DATA-DRIVEN MODEL-BASED DETECTION OF MALICIOUS INSIDERS VIA PHYSICAL ACCESS LOGS

A PREPRINT

Carmen Cheh
Department of Computer Science, University of Illinois
61801 Urbana, Illinois
cheh2@illinois.edu

Uttam Thakore
Department of Computer Science, University of Illinois
61801 Urbana, Illinois
thakore1@illinois.edu

Ahmed Fawaz
Information Trust Institute, University of Illinois
61801 Urbana, Illinois
afawaz2@illinois.edu

Binbin Chen
Advanced Digital Sciences Center, Singapore
1 Create Way, 138602 Singapore
binbin.chen@adsc-create.edu.sg

William G. Temple
Advanced Digital Sciences Center, Singapore
1 Create Way, 138602 Singapore
william.t@adsc-create.edu.sg

William H. Sanders
Department of Electrical and Computer Engineering, University of Illinois
61801 Urbana, Illinois
whs@illinois.edu

February 5, 2019

ABSTRACT

The risk posed by insider threats has usually been approached by analyzing the behavior of users solely in the cyber domain. In this paper, we show the viability of using physical movement logs, collected via a building access control system, together with an understanding of the layout of the building housing the system's assets, to detect malicious insider behavior that manifests itself in the physical domain. In particular, we propose a systematic framework that uses contextual knowledge about the system and its users, learned from historical data gathered from a building access control system, to select suitable models for representing movement behavior. We suggest two different models of movement behavior in this paper and evaluate their ability to represent normal user movement. We then explore the online usage of the learned models, together with knowledge about the layout of the building being monitored, to detect malicious insider behavior. Finally, we show the effectiveness of the developed framework using real-life data traces of user movement in railway transit stations.

Keywords Physical Access · Physical Movement · Cyber-physical Systems · Insider Threat · Intrusion Detection · User Behavior

This material is based upon work supported by the Maryland Procurement Office under Contract No. H98230-18-D-0007. This work was supported in part by the National Research Foundation (NRF), Prime Minister's Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-31) and administered by the National Cybersecurity R&D Directorate, and supported in part by the research grant for the Human-Centered Cyber-physical Systems Programme at the Advanced Digital Sciences Center from Singapore's Agency for Science, Technology and Research (A*STAR). Part of this work was done when Carmen Cheh was a research intern at ADSC. We want to thank Prof Carl Gunter and Jenny Applequist for their feedback on the paper. We also thank the experts from SMRT Trains LTD for providing us with data and domain knowledge.

1 Introduction

Insider threats are a top concern of all organizations because they are common and can have severe consequences. However, insider threats are difficult to detect, since the adversary already has physical and cyber access to the organization's assets. Many state-of-the-art research efforts [1] and state-of-the-practice tools [2, 3] focus on the cyber aspect of insider attacks by analyzing the user's cyber footprint (e.g., logins and file accesses). However, the strength of an organization's defense mechanisms is only as strong as its weakest link. By failing to consider the physical aspect of users' behavior, an organization not only leaves itself unable to detect precursor physical behavior that could facilitate future cyber attacks, but also opens itself up to less tech-savvy attacks, such as vandalism and theft [4].

Thus, physical security plays a crucial role in an organization's overall defense posture. This is especially true for critical infrastructure systems such as power grids and transportation systems in which a physical breach can have major real-world effects. Building access controls [5] are often used to limit the areas that users can access based on their role in the organization; they normally do so through a relatively static assignment of a set of locations to the user's tracking device (e.g., RFID tag or access card). When a user moves between spaces (e.g., swiping a card at a door), information about this movement is logged.

Although building access control restricts the spaces that a user is able to access, it is merely the first step towards physical security. Like other access control solutions, it faces the problem of being overly permissive [6]. However, denying access to rarely accessed rooms is a costly solution, as it places the burden on administrators to grant every access request, which can lead to severe consequences, especially in time-critical situations (e.g., maintenance). Even with a restrictive set of granted permissions, the access control solutions in place do not take into account the context of a user's access.

Thus, we focus on developing a more advanced behavior-monitoring capability that detects abnormalities in a user's movement within an organization's buildings. In Section 3.5, we examine the effect of the addition of such defenses on the overall security posture of an organization. The results of the analysis in that section indicate that this type of defense capability will increase the strength of the organization's security. As such, it is an important defense mechanism that deserves more attention from researchers and security practitioners.

