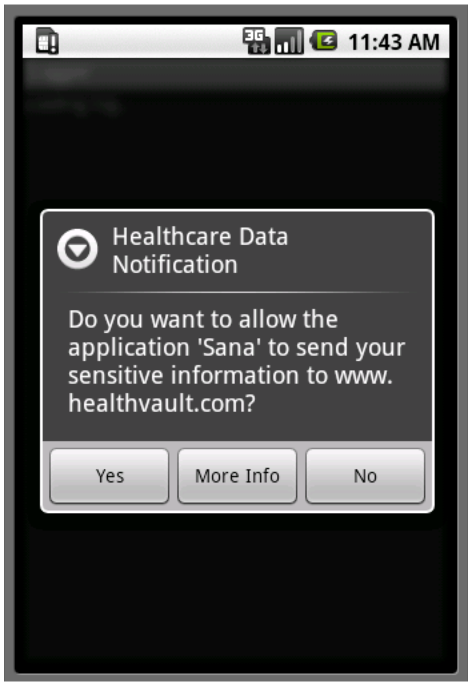
We implemented our framework on a Google Nexus One phone running Android 2.1. We used the TaintDroid [23] framework for tagged data tracking and built our mechanisms over it. For our particular implementation, we assume that data tagging occurs based on the nature of the source repository or based on explicit user input.

We will also assume a single level of sensitivity while describing this section. In other words, we will consider data to be either sensitive or non-sensitive. However, our framework can accommodate 32 levels of sensitivity for data.

The most significant challenge we faced in implementing this framework was understanding the internal interactions of the relevant parts of Android. The Android kernel is a modified Linux kernel for which sufficient documentation exists. However, our mechanisms reside in layers above the kernel for which we did not have sufficient documentation. These layers document the exposed APIs which can be used by the applications that run above them but their internal interactions are not described. This required considerable time on our part to understand these internal interactions before we could enhance the system to include our data protection mechanisms. The middle layers of the Android operating system also include code in several different programming languages. The interactions between code written in these languages and movement of data across them required native code to be implemented within the operating system. Our external data flow mediation was implemented in the dalvik virtual machine in about 400 lines of Java code. Our inter-process communication mediation was implemented in the binder library in about 200 lines of C++ code. Our user consent detection mechanism was implemented in the framework layer in about 250 lines of code.

6.3.1 Background Service

Much of our framework functionality for enforcing a security policy resides within the Android operating system but we also use a background service to support user interactions required for security policy enforcement. This service launches automatically when the device is powered on and listens for any communication from the operating system. Whenever user input is required, a dialog prompt is displayed to the user above any currently active application as shown in Figure 10. The background service


11:43 AM

Healthcare Data Notification

Do you want to allow the application 'Sana' to send your sensitive information to www. healthvault.com?

Figure 10: A prompt generated by our framework.

is only used to display prompts and the response is detected by our user consent detection mechanism which is able to distinguish between scripted responses by malware and actual user input. We describe this mechanism in detail in a later section.

6.3.2 Security Policy Enforcement Engine

Our policy enforcement engine implements mechanisms that can support several policies. We currently use a simple syntax for policy specification and illustrate it via an example but it is possible to utilize more advanced policy specification languages. Our framework implementation allows the structure and source of policy files to be easily changed without affecting the rest of the framework. Our current implementation uses policy files stored on the device. However, the policy files could also be downloaded from an external trusted source. As shown in Figure 11, we maintain a separate policy engine within the virtual machine and at the binder library. This is because the external or network communication channel policy enforcement needs

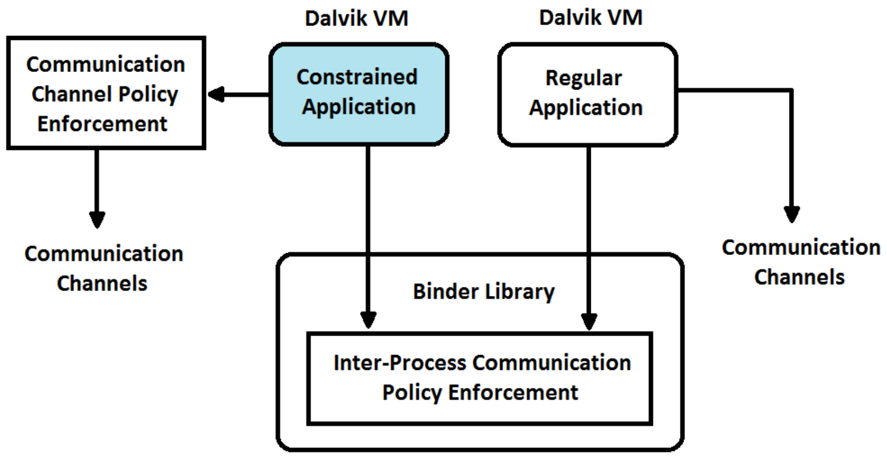


Figure 11: Policy enforcement framework.

to happen within the dalvik virtual machine while the inter-process communication channel enforcement needs to be done within the binder library. Although a single policy engine could enforce both, we decided to avoid the Java-native transition overhead [34] by including two instances of the policy engine. This also allows us to change the policy structure and complexity of either the network or inter-process communication channels without affecting the other.

Policies for different constrained applications are stored separately which allows our enforcement engine to quickly locate the policies that belong to a particular application. Our policies follow the syntax: `DESTINATION:CONTEXT:ACTION`. The `CONTEXT` for a policy rule is optional and only defined if necessary. Examples of `CONTEXT` include `LOCATION`, `SENSITIVITY_LEVEL` and `TIME`. The following are example policy rules for a constrained application on our framework:

www.healthvault.com::allow

www.myphr.com:confidential,0800-1700:allow

com.healthapp.sample:1700-0800:deny

The first rule allows information to flow between this application and the www.healthvault.com repository regardless of any other context for the action. The next rule allows information to flow between this application and the www.myphr.com repository only if the sensitivity level of the data is confidential or lower and between 08:00 and 17:00 hours. The last rule denies this application from communicating via IPC to the application belonging to the package name com.healthapp.sample between 17:00 and 08:00 hours. Similarly, a variety of policies can exist for an application within our framework. A policy may include a large number of such rules.

6.3.3 Controlling External Data Flow

To secure network communication channels, we implement complete mediation for external communication requests over the Internet. It also became apparent that a similar mechanism could be put into place for other communication channels, such as Bluetooth. Our communication channel data mediation monitors all incoming and outgoing data between the constrained application and external entities.

