Learning From Access Logs to Mitigate Insider Threats

By

Wen Zhang

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

May, 2016

Nashville, Tennessee

Approved:

Professor Bradley Malin

Professor Jules White

Professor Gautam Biswas

Professor Yuan Xue

Professor Carl Gunter

This thesis is dedicated to my lovely daughter, Jingyan.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deep gratitude to my advisor, Dr. Bradley Malin, for his support and guidance throughout my PhD study. His broad vision and extraordinary insight are always the source of my inspiration and ideas. Also, his patience and humor made my PhD study not as tough as imaged. I extend my gratitude to Dr. Carl Gunter and Dr. David Liebovitz for providing valuable suggestions and help to my project. My thanks also go to all the collabrators for their help in various form: Steve Nyemba, Jian Tian, Thaddeus Cybulski, Patrick Lawlor and Dr. You Chen. I would also like to thank Dr. Yuan Xue, Dr. Biswas Gautam and Dr. Jules White for serving on my PhD committee. Finally, my heartful thanks go to my wife for her selfless support and understanding, and my parents for their constant love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

The insider threat arises when an employee in an organization abuses their permissions to harm the security of the organization's information system. The insider threat has become one of the greatest security challenges today, due to a number of reasons. First, the insider threat is a common problem. The U.S. State of Cybercrime survey, conducted by the U.S. Secret Service, Carnegie Mellon University, CSO Magazine and Deloitte, found that 23% of electronic crime events are caused by insiders [1]. A Forrester survey in 2013 reports the insider threat is the top cause of data breaches that transpired in organizations who responded to the survey[2]. Moreover, the Forrester survey shows the problem is not localized to any specific country, but is rather dispersed across a range of nations, including the US, Canada, UK, France and Germany. Second, the insider threat leads to substantial monetary cost. Respondents to the U.S. State of Cybercrime survey indicated that insiders' malicious activities are more costly and damaging than incidents perpetrated by outsiders. Indeed, we can see that many security incidents resulting in heavy losses are caused by insiders. For example, in 2011, the UCLA Health System paid the federal government $865,000 for failing to prevent its employees from snooping in the electronic medical records of two celebrity patients [3]. In fact, the number of institutions that are fined due to neglecting insider threat is growing at a fast pace [4].

In general, the insider threat can be addressed by two categories of strategies. The first is what we refer to as a prospective strategy, which makes a decision (i.e., denial or approval) about an user's access at the time of request. Most access control systems fall into this category, including Mandatory Access Control [5], Discretionary Access Control [6], Role-based Access Control (RBAC) [7, 8, 9] and Context-based Access Control [10, 11]. Among them, RBAC is the most widely adopted system [12]. The second strategy to

address the insider threat is what we refer to as a retrospective strategy, which permits the access to proceed but reviews it afterward. Post-hoc audits [13, 14, 15, 16] fall into this category. The retrospective strategy is usually adopted in a mission-critical system [17], such as an electronic medical record (EMR) system.

1.1 Motivation and Research Objectives

When there is a need to design an insider threat mitigation system for an organization, two important questions are often raised. First, what strategy (i.e., prospective vs. retrospective) should be adopted for the target organization? Second, once one strategy is chosen, what specific method should be used to implement it? To the best of our knowledge, there is no clear answer for the first question. As for the second question, most of the existing solutions solely rely on expert knowledge or experience, while ignoring information hidden in the massive access log that may help in the refinement of the system design. In this disseration, we faciliate the design of insider threat mitigation systems by answering both questions from a data-centric perspective. Note this does not preclude the possibility that other questions need to be addressed for the satisfactory design of such a system.

1.1.1 Objective 1: Quantify the Trade-off between Prospective and Retrospective Strategies

Given an access request, an access control system is asked to label it as legitimate or illegitimate. Thus, one access control system can be viewed as a binary classification model. Naturally, the reader may think of using typical performances measures, such as accuracy, F-measure or ROC analysis, to compare classifers (i.e., access control systems). However, these measures are problematic in the context of comparing prospective and retrospective systems for several reasons.

First, accuracy and F-measure are used under the assumption that costs of false positive and false positive are equivalent (assumption 1). Second, although ROC analysis is not

subject to assumption 1, its validity is based on another assumption specifically that costs of false positive (negative) across different classifiers are equivalent (assumption 2). Unfortunately, neither assumption holds in our environment. Thus, one of the objectives in this dissertation is to resolve the problem of comparing two access control systems (classifiers) without assumption 1 and assumption 2 being true.

1.1.2 Objective 2: Data-Driven Security System Design

In most situations, both of the strategies (especially the prospective approach), are implemented based only on expert knowledge. An access log, which records each access to resource in system, is rarely leveraged during the application of those security strategies. Yet an access log could be very useful to either strategy, because the access pattern in the log may reveal some error or weakness in the existing security configuration. Thus, the second problem studied in this thesis is how to diagnose, revise and evolve an existing security system, including prospective and retrospective systems, by applying knowledge discovery methods.

1.2 Contributions

This dissertation introduces several novel approaches to address each of objectives in different scenarios. For objective 1, we propose bispective analysis, a novel decision support framework, to quantify the trade-off between prospective and retrospective strategies. For objective 2, we develop two different role revision algorithms to optimize two different objective functions, and a simple but effective workflow-based audit framework with a machine learning foundation.

- In Chapter 4, we develop another role revision algorithm called Data-Driven Role Evolution (DDRE) algorithm. This algorithm uses knowledge mined from access logs as well as knowledge from experts to faciliate the RBAC design. An objective

function considering both role homogeneity and expert belief is introduced. We also perform an empirical analysis with real and simulated datasets to show that our algorithm can generate appropriate RBAC configurations for various biases of the two competing goals of the objective function.

- In Chapter 5, we devise a procedure called role prediction to measure the quality of role specifications in RBAC, then develop a heuristic-based algorithm, called Role-Up, to abstract existing roles along a pre-defined role hierarchy to achieve high quality roles. Our findings suggest that RBAC for EMR systems can be effectively guided through information mined from audit logs.

- In Chapter 6, we introduce a workflow based audit system to detect suspicious accesses. It consists of two phases. First, we develop methods to extract attributes from workflow sequences. Second, on the labeled accesses characterized by attributes, we build a classification model with a statistical machine learning algorithm. From experimental results, we find it achieves satisfactory results.

- In Chapter 7, we devise a novel cost comparison method called bispective analysis that allows for an explicit comparison of classification models without assumption 1 and 2. Typically, we derive a comparison function whose outputs can reflect correct choices between classifiers and generate its contour plot. Once provided with the knowledge of the variables (i.e., the costs of false positive and false negative for the prospective model, the costs of false positive and false negative for the retrospective model, and the receiver operator characteristic (ROC) curves for both models), bispective analysis allows administrators to determine the better option by studying a contour plot. Moreover, bispective analysis provides insight about the distribution of results under varying cost models, such that administrators can make decisions when their confidence in the variables is uncertain (e.g., only a range of costs are known or only partial costs are known).

The research communicated through this dissertation will be useful to multiple communities. 1) Security researchers or security administrators will be interested in the proposed insider threat mitigation system, and the quantification of the trade-off between the prospective and retrospective systems. 2) Machine learning researchers will be interested in how to use bispective analysis to make a comparison between classification when traidtional cost analysis (e.g., ROC) would not work.

1.3 Dissertation Outline

The reminder of this dissertation is organized as follows. Chapter 2 provides surveys on relevant literatures, and points out their limitations. Chapter 3 introduces a dataset generated from a real-world information system, which will be used for evaluation purposes. Chapter 4 and Chapter 5 introduce two data driven approaches to refine an existing RBAC system. Next, Chapter 6 introduces a novel but simple audit system. After that, Chapter 7 describes a novel technique that can support meaningful decisions between the prospective and retrospective methods. Finally, Chapter 8 concludes this dissertation, discusses the limitations and future directions.

Chapter 2

Related Work

This chapter begins with a survey on the techniques related to quantifying the tradeoff between prospective and retrospective strategies. Then, we review techniques related to each of them.

2.1 Work Related to Comparison of Prospective and Retrospective Strategies

In this section, we review existing cost-based security models. We then review methodologies to compare classification models and the limitations associated with applying them to the prospective versus retrospective model analysis.

2.1.1 Cost-based Security Models

According to the National Institutes of Standards and Technologies (NIST) [18], organizations should rate their information systems in terms of risk across three class: i) low, ii) medium, or iii) high. An organization should then adopt their security protections proportional to such risk. However, the selected security control may not be appropriate in that the rating for impact is highly subjective.

To reduce subjectivity in decision making, several risk-based strategies have been suggested for information security management. In particular, based on the recognition that business processes are often disrupted by static and rigid policies, many of these strategies are focused on access control. Here we review the approaches most related to our own. First, [19] proposed an adaptive access control model to balance the tradeoff between risk and utility in dynamic environments. They create a system that encourages information sharing among multiple organizations while keeping its users accountable for their actions

and capping the expected damage an organization could suffer due to sensitive information disclosure. In relation to our own work, they introduce a method to compute the expected risk based on 1) the uncertainty and 2) the cost associated with an incorrect decision. Second, [20] introduced a policy-based access control model to infer a decision for an incoming access. This is achieved by training classifiers, using machine learning, on known decisions and subsequently inferring the new decision when there is no exact matching pattern. By doing so, each access decision is assigned a certain degree of risk. Third, [21] introduced the Benefit and Risk Access Control (BARAC) system, which identifies a set of correlated access requests as a closed system. Based on this system, this method uses a graph-based model to make a decision for each access, such that the cost of the entire system is minimized. All of these lines of research are significantly different from this dissertation in that they focus on decisions between prospective access control models with constant misclassification costs, whereas we investigate a decision between prospective and retrospective models with varying costs.

2.1.2 Comparison of Classification Models

There are a number of performance measures that can be applied to assess the robustness of a classification model. For instance, one could assess the accuracy; i.e., the proportion of total instances that are correctly labeled by the model. However, accuracy is a biased assessment because it assumes that false positives and negatives occur at the same rate and are equally costly. As such, a more nuanced strategy for assessing classification models is to measure the Receiver operating characteristic (ROC) under a range of acceptance levels for false positive and false negative thresholds. In doing so, the area under the curve (AUC) indicates the agility of a classifier, where the "best" classifier is the one that maximizes this value. The AUC has been invoked as a common approach for assessing various classification models for information security, such as intrusion detection systems (e.g., [22]), malware detection (e.g., [23]), and auditing techniques for EMRs (e.g., [24]).

We recognize the relevance of machine learning (for which AUC is a popular evaluation measure) for information security has been questioned [25]. Yet, we stress that our goal is to assess how misclassification costs, rather than the machine learning algorithm itself, influence information security decisions. AUC also has serious deficencies in itself, 1) it is misleading when ROC curves cross and 2) it makes an unrealistic assumption on costs [26].

[27] proposed using a method to analyze the ROC convex hull to compose a dominant classification strategy over a set of classifiers and class frequencies (in the form of prior probabilities). This method begins by constructing a convex hull from all ROC curves (classifiers) to be compared and then determines which point in the convex hull corresponds to the least overall cost, given the costs of each classifier and prior probabilities. A key advantage of this method is that it needs only the ratio of costs and ratio of class frequencies to compose the optimal classifier, such that it is robust to a changing environment.

Subsequently, [28] introduced an alternative to traditional ROC analysis, which is called a cost curve. In this model, the expected cost of a classifier is represented as a function of costs and class frequencies, such that the expected cost can be computed explicitly. A cost curve provides several benefits over the traditional ROC convex hull, including: 1) given specific cost estimates and prior probabilities, it is easy to "read off" the expected cost, 2) it is immediately clear which, if any, classifier is the dominant strategy, and 3) it is straightforward to determine how much one classifier outperforms another. Building on this work, [26] introduced an approach to compares classifiers by computing their expected overall cost, in terms of a unified assumption on the probability density function of the costs of false positives (negatives).

However, in all of the aforementioned techniques, it is assumed that the costs (or cost distributions) of false positive (false negative) for both classifiers are equivalent. Yet, this is clearly not the case in our situation, which implies that such strategies could incorrectly select a model. In fact, we verify this to be the case in our empirical analysis of Chapter 7.

2.2 Existing Methodologies to Design Role-based Access Control (RBAC) System

This section reviews techniques implementing prospective startegies (i.e., Access Control). It begins with an introduction of the RBAC model, which forms the basis of our data-driven access control evolution algorithms. Then, we review a set of methodologies to design an RBAC system.

2.2.1 Role-based Access Control

RBAC is a framework that has been adopted widely for managing the rights of users in information system. It was designed to simplify the allocation of access rights by mapping users to a set of roles, each of which is associated with a set of permissions. A basic RBAC configuration contains five elements: users, permissions, roles, user-role assignment (*URA*) and role-permission assignment (*RPA*) [8]. *URA* is a Boolean matrix indicating the mapping between users and roles, and *RPA* is a Boolean matrix indicating the mapping between roles and permissions. This basic RBAC is also called $RBAC_0$ model, but there are extensions, such as $RBAC_1$ and $RBAC_2$, which introduce role hierarchies and constraints, respectively [8].

The process of defining roles, which is often referred to as role engineering [29], is a notoriously challenging problem and a core part of RBAC design. In general, role engineering approaches have fallen into two camps: i) *top-down* and ii) *bottom-up*. In the top-down setting, organizational experts (or system administrators) model the workflows associated with an enterprise, which are subsequently decomposed into tasks and roles [30, 31]. Bottom-up approaches (e.g., [32, 33]), on the other hand, discover roles by leveraging information that already exists in the system. Many of these approaches (e.g., [32, 34, 33, 35]) propose roles based on patterns in existing user-permission assignment. Formally, this is represented by a Boolean matrix indicating the mapping between users and permissions, called *UPA* in this dissertaion.

2.2.2 Role Engineering

There have been various approaches proposed for top-down approaches (e.g., [31, 30, 36]), but given the time-consuming and costly nature of this approach, it has limited adoption in real settings [30].

Thus, over the past decade, there has been a growing interest in bottom-up approaches, which enables role engineering to be automated with significantly lower cost. Here, we highlight several approaches that are conceptually similar to our work in that they iteratively build larger permission sets for roles. In [33], the goal is to minimize the number of roles and permissions per role. It was shown that this problem is computationally challenging and so a greedy heuristic-driven algorithm was proposed. The algorithm consists of two phases: in the first phase, *FastMiner* [34] produces a set of candidate roles by intersecting each pair of permission sets of users, and then, in the second phase, candidates with the greatest ability to cover the *UPA* (i.e., 1's in the matrix) are selected until coverage is complete. Alternatively, [32] proposes the ORCA algorithm, which generates roles by performing a hierarchical clustering on permission sets. In this process, the quality of a cluster (role) corresponds to the number of users associated with it. [35] use graphs to represent the relations among users, roles, and permissions, and then employs graph optimization to solve the role mining problem. The process begins with a set of possible roles, which is composed of the permission sets of all users. Next, pairs of roles are iteratively selected and are split or merged, to gain the largest improvement on the optimization measure of the resulting graph. While these strategies propose roles, they do not attempt to maximize homogeneity and minimize the distance to an existing set of roles. [37] proposes a role engineering method that leverages organizational information to generate a set of roles with clear business meaning. The method first partitions the data set (user-permission assignment) according to certain appropriate business information (e.g. an organization unit). Next, they adopt a divide-and-conquer approach that performs role mining on each subset. This approach may produce RBAC that is close to that built by administrators or experts

due to the use of business information that is often used in top-down role engineering. However, like other role mining algorithms, it does not leverage the information recorded in access logs, such that semantically meaningful roles could not be searched.

There have been several approaches proposed which attempt to revise roles and leverage permission utilization patterns (which we empirically compared to in the previous section). [38] defines the minimal perturbation role mining problem, whose objective is to find a set of roles that has both small distance to the original roles and a small number of roles in total. $f(R) = w \cdot k + (1-w) \cdot k \cdot D$ describes the corresponding objective function, where k is the number of roles, D is the distance between old and new role sets, and w is a parameter used to control the balance between k and D. In this method, a role is constantly selected from the candidate role sets produced by *FastMiner* [34] according to its value on a heuristic function $f(r) = w \cdot a + (1-w) \cdot a \cdot d$, where a is the remaining 1's in *UPA* covered by this role and d is the distance between this role and the original role set. The selection process terminates when *UPA* is covered by the selected roles. However, this work is limited in that it neither takes the users' behavior into consideration, nor does it measure the similarity of RBAC configurations. Rather, it only uses the similarity between two role sets. By contrast, [39] takes user behavior into consideration and proposes a simulated annealing approach to mine *URA* and *RPA* with the usage of privileges. This approach begins with a random initialization of *URA* and *RPA*, which is derived from the probability distribution of users over roles, and the probability distribution of roles over permissions calculated by the LDA model learned from the access log. It then iteratively decides if a new pair of *URA* and *RPA* matrices would be accepted to replace the old ones by a λ-distance (a measure of how well they explain the usage of permissions). For simplicity, the resulting *URA* and *RPA* does not necessarily have to be consistent with the original *UPA*. Thus, this work is significantly different than ours in that the resulting RBAC configuration is not necessarily subject to $UPA = URA \otimes RPA$ [1].

[1] x = a $\otimes$ b denotes the boolean matrix product, in which an element is defined as $x_{ij} = \vee_k (a_{ik} \wedge b_{kj})$

2.3 Existing Audit Methods

There have been a number of papers on developing audit system. Unfortunately, most of them detect anomalies on user basis. In other words, they aim to determine whether or not a user is behaving suspiciously. However, in many situations (e.g., healthcare), a finer-grained monitoring system which can evaluate each single access is desirable. There has been limited work in this area, but there are several notable publications that suggest this is feasible. Chen et al. [40] propose a model to monitor an access by measuring the deviation of an access to the collaborative network profile it belongs to. However, due to the utilization of social network, this model does not take advantage of other discriminative attributes but users in the network. Zhang et al. [41] propose a graph based model to detect anomalies. This model learns a graph from data to be the profile of the workflow for each medical service. However, only job title sequence was used in this model. By contrast, WOrkflow Based Audit (WOBA) allows any possible features to function in its framework, thus is much more extensible. In [14, 42], an explanation-based auditing is proposed for EHR system. This audit system proposes a mining algorithm to find reasons from the database for each access. However, this model is specialized at explaining legitimate accesses rather than detecting malicious accesses.

Chapter 3

Electronic Medical Record

Since this dissertation explores using knowledge discovered from an access log to facilitate security system design, it is critical to have a real-world data set to perform empirical investigations over. Thus, in this chapter, we introduce an access log, which is extracted from the electronic medical record (EMR) system in place at Northwestern Memorial Hospital (NMH).

NMH is an 854 bed primary teaching affiliate for the Feinberg School of Medicine at Northwestern University. All clinicians (including physicians and nurses) retrieve clinical content and enter inpatient notes and orders online using the Cerner Corporations PowerChart EMR system. The access logs generated by the system consist of user- and patient-specific information as summarized in Table 3.1.

When approved by an authorizing entity, (e.g., the Medical Staff Office), each user of the system receives a login ID tied to a User Position. The User Positions enable or prevent access to specific EMR functions. As an example, a medical student orders require co-signing by a physician. As another example, specific administrative roles do not provide comprehensive result flow-sheet access.

As an additional safeguard, users select a Chart Access Reason upon first access to a chart for a particular encounter. The available Chart Access Reasons displayed for selection are tied to the individuals User Position. Selected User Positions with minimal use case scenarios have only one potential Chart Access Reason and are therefore not prompted. An encounter in this context is defined as a hospital visit and is more narrowly specified for the research cohort below.

The cohort of accesses for this dissertation covers a 3 month period of time for which patients were either in an inpatient status or an observation encounter status. Observation

Table 3.1: A summary of the data captured in the Northwestern EMR access logs.

	Attribute	Description
1	User ID	Login credentials (de-identified)
2	Encounter ID	Treatment ID for the user
3	Patient ID	Medical record number (de-identified for cohort)
4	User Position	Assigned role within the medical record system
5	Date and Time Stamp	Dates were randomly shifted in a 365 day period for de-identification purposes
6	Chart Access Reason	Option selected when a chart is first accessed by each user during a hospitalization. Options available are tied to the User Position
7	Orders Entered	Indicates the number of order entered by the user during the current chart access
8	Location	General location of the patient within the hospital
9	Service	The hospital service caring for the patient as specified by the doctors caring for the patient. If the field is blank (Obstetrics service, e.g.), the specialty of the attending physician is used

User	Patient	Time	Service	User Position	Reason	Location
u_1	p_1	8/4/10	OBSTETRICS	NMH Physician Office - CPOE	Attending Phys/Prov	Ward A
u_2	p_2	12/14/10	OBSTETRICS	NMH Physician - CPOE	Patient Care	Ward A
u_{23}	p_3	12/14/10	PEDIATRICS	Unit Secretary 2	Unit Secretary Orders	Ward B

Figure 3.1: A fictional example of records in the Northwestern EMR access logs.

status refers to an admission for which discharge is expected within 24 hours. An example of such a log is presented in Figure 3.1.

Each entry in the access logs corresponds to one access to the EHR, including the information on the user, patient, reason for the access, type of service, location where the access happens, and whether orders or notes activity occurred. For the purpose of privacy, the names of patient and users are replaced by pseudonyms. Moreover, for the purposes of our study, the User Position is considered to be a surrogate for the role. There are 8,095 users and 140 different roles involved by this log. Summary statistics for users and roles with respect to Reasons, Locations, Services, and accesses are provided in Table 3.2.

Table 3.2: Statistics for the EMR access log

	Users	Roles	Reasons	Locations	Services	Accesses
Total	8095	140	143	58	43	1,138,555
Average per user	-	-	2	10	9	140
Average per role	-	-	4	23	20	8,132

Chapter 4

Evolving Role Definitions Through Permission Invocation Patterns

Role based access control (RBAC) is an important prospective security strategy to mitigate the insider threat. In RBAC, roles are traditionally defined as sets of permissions. Roles specified by administrators may be inaccurate, however, such that data mining methods have been proposed to learn roles from actual permission utilization. These methods minimize variation from an information theoretic perspective, but they neglect the expert knowledge of administrators. In this chapter, we propose a strategy to enable a controlled evolution of RBAC based on utilization. To accomplish this goal, we extend a subset enumeration framework to search candidate roles for an RBAC model that addresses an objective function which balances administrator beliefs and permission utilization. The rate of role evolution is controlled by an administrator-specified parameter.