In this paper, we explore how physical access logs collected from a railway transit system can be used to develop a more advanced behavior-monitoring capability for the purposes of detecting abnormalities in a user's movement. In particular, we aim 1) to determine the feasibility of characterizing the movement behavior of users in a complex real-world system, 2) to develop techniques that can be applied to this detection problem, and 3) to identify ways to integrate real-time detection into physical security.

We provide a systematic approach to tackling these issues in a way that can be generalized to a diverse set of systems. We observe that since an organization consists of users who have a diverse set of roles, the movement patterns of users in different roles may vary vastly because of their job needs. Instead of proposing a single technique to model all users, we construct a methodical approach that selects the appropriate model based on the context of the organizational role and learns that model from historical data. More specifically, we propose metrics to determine the feasibility of modeling the behavior of certain users in a system. We then construct models that factor in contextual information about location, and show that the model can be used in an online manner. We propose two different models and evaluate their ability to represent normal user movement behavior by comparing the detection rates of the framework when it is using those models. This study is supported by a set of real-life physical access traces that we collected from our industrial collaborator.

In summary, our contributions in this paper are as follows:

- We define a framework that characterizes a user's physical movement behavior and learns models of the user's behavior by using historical data. More specifically, we propose a metric based on the entropy of time series that quantifies the regularity of user movement in order to differentiate user movement behaviors. By choosing different models for users with irregular behavior instead of regular behavior, we show that integrating logs of relevant device state is needed to improve detection performance for those users.
- We propose two different Markov models to represent the physical movement of a user. These Markov models differ with respect to the amount of information represented about the physical paths taken by users.

- We evaluate our framework using fourteen months of real-world physical access data obtained from railway transit stations. We show that we can detect suspicious movement behavior in an online manner before the final destination is reached.

The structure of the paper is as follows. In Section 2, we discuss related work in the domain of anomaly detection for physical movement. Section 3 introduces our case study of railway transit systems, and Section 4 describes our framework for detecting malicious insiders, applying it to the case study as an example. The evaluation results are presented in Section 5. Finally, plans for future work are summarized in Section 6, and the conclusion is given in Section 7.

2 Related Work

In this section, we discuss the related work spanning domains from physical movement tracking and prediction to anomaly detection of physical movement and cyber events.

There has been a substantial amount of work on use of cyber logs (e.g., network flows and system logs) to profile users and detect events of interest. For example, Kent et al. proposed *authentication graphs* [7] to profile user behavior and detect threats using computer authentication logs in an enterprise network. They observe distinguishable graph attributes for different user roles and propose to use such differences to help improve user behavior profiling and misuse detection. In contrast, our work focuses on physical access logs, for which physical-world factors, like space and time, directly impact the correlations among different access events. Despite those differences, we also observe the importance of distinguishing among different user roles.

In the area of physical access control, there has been work in the route anomaly detection area that looked at people or objects moving in a geographical space that was not delineated by rooms [8, 9, 10]. Pallotta and Jousselme [8] and Radon et al. [9] both detect deviations in the trajectory of a vessel in the maritime domain. Their approaches use contextual information, such as the speed of the vessel and weather information, in order to predict the next location of a vessel. However, in the maritime domain, the source and destination of the vessel are already known beforehand, and the anomalies are assumed to arise from the differences in trajectories. This is unlike our work, in which we focus on an indoor setting that has unpredictable destinations for each user.

Dash et al. [10] use mobile data to predict the movement of people in a geographical region. They construct multiple Dynamic Bayesian network models, each of which includes different granularities of context (e.g., day of the week vs. time of day). They predict the next visited location by analyzing the results obtained from each of those models. In contrast to their completely data-driven approach of applying all models before computing the best result, we propose a more guided approach in which we start by choosing an appropriate model based on an understanding of a person's past movement data.