On the Android operating system, every application runs within its own dalvik virtual machine. This virtual machine runs above the kernel layer. Even though data movement between applications and external communication channels is first processed in the kernel, we implemented our mediation within the virtual machine as we are able to use TaintDroid tag values at this layer. It was not possible for us to determine the taint tags within the kernel based on the existing TaintDroid taint framework and the notion of an application does not exist at the kernel.

Whenever a connection is made between an external entity and an application, our framework first checks if this application is constrained based on the process that engages in the network transaction. If the application is constrained, the policy engine will then determine if the location has been previously been classified as trusted or untrusted. For pre-classified locations, then the policy engine will determine if this location is a private repository containing sensitive information. If so, all the incoming information from this repository will be tagged as sensitive. If the location is a public repository, then all incoming information will not be tagged as sensitive. Tags for data coming from locations for which the policy does not contain any classification yet will be defined by the user. The user will be prompted by our background service of the current connection and will be asked to classify this new external location. The user's response will be captured by our user consent detection mechanism.

We also discussed that we can classify data from the same external location as non-sensitive and a number of sensitivity levels based on the metadata from the location itself. This can support finer granularity of data protection and would prevent any non-sensitive information from a private repository to be falsely tagged as sensitive. However, in our current implementation, we assumed that the external repository does not provide such a classification of the data it sends to the device. This is the current case with all existing third-party applications. The sensitivity classification is performed on the device based on the source of the information. We can also use other techniques such as text-mining to determine if the received data contains sensitive information and create a tag based on it. This is currently not implemented in our framework.

When information tries to leave the device to an external location, our framework first determines if the application is constrained. For constrained applications, the taint tag of the information that is leaving the device will determine if it is sensitive or not. Non-sensitive information is allowed to leave the device to any external location. However, if the information is sensitive, our policy enforcement engine will check if a policy exists for the type of data and the external location. If no policy exists, the user will be asked to define one using the background notification service and our user consent detection mechanism. Information flowing out of non-constrained applications will not be mediated.

6.3.4 Controlling Communication with Untrusted Applications

Before we discuss our policy enforcement mechanism for inter-process communication, we will discuss some pertinent details of the Android operating system. All inter-process communication between applications in separate dalvik virtual machines takes place through the Binder library. Binder is a custom Android protocol which uses the core concepts of OpenBinder. The binder kernel module is a very low-level protocol where it is difficult to extract taint tags. Hence, we implement our policy enforcement engine for inter-process communication at a higher layer built on top of the kernel module in the binder library. Binder uses Parcel as a marshaling protocol that writes primitive types into transaction buffers. These binder transactions can either be one-way or two-way involving a request parcel and a response parcel.

Our framework intercepts the parcels between two communicating applications within the Binder system. At this point, we are able to determine which two applications are communicating. We then check if either of the communicating processes is a constrained application. If none of them is constrained, we allow the communication to proceed. Otherwise we check the parcel objects to detect if either the request or the subsequent response within a transaction contains any information tagged as sensitive. Our framework enforces a policy only if any one of the communicating processes is constrained and the information it is sharing within the parcel message is tagged as sensitive. However, a devious receiver in an IPC transaction could unpack the variables within the parcel message in a different way to acquire its value without the taint tag. To prevent this, TaintDroid taints the entire parcel message as sensitive even if only a few variables within the parcel message are actually sensitive. This could potentially introduce false positives within the application.

We discovered while implementing our framework that at launch time, an application communicates with the `system _server` process which provides core system services. This communication is vital for the launched application and hence the `system_server` is allowed to bypass our enforcement engine's mediation at the IPC even if a false positive is detected during a binder transaction with a constrained application. However, for all other transactions involving constrained applications, the parcels are checked to determine if they contain any sensitive information. If a policy does not already exist between a constrained application and the application it is communicating with, the user is allowed to provide a policy rule for the current transaction. This is done by generating a user prompt using our background service and activating the user consent detection mechanism.

6.3.5 User Consent Detection

As mentioned in previous sections, our enforcement engine generates prompts for the user to define policy rules when one does not exist to determine if certain data flows should be allowed. However, if such prompts can be intercepted by malware, it can define policy rules which could lead to a breach of sensitive information. To prevent such a situation, our framework must be able to distinguish between actual user input and potentially malicious scripted events.

We first studied how user input events are generated within the Android operating system. Hardware input such as touchscreen presses are first received at the kernel which translates them into actions at the application layer. Each individual touch to the phone screen is received as a sequence of raw codes from the touchscreen driver. These key presses are stored in a `KeyInputQueue`, shown in Figure 12, which eventually reach the application as events. We extract the x and y co-ordinates of the touchscreen press on the device from this queue. Retrieving the information at this level allows us to be oblivious to the touchscreen hardware drivers present below as all drivers input their key presses to this queue. This also provides the flexibility of working above different hardware drivers without requiring any changes to our consent detection mechanism. At the same time, our experiments verified that we are

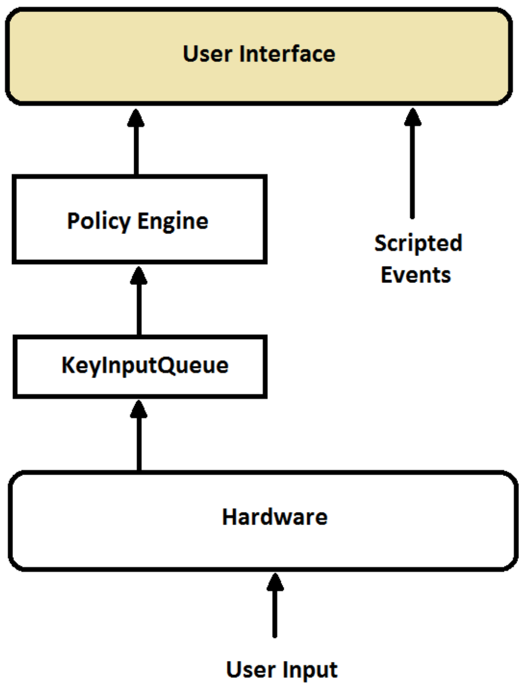


Figure 12: User consent detection mechanism.

below the level at which scripted actions generate events for the operating system. Thus, this enables us to easily distinguish malware scripted actions and actual user input. However, data isolation between different applications complicates this process. The data from hardware drivers can only be accessed by the `system_server` process within the framework layer and is isolated from other applications running within the dalvik virtual machine. To allow the policy engine within the virtual machine to receive the x and y co-ordinates of each touchscreen press, we created a system service which exports only these co-ordinates to the dalvik virtual machine. This is done by using the Android Interface Definition Language(AIDL) to define an interface which different processes agree upon to communicate with each other via IPC.