This chapter is organized as follows. Section 4.1 introduces the motivation and background of our work. Section 4.2 reviews the foundations upon which our method is built, including role mining problem, access logs, distance measures, and outlier detection methods. Section 4.3 then describes our role evolution algorithm. Section 4.4 presents experiments performed to evaluate our approach.

4.1 Introduction

As mentioned in Chapter 2, role engineering approaches have fallen into two camps: top-down and bottom-up. There are benefits and drawbacks to each camp. Top-down approaches, for instance, are based on expert reasoning, in-depth interviews, and tend to reflect organizational expectations [43]. However, these approaches often result in high costs to an enterprise [30] because they require a substantial amount of time to document

the workflows which exist. They may also be subject to the problem of informant inaccuracy [44] and, thus, access control models which are incomplete or contain errors [45]. By contrast, bottom-up approaches enable an RBAC system to be derived automatically, such that their cost is significantly lower than their top-down counterparts. Yet, there is no guarantee that users in the same role, as defined by their permissions, will exhibit similar behavior.

Historically, role engineering strategies have treated these camps independently, but we believe there is merit in combining them into a more comprehensive role engineering framework. Consider, while it may be that expert-specified RBAC configurations are not entirely representative of an enterprise, it is unlikely that such information is completely uninformed. As such, the goal of this chapter is to propose a role engineering approach that evolves roles in a manner that balances 1) the desire to retain an existing RBAC configuration with 2) the need to assign users with similar behavior into common roles.

From a high-level, our evolution strategy consists of mainly two phases as shown in Figure 4.1. In the first phase, we mine a set of candidate roles, which are selected to optimize an objective function that balances distance from the original roles with behaviorial similarity in the form of permission invocation in access logs. In the second phase, each user is assigned to roles according to a criterion that mitigates redundancy in the access control model. There are several primary contributions of this work, including:

- **A new objective function for the role mining problem.** We devise an objective that balances the administrator's belief with the evidence in existing access logs. The function is parameterized, such that a user can bias the resulting RBAC configurations toward belief or evidence as deemed desirable.

- **A hybrid role engineering algorithm.** We propose a new role engineering algorithm that builds on a subset enumeration technique employed in previous role engineering strategies. Our algorithm evolves existing RBAC configurations into new configurations which are are more effective at addressing administrators' beliefs and

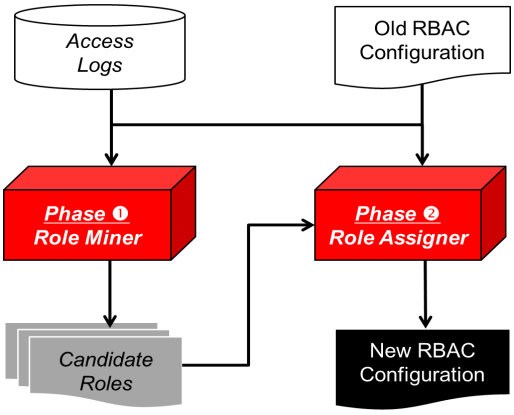


Figure 4.1: An architectural overview of our algorithm

permission utilization goals than current role engineering strategies.

- **A multi-objective empirical evaluation.** To evaluate the resulting RBAC configurations, we compare our algorithm with state-of-the-art role mining techniques using a real dataset derived from a large electronic medical record system, as well as a controlled synthetic dataset. The results show that our role evolution algorithm can produce a range of RBAC configurations in comparison to previous methods. Moreover, we show the resulting configurations follow the expected bias of the algorithm and indicate patterns exist in the real dataset.

4.2 Preliminaries

In this section, we review several topics that inform the development of our role revision method. This section begins with a formalization of a generalized version of the role mining problem. Next, we provide a description of access logs as they are utilized in our method. Then, we introduce a formalization of the objective function invoked in our variation of the role mining problem. Finally, this section concludes with a description of one-class support vector machines, an effective outlier detection algorithm, which we employ as a measure of the quality of RBAC configurations.

4.2.1 Generalized Role Mining Problem

We begin with a generalized perspective of the role mining problem, which will be refined to model the problems studied in this work.

Definition 1 (*Generalized Role Mining Problem*) *Let $t = \langle U, P, UPA \rangle$ denote an access control configuration, where $U = \{u_1, u_2, \ldots, u_m\}$ is a set of users, $P = \{p_1, p_2, \ldots, p_n\}$ is a set of permissions, and UPA is an $m \times n$ Boolean matrix indicating the mapping between U and P.*

The goal of the Generalized Role Mining Problem *is to find an RBAC configuration $c = \langle U, P, R, URA, RPA \rangle$[1], subject to $UPA = URA \otimes RPA$, such that an objective function $f()$ is optimized. In the configuration, $R = \{r_1, r_2, \ldots, r_k\}$ is a set of roles, URA is an $m \times k$ Boolean matrix indicating the mapping between U and R, and RPA is a $k \times n$ Boolean matrix indicating the mapping between R and P.*[2]

The matrices in Figures 4.2(b) and (c) depict an example of an RBAC configuration. It can be seen there are six users, seven permissions, and two roles.

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
u_1	116	485	151	402	249		
u_2	181	797	21	58	33		
u_3	199	819	77	196	91		
u_4			29	81	44	402	174
u_5			108	278	161	530	215
u_6			25	62	31	334	118

(a) *UPIM*

	r_1	r_2
u_1	1	0
u_2	1	0
u_3	1	0
u_4	0	1
u_5	0	1
u_6	0	1

(b) *URA*

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
r_1	1	1	1	1	1	0	0
r_2	0	0	1	1	1	1	1

(c) *RPA*

Figure 4.2: An example of a user-permission invocation matrix (*UPIM*) and an RBAC configuration (*URA* and *RPA*).

Given a role r_l, we can readily extract the corresponding users and associated permissions. We use $\mathbb{P}_l^{(c)} = \{p_x \mid RPA_{lx} = 1\}$ to denote the set of permissions assigned to r_l, and

[1] We only consider the $RBAC_0$ (i.e., we do not consider role hierarchies or constraints).

[2] $x = a \otimes b$ denotes the Boolean matrix product, in which an element is defined as $x_{ij} = \vee_k (a_{ik} \wedge b_{kj})$.

$\mathbb{U}_l^{(c)} = \{u_y \mid URA_{yl} = 1\}$ to denote the set of users under r_l in the RBAC configuration c. When appropriate, we adopt the standard convention of representing a role as its corresponding set of permissions. For example, $\gamma = \{p_1, p_2\}$ represents a role possessing two permissions. In general, all users whose permission set is the superset of γ automatically obtain this role. There are, however, exceptions to this role that will be introduced in Section 4.3.

Various objective functions have been proposed for the role mining problem. Certain functions are based on the size of R [33], while others use variations of structural complexity [35]. With regard to the latter, objective functions have been based on the size of R and the total number of elements in *URA* and/or *RPA*. We define the objective function from the perspective of i) user behavior similarity and ii) distance to the initial RBAC configuration.

4.2.2 Access Log

In this work, an access log is represented as an $m \times n$ user-permission invocation matrix *UPIM*. We use ω_{ij} to denote the number of times user u_i invoked permission p_j. Figure 4.2(a) depicts the *UPIM* that corresponds to the RBAC configuration in Figures 4.2(b) and (c). To mitigate bias which may occur from working with the raw frequency counts, we preprocess *UPIM* through a row-wise normalization (i.e., all numbers are divided by their rowsum) to represent *UPIM* as a set of user-specific probability distributions.

To measure the homogeneity of a role, we need to extract the corresponding access records from *UPIM*. This is accomplished through the application of a projection matrix.

Definition 2 (*Projection Matrix*) *Given an RBAC configuration c and user-permission invocation matrix UPIM, the* projection matrix M_{r_l} *for role* r_l *is an* $p \times q$ *matrix, where* $p = |\mathbb{U}_l^{(c)}|$ *and* $q = |\mathbb{P}_l^{(c)}|$. *Each row (column) of* M_{r_l} *represents a user (permission) associated with* r_l. *Let* β_{ij} *be defined as an element of* M_{r_l} *as follows. If the* i^{th} *user in* $\mathbb{U}_l^{(c)}$ *is* u_f *in U, and the* j^{th} *permission in* $\mathbb{P}_l^{(c)}$ *is* p_g *in P, then* $\beta_{ij} = \omega_{fg}$.

4.2.3 Objective Function

To balance existing beliefs in roles with actual user behavior, we propose a new objective function for the role mining problem, which is based on two goals. The first goal is to enable each role to possess high homogeneity in the rate at which permissions are accessed. The second goal is to ensure the new and pre-existing RBAC are "near" one another. We use functions $h()$ and $j()$ to measure the first and second goal, respectively, and define the objective function as:

$$f(c_{new}) = \alpha \cdot h(c_{new}) + (1-\alpha) \cdot j(c_{old}, c_{new}) \tag{4.1}$$

where c_{new} is the RBAC configuration proposed by a role mining algorithm, c_{old} is the existing RBAC configuration, and α is a real value between 0 and 1 to bias the system from $h()$ to $j()$. The following subsections provide details regarding how the functions $h()$ and $j()$ are computed.

4.2.3.1 RBAC homogeneity

In this section, we formally introduce the notion of homogeneity, which will be applied to characterize the similarity of the users in a role.

Definition 3 (*Homogeneity*) *Given an RBAC configuration $c = \langle U, P, R, URA, RPA \rangle$ and a user-permission matrix UPIM, the* role homogeneity *of r_l is:*

$$\mathsf{ho}(r_l) = m^{-1} \sum_{i=1}^{m} (1 - \mathsf{cosine}(\boldsymbol{x}_i, \boldsymbol{c}_l)), \tag{4.2}$$

where m is the number of row vectors in M_{r_l}, $\boldsymbol{x}_i$ is the i^{th} row vector of M_{r_l}, $\boldsymbol{c}_l$ is the mean vector of all row vectors in M_{r_l}, and $\mathsf{cosine}(\boldsymbol{a}, \boldsymbol{b})$ is the cosine similarity $\frac{\boldsymbol{a} \bullet \boldsymbol{b}}{|\boldsymbol{a}||\boldsymbol{b}|}$.

The RBAC homogeneity *of c is then defined as:*

$$h(c) = |R|^{-1} \sum_{r_l \in R} \mathsf{ho}(r_l) \tag{4.3}$$

Role and RBAC homogeneity (Equations 4.2 and 4.3) have a natural geometric interpretation. Consider, if a role consists of a set of highly similar users, then the vectors representing the behaviors of these users will form a relatively compact cluster in $\mathbb{R}^k$ (where k is the dimensionality of the vectors) and the degree of the angle between each vector and the mean of the cluster, measured by $1 - \mathsf{cosine}(\mathbf{x}_i, \mathbf{c}_l)$, will tend to be small. Conversely, if the users in a role exhibit highly diverse behavior, then the cluster will tend to have a long diameter and the degree of the angle will be large.

4.2.3.2 Distance Between RBAC Configurations

In order to measure how far a new RBAC configuration has migrated from the initial configuration, we introduce a set-based similarity measure. First, we define the distance between two roles.

Definition 4 (*Role Distance*) *Let γ_i and δ_j be roles in RBAC configurations c_1 and c_2, respectively. The* role distance *between the roles is defined as:*

$$\mathsf{jac}(\gamma_i, \delta_j) = 1 - \frac{|(\mathbb{P}_i^{(c_1)} \times \mathbb{U}_i^{(c_1)}) \cap (\mathbb{P}_j^{(c_2)} \times \mathbb{U}_j^{(c_2)})|}{|(\mathbb{P}_i^{(c_1)} \times \mathbb{U}_i^{(c_1)}) \cup (\mathbb{P}_j^{(c_2)} \times \mathbb{U}_j^{(c_2)})|} \tag{4.4}$$

where $A \times B$ is the Cartesian product of sets A and B.

In our setting, a role corresponds to the Cartesian product of its associated set of permissions and set of users. This enables the comparison of two roles to be performed in the joint space of permissions and users. Thus, our definition corresponds to the Jaccard distance, a widely used measure for the comparison of two sets [46].

We leverage the distance between roles to define the distance between a role and a role set.

Definition 5 (*Role Set Distance*) *The* role set distance *from role γ to role set R is the minimum distance to any role in the set:*

$$\mathsf{minjac}(\gamma, R) = \min_{\delta \in R} \mathsf{jac}(\gamma, \delta) \tag{4.5}$$

Finally, we can define the distance from one RBAC configuration to another.

Definition 6 (*RBAC Distance*) *Let c_i and c_j be RBAC configurations. The* RBAC distance *from c_i to c_j is:*

$$j(c_i, c_j) = |R_i|^{-1} \sum_{\gamma \in R_i} \mathsf{minjac}(\gamma, R_j) \tag{4.6}$$

where R_i and R_j are the role sets of c_i and c_j, respectively.

4.2.3.3 Quality of a Role

We further use the metrics above to define a heuristic function that computes a score for a role γ. This function, which we call the *role score rs*, is defined as:

$$\mathsf{rs}(\gamma) = \alpha \cdot \mathsf{ho}(\gamma) + (1 - \alpha) \cdot \mathsf{minjac}(\gamma, R), \tag{4.7}$$

where α is as defined in Equation 4.1 and R is the role set of c_{old} in Equation 4.1. This function will be leveraged to guide our role evolution algorithm (described in Section 4.3).

4.2.4 One-Class SVM

To evaluate the homogeneity of the resulting roles, we employ an outlier detection algorithm. The selection of this strategy is based on the hypothesis that the more homogeneous a role is, the smaller the number of outlying users it will contain. We use a one-class support vector machine (SVM) [47] to detect outlying users for each role. SVMs have been

reported as comparable, and often superior, to other anomaly detection methods in various settings [48], including intrusion detection [49].

One-class SVMs can be applied to learn a region that contains only the training set, which is expected to be typical data for a class. Any data point in a test set that falls out of the region will be predicted as an anomaly. Theoretically, the goal of SVM in this scenario is to find a hyperplane $\mathbf{w} \in F$ that separates the training set from the origin with the maximum margin. This can be formalized as an optimization problem as follows:

$$\begin{aligned} &\min_{\mathbf{w}\in F, \xi\in\mathbb{R}^l, \rho\in\mathbb{R}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{v\cdot l}\sum_i \xi_i - \rho \\ &subject\ to\ (\mathbf{w}\cdot\Phi(\mathbf{x}_i)) \geq \rho - \xi_i, \xi_i \geq 0 \end{aligned}, \tag{4.8}$$

where ξ_i are non-zero slack variables to be penalized in the objective function. When values for $\mathbf{w}$ and ρ can be found which solve the optimization function, the majority of the training set satisfies $\mathsf{sgn}(\mathbf{w}\cdot\Phi(\mathbf{x}_i)) \geq \rho$, while the regularization term $\|\mathbf{w}\|$ remains small. The parameter v determines the tradeoff between these two goals. With $\mathbf{w}$ and ρ, we have a decision function $f(\mathbf{x}) = \mathsf{sgn}(\mathbf{w}\cdot\Phi(\mathbf{x}) - \rho)$ to determine if an new instance $\mathbf{x}$ is anomalous.

In this work, we specifically use one-class SVMs with an RBF kernel, as defined in Equation 4.9.

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathsf{exp}(-g\cdot\|\mathbf{x}_i - \mathbf{x}_j\|^2) \tag{4.9}$$

The parameters g in Equation 4.9 and v in Equation 4.8 are key factors that influence the performance of one-class SVMs. We utilize a grid search technique to find values for g and v that enable a robust SVM [50].

For evaluation, for each role r_l, we split the row vectors of a projection matrix into a training set and a test set. We perform gird search on the training set to obtain g and v, which are then applied to train and test a one-class SVM. The proportion of outlying users identified by the one-class SVM is applied to measure the homogeneity of the role.

4.3 Role Evolution by Permission Utilization

This section begins by formally defining the *Role Evolution By Permission Utilization* (REPU) problem.

Definition 7 (*REPU Problem*) *Given an existing RBAC configuration* $c = \langle U, P, R, URA, RPA \rangle$*, a user-permission assignment UPA (*$UPA = URA \otimes RPA$*) and a user-permission invocation matrix UPIM,* REPU *is to find a new RBAC configuration* $c^* = \langle U, P, R^*, URA^*, RPA^* \rangle$ *subject to* $UPA = URA^* \otimes RPA^*$*, such that the objective function* $f(c^*) = \alpha \cdot h(c^*) + (1-\alpha) \cdot j(c^*, c)$ *is minimized.*

The role mining problem has been shown to be NP-complete [33]. The REPU problem is a variation on role mining and can be reduced to this problem, making it NP-complete as well. Thus, we propose a heuristic-based search strategy based on a two-phase process as defined below.

4.3.1 Algorithm Description

To address the REPU problem, we designed the *Data-Driven Role Evolution* (DDRE) algorithm. Here we provide a walkthrough of the process and refer the reader to Algorithm 1 for specific details. The two phases of the algorithm are: i) candidate role generation and ii) role assignment.

4.3.1.1 Candidate Role Generation

The first phase begins with a set of *unit roles UR* such that there is one permission per role and no roles have the same permission (i.e., a one-to-one mapping of permissions and unit roles). Next, *UR* is copied to a candidate role set *CR*. The algorithm then iterates until a termination condition is satisfied. Each iteration begins by instantiating a set of new roles into an empty pool *DP*, which is based on a pairwise union for all roles in *CR*. For example, the union of $\{p_x\}$ and $\{p_y\}$ yields $\{p_x, p_y\}$.

The roles in *DP* are sorted by their quality scores (defined in Section 4.2.3.3), such that *DP* serves as a priority queue, where the best role is at the top. The algorithm then proceeds through the queue, by moving a role from the top of *DP* to *CR* and flipping the 1's in the user-permissions assignment UPA_{temp} that are covered by the role to 0's. This process continues until every element in UPA_{temp} is set to 0.

4.3.1.2 Role Assignment

Once *CR* is stable or the maximum number of iterations is reached, the algorithm enters the second phase of role assignment, the details of which are in Algorithm 3. The goal of this phase is to ensure that each user is assigned to non-redundant roles. By default, any user whose permission set is a superset of one role will automatically obtain this role. This would result in redundancy and an increasing unnecessary complexity in the system. For example, consider a set of users assigned to roles $\gamma = \{p_1, p_2, p_3\}$ and $\mu = \{p_1, p_2\}$. The latter role μ is redundant because the affiliated users can accomplish their task using only the γ role. Thus μ could be removed for the sake of succinctness.

The problem of winnowing the system down to a minimal set of roles for each user is similar to the set cover problem: given a user who possesses a set of permissions $PMS_i = \{p_{i_1}, p_{i_2}, \ldots, p_{i_k}\}$ and a set of roles $ROLES_i = \{\gamma_1, \gamma_2, \ldots, \gamma_l\}$ whose elements are all subsets of PMS_i, identify the smallest number of roles in $ROLES_i$ whose union equals PMS_i. It has already been shown that this problem is NP-complete [51]. Given the complexity of the problem, we adopt an approximation algorithm [52] to resolve the problem, as shown in Algorithm 3. At each iteration, we select the role in $ROLES_i$ with the largest number of permissions in common with PMS_i. The role is added to R_i and the permissions which were in common with PMS_i are removed from further consideration. This procedure repeats until no elements exist in PMS_i.

Finally, roles in R_i are assigned to user i. After assigning roles for each user, we obtain a final role set R^* and a user-role assignment URA^*. Specifically, we generate a role-

Algorithm 1 Data-Driven Role Evolution

Input: $c = \langle U, P, R, URA, RPA \rangle$, $UPIM$, α, $maxTimes$
Output: $c^* = \langle U, P, R^*, URA^*, RPA^* \rangle$

$t \leftarrow 0, DP \leftarrow \varnothing, CR \leftarrow \varnothing, CR_{old} \leftarrow \varnothing, UR \leftarrow \varnothing, UPA = URA \otimes RPA, UPA_{temp} = UPA$
for each $p_i \in P$ **do**
  $UR \leftarrow UR \cup \{\{p_i\}\}$
end for
$CR \leftarrow UR$
while $|CR_{old} - CR| > 0$ && $t++ \leq maxTimes$ **do**
  $DP \leftarrow \varnothing$
  for each $\mu_i \in CR$ **do**
    for each $\mu_j \in CR$ **do**
      $DP \leftarrow DP \cup \{\mu_i \cup \mu_j\}$
    end for
  end for
  $CR_{old} \leftarrow CR, CR \leftarrow \varnothing$
  Sort(DP, $UPIM$, c, α){Sort roles in DP according to their quality score. See Algorithm 2 for details.}
  for each $\gamma_i \in DP$ **do**
    if every element in UPM_{temp} is 0 **then**
      break
    end if
    if γ_i cannot cover any 1's in UPA_{temp} **then**
      continue
    end if
    $CR \leftarrow CR \cup \gamma_i$
    change all 1's in UPA_{temp} covered by γ_i to 0's
  end for
end while
$\{R^*, URA^*\} = RoleAssignment(U, P, UPA, CR)$
Initialize a $p \times n$ Boolean matrix RPA^* with all elements equal to zero, where $p = |R^*|$ and $n = |P|$.
for each $\mu_i' \in R^*$ **do**
  for each $p_j \in P$ **do**
    if $p_j \in \mu_i'$ **then**
      $RPA^*_{ij} = 1$
    end if
  end for
end for
return $c^* = \langle U, P, R^*, URA^*, RPA^* \rangle$

Algorithm 2 Sort()

Input: DP, $UPIM$, α, $c = \langle U, P, R, URA, RPA \rangle$
Initialize an array of real value, $score[]$, which has the same size as DP
for each $\gamma_i \in DP$ **do**
$score[i] = rs(\gamma_i)$
end for
Sort DP in ascending order according to $score[]$
return

Algorithm 3 RoleAssignment()

Input: U, P, UPA, CR
Output: URA, R
$R \leftarrow \varnothing$, $m = |U|$
for each $u_i \in U$ **do**
$PMS_i \leftarrow \{p_j | \forall p_j \in P, UPA_{ij} = 1\}, ROLES_i \leftarrow \varnothing$
while $PMS_i \neq \varnothing$ **do**
Select role μ_k from CR, such that $\mu_k \subseteq PMS_i$ and $|PMS_i \cap \mu_k|$ is maximized.
$PMS_i \leftarrow PMS_i - \mu_k, ROLES_i \leftarrow ROLES_i \cup \{\mu_k\}$
end while
$R \leftarrow R \cup ROLES_i$
end for
Re-index the roles in R using integers 1 to $h = |R|$, such that $R = \{\mu'_1, \mu'_2, \ldots, \mu'_h\}$
Construct $m \times h$ Boolean matrix URA, such that if $\mu'_j \in ROLES_i$, $URA_{ij} = 1$, otherwise $URA_{ij} = 0$
return R, URA

permission assignment RPA^* from the permission sets of the roles. Thus, a new RBAC configuration $c^* = \langle U, P, R^*, URA^*, RPA^* \rangle$ is returned.