Lin et al. [11] also use mobile data for user verification by constructing *Hidden Markov Models* (HMMs) that represent each user's mobility patterns. Those HMMs are constructed from cell tower data and ambience features of WiFi. Using the sequential probability ratio test, the authors decide whether the current sequence of locations adheres well to the user's HMM model and not to other users' HMM models. Although their approach allows for real-time authentication of the user, the result of the analysis at each time point may be inconclusive if the evidence of the user's movement does not strongly conform to the user's HMM model in comparison to other users' HMM models. Our approach, however, produces a decisive yes-or-no answer for each user movement and does not compare the movement to other users' models. In other words, we take a more conservative approach and treat any deviations from the user's movement model as suspicious behavior. This allows the security practitioner to identify malicious behavior more quickly when it appears.

In contrast to the work described above, we consider the more restrictive setting of indoor location tracking, which reduces the amount of noise in the data and allows us to identify a user's location with more confidence. Because of physical barriers that prevent a user from moving uninhibited from one space to another, the paths that a user can take are also limited.

For indoor physical access, there has been work in both movement prediction and anomaly detection. In the movement prediction domain, Gellert and Vintan [12] use HMMs to predict a user's next location. They use real-world physical access data of four users from a single floor of an office building; notably, the size and topology of the building are very small. Their results show that a simple Markov model of order 1 gives the best performance. Koehler et al. [13] expand on Gellert's work by using ensemble classifiers to predict how long a user will stay at a given location.

In the anomaly detection domain, different techniques to detect differences in a user's movement have been proposed. Graph models have been studied by Eberle and Holder [14] and Davis et al. [15]. Eberle and Holder [14] detect structural anomalies by extracting common subgraph movement patterns [16]. However, they only consider simplified physical layouts and do not distinguish among different user roles. Davis et al. [15] search labeled graphs for both structural and numeric anomalies and apply their approach to physical access logs in an office building.

Other models, ranging from finite state machines to specific rules, have also been studied. Liu et al. [17] model the normal movements of devices as transitions in finite state machines. Unlike us, they focus on the movement of devices (instead of people) in a hospital setting, where their main goal is to detect missing-device events. Biuk-Aghai et al. [18] focus on suspicious behavioral patterns, including temporal, repetitive, displacement, and out-of-sequence patterns. These patterns only involve the time interval between movements and the reachability of locations, rather than the order in which locations were visited.

Finally, patents from IBM [19] and Honeywell [20] present the general designs of ways to use physical access data to detect potential security incidents. However, they do not discuss detailed designs for dealing with complicated building topology and user roles, and do not provide experimental studies involving real-world traces.

In this paper, we build upon our previous work in [21]. In that work, we build Markov models of user movement behavior such that the states of the Markov model represent rooms in a building. While the results obtained in that work were promising, it ignores the paths that a user can take to move from one room to another. In this paper, we improve that Markov model by augmenting the state in the Markov chain to include the common areas that are visited by the user on the path to the rooms in the building. By refining the Markov model, we characterize normal movement behavior not only by the sequence of rooms visited, but also by the sequence of common areas visited on the path to those rooms. In subsequent sections, we formally define this improved Markov model and evaluate its ability to represent normal user movement behavior, in comparison to the Markov model in [21].

3 Motivating Use Case

Physical security is of high priority for industrial control facilities and critical infrastructures. Through a project partnership, we have gained deep knowledge about the physical access control challenges faced by railway transit system operators. We will use this real-world use case to motivate our study.

3.1 Background

The railway transit system is an important component of a nation's transportation system. The impact of an attack or fault in the system can be very severe, ranging from loss of service and station blackouts to derailment. For example, a Polish teenager once rewired a remote control to communicate with wireless switch junctions, causing derailment of a train and injury of twelve people [22]. Since the track was accessible to the public, the attack was easily performed. However, in our case study, the railway system is underground and therefore presents a stronger barrier against such an attack. Potential loss of revenue and human life motivates the need for both physical and cyber security of such systems. In particular, the insider threat is of the utmost importance, as can be seen in a 2006 case in which two traffic engineers hacked into a Los Angeles signal system, causing major traffic disruption [23].

3.2 System Architecture

A railway station consists of a single building that may house one or multiple railway lines through it. The general public accesses the railway lines by passing through fare gates in the concourse area and moving to the platform. Figure 1 depicts the topology of the railway station in our case study. In addition to the concourse and platform area, the railway station contains many rooms hidden from the public eye that house the equipment necessary to maintain the running of the station and its portion of the railway track. Each room serves a specific function, and there are multiple rooms that share the same function. The rooms are distributed throughout the station on multiple levels. The railway staff can access those spaces only by swiping their access cards at readers on the doors. Although most of the doors inside the staff-only spaces have card readers, there are a number of doors that allow free access. Different stations have different floor plans, and the number of rooms within a station may vary. However, all the stations share the same types of rooms (e.g., power supply room).