When user consent detection needs to occur, the background service of our framework launches a notification which details the triggering action. The user is asked to select a button on the screen to allow or deny the action. The x and y co-ordinates generated from the drivers are compared to those corresponding to the allow or deny buttons on the screen. The positions of the allow or deny buttons generated by our background service remain constant for each device and hence the x and y ranges for each button are known beforehand. Based on the response given by the user, the action is permitted or blocked.

Apart from the co-ordinates, we can also retrieve the pressure of each touchscreen press, the area of the finger used to touch the screen and the duration of each press from the hardware driver. All of this information can be used to create user specific profiles similar to a hardened password [69] which provides a stronger guarantee that the owner of the device has responded to the prompt rather than any other user. However, this is not implemented within our framework.

6.4 Evaluation

Our framework includes additional checks when communication takes place between applications or when an application exchanges data with an external entity. This could have some performance impact on the applications. To understand this, we present an evaluation of our framework in the following subsections. We outline the factors which could affect performance and discuss the expected trends. We also provide performance measurements of delays introduced by our framework for a sample application. Following that, we also modify an existing third-party healthcare application to include malicious functionality and evaluate the response of our framework to its actions.

6.4.1 Performance Analysis

To understand performance overheads of our policy enforcement, we examine the sources of overhead which could potentially be added by our framework. Since we use TaintDroid for taint tracking of sensitive data, one source of overhead could be the taint tracking mechanism. However, it has been reported that TaintDroid

Table 17: Performance Evaluation Results

Communication	TaintDroid	Policy Enforcement framework
Upload	161.66 ms	181.83 ms
Download	18.20 ms	49.38 ms
Local Sharing	16.69 ms	18.53 ms

introduces only 14% performance overhead and 4.4% memory overhead on a CPU-bound microbenchmark. This overhead is not significant for third-party healthcare applications such as Sana which are used to access health information.

The next source of overhead comes from the security policy and its enforcement. Since we do not store the policy in memory, our framework will not consume significant amounts of additional memory even if a large number of policy files exist on the device. Also, security policy decisions need to occur infrequently, storing these policies in memory is not as important. However, the drawback of reading a policy file is that policy enforcement decisions may take longer if the size and complexity of the file increases. This is obvious as the files need to be parsed to extract the relevant policy before its rules can be enforced.

Additional overhead could arise due to data movement between the binder library and the dalvik virtual machine. This is due to the cost associated with the Java-native transition to provide the parameters required by the policy engine and to return the policy decision [34]. As mentioned earlier, native code transition is required as the dalvik virtual machine is mainly in Java while the binder library layer is mainly in C++. We avoided such a situation by providing separate policy engines for IPC and network communication.

6.4.1.1 Performance Metrics

To evaluate the performance of our framework, we developed a healthcare application which has the ability to upload, download and locally share a patient's health information. We then measured the time taken to perform these actions on the application in the presence and absence of our framework on the mobile device. First, we measured the time elapsed between the user pressing a button on the application to upload data to a repository to the data actually being written to a socket. Next, we measured the time elapsed between the user pressing a button on the application to download data from a repository to the time the data is placed into a receive buffer from the socket. Finally, we measured the time elapsed between the user pressing a button on the application to share this information with another application and the time it takes to perform the IPC transaction. We summarize our findings for average times taken for the above actions in Table 17.

Our experimental results show that our policy enforcement mechanism added a 20.17 ms overhead while sending data on the network and a 31.18 ms overhead while receiving data. The overhead added by our policy enforcement for IPC transactions was found to be 1.84 ms. This proves that our policy enforcement framework adds acceptable overhead for mobile applications for a policy similar to the one described in Section 4.2.

6.4.2 Threat Analysis

To demonstrate the functionality of our framework and to explore the security provided by it, we downloaded and used an open-source third party healthcare application called Sana Mobile. We created different variants of this application each of which includes a different hidden malicious functionality. Each variant of this application represents a situation where a breach of sensitive healthcare information may occur. We allowed these applications to execute over the modified Android operating system and observed our framework's response which we discuss below.

6.4.2.1 Malware Infected Applications

The Android market is an online software store which allows users to browse and install third-party applications. These applications provide useful features and can often be downloaded for free from the market. However, recent studies have shown that these applications can steal sensitive user information without the user's knowledge [23]. At install time, users are shown a list of permissions, such as Internet access, requested by the application. However, they cannot control what information is sent on the Internet and what subset of remote servers are allowed to receive this information. This requires that the user trust application developers to not send sensitive information on the device to any malicious or unauthorized remote servers.

Apart from this, genuine applications which do not contain any malicious functionality are known to be vulnerable due to variety of reasons such as design flaws and coding errors. To patch these vulnerabilities, developers release updates. User devices which are not regularly updated with these patches could be exploited to harm the user and the device. Hence, even if malicious functionality is not originally included within the application, vulnerabilities can be exploited to steal sensitive information on the device.

Our first variant of Sana imitates both malicious applications and also vulnerable applications which are exploited by malware. We install this variant on a device running our framework and assess our framework's response to its malicious behavior. When the user installs this variant on the device, he or she would check a box on the permissions screen to indicate that the application will deal with sensitive information. Our framework will now tag information flowing into this application from external locations that are defined to be sensitive and check and enforce policy rules on the flow of such information at the application's exit points. This variant contains hidden functionality which periodically uploads sensitive information gathered by the application into a remote malicious server using the Internet as the communication channel. We run this remote malicious server on a desktop and monitor the incoming connections.

When this application first connects to its private repository to download sensitive healthcare information, the user is prompted that a connection has been made to an external location which is not known to be trusted. The user then indicates through the user prompt generated by our framework that the external location contains sensitive information and data from it is tagged as sensitive. When this application tries to discretely connect to the remote malicious server for the first time, our framework looks for a previously defined policy for this application. When no existing policy is found, the user is prompted to allow or deny the Sana application request to send information tagged as sensitive to the malicious server. The user is then aware that the application is sending sensitive information to a remote unknown server which the user did not initiate. They would then deny this action, preventing the loss of sensitive information. Our framework then enters a policy for this particular application which would deny sending any sensitive information to that particular malicious server. All future requests by this application to send sensitive information to that malicious server are subsequently blocked. This successfully prevents a data breach from occurring.