4.3.2 An Example

In this section, we use the RBAC configuration and $UPIM$ in Figure 4.2 with $\alpha = 1$ to illustrate how the DDRE algorithm works in detail.[3]

UPA and RPA indicate there are two roles and six users. The roles are represented by permission sets $\{p_1, p_2, p_3, p_4, p_5\}$ and $\{p_3, p_4, p_5, p_6, p_7\}$. For this example, we create a set of ideal roles as the optimal solution, from which we design a series of generative

[3] $\alpha = 1$ implies the algorithm is completely biased to generate a set of roles with high homogeneity in user behavior (i.e., it ignores the structure of the original roles).

models to construct *UPIM*. This set contains three roles, which correspond to $\{p_1, p_2\}$, $\{p_3, p_4, p_5\}$ and $\{p_6, p_7\}$. The generative model for each of the roles follows a fixed distribution, which for this example is set to $\{0.2, 0.8\}$, $\{0.2, 0.5, 0.3\}$, and $\{0.7, 0.3\}$, respectively. This means, for instance, that for an arbitrary user u_k associated with the first role, $UPIM_{k1}$:$UPIM_{k2}$ is 1:4.

First, the algorithm initializes the system with a set of unit-roles: $\{\{p_1\}, \{p_2\}, \{p_3\}, \{p_4\}, \{p_5\}, \{p_6\}, \{p_7\}\}$. Next, the algorithm performs a pairwise combination of the unit-roles to derive a pool *DP* of the form $\{\{p_1, p_2\}, \{p_1, p_3\}, \ldots, \{p_6, p_7\}\}$. From this pool, four roles, $\{p_1, p_2\}$, $\{p_3, p_4\}$, $\{p_6, p_7\}$, and $\{p_4, p_5\}$, are selected for the next round of pairwise combination because they comprise the top four positions of the pool and are able to recover the *UPA*. When this set of roles is combined, it updates the pool to become $\{\{p_1, p_2\}, \{p_3, p_4\}, \{p_4, p_5\}, \{p_6, p_7\}, \{p_1, p_2, p_3, p_4\}, \{p_1, p_2, p_4, p_5\}, \{p_3, p_4, p_5\}, \{p_4, p_5, p_6, p_7\}\}$.

At this point, we select another four roles, $\{p_1, p_2\}$, $\{p_3, p_4\}$, $\{p_6, p_7\}$ and $\{p_3, p_4, p_5\}$, from the pool because they comprise the top four positions in the pool and are able to recover the *UPA*. Again, these roles are combined to update the pool to become $\{\{p_1, p_2\}, \{p_3, p_4\}, \{p_3, p_4, p_5\}, \{p_6, p_7\}, \{p_1, p_2, p_3, p_4\}, \{p_1, p_2, p_3, p_4, p_5\}, \{p_3, p_4, p_6, p_7\}, \{p_3, p_4, p_5, p_6, p_7\}\}$. At this point, roles in the top four positions of the pool, $\{p_1, p_2\}$, $\{p_3, p_4\}$, $\{p_6, p_7\}$ and $\{p_3, p_4, p_5\}$, are selected to constitute the candidate role set. Since the candidate role set is the same as the previous round, this phase of the DDRE algorithm terminates and returns this candidate role set.

Next, the roles $\{p_3, p_4\}$ are are redundant in the presence of $\{p_3, p_4, p_5\}$, so they are discarded in the second phase.

Finally, the remaining three roles $\{p_1, p_2\}$, $\{p_3, p_4, p_5\}$, and $\{p_6, p_7\}$ constitute the role set in RBAC configuration as a solution, which are the same as the three ideal roles alluded to earlier.

4.4 Experiment

We investigated the performance of the DDRE algorithm on both synthetic and real world datasets. In the process, we varied α to characterize how the resulting RBAC configuration changes. In addition, we compared DDRE with several related role mining algorithms, including the minimal perturbation role mining algorithm [38] and role mining with latent Dirichlet allocation (LDA) [39], which has been introduced in Chapter 2.

4.4.1 Description of Datasets

4.4.1.1 Electronic Medical Record Roles & Access Logs

Although the EMR described in Chapter 3 is not based on RBAC, a reason is an option selected when a chart is accessed by the user during a patient's hospitalization and the options available are tied to the job title of the user. As a result, we believe it is reasonable to utilize the reasons as privileges and job titles as roles in the system. Table 4.1 shows how we acquire an RBAC configuration $c = \langle U,P,R,URA,RPA \rangle$ and a user-permission invocation matrix *UPIM* from the access log.

4.4.1.2 Synthetic Roles & Access Logs

To allow for replication of our study and comparison to the EMR dataset, we created a synthetic dataset which consists of an RBAC configuration $c' = \langle U', P', R', URA', RPA' \rangle$ and a corresponding *UPIM*. As in the example in Section 4.3.2, there are several ideal roles, each of which has a corresponding probability distribution over its affiliated permissions.

To enable a clean analysis, there is no overlap in the permission sets of these roles. We merge the permission sets of several ideal roles to realize an *actual* role in the RBAC system. For each user under one actual role, we utilize the ideal roles hiding in the actual role to generate its corresponding vector in *UPIM*, where the numbers corresponding to one ideal role need to follow the probability distribution of this ideal role. For instance, we can

Table 4.1: A summary of how the RBAC configuration and *UPIM* are derived from the EMR access logs.

Feature	Derivation Process
U	*The set of users in the access logs.*
R	*The set of job titles in the access logs.*
P	*The union of reason sets available to each job title in R.*
URA	$\|U\| \times \|R\|$ *Boolean matrix. If the* i^{th} *user and* j^{th} *job title (role) co-occur in one entry of the access log,* $URA_{ij} = 1$*; otherwise* $URA_{ij} = 0$.
RPA	$\|R\| \times \|P\|$ *Boolean matrix. If the* j^{th} *reason (permission) belongs to the reason set available to* i^{th} *job title (role),* $RPA_{ij} = 1$*; otherwise* $RPA_{ij} = 0$.
$UPIM$	$\|U\| \times \|P\|$ *real value matrix. If the* i^{th} *user and* j^{th} *reason (permission) co-occur in the same entry of the access log* t *times, then* $UPIM_{ij} = t$.

merge two ideal roles $\{p_1, p_2, p_3\}$ and $\{p_4, p_5, p_6\}$ whose distributions are $\{0.2, 0.3, 0.5\}$ and $\{0.1, 0.7, 0.2\}$, respectively, to create an actual role $\{p_1, p_2, p_3, p_4, p_5, p_6\}$. The rates of permissions invoked by each user u_i assigned to this role need to be consistent with the distributions of both ideal roles, which means $UPIM_{i1} : UPIM_{i2}$:$UPIM_{i3}$ = 2:3:5 and $UPIM_{i4} : UPIM_{i5}$:$UPIM_{i6}$ = 1:7:2. The $UPIM$ matrix is constructed by performing this procedure for each user. A more detailed example is reported in the Appendix.

For this study, we created 10 ideal roles, and use 10 actual roles, which are derived by merging different sets of the ideal roles as R'. We synthesize 20 users per role (i.e., 200 users in total) as U'. For each actual role, the ideal roles used for merging are randomly selected from the 10 ideal roles. Since the actual roles are represented by permission sets, RPA' is derived accordingly. In addition, we derive P' by uniting the permission sets of all actual roles. Thus, a synthetic RBAC c' is successfully constructed.

4.4.2 Evaluation Measures

We use two measures to assess the quality of the resulting RBAC system.

RBAC Evolution Distance: This measure characterizes the distance between the old and new RBAC configurations. It directly corresponds to Equation 4.6.

Outlier Rate: This measure characterizes the homogeneity of users' behavior in the resulting roles. For this measure, we use the rate at which users are predicted to be outliers in the system. The outlier rate is computed as follows. For each role r_l, we perform outlier detection on the corresponding projection matrix M_{r_l} using one-class SVM.[4] To do so, the row vectors in M_{r_l} are split into three equally-sized partitions $\{part_1, part_2, part_3\}$. We pick one partition as the test set, and the remaining two partitions as training and validation sets for a one-class SVM. After we obtain a one-class SVM model, we perform the outlier detection on the test set. All vectors classified as negatives are designated as outliers. This process is performed in three-fold cross-validation (i.e., three times with a different test set and training set), so that each row vector in M_{r_l} is evaluated. The outlier rate of a role r_l is computed as:

$$\mathsf{or}_l = \frac{\sum_{i=1}^{3}(\#\ of\ outliers\ in\ part_i)}{\#\ of\ row\ vectors\ in\ M_{r_l}} \tag{4.10}$$

Finally, the outliers from each role are consolidated to calculate the outlier rate for the entire RBAC system:

$$\mathsf{oor} = \frac{\sum_i or_i \cdot n_i}{\sum_i n_i} \tag{4.11}$$

where n_i is the number of users who are members of role r_i.

Detecting the outlier rate is a more intuitive and straightforward way to measure the homogeneity of the entire RBAC system[5] because the two concepts are strongly related.

[4] All SVM calculations were performed in *libsvm* [53].

[5] The RBAC homogeneity in definition 7 could be replaced with the outlier rate. However, RBAC homogeneity has significantly lower time complexity ($O(mn)$, where m is the number of roles which ever exist in DP of algorithm 1, n is the average number of users contained by each role) and can be computed in a feasible amount of time on a commodity server. By contrast, the outlier rate requires computation on the order of ($O(mn^2)$).

Figure 4.3 shows the relationship for both the EMR and synthetic datasets, where each point is derived from the RBAC configuration from the DDRE algorithm over a range of α values. The correlation coefficient (r^2) for a linear regression was found to be 0.912 and 0.837 for the EMR and synthetic datasets, respectively. Thus, we conclude that the outlier rate is positively correlated with RBAC homogeneity and use it to measure the homogeneity of the system in the following evaluation.

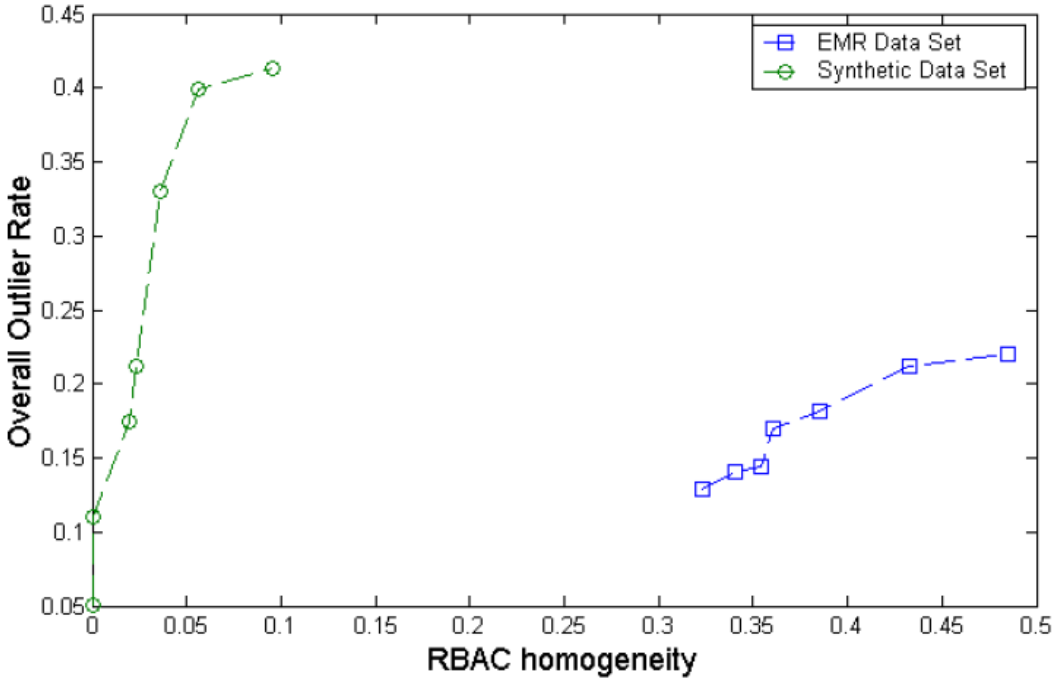


Figure 4.3: Relationship between RBAC homogeneity and outlier rate.

4.4.3 Results

4.4.3.1 Assessing the Tradeoff

All of the following experiments were run an Intel Core i5 2.40GHz CPU with 4G memory and a Windows XP operating system. We ran the DDRE algorithm with a range of α values to assess its efficiency. Table 4.2 shows the time consumed by the algorithm on the EMR dataset. The longest time was 21.7 minutes, which shows the algorithm can

terminate in a practical amount of time. Moreover, the runtime is directly correlated with α.

Next, we investigated how the RBAC configuration yielded by DDRE changes with α. Figure 4.4 summarizes this result, where the number near each point in the curves of DDRE corresponds to the value of α used to generate the corresponding RBAC configuration. In this figure, it can be seen that when α biases the system towards behavior, the overall outlier rate is low, whereas the distance between the old RBAC configuration and the resulting RBAC configuration is large. On the contrary, when α is biased towards the distance to old RBAC configuration, the overall outlier rate is high, while the distance between the two RBAC configurations is significantly smaller. In particular, we find that the outlier rate corresponding to $\alpha = 1$ is 41.1% and 87.7% lower than that corresponding to $\alpha = 0$ on the EMR and synthetic datasets, respectively. The Jaccard distance between the two RBAC configurations when $\alpha = 0$ is 90.9% and 100% lower than that corresponding to $\alpha = 1$ on the EMR and synthetic datasets, respectively. This observation indicates that we can obtain an almost identical RBAC configuration to the initial one when $\alpha = 0$. These results suggest that the DDRE algorithm is effective.

Table 4.2: Runtime of the DDRE algorithm.

α	1	0.97	0.93	0.9	0.8	0.7	0
Runtime(min)	21.7	15.5	12.7	12.5	10.4	7.3	6.4

We also note that the EMR dataset yields a much smaller range of outlier rates and Jaccard distances than the synthetic dataset. We hypothesized that this is because each actual role in the synthetic dataset is composed of more ideal roles than the actual roles in the EMR dataset. For instance, imagine there is an actual role composed of m ideal roles. When we compute the distance from one ideal role to the actual role, a larger m means the ideal role has permission set with a smaller size. This will result in a smaller numerator in Equation 4.4 and, thus, will yield a larger value for $j()$. Moreover, the ideal role exhibits a strong pattern as a single role, but the more ideal roles that aggregate into an actual role, the

faster their patterns are diluted. This leads to a significant increase in outlier detection. By studying the original roles (actual roles) and the resulting roles (ideal roles) yielded by the DDRE algorithm with $\alpha = 1$, we find that each original role in the EMR dataset possesses 1.5 roles on average in the corresponding resulting role set. By contrast, each original role in the synthetic dataset possesses 5.4 roles on average in the corresponding resulting role set. We believe this finding validates our hypothesis.

Figure 4.4 also depicts the results of the minimal perturbation role mining (RM-MP) and role mining with LDA (RM-LDA) algorithms.[6] The number near each point of the RM-MP curve corresponds to the value of *w* that controls the balance between the number of roles generated and the *Roles_Roles Distance*, a set-based distance between new role set and old role set (*D* in the objective function in [38]). It can be seen that the curves for RM-MP have the same tendency as that generated through DDRE. That is an intuitive and expected finding. Consider, when *w* is biased towards the number of generated roles, the algorithm will prefer the roles with larger sizes to those that are closer to original roles. This can lead to low homogeneity and large distances to the original roles. In addition, we notice that RM-MP yields curves that are close to those from DDRE, however, DDRE has a broader range of solutions, which can be seen by observed at the points when α approaches the boundary cases of 0 and 1. This indicates DDRE can yield better results when α be biased toward either sole objective. The result of RM-LDA on the EMR dataset shows it yields an overall outlier rate that is comparable to the results of DDRE when biased towards permission utilization, however, the RBAC it generated is significantly different than the original RBAC. The result of RM-LDA on the synthetic dataset is in the neighboring region of that of DDRE, but it is easy to find a solution from the curve of DDRE that has both a lower RBAC distance and a lower outlier rate than RM-LDA.

[6] As is stating in [39], the number of topics (roles) specified for the LDA is $\sqrt{|U|}$.

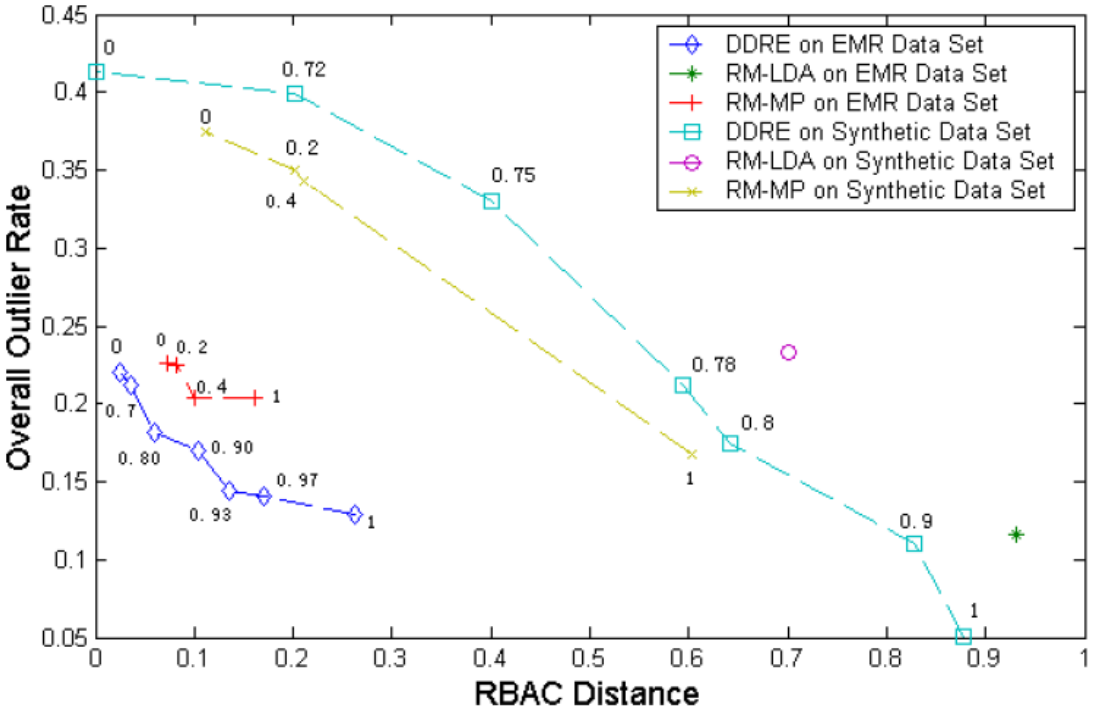


Figure 4.4: Summary of the tradeoff between the distance of old and new RBAC configurations (i.e., RBAC distance) and the rate of outlying behavior for the EMR and synthetic datasets.

4.4.3.2 Influence of SVM Training on Outliers

Next, we investigated how the results of one-class SVM are influenced with respect to the ν parameter.[7] This experiment is performed to determine if there exist patterns in the EMR dataset or if our results are based on random effects. As mentioned earlier, ν controls the tradeoff between the fraction of training instances falling into the learned region and the value of the regularization term. As ν increases, less instances in the training set will fall into the learned region. So, if the entire dataset follows a pattern, the test set will be distributed in approximately the same region as the training set even though ν is decreasing. Otherwise, due to the high diversity of the support vectors, the test set will likely be located in a different region.

To perform this portion of our analysis, we created two uncontrolled versions of *UPIM*

[7]The paratmeter g is determined by the grid search method and is not investigated.

Table 4.3: Role prediction accuracy as a function of v.

	v=0.16	v=0.21	v=0.26	v=0.31	v=0.36
EMR_{UN}	79.48%	75.48%	71.08%	66.25%	61.93%
EMR	82.48%	77.35%	75.10%	71.51%	67.02%
SYN_{UN}	78.10%	72.00%	66.59%	60.75%	54.93%
SYN	77.18%	73.82%	68.73%	64.18%	58.91%

for the two data sets used earlier by assigning a random value to $UPIM_{ij}$ that was originally $UPA_{ij} = 1$. The uncontrolled version of *UPIM* is used for simulating the access log without any pattern. We then employed one-class SVM to compute the accuracy (calculated by $1 - \mathsf{oor}$) for the RBAC configurations with the real and uncontrolled *UPIM* matrices for the EMR dataset (called EMR and EMR_{UN}) and synthetic dataset (called SYN and SYN_{UN}). It is expected that the accuracy on the uncontrolled dataset will decrease more quickly than the controlled dataset.