3.3 Threat Model

Our threat model focuses on users who have gained physical access to the rooms in a railway station. Such a user may be a malicious member of the railway staff or an outsider who has acquired an employee's access control device. The adversary will then use the acquired access control device to move to rooms within the railway station. We assume that this malicious movement will deviate from the normal movement behavior associated with the acquired access control

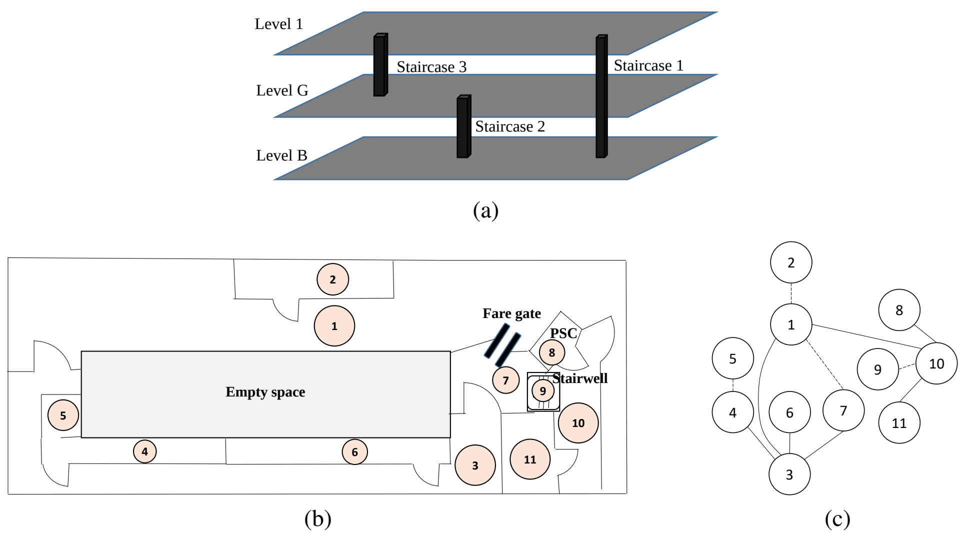

Figure 1: Building topology of a railway station. (a) The different levels of the station. (b) A small sample floor plan of Level G. (c) Graph representation of (b). Each edge in the graph represents a pair of directed edges between two vertices. Non-dotted edges imply that a card reader exists on the door separating the spaces (vertices).

device. (E.g., a staff member who always visits the Passenger Service Center after visiting the staff room suddenly visits the power supply room instead. This would be considered suspicious.) The adversary would then be able to tamper with devices within that room. In a railway station, almost all the rooms house critical assets. Thus, we cannot narrow our focus to any specific portion of the railway station to reduce the space of possible movement trajectories. The level of risk involved in letting an adversary achieve his or her goal is too high. However, restricting a user's access to rooms in a station can also result in severe consequences. Since railway staff require access to rooms in order to conduct maintenance and repairs on devices within those rooms, denying them access could cause disruption of service.

3.4 Opportunities and Challenges

Unlike an enterprise system for which the office building has a simple, systematic layout across all levels (e.g., a single corridor branching out to multiple rooms), a railway station has a complex, asymmetrical layout. There are multiple paths with varying lengths that a user can take to get from one room to another. This implies that topology is an important factor in determining whether a user's physical movement is anomalous. In addition, the railway transit system involves diverse user roles (e.g., station operators and power maintenance staff). The job scopes of such users vary in terms of work shifts, responsibilities, and work locations, all of which affect their physical movement behavior. Even users in the same role exhibit different movement behaviors based on their assigned duties and personal habits. Some users also have very sporadic movement behavior. Therefore, it is challenging to detect both spatial and temporal deviations in a user's movement behavior. In the next sections, we present our approach, which learns spatial models of users' behaviors using knowledge of system layout and historical physical access data. We evaluate the detection rate of our approach and make conclusions about the types of movement behaviors that can and cannot be captured well by our approach.