6.4.2.2 Insecure Programming Practices

Application developers do not always make the most secure choices while developing software. Such insecure design choices could later be exploited for malicious purposes. Our next variant of Sana imitates an insecure design choice which could potentially be exploited by malware. We added a background service to Sana which launches at phone boot time and continues to run in the background even if the application is not active. This background service periodically connects to a private repository containing sensitive information such as communication from a doctor, lab test results, etc. Whenever new health information pertaining to the user of the device is available on the repository, the background service downloads this information and sends this data to Sana.

Since Sana is an open source application, all the services launched by it can be found in the `ApplicationManifest.xml` file including the background service we added. We then created a separate weather application with hidden malicious functionality. As the name suggests, the weather application displays weather information to the user. But this application also requests permissions to bind to the background service of Sana. This can be easily done by adding a `service` tag within the `ApplicationManifest.xml` file of the weather application. Within this `service` tag, the weather application specifically requests permissions to bind to the Sana's background service. Once this permission has been requested in the manifest file, Android will allow the weather application to bind to the service. Although Android will display the required permissions to the user at install time, the service permissions are not shown to the user. The user is completely unaware that the weather application is maliciously binding to Sana's background service.

The weather application can now conveniently bind to the background service periodically and collect sensitive health information from our variant's background service. When this weather application binds to our variant and tries to download this information, an IPC transaction is initiated. Our framework checks to see if any one of the communicating applications is constrained. Once it determines that the Sana variant is constrained, it checks every parcel request and response between these applications to assess if they contain any sensitive information. This is done by checking the tags of the parcels in every transaction. When the background service responds to the malicious weather application by sending sensitive information, our framework detects a parcel containing sensitive information being sent from a constrained application and immediately checks if a previous policy rule exists. If one does not exist, which is the case here, the user is notified of this transaction through a prompt generated by the background service. The user consent detection framework is activated to capture the response of the user. Thus, our framework is able to successfully prevent loss of sensitive information via IPC.

6.4.2.3 Malicious Scripted Actions

Our third variant emulates a malicious application which is aware that the user will be prompted to confirm any potential disclosure of sensitive information such as that provided by our framework. As an attempt to subvert this process, this variant inserts scripted events into the operating system to hijack any such prompts and prevent the user from being notified of potential disclosures. This scripted input can automatically select the button on the prompt which will allow the breach of sensitive information to occur.

When this variant tries to steal sensitive information from the device, our policy enforcement engine detects sensitive information leaving a constrained application and triggers a prompt detailing the action from the background service and the user consent detection mechanism is activated. This prompt will wait for input from the user to either allow or deny the particular action in question. At this point, the variant quickly responds to the prompt by inserting simulated key events. However, our user consent detection mechanism ignores any such event and looks at the `KeyInputQueue` where actual hardware key events are present. Scripted events are not inserted in this queue and can be used to easily distinguish between actual user generated hardware events and potentially malicious scripted events. Our framework extracts the x and y co-ordinates of the touchscreen press by the user and is able to determine which specific button was selected at the prompt generated by our background service. When the user responds to block this unknown movement of sensitive data outside the constrained application, our consent detection mechanism notifies the policy enforcement framework which inserts the appropriate policy into the application's policy file. Thus, this mechanism helps defeat any attempts at subverting our policy enforcement by using scripted actions.

6.5 Conclusions

In this chapter, we secured the devices that connect to the health information sharing architecture. We achieved this by introducing the concept of a constrained application for mobile devices which can be used to safeguard sensitive data and prevent its flow to unauthorized entities. A constrained application's sensitive data can be protected even in the presence of attacks that can successfully compromise the application. Our mechanisms accommodate for varying levels of sensitivity of data within the constrained application which can be governed by different security policies. Non-sensitive data within a constrained application is unaffected by our mechanisms. Also, applications that do not deal with sensitive data are unaffected. We also developed a user consent detection mechanism which can help distinguish actual user input from scripted events that can be generated by malware. Such secure consent can enable user awareness and control over how health information is shared.

CHAPTER VII

CONCLUSIONS & FUTURE WORK

As electronic medical information is shared across the United States, backed by the incentives provided by the government, it will be exposed to a wide variety of online threats. In this dissertation, we explored the hypothesis that middleware in systems which exchange health information can be augmented to support better accountability and security of health data and reduce losses due to fraud. To this effect, we introduced several techniques that help improve the security of the exchange of health information.

- **Enhanced Auditing & Awareness:** We augmented the eHealth Exchange as an example to enhance accountability and patient awareness at each node. We introduced two new types of records called the *Sharing Record Log (SRL)* and *Audit Record (AR)*. These are created, propagated, signed and stored by two additional architectural components, namely the *CONNECT Monitoring Agent (CMA)* and the *Component Logging Agent (CLA)*. In addition to this another architectural component called the *Patient Agent (PA)* helps inform the patient of how their data is being shared within the eHealth Exchange.

  These enhancements help us ensure that we log all inter-nodal and inter-component interactions which cannot be subverted or corrupted without detection unless both interacting parties are compromised or malicious. Our audit records are also redundantly distributed throughout the node and eHealth Exchange which helps in detecting deletion of records. In addition to this, we also help alleviate patient privacy concerns by providing greater awareness to patients about how their personal medical information is being shared. These enhancements enable early detection of unauthorized and malicious sharing of health data, which can help limit the resulting damage.

In case a breach or compromise occurs, our introduced concept of sharing provenance helps us identify the medical practitioner or healthcare organization that may be a source of leak of information or the unauthorized node that fraudulently releases or acquires a particular patient's data. We can use these records to determine all the other actions that were performed by those identities within the eHealth Exchange which may have also been malicious. Reversing these actions would give us a starting point to rectify any patient's medical history that may have been corrupted by fraud.

- **Fraud, Waste and Abuse Detection:** We also described FraudScope, our fraud, waste and abuse detection system that helps detect suspicious claims. FraudScope also provides a rank and risk score to the most suspicious claims, providers, beneficiaries, etc. to prioritize their investigation to maximize savings. It generates output very quickly which allows it to run prospectively to detect fraud before claims are adjudicated. It can also be run retrospectively to detect the most suspicious claims after adjudication.

  Unlike existing methods, it does not require any input in the fraud identification stage which frees up precious fraud analyst time to investigate a greater number of claims. This also enables our technique to detect newly emerging as well as previously unidentified fraud schemes. It can also adapt to natural changes in the treatment patterns or change in the coding system and will not require any additional work by the user to include these changes.