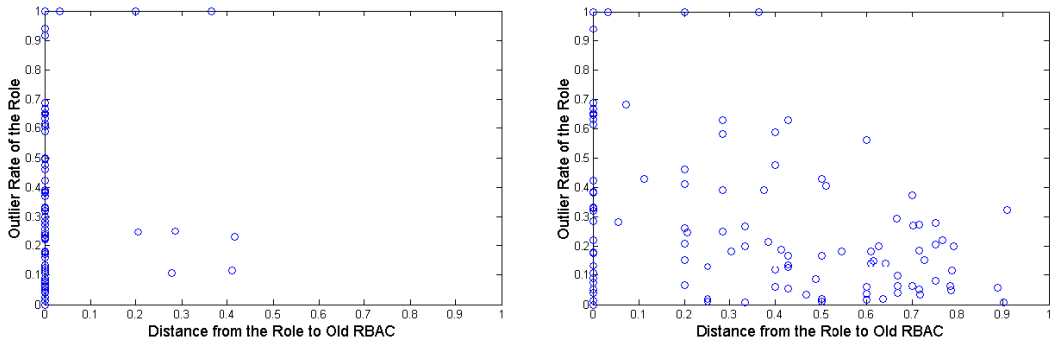
Table 4.3 shows the accuracy of one-class SVM with different v on the resulting RBAC configurations. Here it can be seen that the accuracy of SYN_{UN} decreases by 29.67%, while the accuracy on SYN decreases by 23.67%. By performing a proportion test, the latter accuracy decrease rate is slower than the former one with 90% confidence. This observation confirms our suspicion. We further note that the accuracy on EMR_{UN} decreases by 22.08%, while the accuracy on EMR decreases by 18.74%, and the difference between them is also proven statistically significant with 90% confidence by the proportion test, which suggests patterns exist in the real EMR dataset.

4.4.3.3 Statistics of Generated Roles

Finally, Figures 4.5 and 4.6 provide summary statistics of the roles generated when DDRE is applied to the EMR dataset. In Figure 4.5, each circle (x,y) represents one role γ. x is calculated by $\mathsf{minjac}(\gamma, R)$ (see Equation 4.5), where R is the role set of the original RBAC, while y is the outlier rate (see Equation 4.10) detected for this role. From Figure 4.5, it can be seen there is a major difference between the distributions of roles yielded by

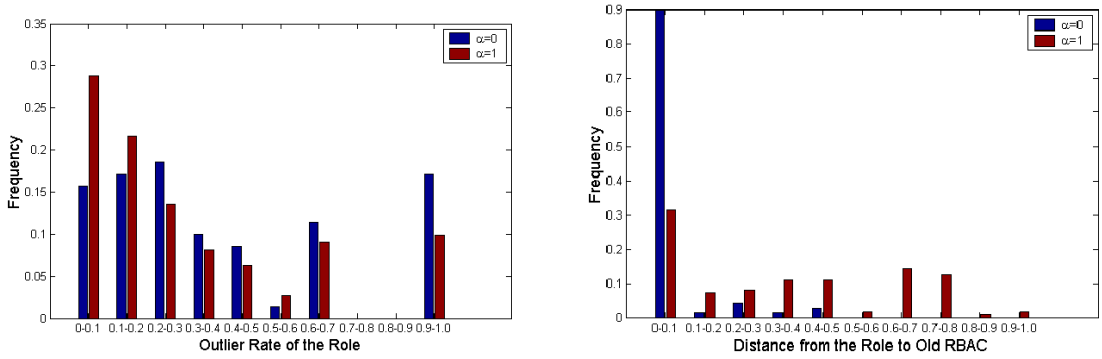
the algorithm with α set to 1 and 0.

Moreover, we show the marginal distributions of $\mathsf{minjac}(\gamma, R)$ and the outlier rate under different α in Figure 4.6. The histogram in Figure 4.6(a) demonstrates that the roles generated by $\alpha = 1$ tend to have less outlying users than the roles generated by $\alpha = 0$. By contrast, the histogram in Figure 4.6(b) demonstrates that the roles generated by $\alpha = 0$ tend to be closer to the role set in the original RBAC than the roles generated by $\alpha = 1$. These observations further validate the effectiveness of the DDRE algorithm.



(a) Plot of $\alpha = 0$ (b) Plot of $\alpha = 1$

Figure 4.5: Plots of roles denoted by corresponding distance to old RBAC and outlier rate



(a) Distribution of Outlier Rate (b) Distribution of RBAC Distance

Figure 4.6: Frequency distributions of (a) outlier and (b) distance rates under $\alpha = 0$ and $\alpha = 1$

Chapter 5

Role Prediction and Role Revision using EMR

This chapter introduces an approach, called *Role-Up*, to revise roles by abstracting existing roles along a pre-defined role tree. This approach differs from the DDRE algorithm in two aspects. First, it runs without the constraint that $UPA = URA \otimes RPA$. Second, we introduce the concept of "role prediction", which classifies users into roles and obtain accuracy, instead of "homogeneity" to measure the quality of roles. The purpose we state the differences is not to claim one approach is superior to the other, but that each one has different scope of application.

This chapter is structured as follows. Section 5.1 introduces the motivation. In the Section 5.2 we introduce the background, as well as the Role-Up algorithm designed for this study. We then report on an extensive experimental analysis of role prediction and the recommendations made by the algorithm in Section 5.3.

5.1 Introduction

There are two dominant strategies for limiting access to Electronic Medical Records (EMRs) within enterprises such as hospitals. One strategy is RBAC. This is commonly accomplished by looking at the job positions in the enterprise and the tasks the employees in these positions need to perform, then assigning privileges to positions, or variants of them, to enable the employees to accomplish their assigned tasks. A second strategy, which we group under the general heading of Experience Based Access Management (EBAM) [54], emphasizes accountability and the use of audit data to punish abuse (i.e., EBAM is a retrospective strategy). An often referenced strategy for EBAM is to manually review the audit logs of VIPs to determine when abuses transpire [55, 56, 57]. Another strategy, often

called break-the-glass security, discourages abuse by warning users that certain types of access are likely to be manually reviewed [58].

However, at the current point in time, RBAC and EBAM are used without much common foundation. Yet there are significant opportunities for synergy between the techniques. Consider, audit data may provide valuable information about roles, such as whether a new role would be beneficial or whether two existing roles should be merged. More appropriate role definitions, or roles that are context-specific, driven by auditing analytics, may be applied to restrict access so that fewer checks are required in the auditing process.

The aim of this chapter is to investigate a key step that could lead to such a synergy between RBAC and EBAM. We call the concept role prediction and it refers to the ability to use audit logs to predict whether a given user is associated with a given role. Role prediction can be a valuable tool for the role engineer, that is, the security administrator responsible for creating roles and managing assignments to them. For instance, a pair of roles that are often confused in the role prediction process might be good candidates for merging. Moreover, role prediction can provide insights into role hierarchies, such as indicating whether the right relationships have been allocated.

This chapter has three specific goals:

- **Hospital Role Classification** First, we aim to determine the extent to which expert-defined job titles in a large academic medical center help to distinguish between roles. To perform this part of the investigation, we train a machine learning-based classifier over the various features invoked by users acting in a role while accessing a patient record, and classify a test set of users. The accuracy acquired is used to measure the quality of the role specifications.

- **Intelligent Role Abstraction** Second, we hypothesize that certain abstractions of roles can permit more accurate differentiation of roles in the system. To answer this hypothesis we developed and applied role hierarchies to determine appropriate levels of role auditing. Moreover, we develop a heuristic-based algorithm, called Role-Up,

to execute a "rolling-up" procedure for the hierarchy.

- **Empirical Evaluation** Third, we apply our methods to three months of access logs from a large academic medical center, Northwestern Memorial Hospital. From these results we judge whether the role specification performs well and how role specification might be optimally informed.

Our findings suggest that RBAC for EMR systems can be effectively guided through information mined from audit logs. We demonstrate generalization of roles can improve the predictability of role behavior with minimal sacrifices to the specificity of the system.

5.2 Methods

5.2.1 Roles and Hierarchies

One of the specific aims of this chapter is to determine how generalizations of roles in the EMR system could permit more effective access control. However, at the time this study was conducted, there was no explicit relationship established between the user positions in the Cerner EMR. Thus, we collaborated with several clinicians at Northwestern to design a role generalization hierarchy. This hierarchy, a section of which is depicted in Figure 5.1, was designed as a tree data structure and consists of four levels: 1) Specific-Position, 2) General-Position, 3) Conceptual-Position, and 4) Employee. The lowest level in the hierarchy, termed Specific-Position, consist of the 140 user positions (i.e., job titles) defined for the current EMR system. The next level up, termed the General-Position level, was established by suppressing semantic qualifiers from the user positions. This level consists of 62 nodes in the hierarchy. The qualifiers that were removed represented certain administrative pay (or responsibility) grades or specializations of particular job titles. For instance, the job titles "Dietary 1" and "Dietary 2" were generalized to the common "Dietary".

The next level up is called the Conceptual-Position level, and was defined with the assistance of the clinicians. This level is composed of five roles defined to capture the

anticipated workflow of the healthcare domain. These roles are: i) Doctor: all users whose workflow is most consistent with that of a physician and includes entering orders/notes using the physician tools; ii) General Clinician: all non-physician clinical staff who do not have a restricted domain of work (e.g., nurses who rotate among various care areas); iii) Specific Clinician: all non-physician clinical staff who work in a specific clinical care domain (e.g., Oncology, Cardiology, and Gastroenterology). This group likely represents a more diverse set of users in comparison to the other roles at this level; iv) Billing: users who interact with charts from a billing specific perspective; and v) Admin: users who interact with charts from an administrative and not immediate clinical care perspective.

One of the key reasons why these roles deviate from the terms used in the lower levels is that the "User Positions" address concerns that are less characteristic of the user and instead reflect system design nuances at the time the user was enrolled. An example of this somewhat artifactual name distinction is the existence of positions reflecting whether or not a user had access to CPOE when the user was first enrolled. Now, all physician users have CPOE capability whether or not their user role at inception indicated this was available. Thus, it is anticipated that this higher level view should help mitigate outliers of particular users or job titles. And, at the same time, we believe this level should provide a structure for other healthcare organizations, and EMR systems, to adopt for similar role assignment endeavors.

Finally, and for the purposes of completeness, the highest level in the hierarchy corresponds to the root of the tree and consists of a single role, namely Employee or Affiliate.

5.2.2 A Formal Representation of the Users

Before delving into the details of the hierarchy-based role assignment process, we take a moment to formalize the EMR access log system and the resulting transformations. Let $U = \{u_1, ..., u_m\}$ be the set of EMR users and let $Role = \{role_1, ..., role_n\}$ be the set of roles. For reference, we use $|\cdot|$ to represent the number of elements in a set.

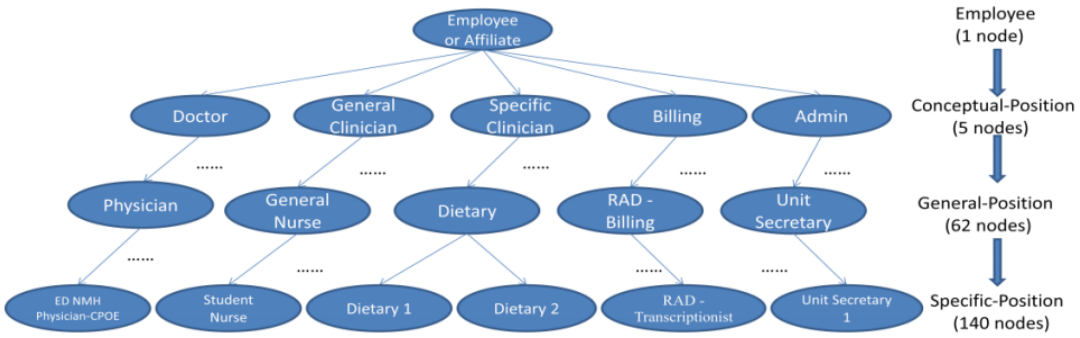


Figure 5.1: A selection of the role generalization hierarchy designed for this study

Given a database of EMR access transactions, we construct a vector space model for each user. Specifically, let $V = \{v_1, ..., v_m\}$ be a set of vectors, where v_i is the corresponding vector for u_i. Each vector is composed of three subvectors, r_i, s_i, and l_i, which represent the access features (i.e., reason, service, and location). Each of these subvector is defined over the domain of categorical values the feature to which it is associated. For instance, r_i contains a position for each of the 143 specific reasons that could have been selected by a user during a session with a patients record. For each reason, and for each user, we weight the j^{th} reason and refer to it r_{ij}.

For the purposes of this study, we represent r_{ij} using the term frequency inverse document frequency (TF-IDF) weighting model, which is widely used in text mining:

$$r_{ij} = TF_{ij} \cdot IDF_i = \frac{n_{ij}}{N_i} \cdot log\frac{|U|}{d_j} \tag{5.1}$$

where n_{ij} is the number of times r_i was invoked by u_i during their EMR sessions , N_i is the total number of accesses issued by u_i, and d_j is the number of users in the system who invoked reason r_j. We apply the TF-IDF schema based on the premise that the more times a user invokes a reason, the more likely the reason is indicative of the user (i.e., TF) and that the smaller the number of users that invoke a reason, the more closely related they are (i.e., IDF). We define s_i and l_i similarly.

5.2.3 A Machine Learning Approach to Role Prediction

The aforementioned vectors provide a summarized view of EMR users behavior in the healthcare system. We use the vectors as the basis of our role prediction procedure. Specifically, we train a Naive Bayes classifier [59] with roles as the class labels and the user vectors as input. The task of predication is to determine the class label (i.e., role) for a new user vector. For the Naive Bayes classifier, the new instance will be assigned the class label according to equation 5.2.

$$role_{MAP} = argmax_{role_j \in Role} P(role_j | r_i, s_i, l_i) \tag{5.2}$$

Using Bayes theorem and assuming conditional independence over the features, we can rewrite the expression as:

$$role_{MAP} = argmax_{role_j \in Role} P(role_j) \Pi_x P(r_{ix} | role_j) \Pi_y P(s_{iy} | role_j) \Pi_z P(l_{iz} | role_j) \tag{5.3}$$

In this work, however, the features are continuous variables, which makes it difficult to estimate $P(r_{ix}|role_j)$, $P(s_{iy}|role_j)$ and $P(l_{iz}|role_j)$ directly. As a result, we replace the conditional distribution function with the conditional probability density function, for which a Gauss distribution is used.

Hence, $P(role_j)$ is estimated as the proportion of users in $role_j$, while $P(r_{ix}|role_j)$ is estimated by the Gaussian density function [59]. For the latter, the parameters of μ and σ are estimated by calculating the mean and standard deviation of feature r_{ix} of the users in $role_j$, respectively. $P(s_{iy}|role_j)$ and $P(l_{iz}|role_j)$ are estimated similarly.

5.2.4 The Role-Up Algorithm

The primary goal of this work is to apply EBAM in the context of EMRs to discover, and assess, the appropriateness of usersroles. To achieve this goal, we developed an algorithm

called *Role-Up*. The algorithm is based on two foundational premises. First, the more roles in the system, the greater the ability to ultimately manage user groups and achieve a key security goal of separation of duty. Second, the more homogenous the user behavior is in a role, the easier it will be to monitor and audit users with respect to their actions. Pseudocode for Role-Up is provided in Algorithm 4.

Here, we provide a high-level walkthrough of the algorithm. First, in step 1, we extract the roles in the middle levels of the hierarchy. Next, in step 2, we employ the Naive Bayes classifier to predict roles in all of levels of the hierarchy. We use a leave-one-out cross-validation approach to evaluate the predictions. Specifically, role prediction is executed such that the classifier is trained with all, but one, user vectors. The remaining vector is then classified into a role. This procedure is repeated for each user until all users receive predictions. Then, to measure how well the roles are specified, we compute the accuracy of the system:

$$Accuracy = \frac{\#Correct\ Predictions}{\#Predictions} \tag{5.4}$$

In step 3, we initialize the set of roles to be returned to the administrator as null. In step 4, we calculate a score for each role at the General-Position and Conceptual-Position levels using the evaluation function of equation 5.5:

$$S = \alpha R + (1-\alpha)A \tag{5.5}$$

R is computed by $(|U| - N_{role})/|U|$, where N_{role} is the number of users covered by this role, and reflects the specificity after generalizing this role to its parent in the hierarchy. A is computed as $(Accuracy_{role} - \overline{Accuracy_{sub(role)}})$, where $\overline{Accuracy_{sub(role)}}$ is the average accuracy of all subroles of role at the Specific-Position level.

Then in Steps 5 and 6, we use a greedy procedure to roll-up the hierarchy. We iteratively select the role with the highest score and implement the corresponding generalization for all of its subroles. This procedure iterates until the highest score is greater than a certain

Algorithm 4 Pseudocode for the Role-Up Algorithm.

Input: *Vectors*: A set of EMR user access vectors, *Hierarchy*: an EMR user role hierarchy, α: a real-valued weighting parameter in the range (0, 1), τ: a threshold

Output: *ROLES*: The roles an EMR security administrator should apply for system management

1: Let H be the set of roles in the *GeneralPosition* and *ConceptualPosition* levels of *Hierarchy*
2: Let $Accuracy_{role}$ be the predictive accuracy score for each role in *Hierarchy* (the reader is referred to the main text for the details on how *Vectors* is applied in the accuracy computation)
3: $ROLES \leftarrow NULL$
4: **for** each $role \in H$ **do**
5: $R_{role} = (|U| - N_{role})/|U|$, where N_{role} is the number users in this role
6: $A_{role} = (Accuracy_{role} - \overline{Accuracy}_{sub(role)}$
7: $S_{role} = \alpha R_{role} + (1 - \alpha) A_{role}$
8: **end for**
9: Sort H by the corresponding scores S_{role} in descending order
10: **for** each $role \in H$ **do**
11: If $S_{role} < \tau$, then break
12: Else
13: $ROLES \leftarrow ROLES \bigcup role$
14: $ROLES \leftarrow ROLES$ - the children of *role* in *Hierarchy*
15: $H \leftarrow H$ - the children of *role* in *Hierarchy*
16: **end for**
17: **return** *ROLES*

threshold value. At this point, the set of roles is returned to the administrator and the algorithm terminates.

5.3 Experiments and Results

5.3.1 Initial Role Prediction

Before applying the Role-Up algorithm, we first investigated the predictability of the roles when the system i trained and tested at each level of the role hierarchy. The results of this experiment are reported in Table 5.1. First, we observe that when the system is trained and tested at the initial Specific-Position level (i.e., with 140 user positions we observe that the system is 51% accurate. In other words, a little more than half of the users can be

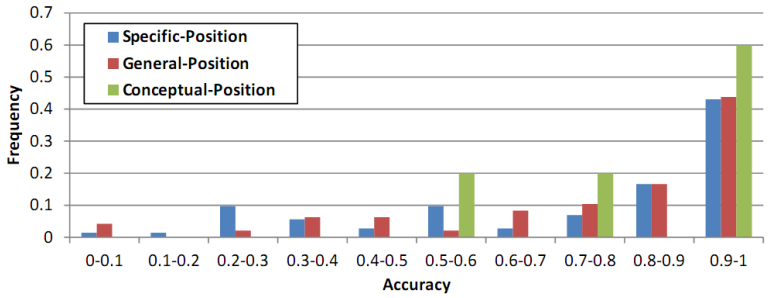


Figure 5.2: The distribution of role predictability (i.e., accuracy) at various level of the role hierarchy.

accurately predicted as their corresponding roles.

Table 5.1: Predictability of the roles when the system is trained and tested at various levels of the hierarchy.

Level of Role Hierarchy	**Accuracy**
Specific-Position(original role design)	51.34%
General-Position	52.45%
Conceptual-Position	82.38%

When we step up the hierarchy one level to General-Position, we find there is only a marginal gain in performance. We observed that the accuracy increased by approximately 1% to 52.5%. This was somewhat surprising because this level has less than half the number of roles than Specific-Position. However, when stepped up one more level to Conceptual-Position, we find that the system became significantly more predictable. Notably, the accuracy increased by approximately 30% to 82%.

However, it should be noted that the accuracy of Specific-Position and General-Position is not uniformly distributed across roles. Rather, there are a significant number of roles that are highly predictable. To illustrate this observation, Figure 5.2 depicts the distribution of accuracy scores for the roles at each level in the role hierarchy. Notice that for the Specific-Position and General-Position levels, an accuracy of 0.5 or greater is achieved for a 57 and 39 (or 79% and 81.2%) roles, respectively.

To make this result more concrete, Table 5.2 provides a summary of the five most and

five least predictable roles in the Specific-Position level. There were ten roles that achieved 100% prediction, so for presentation purposes we randomly selected five roles.

Table 5.2: The most predictable roles and the least predictable roles in the system.

Rank	Most Predictable	Accuracy	Users
1(tie)	AP-Technologist	100%	54
1(tie)	ED Assistant	100%	26
1(tie)	ED NMH Physician-CPOE	100%	43
1(tie)	NMH Resident/Fellow Clinic-CPOE	100%	10
1(tie)	Patient Care Staff Nurse - Lactation	100%	14

Rank	Least Predictable	Accuracy	Users
140	Patient Care Staff Nurse	7.6%	1554
139	Rehab OT	14.3%	28
138	TransferE	20%	20
137	View Only PC 3	21.4%	14
136	Patient Care Staff Nurse (Pilot)	22.1%	217

Despite the finding that a significant number of roles received accuracy greater than the system average of 0.5, many of these roles were smaller in terms of the number of users that they cover. Thus, although there are a few outliers, it appears that roles in the EMR system with a small number of users tend to obtain a high predictability while roles with a large number of users are less predictable. It is not surprising too much, because a small number of users implies the role is more specific, and have more specialized responsibilities compared to other roles. As such, roles with a small number of users can be distinguished from other roles more easily. Figure 5.3 provides a visual depiction of the relationship between the number of users in a role and the prediction accuracy. As also noted in Table 5.2, the largest role, Patient Care Staff Nurse, was also the least predictable. However, some of the larger roles, such as Med Student CPOE, which contained about 500 users, achieved very high prediction rates (i.e., over 80%). This is a clear illustration of why there is no one-size-fits all approach to role engineering or role mining.