Our approach is limited by the abilities of the building access control system. In particular, the building access control system offers a limited view of users' physical movements. Since card readers may fail and certain doors are not outfitted with card readers, we are unable to determine a user's full movement trajectory. A user may also tailgate, i.e., follow closely behind, another user, and thus the access will remain invisible to us and undetectable by our approach.

3.5 Envisioned Monitoring

We envision that when our detection system is put into use by a company, past movement data logs will already have been collected for the present users in that company. Those logs will be used as training data by our algorithm in building the user movement models. The user movement models will be built during the installation of the intrusion detection system by feeding in the existing data logs. Then, our intrusion detection system can be put into use immediately and will be able to process door accesses in real time.

Currently, a railway system staff member would need to look through the physical access logs manually in order to detect malicious behavior. We aim to reduce the amount of manual effort with our detection system, which automatically presents to the staff member a smaller subset of potentially malicious physical accesses in real time. The staff member can then focus his or her attention on the smaller subset, using video surveillance to corroborate evidence of malicious activity. To aid in decision-making, we can also supplement the information on suspicious accesses with a model of the users' normal behavior.

We envision that use of this additional defense mechanism will improve security. To substantiate our claim, we constructed a small case study of a railway system and evaluated the security of the system with and without the addition of our defense mechanism [24]. We used the *ADversary VIew Security Evaluation* (ADVISE) tool to perform a quantitative security assessment of the system. The quantitative security analysis revealed that the addition of our defense mechanism can help in the detection of non-stealthy attackers or, at the very least, increase the overhead for stealthy attackers.

4 Malicious Insider Detection Framework

In this section, we describe our framework, which systematically analyzes users' physical movement logs to detect malicious insiders. The framework dissects the problem into three parts: understanding the characteristics of users' behaviors, learning a suitable model representation, and using the model together with the system layout to estimate the probability of an abnormal access.

4.1 Preliminaries and Definitions

We define a system $sys = (U, Env)$ (e.g., an enterprise organization or critical infrastructure) as the collection of users U who work for it, and the environment Env that contains the system's assets. The environment Env consists of both the physical and cyber aspects of the system. The physical aspect is composed of the building and the physical assets within it. The cyber aspect consists of the networked computer system and its digital assets. The cyber and physical aspects are interrelated, but we focus only on the physical aspect in this paper.

We represent the building topology as a directed graph $G = (S, E)$ in which the set of vertices represents the spaces in the building. A directed edge $e(v_1, v_2)$ represents possible movement from v_1 to v_2.[1] For example, the floor plan in Figure 1b is represented as the graph in Figure 1c. The set of spaces S can be divided into two partitioning subsets: rooms $\mathbf{R}$, and common areas $\mathbf{C}$ (e.g., staircases, corridors), i.e., $S = \mathbf{R} \cup \mathbf{C}$, and $\mathbf{R} \cap \mathbf{C} = \phi$. The edges are labeled with the access door codes that are associated with user access. The edges can be weighted to reflect a metric quantifying the relationship between the spaces, i.e., $w : E \to \mathbb{R}$. For example, $w(e(u, v))$ can represent the physical distance between spaces u and v. In Section 4.3, we will define w more precisely and use the edge weights to determine the shortest paths between spaces.

The state of the system at time t is defined as $State_t(sys) = (L^t, Access_t, Q_t)$, where

- $L^t \subseteq S^{|U|}$ is the location of the users in the system,
- $Access_t \subseteq U \times S \times S$ is the set of accesses, where an access is a triple (u_i, s_1, s_2) or $s_1 \to_{u_i} s_2$, and
- Q_t is the state of the environment, including the condition of the physical and cyber topology and assets (e.g., malfunctioning devices, changes in network access).

4.2 Phase 1: Offline

The framework consists of an offline phase and an online phase as shown in Figure 2. The offline phase consists of two stages: characterization of users based on their past movement behavior, and construction of models based on users' characteristics and past movement. The input to this phase is the training data that describe the historic system states $State_{Hist} = State_{t_{-1}}(sys) \ldots State_{t_{-n}}(sys)$, and the output of this phase is a collection of tuples $(\mathcal{M}, \mathbf{Thresh})$, in which $\mathcal{M}$ is a model representing the movement behavior of a user and $\mathbf{Thresh} : Access_{Past} \to \mathbb{R}$ is a function that takes in a set of accesses and returns a real number that represents the threshold value, below which the probability score of a movement will be considered anomalous.