- **Securing Health Data at End Devices:** We also secured the devices that connect to the health information sharing architecture. We achieved this by introducing the concept of a constrained application for mobile devices which can be used to safeguard sensitive data and prevent its flow to unauthorized entities. A constrained application's sensitive data can be protected even in the presence of attacks that can successfully compromise the application. Our mechanisms accommodate for varying levels of sensitivity of data within the constrained application which can be governed by different security policies. Non-sensitive data within a constrained application is unaffected by our mechanisms. Also, applications that do not deal with sensitive data are unaffected. We also developed a user consent detection mechanism which can help distinguish actual user input from scripted events that can be generated by malware. Such secure consent can enable user awareness and control over how health information is shared.

7.1 Future Work

In this dissertation, we have enhanced the existing eHealth Exchange architecture to improve its security guarantees, however, there are several more enhancements that can be made in the future:

- **Auditing System:** In our future work we would like to discuss how a *SRL* of an existing document can be merged with a modified version of the same document from another node that contains a different *SRL*. This would help maintain the sharing provenance when two documents are modified by separate nodes and then merged at some point in the future. In addition to this, we would also like to explore how the sharing record logs may be affected if different sections of a medical document are redacted for different users on the same node.

- **Fraud, Waste and Abuse Detection:** The current datasets available to us contained aggregated data from the Centers for Medicare & Medicaid Services (CMS). These datasets did not contain any timestamps to simulate prospective detection of suspicious claims using FraudScope. For our future work, we would like to obtain detailed claims data that is timestamped so that we can apply our technique prospectively and show the effectiveness of FraudScope in such a setting as well.

- **Secure Devices:** In our future work to secure end devices we would like to explore other techniques such as text-mining to determine the sensitivity level of information received by the constrained application. This will allow us to automatically identify specific sensitivity levels of information received from a repository. We also want to enable social networking to share permissions for external repositories or applications by other users of our secured end device system. In addition to this, we also want to develop methods to capture the context of sensitive information which will help the users make better policy decisions. We will also explore how other data governance policies can be supported by our mechanisms.

REFERENCES

[1] Agrawal, S. and Budetti, P., "Physician medical identity theft," *JAMA*, vol. 307, no. 5, pp. 459–460, 2012.

[2] Al Ameen, M., Liu, J., and Kwak, K., "Security and privacy issues in wireless sensor networks for healthcare applications," *Journal of medical systems*, vol. 36, no. 1, pp. 93–101, 2012.

[3] American Health Information Management Association, "Report on the use of health information technology to enhance and expand health care anti-fraud activities." Sept 30, 2005.

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

[5] Becker, M. Y. and Sewell, P., "Cassandra: Flexible trust management, applied to electronic health records," in *Computer Security Foundations Workshop, 2004. Proceedings. 17th IEEE*, pp. 139–154, IEEE, 2004.

[6] Benaloh, J., Chase, M., Horvitz, E., and Lauter, K., "Patient controlled encryption: ensuring privacy of electronic medical records," in *Proceedings of the 2009 ACM workshop on Cloud computing security*, pp. 103–114, ACM, 2009.

[7] Blocki, J., Christin, N., Datta, A., and Sinha, A., "Audit mechanisms for privacy protection in healthcare environments.," in *HealthSec*, 2011.

[8] Blue Cross Blue Shield Association, "Understanding healthcare fraud," `http://www.bcbs.com/report-healthcare-fraud/`.

[9] Bock, B., Huemer, D., and Tjoa, A. M., "Towards more trustable log files for digital forensics by means of "trusted computing"," in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 1020–1027, IEEE, 2010.

[10] Bose, A. and Shin, K., "On mobile viruses exploiting messaging and bluetooth services," in *Securecomm and Workshops*, 2006.

[11] Cattell, R. B., "The scree test for the number of factors," *Multivariate behavioral research*, vol. 1, no. 2, pp. 245–276, 1966.

[12] CBS, "Dozens charged in $163m medicare fraud scheme," *CBSNews*, `http://www.cbsnews.com/news/dozens-charged-in-163m-medicare-fraud-scheme/`.

[13] Centers for Medicare & Medicaid Services, "Medicare fee-for service provider utilization and payment data part d prescriber public use file: A methodological overview," *2015*, `https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Downloads/Prescriber_Methods.pdf`.

[14] Centers for Medicare & Medicaid Services, "Medicare fee-for service provider utilization and payment data physician and other supplier public use file: A methodological overview," *2014*, `http://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Downloads/Medicare-Physician-and-Other-Supplier-PUF-Methodology.pdf`.

[15] Centers for Medicare & Medicaid Services, "Medicare provider utilization and payment data: Part d prescriber," *2015*, `http://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Part-D-Prescriber.html`.

[16] Centers for Medicare & Medicaid Services, "Medicare provider utilization and payment data: Physician and other supplier," *2014*, `http://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Physician-and-Other-Supplier.html`.

[17] Chowdhury, A. R., Falchuk, B., and Misra, A., "Medially: A provenance-aware remote health monitoring middleware," in *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pp. 125–134, IEEE, 2010.

[18] Cleveland, "University hospitals: Employee gained unauthorized access to 692 patient files in breach," *2014*, `http://www.cleveland.com/metro/index.ssf/2014/11/uh_employee_gained_improperly.html`.

[19] CNN, "Dozens arrested in health care fraud schemes," *2010*, `http://www.cnn.com/2010/CRIME/10/13/health.care.fraud/`.

[20] Das, K. and Schneider, J., "Detecting anomalous records in categorical datasets," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 220–229, ACM, 2007.

[21] Deora, V., Contes, A., Rana, O. F., Rajbhandari, S., Wootten, I., Tamas, K., and Varga, L. Z., "Navigating provenance information for distributed healthcare management," in *Proceedings of the 2006 IEEE/WIC/ACM*

International Conference on Web Intelligence, pp. 859–865, IEEE Computer Society, 2006.

[22] Dhopeshwarkar, R. V., Kern, L. M., O'Donnell, H. C., Edwards, A. M., and Kaushal, R., "Health care consumers' preferences around health information exchange," *The Annals of Family Medicine*, vol. 10, no. 5, pp. 428–434, 2012.

[23] Enck, W., Gilbert, P., Chun, B., Cox, L., Jung, J., McDaniel, P., and Sheth, A., "Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones," in *Proceedings of OSDI*, 2010.

[24] Enck, W., Ongtang, M., and McDaniel, P., "On lightweight mobile phone application certification," in *Proceedings of the 16th ACM conference on Computer and communications security*, ACM, 2009.

[25] Federal Bureau of Investigation, "Financial crimes report to the public," *2011*, `http://www.fbi.gov/stats-services/publications/financial-crimes-report-2010-2011/financial-crimes-report-2010-2011`.