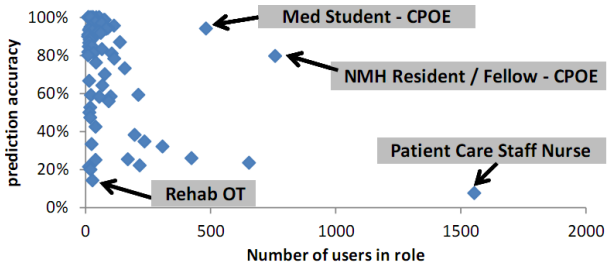


Figure 5.3: A plot of accuracy of role as a function of the number of users in the role

5.3.2 A Case Study in Incorrect Predictions

Accuracy provides an indication of how predictable each role (and the system) is, but it obscures the intuition behind why the system is failing to predict roles correctly. Thus, we take a moment to illustrate a case study in the types of mispredictions that occur in the system.

As Table 5.2 shows, the role of Patient Care Staff Nurse is the least predictable role among all 140 roles in the Specific-Position level. Thus, it is useful to know which roles the system has predicted these users belong to. Table 5.3 depicts the probabilities for the five least correct predictions for Patient Care Staff Nurse and Transfer, respectively. Semantically (and literally), they are very similar roles, and we may infer that these roles are often assigned the same tasks as Patient Care Staff Nurse. Hence, merging Patient Care Staff Nurse with Patient Care Staff Nurse Lactation, RAD Nurse or other similar roles in this table should lead to a more predictable role. This was one of the inspirations for the expert design of reasonable role hierarchies.

Table 5.4 provides an indication of which roles were being confused in the prediction process. Specifically, it reports on the conditional probability of predicting a role given the original role. For instance, there was an 85% chance of predicting Rehab PT if the original role was Rehab OT. Similarly, there was a 60% chance of predicting Rehab OT if the original role was Rehab PT. This is further justification for generalizing roles for EMR management purposes.

Table 5.3: Most likely incorrect role predictions for Patient Care Staff Nurse and Transfer

Predicted Role	**Percent**
Patient Care Staff Nurse - Lactation	19.6%
View Only PC 1	14.3%
RAD Nurse	14%
Patient Care Staff Nurse (Pilot)	10.4%
SN-RN/Customer Service	5.8%
Predicted Role	**Percent**
Patient Care Staff Nurse - Lactation	15%
Unspecified	10.0%
Unit Secretary 1	10.0%
Patient Care Staff Nurse (Pilot)	10.0%
SN-Management	5.0%

Table 5.4: Most likely incorrect predictions among all of the predictions.

Original Role	**Predicted Role**	**Probability**
RehabOT	Rehab PT	85.7%
Patient Care Staff Nurse - Agency	Patient Care Staff Nurse - Lactation	75.0%
Rehab PT	Rehab OT	60.0%
View Only PC 3	Patient Care Staff Nurse - Lactation	50.0%
Medical Records - Scanner	Medical Records	47.4%

5.3.3 Rolling-Up Role Prediction

The following set of experiments report on the application of the Role-Up algorithm. For the purposes of this work, we set the threshold in the algorithm equal to α. In contrast to the earlier experiments, Role-Up permits the hierarchy to allow for roles managed at different levels in the hierarchy. Table 5.5 shows the number of roles recommended by the approach and the accuracy of the resulting system under different values of α. From this table we wish to highlight three findings. First, there is a tradeoff in specificity in roles and accuracy of the system. Notice that when α is low, between 0.1 and 0.4, the number of roles is relatively small (i.e., 27), but the accuracy of the system is relatively high (i.e., approximately 63%). And, when α is higher, such as at 0.8, the specificity of the system is relatively high (i.e., 60 roles), but the accuracy is lower (i.e., approximately 52%).

Second, we note that $\alpha \geq$ 0.8 appears to be the most appropriate choice. When the

Table 5.5: Results of rolling-up the hierarchy under different α

α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
# of Roles Recommended	27	27	27	27	54	55	55	60	64
Accuracy of Role Predictions	63.3%	63.3%	63.3%	63.3%	49.9%	50.2%	50.2%	51.8%	51.3%

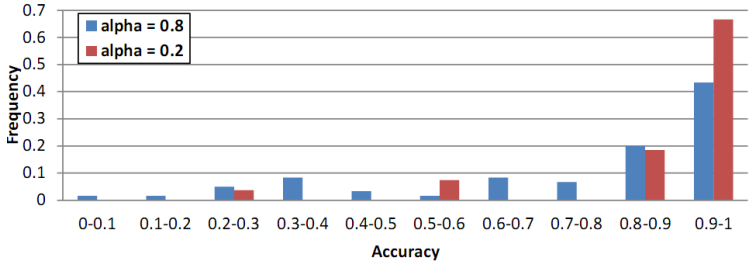


Figure 5.4: Distribution of the accuracy for the system when α is set to 0.2 and 0.8.

system is set at this level, Role- Up achieves an accuracy that is slightly better than that of original role designation while maximizing the number of roles retained for all α settings.

Third, a security administrator might also wish to consider $\alpha \leq 0.4$. At this setting, the system proposed by Role-Up achieves high accuracy, but at the loss of a significant number of roles.

As an illustration of the tradeoff between high and low α's, Figure 5.4 provides a discussion of the accuracy per role.

Chapter 6

WOBA: WOrkflow Based Audit System

6.1 Introduction

As mentioned earlier, organizations have attempted to prevent malicious insider activities through two types of technical strategies, which are audit and access control. The two strategies have different applicable scenarios. Recent efforts even shows it is possible to determine which one to be adopted by a quantitative way [60]. In this chapter, we focus on an audit method. While most of audit strategies are on the basis of machine learning methods [40, 41, 14], there has been little investigation into incorporating workflow information while selecting features. This seems like a missed opportunity because there could be valuable information for charactering a user or their accesses. A realistic assumption is that a user tends to appear in relevant workflow (e.g., It is not likely a cardiologist will treat a HIV patient), such that the attributes extracted from the workflow can be discriminative with respect to the legitimacy of accesses.

The aim of this chapter is to introduce a workflow based audit (WOBA) framework that takes advantage of workflow information to detect insider threat. This framework consists of two phases. First, a feature extraction algorithm is applied on the actual workflows to obtain useful features (workflow features). In this chapter, we generate two different types of workflow features. Second, we combine workflow features with additional features that are commonly used for anomaly detection (ordinary feature) to characterize accesses. On the accesses, we build a classification model using a statistical machine learning algorithm. There are several primary contributions of this report, including:

Specifically, there are several primary contributions of the paper, including:

- **An audit framework leveraging workflow information** We devise a novel but sim-

ple framework that can integrating workflow features with more traditional features. Under this framework, any workflow information could be translated into features suitable for classical machine learning model. Due to its simplicity, this technique is highly adaptable to most information systems involving workflow. In addition, since each access is considered as the unit of suspiciousness, this model is capable of accomplishing finer-grained anomaly detection.

- **Empirical Analysis** We illustrate how to apply and experiment with SPA on a real electronic health record system in a large medical center. To evaluate the effectiveness of SPA, we compare it with the classification model built without workflow features. The results show SPA can achieve higher accuracy and AUC on detection than classification with only ordinary features.

The reminder of this chapter is organized as follows. Section 6.2 reviews the foundations upon which our method is built, including traditional features, workflow features and sequential patterns. Section 6.3 gives a detailed introduction about the entire WOBA framework. Section 6.4 presents the dataset preparation and experiment design. Section 6.5 presents detailed results of the experiments.

6.2 Preliminaries

In this section, we introduce preliminary concepts and techniques that form the foundation of the WOBA framework. First, we give a description of workflow in a healthcare system. Second, a context-based classification model is introduced. Finally, we review the sequential pattern mining.

6.2.1 Workflow

In this chapter, we refer to the access event that is under review as the target. We assume that the target access takes place in the midst of a workflow, which we represent

as a sequence of accesses, such that each is associated with the same underlying resource. We will represent a workflow as $\varepsilon = \langle e_1, e_2, \ldots, e_i, \ldots, e_l \rangle$. For illustration, Figure 6.1 depicts a series of accesses to a specific patient's EMR from the point of admission to discharge from a hospital. Here, e_3 is the target access and the corresponding workflow is $\langle e_1,e_2,e_3,e_4,e_5,e_6 \rangle$. Workflow features are extracted from all accesses except target access that transpires in the workflow.

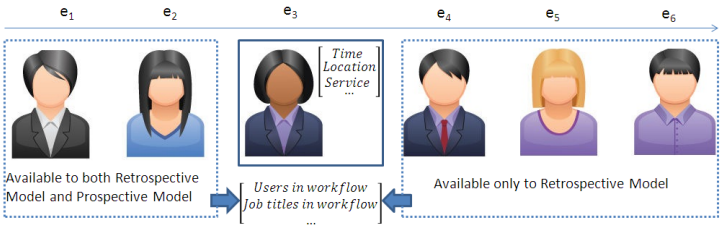


Figure 6.1: An example of a workflow of accesses to a patient's medical record. Here, the target access e_3 is surrounded by a solid rectangle. The other accesses in the workflow are surrounded by a dashed rectangle. The parts contained by brackets represent context.

6.2.2 Sequential Rules

We use Table 6.1 as a running example to explain concepts in this section. Table 6.1 is a sequence database which consists of a set of sequences. Each sequence consists of ordered items. A **subsequence** of a sequence s is defined as sequences generated by removing one or multiple items in s. For example, $< ace >$ is a subsequence of the first sequence in the database because it can be derived by removing 2^{nd}, 3^{rd} and 5^{th} items in the original sequence. On the other hand, s is defined as a **supersequence** of its subsequences.

A sequence's **support** is the number of its supersequences in sequence database. $A \rightarrow B$ is a sequential rule where A and B are both sequences. **confidence** of $A \rightarrow B$ is defined by $support(AB)/support(A)$, where AB is a sequence constructed by joining A and B. $A \rightarrow B$ is named **frequent sequential rule** when confidence($A \rightarrow B$) and support(AB) are both larger than thresholds. Consider an example of the rule $ac \rightarrow e$. Its support is 2 because 1^{st} and 2^{nd} are its supersequences. Also, since the support of ac is 3 (sequence 1, 2 and

4 contains it), the confidence of the rule is 2/3. Assume we set threshold for support and confidence as 2 and 0.5 respectively, then $ac \rightarrow e$ is considered a frequent rule.

Table 6.1: Sequence Database: An Example

SID	Sequences
1	⟨abacde⟩
2	⟨bcdeabacde⟩
3	⟨edbbbae⟩
4	⟨ceaaabc⟩

6.3 Framework

6.3.1 Context-based Classification

We use $C = \{C_1, C_2, \ldots, C_h\}$ to denote the set of context that is associated with a target access. C_r is composed of elements from $dom(C_r)$, which is the *domain* of elements associated with this type of context. For example, let $U \in C$ denote all users that attend the workflow of target access. As such, we have $dom(U) = \{u_1, u_2, \ldots, u_d\}$, such that u_i is a certain user in the system. As mentioned before, a workflow is represented as $\varepsilon = \langle e_1, e_2, \ldots, e_i, \ldots, e_l \rangle$, from which features would be generated to characterize e_i. For e_i, we can use vectors as representations of all h types of context. Equation 7.12 denotes $V(U)$, the vector corresponding to context U.

$$V(U) = (v_{u_1}, v_{u_2}, \ldots, v_{u_d}) \tag{6.1}$$

In this model, v_{u_x} is set to 1 if u_x is observed when at least one $e_j \in \varepsilon_1$ transpires, otherwise it is set to 0.

For example, imagine we want to construct a vector corresponding to U (i.e., $V(U)$), for the target access e_3 in Figure 6.1. Let $dom(U) = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ in the system and $\langle u_2, u_4, u_5, u_1, u_3, u_8 \rangle$ be the user sequence corresponding to the workflow in Figure 6.1. $\varepsilon_1 = \langle e_1, e_2, e_2, e_3, e_4, e_5, e_6 \rangle$ is the workflow containing e_3, where all accesses (except

e_3) are executed by u_2, u_4, u_1, u_3 and u_8 respectively. Thus, the vector corresponding to U for target user is $(1,1,1,1,0,0,0,1)$.

We use $\oplus$ to denote the union of two vectors[1]. As such, the vector for all h context can be represented as $CV = V(C_1) \oplus V(C_2) \oplus \ldots \oplus V(C_h)$.

6.3.2 Use Sequential Rule as Feature

Building on the model in the previous section (6.3.1), it is straightforward to use some features to represent a target access, such as user IDs and job titles that occur in the surrounding workflow. In this section, we introduce a more complex feature for the access representation. Consider a simple example, in which a radiologist tends to come after specialist in a common healthcare workflow. It is rare to see the reverse case for the example. Thus, sequential patterns could be a potentially useful feature to charaterize a target access. In this section, we first introduce how to learn sequential pattern from workflows, and then describle how we represent an access using these patterns.

6.3.2.1 Generating Rules

In this section, we introduce a procedure named **workflow serilaization** to construct a sequence database from a single workflow. Let us assume there exists a workflow as shown in Figure 6.2 (i.e. ebacdebesejklkfeeebaecde). At the beginning, a window with predefined size k covers the first k items in the workflow. Then, the following procedure accomplishes **workflow serilaization**: 1) The subsequence covered by the window is extracted and stored into the database, 2) Slide the window forward by one position, and 3) If window currently covers the last item in the workflow then exit, otherwise, go to step 1.

[1]For example, vector $C = \langle a_1, a_2, \ldots, a_m, b_1, b_2, \ldots, b_n \rangle$ is the union of vector $A = \langle a_1, a_2, \ldots, a_m \rangle$ and vector $B = \langle b_1, b_2, \ldots, b_n \rangle$ (i.e., $C = A \oplus B$)

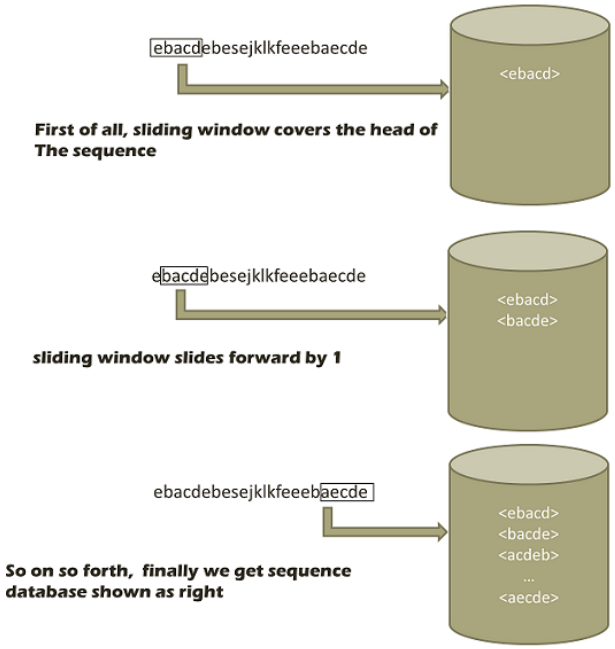


Figure 6.2: An example of workflow serilaization

We then use the procedure in Figure 6.3 to obtain sequential rules. Each sequential database is constructed from a training sequence (workflow) by **workflow serilaization**. Then each of them is taken as input of a sequential rule mining algorithm, with a set of sequential rules as the output. It is easy to see that each workflow yields a set of rules. Let us use set_i to denote the rule set corresponding to the i^{th} workflow. We can obtain an integrated rule set named $allRules$ by performing following operation.

$$allRules = set_1 \cup set_2 ... \cup set_n \tag{6.2}$$

6.3.2.2 Use Rules as Features

Algorithm 5 describes the procedure of using sequential rules as features to charaterize an access. The inputs include $allRules$, the workflow surrounding the target access (i.e., seq, usually consists of the sequence of user ID or job titles in the workflow), and a vector that will become the representation of the access. First, we use **workflow serilaization** to

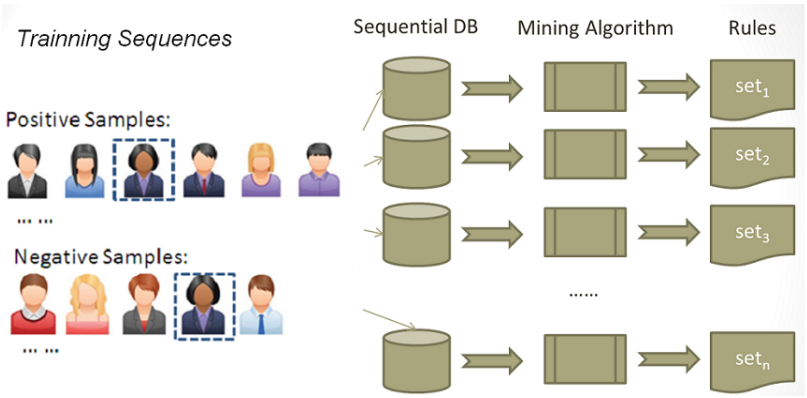


Figure 6.3: Rule generator

form the sequence database *sdb* corresponding to *seq*. Then, for the i^{th} rule in *allRules*, we compute its confidence and support in *sdb*, which are assigned to c_i and s_i of V, respectively. The resulting V is a feature vector capturing the sequential information in *seq*.

Algorithm 5 Use Rules as Feature

Input: $allRules$, seq, $V = [c_1, s_1, c_2, s_2, ..., c_m, s_m]$
Output: V

1: $sdb \leftarrow workflowSerilaization(seq)$
2: **for** each $rule_i \in allRules$ **do**
3: $\quad c_i = confidence(rule_i, sdb)$
4: $\quad s_i = support(rule_i, sdb)$
5: **end for**
6: **return** V

6.4 Experimental Design

This section provides an overview of the experiments designed for this study. It begins with a description of the real electronic medical record (EMR) data. This is followed by an explanation of how context was modeled to train the prospective and the retrospective security models. We then introduce the machine learning algorithm used for training the models and the specific measures used for assessing their performance.

6.4.1 Electronic Medical Record

In the EMR of Northwestern Memorial Hospital, we assume each (*patient-id*, *encounter-id*) pair defines a unique workflow for patient treatment. This encounter begins when the patient is admitted to the hospital and ends two weeks after discharge (to ensure that accesses associated with medical billing are captured). Of the remaining information, there are five types of context: i) the time a target access was issued (Time)[2], ii) the hospital service the patient was on at the time of the target access (e.g., General Medicine vs. Obstetrics), iii) location in the medical center where the patient resided when the target access was issued, iv) the users who commit accesses in the workflow of target access and v) the job titles associated with these users.

We are also interested in seeing how sequential rules (features) improve the performance. Thus we use the procedure in Section 6.3.2 to generate sequential rules as additional features. In the experiment, we use the ClaSP [61] algorithm in SPMF package [62] to mine sequential rules (features).

6.4.2 Dataset Preparation

Without loss of generality, we assume target user t participates in N patient workflows. The corresponding context vectors are $CV_1^+, CV_2^+, \ldots, CV_N^+$, which are composed using the approach described in Section 6.3.1. These vectors are associated with a *positive* label class. We use the following process to generate a corresponding set of N *negative* labeled instances. We randomly select a workflow in which user t failed to issue an access. From this workflow, we randomly select an access and build a corresponding context vector. Doing so N times yields a set of vectors $CV_1^-, CV_2^-, \ldots, CV_N^-$, which are associated with the negative class. Note that we create different CV_i^+ and CV_i^- for prospective model and retrospective model respectively.

[2]For this work, $dom(Time)$ consists of four values: a) Morning (6am - 12pm), b) Afternoon (12pm - 6pm), c) Evening (6pm - 12am), and d) Night (12am - 6am)

To conduct our evaluation, we construct 10 datasets, each of which corresponds to a different job title. Let us use *Patient Care Staff Nurse* as an example. We randomly pick 10 users whose job titles are *Patient Care Staff Nurse*. For each user, we construct N positive samples and N negative samples using the process described above. We select 80% of the vectors from the positive and negative samples, respectively, for the training set, and use the remaining 20% as the test set. The samples generated for all 10 users are then combined to form a single dataset for this job title and the overall performance across the 10 users is measured to evalute the entire dataset. To ensure the results are representative, we select job titles from 10 different hospital departments. The job titles and summary statistics are shown in Table 4.3.

We train a classifier for each user using a support vector machine (SVM) using an RBF kernel [50]. We utilize a grid search technique [50] to find values for parameters to enable a robust SVM. For each user in the job title, we use the classifier trained on the training set of this user to assess the corresponding test set.

6.5 Experimental Result

This section shows the results of experiments. First, we introduce the results of WOBA without sequential feature. Then we show how sequential rules can help to improve the performance.

Table 6.2 shows how the workflow feature can help the prediction. The second column corresponds to accuracy of WOBA, while the third column corresponds to the accuracy of the predictive model with only ordinary features. The third column corresponds to the accuracy of the predictive model with only workflow features. There are several interesting observations. First of all, WOBA can achieve averagely 90% accuracy, which is very satisfactory. Second, ordinary features and workflow features are both important to the prediction performance. Absence of any of them could lead to a significant reduction in accuracy. Third, it is observed that one type of feature would hardly constantly beat the

other. In other word, different types of features can be useful in different settings.

Table 6.2: Accuracies

Job Title	WOBA	Workflow Feature	Ordinary Feature
NMH Physician CPOE	0.940	0.915	0.831
Resident/Fellow CPOE	0.922	0.890	0.816
Emergency Department Patient Care Staff Nurse	0.929	0.883	0.871
Utilization Review/Quality Assurance 1	0.933	0.897	0.881
Unit Secretary	0.977	0.948	0.930
Anesthesia CPOE	0.905	0.877	0.817
Radiology Resident/Fellow	0.875	0.842	0.765
Rehabilitation - Physical Therapist	0.919	0.897	0.847
Patient Care Assistive Staff	0.965	0.940	0.938
Patient Care Staff Nurse	0.918	0.866	0.846

Table 6.3 shows how sequential features improve WOBA. In the expriment, we use only one type of workflow feature: the set of role that occur in the workflow surrounding the target access. In the meanwhile, we extract sequential rules from the sequence database formed by role sequences as the sequential features. In the second column of the table, there are accuracies of WOBA with only role features, while in the third column there are accuracies of WOBA with both role features and sequential features. It shows that sequential features can improve the classification accuracy.