[1]This implies that if $e(v_1, v_2)$ exists, the backward edge $e(v_2, v_1)$ also exists in G.

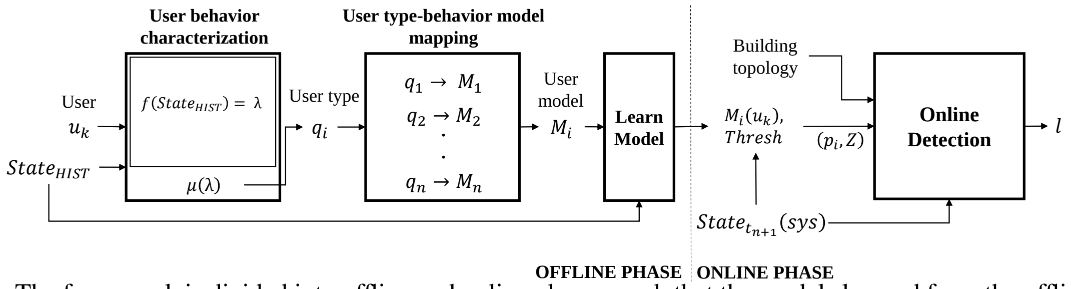

Figure 2: The framework is divided into offline and online phases, such that the models learned from the offline phase are fed into the online phase.

4.2.1 User Types

The first stage of the offline phase is to distinguish between different users based on their past movement behavior. Typical access control systems assign roles to users based on the sets of rooms that they need to access. However, these roles do not directly reflect the user behavior. Instead, we propose to categorize users according to how they move within a building.

We define the different types of user behavior $\mathbb{T}$ based on the users' "reasons" for movement, where "reason" refers to the context that facilitates users' movement patterns. These reasons can be inferred from the structure and numerical values of the historical system state pertaining to the users. We define a function $f : State_{Hist} \rightarrow \mathbf{F}$ that extracts features from the historical system state. We then calculate a function $\mu : \mathbf{F} \rightarrow \mathbb{T}$ over the feature vector returned by f and output the user type or "reason".

Application: In our railway station case study, there are two main types of user movement behavior, $\mathbb{T} = \{q_1, q_2\}$. The first type of users, q_1, is those who have very regular movement behavior. This applies to station operators who work a fixed set of hours in the station; because of their job scope, their movement patterns are fairly consistent. They remain in the *Passenger Service Center* (PSC) to assist the public and monitor the state of the station, visit storerooms and staff rooms, and clock in and out.

The second type of users, q_2, is those whose movement is triggered when an event occurs. This applies to maintenance staff who visit rooms to conduct maintenance and repairs of the equipment. Different maintenance staff members are in charge of different subsystems (e.g., power supply or signaling); thus, they access different sets of rooms in the station.[2]

In order to categorize a user into q_1 or q_2, we extract a feature based on the approximate entropy of a time series [25] constructed using the collected historical access data, or training data, $f(Access_{Hist}) = ln(C_m/C_{m+1}) = \lambda_e \in \mathbf{F}$, where C_m is the prevalence of repetitive patterns of length m in the time series. Each subsequence of length m in the sequence is compared to other subsequences. If the number of similar subsequences is high, then C_m is large. This metric has been shown to be able to quantify the predictability of user movement [26, 27]. We choose m to be 3, which provides a good metric for characterizing our trace, as shown in Section 5. If the user's entropy value is lower than $\mathbf{E}$, then the user has more predictable movement. Therefore, the user belongs to q_1; otherwise, the user belongs to q_2. In other words,

$$\mu(\lambda_e) = \begin{cases} q_1 & \text{if } \lambda_e < \mathbf{E} \\ q_2 & \text{otherwise} \end{cases} \tag{1}$$

where $\mathbf{E}$ is a numerical threshold. The choice of parameter $\mathbf{E}$ is discussed in Section 5.

4.2.2 User Behavior Models

Next, we construct behavior models for each behavior type $q \in \mathbb{T}$ defined earlier. Since each user is motivated to move within the building for different reasons, it is not possible to specify a single model for all users' behavior. Such a model would be inherently biased towards a certain set of users and perform badly for others.