[26] Forbes, "Data breaches in healthcare totaled over 112 million records in 2015," *2015*, `http://www.forbes.com/sites/danmunro/2015/12/31/data-breaches-in-healthcare-total-over-112-million-records-in-2015`.

[27] Francis, C., Pepper, N., and Strong, H., "Using support vector machines to detect medical fraud and abuse," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pp. 8291–8294, IEEE, 2011.

[28] Freire, J., Koop, D., Santos, E., and Silva, C. T., "Provenance for computational tasks: A survey," *Computing in Science & Engineering*, vol. 10, no. 3, pp. 11–21, 2008.

[29] Friedman, J. and Hoffman, D., "Protecting data on mobile devices: A taxonomy of security threats to mobile computing and review of applicable defenses," *Information, Knowledge, Systems Management*, vol. 7, no. 1, 2008.

[30] Gardner, R., Garera, S., Rubin, A., Rajan, A., Rozas, C., and Sastry, M., "Protecting patient records from unwarranted access," *Future of Trust in Computing*, pp. 122–128, 2009.

[31] Gardner, R., Garera, S., Pagano, M., Green, M., and Rubin, A., "Securing medical records on smart phones," in *Proceedings of the first ACM workshop on Security and privacy in medical and home-care systems*, pp. 31–40, ACM, 2009.

[32] Geambasu, R., John, J., Gribble, S., Kohno, T., and Levy, H., "Keypad: an auditing file system for theft-prone devices," in *Proceedings of the sixth conference on Computer systems*, pp. 1–16, ACM, 2011.

[33] Ghani, R. and Kumar, M., "Interactive learning for efficiently detecting errors in insurance claims," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 325–333, ACM, 2011.

[34] Google, "Designing for performance." `http://developer.android.com/guide/practices/design/performance.html`.

[35] Google, "Custom search," `https://developers.google.com/custom-search/`.

[36] Google, "Web search api," `https://developers.google.com/web-search/`.

[37] Government Accountability Office, "U.s. government accountability office - fraud awareness week," *2015*, `http://blog.gao.gov/2015/11/18/fraud-awareness-week/`.

[38] Government Computer News, "Health care fraud: We can't afford to pay and chase," *2015*, `https://gcn.com/articles/2015/09/03/health-care-fraud.aspx`.

[39] Grimaila, M. R., Myers, J., Mills, R. F., and Peterson, G., "Design and analysis of a dynamically configured log-based distributed security event detection methodology," *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, vol. 9, no. 3, pp. 219–241, 2012.

[40] Haas, S., Wohlgemuth, S., Echizen, I., Sonehara, N., and Müller, G., "Aspects of privacy for electronic health records," *International journal of medical informatics*, vol. 80, no. 2, pp. e26–e31, 2011.

[41] Hasan, R. and Yurcik, W., "A statistical analysis of disclosed storage security breaches," in *Proceedings of the second ACM workshop on Storage security and survivability*, ACM, 2006.

[42] Hasan, R., Sion, R., and Winslett, M., "Introducing secure provenance: problems and challenges," in *Proceedings of the 2007 ACM workshop on Storage security and survivability*, pp. 13–18, ACM, 2007.

[43] Hasan, R., Sion, R., and Winslett, M., "The case of the fake picasso: Preventing history forgery with secure provenance.," in *FAST*, vol. 9, pp. 1–14, 2009.

[44] Healthcare Information and Management Systems Society, "Creating a trusted environment: Reducing the threat of medical identity theft." `http://himss.files.cms-plus.com/HIMSSorg/content/`

files/CreatingaTrustedEnvironment_Reducing_the_Threat_of_Medical_Identify_TheftFINAL.pdf.

[45] HealthITSecurity, "Hacking accounts for 98% of healthcare data breaches in 2015," *2016*, http://healthitsecurity.com/news/hacking-accounts-for-98-of-healthcare-data-breaches-in-2015.

[46] Hupperich, T., Löhr, H., Sadeghi, A.-R., and Winandy, M., "Flexible patient-controlled security for electronic health records," in *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pp. 727–732, ACM, 2012.

[47] Integrating the Healthcare Enterprise, "Ihe it infrastructure technical framework volume 2a." http://www.ihe.net/uploadedFiles/Documents/ITI/IHE_ITI_TF_Vol2a.pdf.

[48] Jafari, S., Mtenzi, F., Fitzpatrick, R., and O'Shea, B., "An approach for developing comparative security metrics for healthcare organizations," in *Internet Technology and Secured Transactions, 2009. ICITST 2009. International Conference for*, pp. 1–6, IEEE, 2009.

[49] Jangde, P. and Mishra, D., "A secure multiparty computation solution to healthcare frauds and abuses," in *Intelligent Systems, Modelling and Simulation (ISMS), Second International Conference on*, IEEE, 2011.

[50] JASON, "A robust health data infrastructure," *2014*, https://www.healthit.gov/sites/default/files/ptp13-700hhs_white.pdf.

[51] Johnson, M. E., "Data hemorrhages in the health-care sector," in *Financial Cryptography and Data Security*, pp. 71–89, Springer, 2009.

[52] Jueneman, R., "Securing wireless medicine confidentiality, integrity, nonrepudiation, & malware prevention," in *Emerging Technologies for a Smarter World (CEWIT), International Conference on*, IEEE, 2011.

[53] Kifor, T., Varga, L., Álvarez, S., Vázquez-Salceda, J., and Willmott, S., "Privacy issues of provenance in electronic healthcare record systems," in *Proceedings of the 1st Int. Workshop on Privacy and Security in Agent-based Collaborative Environments (PSACE 2006)*, 2006.

[54] King, J., Smith, B., and Williams, L., "Audit mechanisms in electronic health record systems: Protected health information may remain vulnerable to undetected misuse," *International Journal of Computational Models and Algorithms in Medicine (IJCMAM)*, vol. 3, no. 2, pp. 23–42, 2012.

[55] King, J. and Williams, L., "Secure logging and auditing in electronic health records systems: what can we learn from the payment card industry," in *Proceedings of the 3rd USENIX conference on Health Security and Privacy*, pp. 13–13, USENIX Association, 2012.

[56] King, J. T., Smith, B., and Williams, L., "Modifying without a trace: general audit guidelines are inadequate for open-source electronic health record audit mechanisms," in *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pp. 305–314, ACM, 2012.

[57] Kirlidog, M. and Asuk, C., "A fraud detection approach with data mining in health insurance," *Procedia-Social and Behavioral Sciences*, vol. 62, pp. 989–994, 2012.