Table 6.3: WOBA with Sequential Feature

Job Title	R	R + S
NMH Physician CPOE	0.803	0.819
Resident/Fellow CPOE	0.807	0.807
Emergency Department Patient Care Staff Nurse	0.862	0.863
Utilization Review/Quality Assurance 1	0.831	0.859
Unit Secretary	0.879	0.902
Anesthesia CPOE	0.841	0.851
Radiology Resident/Fellow	0.826	0.838
Rehabilitation - Physical Therapist	0.843	0.845
Patient Care Assistive Staff	0.816	0.844
Patient Care Staff Nurse	0.818	0.825

Chapter 7

Quantifying the Tradeoff between Prospective and Retrospective Access Decisions

As mentioned earlier, the insider threat can be addressed through two technical strategies: i) *prospective* methods, such as access control, that make a decision at the time of a request, and ii) *retrospective* methods, such as *post hoc* auditing, that make a decision in light of the knowledge gathered afterwards. While it is recognized that each strategy has a distinct set of benefits and drawbacks, there has been little investigation into how to provide system administrators with practical guidance on when one or the other should be applied. To address this problem, we introduce a framework to compare these strategies on a common quantitative scale. In doing so, we translate these strategies into classification problems using a context-based feature space that assesses the likelihood that an access request is legitimate. We then introduce a technique called *bispective analysis* to compare the performance of the classification models under the situation of non-equivalent costs for false positive and negative instances, a significant extension on traditional cost analysis techniques, such as analysis of the receiver operator characteristic (ROC) curve. Using domain-specific cost estimates and access logs of several months from a large Electronic Medical Record (EMR) system (see Chapter 3), we demonstrate how bispective analysis can support meaningful decisions about the relative merits of prospective and retrospective decision making for specific types of hospital personnel.

We begin by introducing the motivation and background for our tradeoff quantification in Section 7.1. Then, some preliminary knowledges relevant to proposed approach are introduced in Section 7.2. In Section 7.3, we introduce a novel framework to support decisions between security strategies. In Section 7.4, we describe the experimental models for assessing the proposed framework. Finally, we report on empirical results for the traditional and proposed methods in Section 7.5 and 7.6, respectively.

7.1 Introduction

A fundamental tradeoff in authorization pits making a decision prospectively, before access is granted, against making a decision retrospectively, when an audit is carried out. Much of the work on access control has focused on the prospective decision making, but it has often been pointed out [63, 64] that retrospective decision making, in which users beg for forgiveness rather than permission, has some significant advantages. In many applications: (1) it is difficult to determine what access a user requires in advance, (2) denying access to a user with a legitimate need could result in significant inconvenience, expense, or loss, (3) most users are responsible and can be trusted to access resources for legitimate reasons, and (4) accountability (such as disciplinary action) is effective in deterring abuses. An iconic example of such a situation is access to patient records in Electronic Medical Record (EMR) systems, where (1) hospital workflows are complex and commonly involve emergencies and unexpected events, (2) lack of timely access could result in the loss of a patient's life, (3) most healthcare providers are highly trained and ethical professionals, and (4) there are strong penalties for abuse. These four criteria (and others, such as the ability in certain cases to roll back an illegitimate action) provide a good qualitative story for when retrospective decision-making based on audit may be better than prospective decision-making based on preventing access to a resource. We see the phenomena in many non-computer contexts already. For example, a red light tells a driver not to cross an intersection, but it does not prevent the driver from crossing it. On the other hand, there are instances where retrospective techniques are inadequate or too risky: the honor system may not be sufficient if the stakes for abuse are too high and the effectiveness of accountability is too low.

Given the recognition that retrospective techniques will have their place, we are led to ask: is there any systematic way to determine when retrospective techniques are better than prospective ones? Ideally this would be done quantitatively by measuring the tradeoff between the risks of addressing an abuse at audit time versus denying access to user when

it is requested. If we accept the idea that the implementation of access control provides, in general, only an approximation of the desired access rules, then we may be able to quantify the rules with a ROC that compares false positives to true positives (a technique commonly used already for biometric authentication systems [65]). Better decision making then means better Area Under the ROC Curve (AUC) values. For example, if we are able to estimate that a prospective access control system gives proper access 95% of the time (true positives), but only if we accept that 10% of the time it will grant access where access should not have been granted (false positives), then we are on the path to quantify whether one type of prospective access is better than another. However, this does not offer a clear way to compare prospective techniques with retrospective ones. The latter, which can use information from both before and after a user has accessed a record, is expected to have better AUC values. The problem is that we do not have a cost model that allows us to judge tradeoffs between a pair of ROCs.

The aim of this chapter is introduce a technique called *bispective analysis* that can be used to compare prospective and retrospective techniques for access control via a model that accounts for the different costs associated with false positives and negatives associated with each model. This is accomplished by weighting the ROC models for prospective and retrospective techniques by their costs and, subsequently, combining these in a way that enables direct comparison to see which is better in which circumstance. The primary contributions are:

- **A Novel Cost Analysis Technique** We devise a novel cost comparison method called bispective analysis that allows for an explicit comparison of classification models with different costs. Once provided with the knowledge of the variables (i.e., the costs of false positive and false negative for prospective model, the costs of false positive and false negative for retrospective, and the ROC curves for both models), bispective analysis allows administrators to calculate which is the better option. Moreover, bispective analysis provides insight about the distribution of results under

varying cost models, such that administrators can make decisions when their confidence in the variables is uncertain (e.g., only a range of costs are known or only partial costs are known).

- **Classification Models for Prospective and Retrospective Security** We develop a technique to represent and evaluate both prospective and retrospective models. To do so, we translate the context associated (e.g., other users who accessed a record, when the access was committed, and where the entity associated with the record was located) with each access into a vector space representation. We then subject such vectors to a classical machine learning model to build classifiers. In this way, prospective model and retrospective model are mapped to a common framework, such that comparable results can be generated. In addition, due to its simplicity and compactness in representation, this technique is scalable and adaptable to most information systems.

- **Empirical Analysis and Case Study** We illustrate how to apply bispective analysis to analyze tradeoffs for a large urban hospital system based on its EMR audit logs to provide assessments for various positions at the hospital. We deploy prospective and retrospective models implemented by the proposed technique in this system, and then obtain detection results (i.e, false positive rate, false negative rate) respectively. With bispective analysis and our detection results, we conduct illustrative case studies about the model selection with different assumptions on costs. In doing so, we assess how the model plays out for ten care provider positions in the system. The results show how cost weighting can yields different guidance in comparison to a standard ROC analysis.

7.2 Preliminaries

This section begins by reviewing basic concepts in classifier performance evaluation that are relevant to our strategy. Next, we introduce the definition of the cost of a classifier. This is followed by a review of the concept of an ROC curve, and several ROC-based comparison methods for classifiers. Finally, we review the notion of *context*, which is used in the implementation of our prospective and retrospective models.

7.2.1 Basic Concepts

The application of a classifier to a test instance results in either a correct or an incorrect decision. To assess the performance of a classifier, we consider the rates of these results over a set of cases. In doing so, the following simple measures are relevant: 1) True Positive Rate (tpr): the fraction of positive samples correctly classified; 2) False Negative Rate ($fnr = 1 - tpr$): the fraction of positive samples misclassified; 3) True Negative Rate (tnr): the fraction of negative samples correctly classified; and 4) False Positive Rate ($fpr = 1 - tnr$): the fraction of negative samples misclassified. Finally we report 5) *Accuracy*: the fraction of all samples correctly classified.

For orientation, it should be made clear that false positive and negatives have different implications (and thus different costs) in prospective and retrospective systems. In the prospective system, a false positive indicates the system approves an illegitimate access, while a false negative indicates the system denies access to a legitimate request. In the retrospective system, a false positive indicates that no investigation is performed for an illegitimate access, while a false negative means the system recommends an investigation for a legitimate access.

7.2.2 Cost Function

The *cost* of a classifier can be represented by Equation 7.1 [27]. Let π_1 and π_0 be the prior probabilities of positive and negative cases, respectively, such that $\pi_0 = 1 - \pi_1$. Let p_{10} and p_{01} be the fnr and fpr, respectively. And, let $c_{10} \in (0, \infty)$ and $c_{01} \in (0, \infty)$ be the associated costs for the fnr and fpr, respectively. In the remainder of this chapter, we refer to c_{10} and c_{01} as the *false negative cost* and *false positive cost*, respectively.

$$cost = \pi_1 p_{10} c_{10} + \pi_0 p_{01} c_{01} \tag{7.1}$$

7.2.3 ROC Curve

The result of a probabilistic classifier is dependent on its parameterization. For example, the naïve Bayes classifier incorporates a threshold for the probability with which it claims a class label (e.g., negative versus positive) corresponds to a certain instance. Traditionally, the result of a classifier is represented by a (fpr, tpr) pair. The ROC curve can be obtained by plotting these pairs with respect to a range of parameterizations of the classifier. And, the AUC [66] is a commonly used measure for the evaluation of classification models. The larger the *AUC* of a classifier, the better its performance.

Now, in this setting, a classifier A is said to dominate another classifier B if for any point (fpr_A, tpr_A), there exists a point (fpr_B, tpr_B), such that $tpr_B > tpr_A$ and $fpr_B < tpr_A$. For example, in Figure 7.5(a), it can be seen that the ROC of the retrospective model dominates the ROC of the prospective model.

Given any combination of π_1, π_0, c_{10} and c_{01}, $MIN(cost_A) < MIN(cost_B)$ will be true if A dominates B [27], where $MIN(cost_X)$ is the minimal value of cost over the ROC curve of classifier X. This proposition is true because the ROC of A forms the convex hull for both A and B, and the point (fpr, tpr) that minimizes cost, for any combination of π_1, π_0, c_{10} and c_{01}, is only located on the convex hull [27]. As noted in Section 2.1.2, a premise for

the convex hull method is that the cost of a false positive (negative) is equivalent for both classifier A and B. However, as we will show in our empirical analysis, selecting security models by identifying dominance is inappropriate in situations for which this premise fails to hold.

7.2.4 Cost Curve

In this section, we review the cost curve introduced in [28]. As mentioned in the Chapter 2, the cost curve retains all the merits of the ROC curve, but provides for several notable benefits. Though it is also hampered by the assumption of equivalent costs (as mentioned above), it serves as a foundation of our cost analysis.

Given estimates for π_1, c_{10}, π_0 and c_{01}, we can discover a point on the ROC curve to minimize *cost*. It has been proven that only $W = \frac{\pi_0 c_{01}}{\pi_1 c_{10}}$ is needed to determine the point $(1-\bar{p}_{10}, \bar{p}_{01})$ of ROC that can minimize *cost*[27].

[28] introduced the concept of a normalized expected cost, which is defined in equation 7.2. $(\pi_1 c_{10} + \pi_0 c_{01})$ in Equation 7.2 is the maximized *cost* because it indicates both p_{10} and p_{01} are equal to 1. In other words, the classifier has misclassified all samples. Thus, computing *normcost* corresponds to normalizing *cost* into the (0,1) range. In this model, $(1-\bar{p}_{10}, \bar{p}_{01})$ in the ROC minimizes *normcost* as well.

From Equation 7.2, we can state $K = W/(W+1) = \pi_0 c_{01}/(\pi_1 c_{10} + \pi_0 c_{01})$, which means K and W constitute a one-to-one mapping. So, the values for $\bar{p}_{10}$ and $\bar{p}_{01}$ can be determined by K. Thus, the minimized *normcost*, denoted by $normcost^*(K)$, can be represented by Equation 7.3. [28] provides a detailed method for deriving the curve of $normcost^*$ (i.e., the cost curve). We directly employ this method when a computation of $normcost^*$ is required,

but, due to space limitations, we refer the reader to [28] for the details.

$$\begin{aligned} normcost &= \frac{\pi_1 p_{10} c_{10} + \pi_0 p_{01} c_{01}}{\pi_1 c_{10} + \pi_0 c_{01}} \\ &= p_{10} \cdot \frac{1}{W+1} + p_{01} \cdot \frac{W}{W+1} \\ &= p_{10} \cdot (1-K) + p_{01} \cdot K \end{aligned} \tag{7.2}$$

$$normcost^*(K) = \bar{p}_{10} \cdot (1-K) + \bar{p}_{01} \cdot K \tag{7.3}$$

K can be interpreted as the *false positive cost ratio*. Informally, this corresponds to the proportion of cost resulting from false positives.

7.2.5 Context

In chapter 6.2.1, concept of workflow is introduced. This section will revisit it and introduce the concept of *context* based on it. Again, we refer to the access event that is under review as the *target*. This event can be associated with a wide range of semantics, which we call the *context* around the target access. The access itself is a request to a resource that is issued by a user, but there is a variety of contextual information that surrounds the target.

We assume that the target access takes place in the midst of a workflow, which we represent as a sequence of accesses, such that each is associated with the same underlying resource. We will represent a workflow as $\varepsilon = \langle e_1, e_2, \ldots, e_i, \ldots, e_l \rangle$. For illustration, Figure 7.1 depicts a series of accesses to a specific patient's EMR from the point of admission to discharge from a hospital. Here, e_3 is the target access and the corresponding workflow is $\langle e_1,e_2,e_3,e_4,e_5,e_6 \rangle$. Context can be extracted from the target access itself (e.g., the time this access occurs). It can also be extracted from the corresponding workflow (e.g., users participated in the workflow). Note the availability of context in a workflow for the prospective model and the retrospective model are different. The retrospective model can take advantage of the entire workflow, while the prospective model can only take advantage

of the parts of the workflow that occur before the target access.

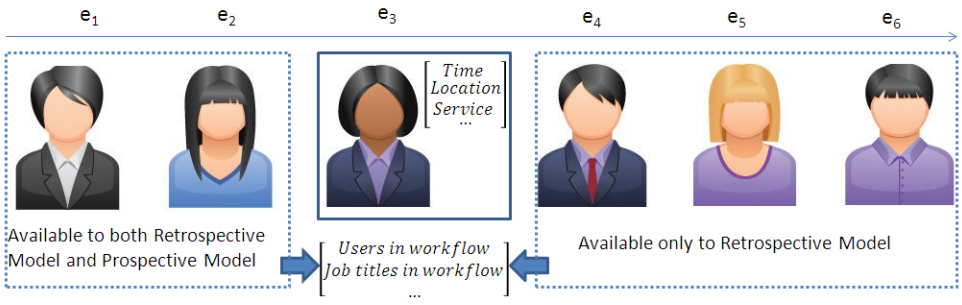


Figure 7.1: An example of a workflow of accesses to a patient's medical record. Here, the target access e_3 is surrounded by a solid rectangle. The other accesses in the workflow are surrounded by a dashed rectangle. Parts contained by brackets represent context.

7.3 A Framework to Quantify the Tradeoff between Two Strategies

7.3.1 Framework Overview

To orient the reader, Figure 7.2 provides a high-level view of the proposed decision process for a specific user. As previous work has shown [20], reliable access control policies (i.e., a prospective model) can be learned by a machine learning algorithm. We extend this notion for implementation of both the prospective model and the retrospective model. To do so, first, we extract workflows of targeted user from a database of transactions. Next, we construct vectors from the workflows to represent all accesses issued by the user. For the prospective model, the vectors are composed of contextual information that occurs at or before the point of a target access. For the retrospective model, the vectors are composed of context observed at any time (i.e., before, at or after the time of the target access). Next, the vectors are subject to a standard machine learning framework to build classifiers that are representative of prospective and retrospective models. Finally, a decision support system uses the ROC curves for the classifiers and their associated costs and returns an answer for which classifier (model) should be adopted to manage this specific user.

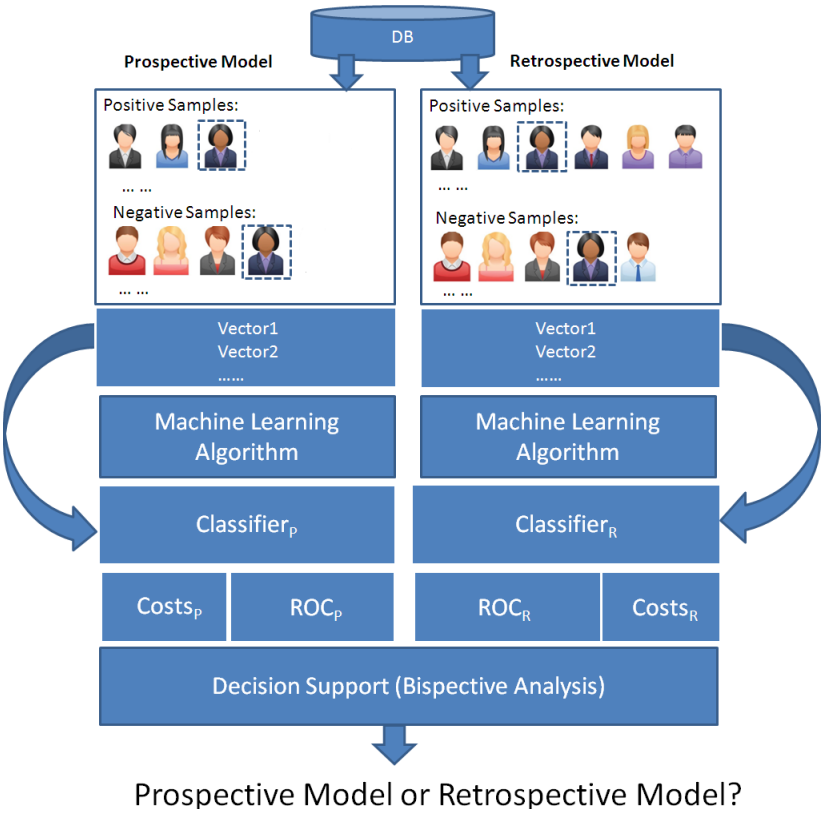


Figure 7.2: An architectual view of the Bispective Analysis

7.3.2 Decision Support

7.3.2.1 Bispective Analysis

As mentioned earlier, the prospective and retrospective security models are based on as machine learning algorithms. Traditional methods (e.g., ROC analysis) for comparing classifiers work under the belief that the costs for false positives (false negatives) are equivalent. However, this premise does not hold in the prospective versus retrospective security decision. Thus, we propose an analytic method called bispective analysis that extends cost curves to account for classifiers with differing misclassification costs. As will be illustrated, this method has a natural visual interpretation that can facilitate the decision making process.

To begin, equations 7.4 and 7.5 provide formulations for the overall cost of a prospec-

tive (P) and retrospective (R) model, respectively.

$$cost_P = \pi_1 p_{10}^{(P)} c_{10}^{(P)} + \pi_0 p_{01}^{(P)} c_{01}^{(P)} \tag{7.4}$$

$$cost_R = \pi_1 p_{10}^{(R)} c_{10}^{(R)} + \pi_0 p_{01}^{(R)} c_{01}^{(R)} \tag{7.5}$$

These functions allow us to derive a comparison function to compare the costs caused by the two models, denoted by equation 7.6.

$$comp(P,R) = ln(\frac{cost_P^*}{cost_R^*}) \tag{7.6}$$

Here, $cost_P^*$ and $cost_R^*$ correspond to the minimized overall costs given: i) the false positive (negative) costs estimates and ii) the prior distributions of positives and negatives. iii) the ROC curves. When $comp(P,R) > 0$, the prospective model incurs greater cost than the retrospective model (denoted by R $\succ$ P). When $comp(P,R) < 0$, the retrospective model incurs greater cost than the prospective model (denoted by R $\prec$ P). And, when $comp(P,R) = 0$, the prospective and retrospective models have equivalent costs (denoted by R $\simeq$ P).

The comparison function contains too many variables to be visualized in an interpretable manner. Thus, we reduce the number of variables via a mathematical deduction in Equation 7.7. Note we use the cost curve $normcost^*(K)$ in Equation 7.7. It can be seen that $comp(P,R)$ is a function of $K_P = \pi_0 c_{01}^{(P)} / (\pi_1 c_{10}^{(P)} + \pi_0 c_{01}^{(P)})$, $K_R = \pi_0 c_{01}^{(R)} / (\pi_1 c_{10}^{(R)} + \pi_0 c_{01}^{(R)})$ and $ratio = c_{01}^{(P)} / c_{01}^{(R)}$. When $ratio$ is a constant z, the comparison function can be represented as $Magnitude(K_P, K_R)$, as shown in Equation 7.8. Given this representation, we can then compose a contour for $Magnitude(K_P, K_R)$ to investigate the tradeoffs under various cost conditions.