[58] Kotz, D., Avancha, S., and Baxi, A., "A privacy framework for mobile health and home-care systems," in *Proceedings of the first ACM workshop on Security and privacy in medical and home-care systems*, 2009.

[59] Lakhina, A., Crovella, M., and Diot, C., "Diagnosing network-wide traffic anomalies," in *ACM SIGCOMM Computer Communication Review*, vol. 34, pp. 219–230, ACM, 2004.

[60] Lee, J.-H., Park, M.-W., Eom, J.-H., and Chung, T.-M., "Multi-level intrusion detection system and log management in cloud computing," in *Advanced Communication Technology (ICACT), 2011 13th International Conference on*, pp. 552–555, IEEE, 2011.

[61] Li, J., Huang, K.-Y., Jin, J., and Shi, J., "A survey on statistical methods for health care fraud detection," *Health Care Management Science*, vol. 11, no. 3, pp. 275–287, 2008.

[62] Li, M., Yu, S., Ren, K., and Lou, W., "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," in *Security and Privacy in Communication Networks*, pp. 89–106, Springer, 2010.

[63] Ma, D. and Tsudik, G., "A new approach to secure logging," *ACM Transactions on Storage (TOS)*, vol. 5, no. 1, p. 2, 2009.

[64] Markle, "The architecture for privacy in a networked health information environment." `http://www.markle.org/sites/default/files/P1_CFH_Architecture.pdf`.

[65] Mashima, D. and Ahamad, M., "Enabling robust information accountability in e-healthcare systems.," in *HealthSec*, 2012.

[66] Medicare, "Whats medicare?," `https://www.medicare.gov/sign-up-change-plans/decide-how-to-get-medicare/whats-medicare/what-is-medicare.html`.

[67] Microsoft, "Bing search api," *Azure*, `https://datamarket.azure.com/dataset/bing/search`.

[68] Mohan, A., Bauer, D., Blough, D. M., Ahamad, M., Bamba, B., Krishnan, R., Liu, L., Mashima, D., and Palanisamy, B., "A patient-centric, attribute-based, source-verifiable framework for health record sharing," 2009.

[69] Monrose, F., Reiter, M., and Wetzel, S., "Password hardening based on keystroke dynamics," *International Journal of Information Security*, vol. 1, no. 2, pp. 69–83, 2002.

[70] Morris, L., "Combating fraud in health care: an essential component of any cost containment strategy," *Health Affairs*, vol. 28, no. 5, pp. 1351–1356, 2009.

[71] Moss, L., Corsar, D., and Piper, I., "A linked data approach to assessing medical data," in *Computer-Based Medical Systems (CBMS), 2012 25th International Symposium on*, pp. 1–4, IEEE, 2012.

[72] Myers, J., Grimaila, M., and Mills, R., "Towards insider threat detection using web server logs," in *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*, ACM, 2009.

[73] National Health Care Anti-Fraud Association, "Combating health care fraud in a post-reform world: Seven guiding principles for policymakers." October 6, 2010.

[74] National Health Care Anti-Fraud Association, "The challenge of health care fraud," http://www.nhcaa.org/resources/health-care-anti-fraud-resources/the-challenge-of-health-care-fraud.aspx.

[75] Nauman, M., Khan, S., and Zhang, X., "Apex: extending Android permission model and enforcement with user-defined runtime constraints," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ACM, 2010.

[76] Nematzadeh, A. and Camp, L., "Threat analysis of online health information system," in *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, ACM, 2010.

[77] Ng, K., Shan, Y., Murray, D., Sutinen, A., Schwarz, B., Jeacocke, D., and Farrugia, J., "Detecting non-compliant consumers in spatio-temporal health data: A case study from medicare australia," in *Data Mining Workshops (ICDMW), IEEE International Conference on*, IEEE, 2010.

[78] Ongtang, M., McLaughlin, S., Enck, W., and McDaniel, P., "Semantically Rich Application-Centric Security in Android," in *2009 Annual Computer Security Applications Conference*, IEEE, 2009.

[79] Ortega, P. A., Figueroa, C. J., and Ruz, G. A., "A medical claim fraud/abuse detection system based on data mining: A case study in chile.," *DMIN*, vol. 6, pp. 26–29, 2006.

[80] Pamila, J. and Thanushkodi, K., "Log management support for recovery in mobile computing environment," *arXiv preprint arXiv:0908.0076*, 2009.

[81] Parks, R., Chu, C.-H., and Xu, H., "Healthcare information privacy research: Iusses, gaps and what next?," in *AMCIS*, Citeseer, 2011.

[82] Peng, Y., Kou, G., Sabatka, A., Chen, Z., Khazanchi, D., and Shi, Y., "Application of clustering methods to health insurance fraud detection," in *Service Systems and Service Management, 2006 International Conference on*, vol. 1, pp. 116–120, IEEE, 2006.

[83] Pepper, N., Strong, H., and Lynagh, K., "Qyz: A platform for visual analysis of error, abuse, and fraud in medical bills," in *IEEE VisWeek Workshop on Visual Analytics in Health Care*, 2010.

[84] Phua, C., Lee, V., Smith, K., and Gayler, R., "A comprehensive survey of data mining-based fraud detection research," *arXiv preprint arXiv:1009.6119*, 2010.

[85] Ponemon Institute, "Fifth annual benchmark study on privacy & security of healthcare data." 2015.

[86] Ponemon Institute, "Third annual survey on medical identity theft." 2012.

[87] Prasad, A., Peterson, R., Mare, S., Sorber, J., Paul, K., and Kotz, D., "Provenance framework for mhealth," in *Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on*, pp. 1–6, IEEE, 2013.

[88] Rabkin, A. and Katz, R., "Chukwa: A system for reliable large-scale log collection," in *Proceedings of the 24th international conference on Large installation system administration*, USENIX Association, 2010.

[89] Reuters, "Phantom firms bleed millions from medicare," *Special Report*, `http://www.reuters.com/article/2011/12/21/us-shellcompanies-medicare-idUSTRE7BK0PY20111221`.

[90] Rostad, L., "An initial model and a discussion of access control in patient controlled health records," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, pp. 935–942, IEEE, 2008.

[91] Sana, "Sana mobile." `http://www.sanamobile.org/`.

[92] Shabtai, A., Fledel, Y., and Elovici, Y., "Securing Android-Powered Mobile Devices Using SELinux," *IEEE Security and Privacy*, 2009.

[93] Simmhan, Y. L., Plale, B., and Gannon, D., "A survey of data provenance techniques," *Computer Science Department, Indiana University, Bloomington IN*, vol. 47405, 2005.