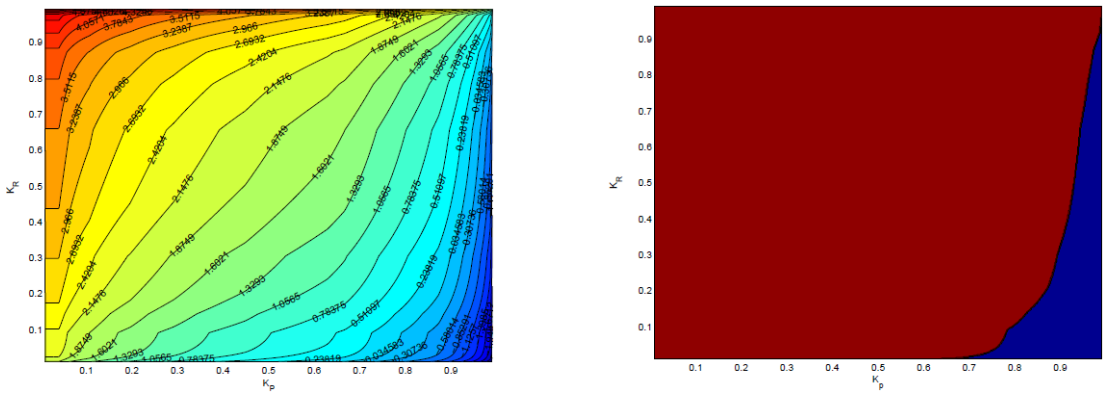
$$
\begin{aligned}
comp(P,R) &= ln(\frac{\pi_1 \bar{p}_{10}^{(P)} c_{10}^{(P)} + \pi_0 \bar{p}_{01}^{(P)} c_{01}^{(P)}}{\pi_1 \bar{p}_{10}^{(R)} c_{10}^{(R)} + \pi_0 \bar{p}_{01}^{(R)} c_{01}^{(R)}}) \\
&= ln(\frac{\pi_1 c_{10}^{(P)} + \pi_0 c_{01}^{(P)}}{\pi_1 c_{10}^{(R)} + \pi_0 c_{01}^{(R)}} \cdot \frac{\frac{\pi_1 \bar{p}_{10}^{(P)} c_{10}^{(P)} + \pi_0 \bar{p}_{01}^{(P)} c_{01}^{(P)}}{\pi_1 c_{10}^{(P)} + \pi_0 c_{01}^{(P)}}}{\frac{\pi_1 \bar{p}_{10}^{(R)} c_{10}^{(R)} + \pi_0 \bar{p}_{01}^{(R)} c_{01}^{(R)}}{\pi_1 c_{10}^{(R)} + \pi_0 c_{01}^{(R)}}}) \\
&= ln(\frac{c_{01}^{(P)}}{c_{01}^{(R)}} \cdot \frac{K_R}{K_P} \cdot \frac{normcost_P^*(K_P)}{normcost_R^*(K_R)})
\end{aligned}
\tag{7.7}
$$

$$
\begin{aligned}
Magnitude(K_P, K_R) &= comp(P,R)|_{ratio=z} \\
&= ln(z \cdot \frac{K_R}{K_P} \cdot \frac{normcost_P^*(K_P)}{normcost_R^*(K_R)})
\end{aligned}
\tag{7.8}
$$

Figure 7.3(a) depicts an example of such a contour for one user associated with the job title of *NMH Physician CPOE* (Computerized Provider Order Entry) in the EMR dataset of our case study. Each line in the contour plot, which we call a *contour line*, consists of the points (K_P, K_R) for which $Magnitude(K_P, K_R)$ has a constant value. This value is represented by the number on the contour line.

$$
Threshold(K_P, K_R) = sgn(Magnitude(K_P, K_R)) \tag{7.9}
$$

To further simplify the decision making process, we can compose a contour using Equation 7.9, where $sgn(\cdot)$ is the sign function. The value of *Threshold()* must be drawn from $\{-1, 0, 1\}$, which corresponds to $R \prec P$, $R \simeq P$ and $R \succ P$, respectively. Figure 7.3(b) provides an example of the contour after applying this threshold, where the red region corresponds to $R \succ P$, the blue region corresponds to $R \prec P$ and the boundary between them corresponds to $R \simeq P$. To provide guidance, the former contour should be utilized when the magnitude of difference between the prospective and respective models is of interest to an administrator (e.g., the trends of comparison results when K_P and K_R changes), while



(a) Contour Plot for $Magnitude(K_P, K_R)$ (b) Contour Plot for $Threshold(K_P, K_R))$

Figure 7.3: Contour plots for the *NMH* Physician CPOE role in the NMH dataset. The red and blue regions correspond to when the prospective and retrospective models dominate, respectively.

the latter should be chosen when the administrator is interested only in which model is dominant.

7.3.2.2 Probability Computation with Comparison Function

Intuitively, in a contour plot, the proportion of the area determined by $Threshold() = 1$ reflects the probability that the retrospective model will be the dominant strategy. For illustration, in Figure 7.3(b), the region shaded in red indicates the probability that retrospective is the dominant solution for the *NMH Physician CPOE* is very high.

This type of contour can enable an administrator to ascertain which model has a higher probability of effectiveness. To understand how, let us assume that $f(K_P, K_R)$ corresponds to the joint density function of K_P and K_R. Now, K_P and K_R can be considered independent because they are derived from two distinct classification models. As a consequence, the probability that the retrospective model dominates the prospective model can be represented by Equation 7.10, where $f_P()$ and $f_R()$ indicate the density functions of K_P and K_R,

respectively.

$$\begin{aligned} Pr(R \succ P) &= \int_{Threshold(K_P,K_R)=1} f(K_P,K_R)dK_PdK_R \\ &= \int_{Threshold(K_P,K_R)=1} f_P(K_P)f_R(K_R)dK_PdK_R \end{aligned} \tag{7.10}$$

A common and reasonable assumption for $f_P()$ and $f_R()$ is the density function of the uniform distribution with range (0,1) [28, 67]. This is useful because, in combination with Equation 7.10, it follows that $Pr(R \succ P)$ corresponds to the proportion of the contour where $Threshold(K_P,K_R) = 1$. More formally, this is derived as follows

$$\begin{aligned} Pr(R \succ P) &= \int_{Threshold(K_P,K_R)=1} f_P(K_P)f_R(K_R)dK_PdK_R \\ &= \int_{Threshold(K_P,K_R)=1} 1 \cdot 1 dK_PdK_R \\ &= \int_{Threshold(K_P,K_R)=1} dK_PdK_R. \end{aligned} \tag{7.11}$$

7.3.3 Context-based Classification

WOBA framework in chapter 6 will be used to translate strategies into context-based classification models. To make this section more clear, some materials in chapter 6 will be repeated in this section. The context-based classification consists of three steps: i) construct vectors from the workflows; ii) train a classifier on a subset of the vectors; and iii) test the classifier on the remainder of the vectors. Since the work of this chapter does not focus on a specific machine learning algorithm, here we focus on the process by which we construct vectors used for prospective and retrospective models.

7.3.3.1 Prospective Model

We use $C = \{C_1, C_2, \ldots, C_h\}$ to denote the set of context that is associated with a target access. C_r is composed of elements from $dom(C_r)$, which is the *domain* of elements associated with this type of context. For example, let $U \in C$ denote all users that attend the

workflow of target access. As such, we have $dom(U) = \{u_1, u_2, \ldots, u_d\}$, such that u_i is a certain user in the system.

In a prospective model, the system needs to make a decision once the target access e_i has been issued. At the moment e_i is issued, we only know the accesses transpiring beforehand, which corresponds to $\varepsilon_1 = \langle e_1, e_2, \ldots, e_{i-1} \rangle$. For e_i, we can use vectors as representations of all h types of context. Equation 7.12 denotes $V(U)$, the vector corresponding to context U.

$$V(U) = (v_{u_1}, v_{u_2}, \ldots, v_{u_d}) \tag{7.12}$$

In this model, v_{u_x} is set to 1 if u_x is observed when at least one $e_j \in \varepsilon_1$ transpires, otherwise it is set to 0.

For example, imagine we want to construct a vector corresponding to U (i.e., $V(U)$), for the target access e_3 in Figure 7.1. Let $dom(U) = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ in the system and $\langle u_2, u_4, u_5, u_1, u_3, u_8 \rangle$ be the user sequence corresponding to the workflow in Figure 7.1. $\varepsilon_1 = \langle e_1, e_2 \rangle$ is the access sequence occurring before e_3, where e_1 and e_2 are executed by u_2 and u_4 respectively. Thus, the vector corresponding to U for target user is $(0,1,0,1,0,0,0,0)$.

We use $\oplus$ to denote the union of two vectors[1]. As such, the vector for all h context can be represented as $CV = V(C_1) \oplus V(C_2) \oplus \ldots \oplus V(C_h)$.

7.3.3.2 Retrospective Model

A retrospective model is employed to review the target access using accesses occurring in the entire workflow. These accesses correspond to $\varepsilon_0 = \langle e_1, e_2, \ldots, e_{i-1}, e_{i+1}, \ldots, e_l \rangle$. In this case, during construction of $V(U)$, v_{u_x} is assigned 1, if user u_x exists when at least one $e_j \in \varepsilon_0$ transpires (i.e., e_j is executed by u_x). In Figure 7.1, the user context vector of the retrospective model is $(1,1,1,1,0,0,0,1)$. It is not necessary for the vector $V(C_r)$ in

[1]For example, vector $C = \langle a_1, a_2, \ldots, a_m, b_1, b_2, \ldots, b_n \rangle$ is the union of vector $A = \langle a_1, a_2, \ldots, a_m \rangle$ and vector $B = \langle b_1, b_2, \ldots, b_n \rangle$ (i.e., $C = A \oplus B$)

the prospective model and retrospective model to be different. For example, $V(C_r)$ will be identical for two models when C_r denotes the time the target access was issued.

7.4 Experiment Design

This section provides an overview of the experiments designed for this study. It begins with a description of the context extracted from real electronic medical record (EMR) data introduced in Chapter 3. This is followed by an explanation of how context was modeled to train the prospective and the retrospective security models. We then introduce the machine learning algorithm used for training the models and the specific measures used for assessing their performance. Note there is a major overlap with section 6.4 in this section, which would not be avoided for the ease of read.

7.4.1 Extract Context

In the EMR of Northwestern Memorial Hospital, each (*patient-id*, *encounter-id*) pair defines a unique workflow for patient treatment. This encounter begins when the patient is admitted to the hospital and ends two weeks after discharge (to ensure that accesses associated with medical billing are captured). Of the remaining information, there are five types of context: i) the time a target access was issued (Time)[2], ii) the hospital service the patient was on at the time of the target access (e.g., General Medicine vs. Obstetrics), iii) location in the medical center where the patient resided when the target access was issued, iv) the users who commit accesses in the workflow of target access and v) the job titles associated with these users.

[2]For this work, $dom(Time)$ consists of four values: a) Morning (6am - 12pm), b) Afternoon (12pm - 6pm), c) Evening (6pm - 12am), and d) Night (12am - 6am)

7.4.2 Dataset Preparation

Without loss of generality, assume target user t participates in N patient workflows. The corresponding context vectors are $CV_1^+, CV_2^+, \ldots, CV_N^+$, which are composed using the approach described in Section 7.3.3. These vectors are associated with a *positive* label class. We use the following process to generate a corresponding set of N *negative* labeled instances. We randomly select a workflow in which user t failed to issue an access. From this workflow, we randomly select an access and build a corresponding context vector. Doing so N times yields a set of vectors $CV_1^-, CV_2^-, \ldots, CV_N^-$, which are associated with the negative class. Note that we create different CV_i^+ and CV_i^- for prospective model and retrospective model respectively.

To conduct our evaluation, we construct 10 datasets, each of which corresponds to a different job title. Let us use *Patient Care Staff Nurse* as an example. We randomly pick 10 users whose job titles are *Patient Care Staff Nurse*. For each user, we construct N positive samples and N negative samples using the process described above. We select 80% of the vectors from the positive and negative samples, respectively, for the training set, and use the remaining 20% as the test set. The samples generated for all 10 users are then combined to form a single dataset for this job title and the overall performance across the 10 users is measured to evalute the entire dataset. To ensure the results are representative, we select job titles from 10 different hospital departments. The job titles and summary statistics are shown in Table 7.1.

We train a classifier for each user using a support vector machine (SVM) using an RBF kernel [50]. We utilize a grid search technique [50] to find values for parameters to enable a robust SVM. For each user in the job title, we use the classifier trained on the training set of this user to assess the corresponding test set.

7.5 Experiment by Traditional Methods

In this section, we compare prospective and retrospective security models using traditional evaluation strategies to set a baseline. We observe what kind of decision would be made by these traditional strategies, and figure out they may make unwise decision sometimes.

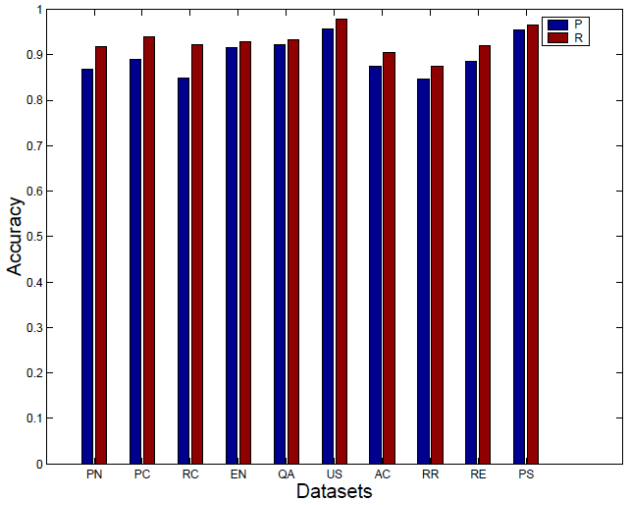


Figure 7.4: Accuracy of the prospective and retrospective security models for various NMH job titles

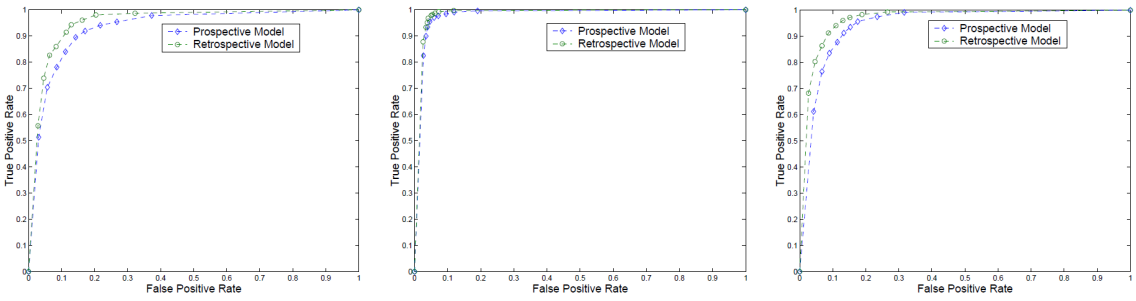
First, Figure 7.4 presents the accuracy of both the prospective model and the retrospective model on 10 datasets. It can be seen that the retrospective model has a higher accuracy than the prospective model for each job title. This evidence supports the hypothesis that contextual information obtained after a target access can lead to better classification performance. Simply put, an retrospective model can yield a more correct assessment of an access request. Moreover, from Table 4.3 it can be observed that AUC_R is larger than AUC_P for every job title, which further indicates that retrospective security models are better than prospective security models under a traditional assumption of costs.

Next, we inspected the ROC curves of the prospective and retrospective models. The curves for three of the job titles are depicted in Figure 7.5. From the ROC curves, we find that the retrospective model dominates the prospective model for the three datasets. This

Table 7.1: Datasets per job titles and the *AUC* for their corresponding prospective and retrospective models.

Abbrev.	Job Title	Instances Per Class	AUC_P	AUC_R
US	Unit Secretary	1839	0.984	0.994
QA	Utilization Review/Quality Assurance 1	1069	0.959	0.972
PS	Patient Care Assistive Staff	777	0.979	0.983
RE	Rehabilitation - Physical Therapist	712	0.944	0.964
RC	Resident/Fellow CPOE	504	0.925	0.967
AC	Anesthesia CPOE	456	0.932	0.953
PC	NMH Physician CPOE	448	0.953	0.979
PN	Patient Care Staff Nurse	382	0.939	0.959
EN	Emergency Department Patient Care Staff Nurse	366	0.961	0.976
RR	Radiology Resident/Fellow	364	0.919	0.944

indicates that, if the assumption of equal costs for false positive (negative) holds true, then the retrospective model will always be chosen regardless of the false positive (negative) cost estimation and prior positive (negative) probability. The cost curve is considered a dual representation of the ROC curve. This means using cost curve would reach the same conclusion (i.e., retrospective model wins) as the ROC curve for the job titles studied. As such, we do not present the cost curve in this section.



(a) ROC of *AC* (b) ROC of *PS* (c) ROC of *RE*

Figure 7.5: ROC curves for the prospective and retrospective models of three job titles.

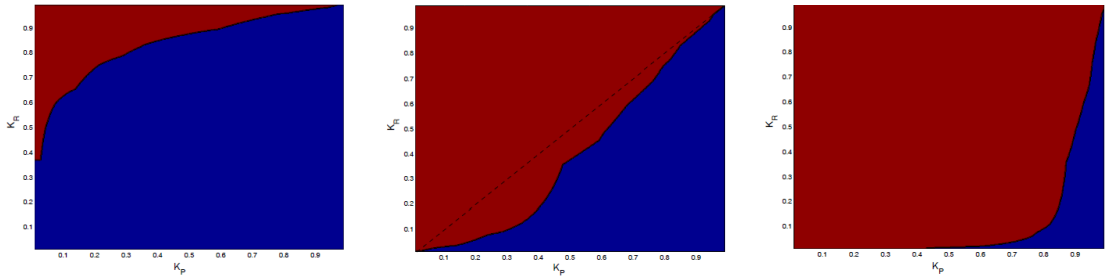
The assumption of equal costs for security-related classifiers is made in almost all previous research. And, if a security professional worked under this belief, then retrospective protections would almost be utilized over prospective models. However, as has been alluded to, this assumption certainly does not hold and, as the following results will illustrate, can unnecessarily justify costly behavior.

7.6 Experiment by Bispective Analysis

This section shows how our proposed technique affects the prospective versus retrospective decision model. First, we draw a series of contour plots for $Magnitude(K_P, K_R)$ or $Threshold(K_P, K_R)$ under a different $ratio = c_{01}^{(P)}/c_{01}^{(R)}$ for job title *Radiology Resident/Fellow*. We demonstrate how prospective and retrospective models can be compared from various pespective. Then, we present several case studies to show the application of our cost analysis technique in real environments, which demonstrates our technique can make a more reasonable decision than traditional methods.

7.6.1 Make Decision with Bispective Analysis

Figure 7.6 shows the contour plots of $Threshold(K_P, K_R)$ for the *Radiology Resident/Fellow* job title. With full knowledge about costs and the prior distribution of positive and negative instances, we can determine which security is best by pinpointing the corresponding coordinate in the plot. We will present case studies later to show this process in detail. With uncertainty in costs and prior distributions, bispective analysis can still be conducted through the contour plots from various perspectives, as we now illustrate.



(a) $raito = 0.3$ (b) $ratio = 1.0$ (c) $ratio = 3.0$

Figure 7.6: Contour plots for $Threshold(K_P, K_R)$ with different $ratio$ for the *Radiology Resident/Fellow*. The red and blue regions correspond to when the retrospective and prospective models dominate, respectively.

7.6.1.1 Probability Analysis

According to section 7.3.2.2, the area of the region in the contour plot determined by $Threshold(K_P, K_R) = 1$ equals the probability that $R \succ P$. Now, assume that we already know $ratio = 0.3$. Then, if we look at the contour plot corresponding to $ratio = 0.3$ in Figure 7.6(a), it is clear that $P(R \succ P) < 0.5$. This means that an administrator should choose a prospective model to manage the accesses from *Radiology Resident/Fellow* when only $ratio = 0.3$ is known.

7.6.1.2 Range Narrowing Analysis

In certain instances, with limited knowledge of costs and prior distributions, the search space can be narrowed into a small area. When this is possible, it can provide a clear solution to which model should be selected, even if such a decision was not possible in general. For instance, in an hospital system, the following assumptions about costs for misclassification in prospective and retrospective systems:

$$c_{01}^{(P)} \approx c_{01}^{(R)} \tag{7.13}$$

$$c_{10}^{(P)} > c_{10}^{(R)} \tag{7.14}$$

The first assumption (Equation 7.13) states that the cost of the prospective system allowing a malicious access and the cost of the retrospective system failing to identify a malicious access are approximately equal. The second assumption (Inequation 7.14) states that the cost of a prospective system blocking an access from *Radiology Resident/Fellow* would be greater than that of a retrospective system incorrectly identifying a normal and historical access from this job title as malicious. We will discuss how these assumptions are justified in our case studies. When such an assumption holds, we should look at Figure

7.6(b), which is a contour plot of $Threshold(K_P, K_R)$ given $ratio = c_{01}^{(P)}/c_{01}^{(R)} = 1.0$. Additionally, based on these assumptions, it follows that $K_P - K_R < 0$ because the numerator of K_P and K_R are equal according to $c_{01}^{(P)} \approx c_{01}^{(R)}$, and denominator of K_P would be larger than that of K_R according to $c_{01}^{(P)} \approx c_{01}^{(R)}$ and $c_{10}^{(P)} > c_{10}^{(R)}$. In Figure 7.6(b), it can be seen that the $K_P - K_R < 0$ is always located at the left of the diagonal (i.e., the black dashed line in the figure), a region where the retrospective security model is always dominant.

Note that when $c_{01}^{(P)} = c_{01}^{(R)}$ and $c_{10}^{(P)} = c_{10}^{(R)}$ (i.e., the premise that false positive (negative) costs are equal across two models holds), we have $K_P - K_R = 0$, which corresponds to the dashed line in Figure 7.6(b). That means our bispective analysis can still work under the permise as is believed in traditional ROC analysis.

7.6.2 Case Studies

In this section, we show three examples of bispective analysis in the domain of healthcare. We consider three job titles, Patient Care Assistive Staff and Anesthesia CPOE, and Rehabilitation - Physical Therapist, estimating $c_{01}^{(P)}$, $c_{01}^{(R)}$, $c_{10}^{(P)}$, and $c_{10}^{(R)}$ for each job title, and then apply bispective analysis to determine if a prospective or a retrospective models should be applied on this job title. We show that, for some jobs, choosing a prospective model will minimize cost, disagreeing with techniques that do not take cost into account. The estimations described are by no means exhaustive; rather they exist to demonstrate the utility of a cost-based decision support.

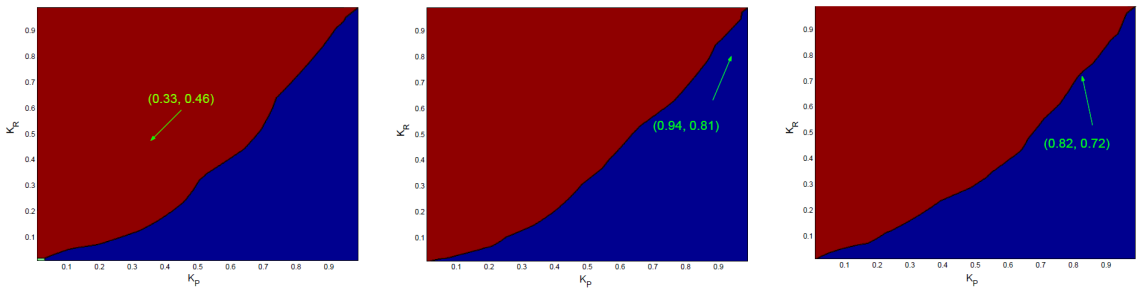
7.6.2.1 Cost Estimation

$c_{01}^{(P)}$ represents the costs of allowing an inappropriate access under a prospective model, while $c_{01}^{(R)}$ represents the costs of deciding not to review an illegitimate access under a retrospective model. These costs are generally the result of fines under HIPAA, HITECH, and other heathcare security statues. As the fines associated with inappropriate access are likely relatively independent of the security model that they were performed under, we as-

sume equality of $c_{01}^{(P)}$ and $c_{01}^{(R)}$. We also assume that fines due to inappropriate accesses are equivalent regardless of who makes them. For the sake of example, fines for inappropriate access over eight separate incidents in California hospitals ranged from $5,000 to $225,000, averaging $18,546 per inappropriate access [68, 69]. Though costs associated with inappropriate access will vary due to jurisdiction and individual details, we use this average as both $c_{01}^{(P)}$ and $c_{01}^{(R)}$ for the three job titles.

$c_{10}^{(P)}$ represents denying a legitimate access under a prospective model. This is likely the most difficult cost to estimate, as it alters behavior in a way that is not currently present in medical settings. For Patient Care Assistive Staff, which generally would be assisting another employee that has chart access permission, we can estimate $c_{10}^{(P)}$ as an hour of personnel time with no other costs. The national average wage for medical assistive staff is $11.73 [70]. For Anesthesia CPOE, in the best case, withholding physician access to a patient chart would cause the physician to wait, incurring a cost of only an hour of personnel time. The national average hourly compensation for anesthesiologists is $183 [71]. However, withholding access during a high-risk, high-urgency situation could result in a number of adverse outcomes, such as misdiagnosis or drug interactions, reducing quality of care and introducing the prospect of legal action. There is very little data on such a scenario. We estimate $c_{10}^{(P)}$ for Anesthesia CPOE to be $500, although it could range from our conservative estimate of $183 to something orders of magnitude higher depending on physician behavior. Physical therapists generally work in low-urgency situations, so adverse outcomes are significantly less likely. We estimate $c_{10}^{(P)}$ for them as $39.51, the national average wage [70].

$c_{10}^{(R)}$ represents the costs associated with auditing a legitimate access. We assume that this decision only incurs costs related to personnel time, specifically an hour of auditor time at $32.10 [70], again the national average for compliance officers, and an hour of time from the individual being audited. Thus $c_{10}^{(R)}$ for Patient Care Assistive Staff is approximately $43.83, while $c_{10}^{(R)}$ for Anesthesia CPOE is approximately $215, and $c_{10}^{(R)}$ for Rehabilitation



(a) Contour for AC when $ratio = 1$ (b) Contour for PS when $ratio = 1$ (c) Contour for RE when $ratio = 1$

Figure 7.7: Case Study Contour Plots

- Physical Therapist is $71.61.

7.6.2.2 Bispective Analysis on Three Job titles

The resulting values of K_P and K_R for Patient Care Assistive Staff (PS), Anesthesia CPOE (AC) and Rehabilitation-Physical Therapist (RE) are in Table 7.2, assuming 1% of accesses are inappropriate. Using the contour plots in Figure 7.7, we can make the following observations. For AC, a retrospective model minimizes cost. For PS, a prospective model minimizes cost. For RE, bispective analysis shows the prospective model minimizes cost (or at least no preference between the two). Remember if we use traditional methods, retrospective models would be chosen for all three job titles.

Table 7.2: Cost Estimation

	$c_{01}^{(P)}$	$c_{01}^{(R)}$	$c_{10}^{(P)}$	$c_{10}^{(R)}$	K_P	K_R
PS	$18,546	$18,546	$11.73	$43.84	0.94	0.81
AC	$18,546	$18,546	$183.00	$215.10	0.33	0.46
RE	$18,546	$18,546	$39.51	$71.61	0.82	0.72

Chapter 8

Conclusion

8.1 Summary of Research

This dissertation introduced a set of data-driven techniques to faciliate insider threat mitigation, which are summarized in this section.

First, we introduced a role prediction and a role revision method with EMR data. This study illustrated that usage patterns of a commercial EMR system can enable accurate prediction of certain roles in a healthcare system. Additionally, we illustrated that an automated approach can be leveraged to integrate role hierarchies with information learned from EMR access logs to improve role management. These findings are notable because they suggest that RBAC, in combination with some EMR usage mining, may assist in minimizing the management of access to an EMR system. Moreover, the increased specificity provided by User Positions versus higher levels within the role hierarchy enables more detailed access pattern analysis. These results are further notable because a recent report from the Presidents Council of Advisors on Science and Technology (PCAST) recommended that emerging health information architectures should leverage security principles that have proven successful in a range of industries beyond healthcare [72]. In particular, the PCAST report alludes to RBAC as a foundation upon which such policies can be defined. With respect to the healthcare domain, RBAC is intended to be a scalable framework for commissioning (and decommissioning) users with access rights to functions (e.g., order issuance) or elements of a clinical information system (e.g., a specific patients record). And, notably, various commercial EMR systems have integrated such security frameworks into their design. Yet, as the PCAST report acknowledges, healthcare organizations rarely execute RBAC on the scale found in other domains.

Second, this dissertation proposed a novel role engineering algorithm that enables a controlled evolution of RBAC based on the utilization of permissions (as documented in access logs). We devised an objective function that balances an administrator's beliefs and actual permission utilization, and defined a role mining problem for finding an RBAC configuration that optimizes this objective. To solve this problem, we proposed a two-phase heuristic algorithm. We then performed an empirical analysis with real and simulated datasets to show that our algorithm can generate appropriate RBAC configurations for various biases of the two competing goals of the objective function.

Third, this dissertation introduces a workflow based audit (WOBA) framework to audit accesses. The empirical results show WOBA could yield a satisfactory accuracy on the dataset, and workflow features play a vital role in the performance. In addition, adding sequential features improved the performance considerably.

Fourth, we propose a novel framework that enables organizations to perform comparison between prospective and retrospective models on a quantitative scale. Developing such a framework addresses two challenges. First, existing prospective and retrospective models are semantically different such that their results are not directly comparable. Second, the assumption that costs of false positive (and false negative) are equivalent across the classifiers needs to hold for existing technique to conduct cost analysis of multiple classifiers. To address the first challenge, we converted the two security models (i.e., prospective and retrospective) into a unified classification models by training the same classifiers on the data represented by the same set of features (contexts). To address the second challenge, we devise a visualized analysis method, named bispective analysis, that leverage contour plot of a comparison function to provide a direct decision support for administrator. We then experimented on a real hospital information system with this framework to show that it can provide good decision support quality. Somewhat surprisingly, we also found it can provide decision support even when knowledge about costs are insufficient.

8.2 Limitations

This section discusses the limitations for the techniques proposed in this dissertation.

For the first technique, there are two drawbacks. The first drawback of this study to note is that the original roles (i.e., User Positions), were defined over time and not in a single security engineering design. As a consequence, in certain cases, User Position designations represent vestigial remnants of a prior CPOE roll-out strategy. That is, for a time, selected physician user roles were not entering orders online, although now all physician User Positions include this functionality. Additionally, User Position assignments fail to take into account some workflow idiosyncrasies. For example, hospital medicine physicians, or hospitalists, often serve as a pilot physician group requiring their User Position to be distinct from other internal medicine physicians. Hospitalists may also work as a non-hospitalist (e.g., as a teaching attending), however, and, at those times, their chart access patterns would differ from their hospitalist service rotations. The second drawback of this study is a function of the Role-Up algorithm. Currently, the roll-up procedure is guided by a greedy heuristic. Specifically, in each iteration, the algorithm generalizes the set of sibling roles (i.e., roles with a common parent) that provide the greatest gain in predictive accuracy without sacrificing much role specificity. However, this process does not guarantee the discovery of a system that maximizes the number of roles and system accuracy.

The second limitation of this dissertation is with respect to DDRE algorithm. First, our strategy is based on permission utilization patterns in an atemporal fashion. This is a simplification of the access logs and neglects that the order in which permissions are invoked may be correlated. Second, our approach is predicated on the hypothesis that there is only one pattern (in the form of a distribution of permission rates) associated with the underlying roles. Yet, it is possible there could be multiple patterns.

The third limitation of this dissertation is about time complexity of WOBA. Although the introduction of sequential feature improves the auditing performance, the extraction of sequential features would take prohibitively long time with the increment of sequence

length. This would be a obstacle to take advantage of all possible workflow information (e.g. roles and users).

Finally, there are two limitations for bispective analysis that should be acknowledged. First, its decision support method relies heavily on the contour plot of comparison function of two models. That means we may need C_n^2=$n(n-1)/2$ contour plots when there are options of n models. When n is a large number, we would need to study too many contour plots to make a decision, which would offset the visual convenience of contour plot. Another limitation is that the cost function used in this paper assumes correct classification does not incur cost, which however is not the case in reality. For example, let us consider retrospective model in hospital system. Assume a user issued a malicious access to a patient's record in the system, and was identified later by retrospective system. Even though the user would be penalized, it is possible the patient's information has already been leaked to the public, which would lead to costly consequence.

8.3 Future Research

This section provides intuition into how to extend the research reported in this dissertation.

First, the data studied in this dissertation is simplified with respect to the settings in which it was captured. For instance, it is assumed that each user appears in a workflow only once. This is obviously not the case in real world. Additionally, the time span of the data is only three months, such that the patterns (and anomalies) detected may not be completely indicative of the functions of an organization over time. Resolving these two issues would be helpful in improving the usability of techniques in this dissertation in practice.

Second, it is important to recognize that there could be a gap between the objective function defined in this dissertation and real world security requirements. It would be worthwhile to investigate the extent those objective functions reflect the real requirements.

Third, this dissertation assumes that only one security strategy (prospective strategy vs. retrospective strategy) would be adopted in real world. However, it is possible that an organization would have an interest in combining these approaches in practice. Investigating how to combine these approaches into a sequential decision making setting would be worthwhile.

8.4 Conclusion

The insider threat has become one of the greatest threat to information security. Two questions are raised when there is a need to design an insider threat mitigation system: 1) What strategy (i.e., prospective vs. retrospective) should be adopted for the target organization? 2) Once one strategy is chosen, what specic method should be used to implement it? To answer these two questions, this dissertation proposed a set of novel data-driven techniques, which are validated in a real dataset. We believe techniques in this dissertation will empower security expert to design more secure and less costly insider threat mitigation system.

BIBLIOGRAPHY

[1] CSO Magazine, U.S. Secret Service et al. 2013 US State of Cybercrime Survey, 2013.

[2] Forrester Research Inc. Understand The State Of Data Security And Privacy: 2013 To 2014, 2013.

[3] C. Ornstein. http://www.propublica.org/article/ucla-health-system-pays-865000-to-settle-celebrity-privacy-allegations.

[4] U.S. Department of Health and Human Services. http://www.hhs.gov/ocr/privacy/hipaa/administrative/breachnotificationrule/breachtool.html.

[5] R. S. Sandhu. Lattice-based access control. *IEEE Computer*, 26:9–19, 1993.

[6] R. S. Sandhu and P. Samarati. Access control: Principles and practice. *IEEE Communication Magazine*, 32:40–48, 1994.

[7] D. F. Ferraiolo, R. S. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed nist standard for role-based access control. *ACM Transactions on Information and System Security*, 4:224–274, 2001.

[8] R. S. Sandhu. Role-based access control. *Advances in Computers*, 46:237–286, 1998.

[9] D. F. Ferraiolo, J. F. Barkley, and D. R. Kuhn. A role-based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information and System Security*, 2:34–64, 1999.

[10] X. Jin, R. Krishnan, and R. Sandhu. A unified attribute-based access control model covering dac, mac and rbac. In *Proceedings of the 26th Annual IFIP WG 11.3 Conference on Data and Applications Security and Privacy*, pages 41–55, 2012.

[11] M. Peleg, D. Beimel, D. Dori, and Y. Denekamp. Situation-based access control: Privacy management via modeling of patient data access scenarios. *Journal of Biomedical Informatics*, 41:1028–1040, 2008.

[12] A. C. O'Connor and R. J. Loomis. 2010 economic analysis of role-based access control. Technical report, 2010.

[13] Y. Chen, S. Nyemba, and B. Malin. Detecting anomalous insiders in collaborative information systems. *IEEE Transactions on Dependable and Secure Computing*, 9:332–344, 2012.

[14] D. Fabbri and K. LeFevre. Explanation-based auditing. In *Proceedings of the VLDB Endowment*, pages 1–12, 2011.

[15] M. Dekker and S. Etalle. Audit-based access control for electronic health records. In *Proceedings of the Second International Workshop on Views on Designing Complex Architectures*, pages 221–236, 2006.

[16] A. K. Menon, X. Jiang, J. Kim, J. Vaidya, and L. Ohno-Machado. Detecting inappropriate access to electronic health records using collaborative filtering. *Machine Learning*, 95, 2014.

[17] D. Povey. Optimistic security: A new access control paradigm. In *Proceedings of the Workshop on New Security Paradigms*, pages 40–45, 1999.

[18] K. Dempsey, G. Witte, and D. Rike. Security and privacy controls for federal information system and organizations. Technical Report Special Publication 800-53, Revision 4, Washington, DC, 2014.

[19] P. C. Cheng, P. Rohatgi, C. Keser, P.A. Karger, G.M. Wagner, and A.S. Reninger. Fuzzy multi-level security: An experiment on quantified risk-adaptive access control.

In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 222–230, 2007.

[20] I. Molloy, P. C. Cheng, J.L.L. Dicken, A. Russo, and C. Morisset. Risk-based access control decisions under uncertainty. In *Proceedings of the* 2^{nd} *ACM conference on Data and Application Security and Privacy*, pages 157–168, 2012.

[21] L. Zhang, A. Brodsky, and S. Jajodia. Toward information sharing: Benefit and risk access control. In *Proceedings of the* 7^{th} *IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 45–53, 2006.

[22] R. Lippmann, D. Fried, I. Grad, et al. Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, pages 12–26, 2000.

[23] J. Z. Kolter. and M. A. Maloof. Learning to detect malicious executables in the wild. In *Proceedings of the* 10^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 470–478, 2004.

[24] A. A. Boxwala, J. Kim, J. M. Grillo, and L. Ohno-Machado. Using statistical and machine learning to help institutions detect suspicious access to electronic health records. *Journal of the American Medical Informatics Association*, 18(4):498–505, 2011.

[25] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 305–316, 2010.

[26] D.J. Hand. Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine Learning*, 77:103–123, 2009.

[27] F. Provost and T. Fawcett. Analysis and visualization of classifier performance: Com-

parison under imprecise class and cost distributions. In *Proceedings of the* 3^{rd} *international conference on Knowledge Discovery and Data Mining*, pages 43–48, 1997.

[28] C. Drummond and R.C. Holte. Explicitly representing expected cost: an alternative to roc representation. In *Proceedings of the* 6^{th} *international conference on Knowledge Discovery and Data Mining*, pages 198–207, 2000.

[29] E. J. Coyne. Role engineering. In *Proceedings of the* 1^{st} *ACM Workshop on Role-Based Access Control*, 1995.

[30] A. Schaad, J. Moffett, and J. Jacob. The role-based access control system of a european bank: a case study and discussion. In *Proceedings of the* 6^{th} *ACM Symposium on Access Control Models and Technologies*, pages 3–9, 2001.

[31] D. Shin, G.J. Ahn, S. Cho, and S. Jin. On modeling system-centric information for role engineering. In *Proceedings of the* 8^{th} *ACM Symposium on Access Control Models and Technologies*, pages 169–178, 2003.

[32] J. Schlegelmilch and U. Steffens. Role mining with ORCA. In *Proceedings of the* 10^{th} *ACM Symposium on Access Control Models and Technologies*, pages 168–176, 2005.

[33] J. Vaidya, V. Atluri, and Q. Guo. The role mining problem: A formal perspective. *ACM Transactions on Information and System Security*, 13:27, 2010.

[34] J. Vaidya and V. Atluri. Roleminer: mining roles using subset enumeration. In *Proceedings of the* 13^{th} *ACM Conference on CCS*, pages 144–153, 2006.

[35] D. Zhang and T. Ebringer K. Ramamohanarao. Role engineering using graph optimisation. In *Proceedings of the* 10^{th} *ACM Symposium on Access Control Models and Technologies*, pages 139–144, 2007.

[36] H. Roeckle, G. Schimpf, and R. Weidinger. Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization. In *Proceedings of the* 5^{th} *ACM Workshop on Role-Based Access Control*, pages 103–110, 2000.

[37] A. Colantonio, R. D. Pietro, A. Ocello, and N. V. Verde. A new role mining framework to elicit business roles and to mitigate enterprise risk. *Decision Support Systems*, 50:715–731, 2011.

[38] J. Vaidya, V. Atluri, Q. Guo, and N. Adam. Migrating to optimal RBAC with minimal perturbation. In *Proceedings of the* 13^{th} *ACM symposium on Access Control Models and Technologies*, pages 11–20, 2008.

[39] I. Molloy, Y. Park, and S. Chari. Generative models for access control policies: applications to role mining over logs with attribution. In *Proceedings of the* 17^{th} *ACM SACMAT*, pages 45–56, 2012.

[40] Y. Chen, S. Nyemba, W. Zhang, and B. Malin. Leveraging social networks to detect anomalous insider actions in collaborative environments. In *Proceedings of IEEE Intelligence and Security Informatics*, pages 119–124, 2011.

[41] H. Zhang, S. Mehotra, D. Liebovitz, C. A. Gunter, and B. Malin. Mining deviations from patient care pathways via electronic medical record system audits. *ACM Transactions on Management Information Systems*, 4, 2013.

[42] D. Fabbri and K. LeFevre. Explaining accesses to electronic medical records using diagnosis information. *Journal of the American Medical Informatics Association*, 20:52–60, 2013.

[43] M. Strembeck. Scenario-driven role engineering. *IEEE Security and Privacy Magazine*, 8:28–35, 2010.

[44] L. Freeman, A. Romney, and S. Freeman. Cognitive structure and informant accuracy. *American Anthropologist*, 89:310–325, 1987.

[45] S. Mehrotra, C. Butts, D. V. Kalashnikov, N. Venkatasubramanian, R. Rao, and et al. Project RESCUE: challenges in responding to the unexpected. In *Proceedings of SPIE*, volume 5304, pages 179–192, 2004.

[46] T. H. Haveliwala., A. Gionis, Dan D. Klein, and P. Indyk. Evaluating strategies for similarity search on the web. In *Proceedings of the* 11^{th} *International Conference on the World Wide Web*, pages 432–442, 2002.

[47] B. Scholkopf, J.C. Platt, J.C. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[48] L. M. Manevitz and M. Yousef. One-class SVMs for document classification. *The Journal of Machine Learning Research*, 2:139–154, 2002.

[49] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In *Proceedings of Applications of Data Mining in Computer Security*, pages 78–100. Kluwer, 2002.

[50] C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. Technical report, Dept. of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2003.

[51] M. R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, NY, 1990.

[52] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4:233–235, 1979.

[53] C. Chang and C. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27, 2011.

[54] C. Gunter, D. Liebovitz, and B. Malin. Experience-based access management: A life-cycle framework to evolve identity and access management systems. *IEEE Security and Privacy Magazine*, 9:48–55, 2011.

[55] P. V. Asaro and J. E. Ries. Data mining in medical record access logs. In *Proceedings of AMIA Symposium*, page 855, 2001.

[56] R. J. Gallagher, S. Sengupta, G. Hripcsak, R.C. Barrows, and P.D. Clayton. An audit server for monitoring usage of clinical information systems. In *Proceedings of AMIA Symposium*, page 1002, 1998.

[57] Z. Zhou and B.J. Liu. Hipaa compliant auditing system for medical images. *Computerized Medical Imaging and Graphics*, 29:235–241, 2005.

[58] A. Ferreira, R.C. Correia, L. Antunes, P. Farinha, E.O. Palhares, D.W. Chadwick, and A.C. Pereira. How to break access control in a controlled manner. In *Proceedings of the 19th IEEE Symposium on Computer-Based Medical System*, pages 847–854, 2006.

[59] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2011.

[60] W. Zhang, Y. Chen, T. Cybulski, D. Fabbri, C. Gunter, P. Lawlor, D. Liebovitz, and B. Malin. Decide now or decide later?: Quantifying the tradeoff between prospective and retrospective access decisions. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communication Security*, pages 1182–1192, 2014.

[61] A. Gomariz, M. Campos, R. Marin, and B. Goethals. Clasp: An efficient algorithm

for mining frequent closed sequences. *Advances in Knowledge Discovery and Data Mining*, 7818:50–61, 2013.

[62] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C. Wu, and V. S. Tseng. Spmf: a java open-source pattern mining library. *Journal of Machine Learning Research*, 15:3389–3393, 2014.

[63] D. Povey. Optimistic security: a new access control paradigm. In *Proceedings of the Workshop on New Security Paradigms*, pages 40–45, 1999.

[64] D. Weitzner. Information accountability. *Communications of the ACM*, 37(6):82–87, 2008.

[65] R. Snelick, U. Uludag, A. Mink, M. Indovina, and A. Jain. Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(3):450–455, 2005.

[66] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159, 1997.

[67] J. H. Orallo, P. Flach, and C. Ferri. A unified view of performance metrics: Translating threshold choices into expected classification loss. *Journal of Machine Learning Research*, 13:2813–2869, 2012.

[68] California Department of Public Health. California Department of Public Health Issues Privacy Breach Fines to 7 California Health Facilities, 2010.

[69] K. Robertson. Kaiser fined for employees checking medical records of octuplets - Sacramento Business Journal, 2009.

[70] Bureau of Labor Statistics and U.S. Department of Labor. Occupational outlook handbook, 2014-15 edition, 2014.

[71] B. Herman. 72 Statistics on Hourly Physician Compensation, 2013.

[72] Office of Science and Technology Policy. Presidents Council of Advisors on Science and Technology Report to the President: realizing the full potential of health information technology to improve healthcare for Americans: the path forward, 2010.