[94] Simon, S. R., Evans, J. S., Benjamin, A., Delano, D., and Bates, D. W., "Patients' attitudes toward electronic health information exchange: qualitative study," *Journal of medical Internet research*, vol. 11, no. 3, 2009.

[95] Smith, B., Austin, A., Brown, M., King, J. T., Lankford, J., Meneely, A., and Williams, L., "Challenges for protecting the privacy of health information: required certification can leave common vulnerabilities undetected," in *Proceedings of the second annual workshop on Security and privacy in medical and home-care systems*, pp. 1–12, ACM, 2010.

[96] Stathopoulos, V., Kotzanikolaou, P., and Magkos, E., "Secure log management for privacy assurance in electronic communications," *computers & security*, vol. 27, no. 7, pp. 298–308, 2008.

[97] Syalim, A., Nishide, T., and Sakurai, K., "Securing provenance of distributed processes in an untrusted environment," *IEICE TRANSACTIONS on Information and Systems*, vol. 95, no. 7, pp. 1894–1907, 2012.

[98] Tagaris, A., Mnimatidis, P., and Koutsouris, D., "Implementation of a prescription fraud detection software using rdbms tools and atc coding," in *Information Technology and Applications in Biomedicine, 2009. ITAB 2009. 9th International Conference on*, pp. 1–4, IEEE, 2009.

[99] Tharaud, J., Wohlgemuth, S., Echizen, I., Sonehara, N., Müller, G., and Lafourcade, P., "Privacy by data provenance with digital watermarking-a proof-of-concept implementation for medical services with electronic health records," in *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2010 Sixth International Conference on*, pp. 510–513, IEEE, 2010.

[100] The Economist, "The $272 billion swindle," *2014*, `http://www.economist.com/news/united-states/21603078-why-thieves-love-americas-health-care-system-272-billion-swindle`.

[101] The Health Information Trust Alliance (HITRUST), "U.s. healthcare data breach trends." December 2012.

[102] The New York Times, "Pursuit of cash taints promise of gene tests," *2015*, `http://www.nytimes.com/2015/06/25/technology/genetic-testing-case-highlights-the-fields-hope-and-hype.html?\_r=0`.

[103] The New York Times - City Room Blog, "44 charged in huge medicare fraud scheme," *2010*, `http://cityroom.blogs.nytimes.com/2010/10/13/44-charged-in-huge-medicare-fraud-scheme/`.

[104] The Office of Inspector General, "Questionable billing for medicare part b clinical laboratory services questionable billing for medicare part b clinical laboratory services questionable billing for medicare part b clinical

laboratory services," *2014*, http://oig.hhs.gov/oei/reports/oei-03-11-00730.pdf.

[105] The Office of Inspector General, "Office of evaluation and inspections," *Office of Inspector General*, http://oig.hhs.gov/reports-and-publications/oei/b.asp.

[106] The Sequoia Project, "ehealth exchange," http://sequoiaproject.org/ehealth-exchange/.

[107] The United States Department Of Justice, "Two cardiovascular disease testing laboratories to pay $48.5 million to settle claims of paying kickbacks and conducting unnecessary testing," *2015*, http://www.justice.gov/opa/pr/two-cardiovascular-disease-testing-laboratories-pay-485-million-settle-claims-paying.

[108] The Wall Street Journal, "Lab nears settlement over pricey medicare drug tests," *2015*, http://www.wsj.com/articles/lab-nears-settlementover-pricey-medicare-drug-tests-1434326131.

[109] The Washington Post, "2015 is already the year of the health-care hack and its only going to get worse.," *2015*, https://www.washingtonpost.com/news/the-switch/wp/2015/03/20/2015-is-already-the-year-of-the-health-care-hack-and-its-only-going-to-get-worse/.

[110] Tomono, A., Uehara, M., Murakami, M., and Yamagiwa, M., "A log management system for internal control," in *Network-Based Information Systems, 2009. NBIS'09.*, pp. 432–439, IEEE, 2009.

[111] Tukey, J. W., "Exploratory data analysis," 1977.

[112] United States, "Department of justice," http://www.justice.gov/.

[113] United States Department of Health & Human Services, "Summary of the health insurance portability and accountability act (hipaa) security rule." http://www.hhs.gov/ocr/privacy/hipaa/understanding/srsummary.html.

[114] United States Department of Health & Human Services, "Summary of the hipaa privacy rule." http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/index.html.

[115] Van Deursen, N., Buchanan, W. J., and Duff, A., "Monitoring information security risks within health care," *computers & security*, vol. 37, pp. 31–45, 2013.

[116] Viswanath, B., Bashir, M. A., Crovella, M., Guha, S., Gummadi, K. P., Krishnamurthy, B., and Mislove, A., "Towards detecting anomalous user behavior in online social networks," in *Proceedings of the 23rd USENIX Security Symposium (USENIX Security)*, 2014.

[117] Waters, B., Balfanz, D., Durfee, G., and Smetters, D., "Building an encrypted and searchable audit log.," in *NDSS*, vol. 4, 2004.

[118] Williams, G. J., "Evolutionary hot spots data mining," in *Methodologies for Knowledge Discovery and Data Mining*, pp. 184–193, Springer, 1999.

[119] Yang, W. and Hwang, S., "A process-mining framework for the detection of healthcare fraud and abuse," *Expert Systems with Applications*, 2006.

[120] Yoo, I., Alafaireet, P., Marinov, M., Pena-Hernandez, K., Gopidi, R., Chang, J.-F., and Hua, L., "Data mining in healthcare and biomedicine: a survey of the literature," *Journal of medical systems*, vol. 36, no. 4, pp. 2431–2448, 2012.

[121] Zeldovich, N., Boyd-Wickizer, S., Kohler, E., and Mazières, D., "Making information flow explicit in histar," in *Proceedings of the 7th symposium on Operating systems design and implementation*, pp. 263–278, USENIX Association, 2006.

[122] Zhang, R. and Liu, L., "Security models and requirements for healthcare application clouds," in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pp. 268–275, IEEE, 2010.

[123] Zimek, A., Schubert, E., and Kriegel, H.-P., "A survey on unsupervised outlier detection in high-dimensional numerical data," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 5, pp. 363–387, 2012.

[124] Zulman, D. M., Nazi, K. M., Turvey, C. L., Wagner, T. H., Woods, S. S., and An, L. C., "Patient interest in sharing personal health record informationa web-based survey," *Annals of internal medicine*, vol. 155, no. 12, pp. 805–810, 2011.