Instead, for each $q \in \mathbb{T}$, we select an appropriate modeling technique $\mathcal{M} \in \mathbb{U}_{\mathcal{M}}$ from a large set of possible modeling techniques $\mathbb{U}_{\mathcal{M}}$. The model should leverage q's distinct characteristics and provide insight into the likelihood that a user will access a room given the current system state $State_{t_{n+1}}(sys)$.

[2]The same principle may apply to other systems too. E.g., a security guard doing rotations in a building belongs to q_1, and a technical support staff member who goes to an office when his or her assistance is required belongs to q_2.

For each user u of type $q \in \mathbb{T}$, we learn the model by analyzing the historic system states $State_{Hist}$ described by the training data in order to assign probabilities to the rooms in $\mathbf{R}$. Finally, we describe the construction of the $\mathbf{Thresh} : Access_{Past} \rightarrow \mathbb{R}$ function in Section 4.3 for ease of understanding.

Application: Users belonging to q_1 have a lower entropy value $\lambda_e < \mathbf{E}$ than those belonging to q_2. This implies that their movement patterns are more predictable and repetitive. Thus, we choose to represent a user's movement behavior with a Markov model.

Given the historic system states $State_{Hist}$ (obtained from the training data), we learn the Markov model of a user u. We can reconstruct the full movement sequence $Seq_{Hist}(u) = S_1 \ldots S_n$ from $Access_{Hist}$. The sequence $Seq_{Hist}(u)$ (or $Seq(u)$ for short) can be divided into segments such that a period of inactivity (of more than 3 hours) separates any two segments , i.e., if $Seq(u) = Seg^1 Seg^2 \ldots Seg^m$ where $\sum_{i \in \{1,\ldots,m\}} |Seg^i| = n$ and $\forall i \in \{1, \ldots, m\}, Seg^i = S_g \ldots S_h, Seg^{i+1} = S_{h+1} \ldots S_j$, then the accesses $(u, S_{h-1}, S_h) = a_t, (u, S_h, S_{h+1}) = a_{t+T}$ occur $T \geq 3$ hours apart.

There are many ways to represent that movement sequence as a Markov model. In this paper, we present two different ways to express a user's movement sequence in the form of a Markov model. Specifically, we define (1) a Markov model that captures only the sequence of rooms that are visited, $\mathcal{M}_r$, which is the model defined in [21], and (2) a Markov model that, in addition to the visited rooms, also captures the sequence of common areas visited on the path to those rooms, $\mathcal{M}_s$. The model $\mathcal{M}_r$ is less restrictive than $\mathcal{M}_s$ because, unlike $\mathcal{M}_s$, it does not describe the likelihood of a user's favoring different paths to a room. As such, $\mathcal{M}_r$ allows more freedom for a user to move within a building without having his or her movement flagged as suspicious. (E.g., a user could take a different path to the staff room because he or she stopped by a store in the opposite direction.) However, such freedom comes at the cost of potentially missing malicious movement that deviates from the user's normal path. For example, an attacker may choose to take a path that is less populated to avoid meeting other staff members. We will see in the next section how these two models, $\mathcal{M}_r$ and $\mathcal{M}_s$, compare in terms of false positive and false negative rates.

We define the Markov model $\mathcal{M}_r$ as having:

- A state space that is the set of rooms $\mathbf{R}$,
- A transition probability matrix P with the entry $p_{ij} = \frac{\#r_i c_1 \ldots c_n r_j \in Seq}{\#r_i \in Seq}$ as the normalized frequency with which the user visits r_i and then r_j, and
- An initial probability vector π with $\pi_i = \frac{\#Seg^j = (c_j \ldots c_n r_i \ldots)}{\#Seg}$,

where $c_1 \ldots c_n \in \mathbf{C}$.

On the other hand, the states in the Markov model $\mathcal{M}_s$ are represented by a tuple (r_{prev}, s), where r_{prev}, just as in $\mathcal{M}_r$, is the previous room visited, and $s \in S$ is the space that the user is currently in. For example, a movement sequence $c_0 \ldots c_n r_0 c_{n+1} \ldots c_m r_1$ in which $c_i \in \mathbf{C}, i \in \{0, \ldots, m\}$ can be expressed as the following state transitions: $(\cdot, c_0) \rightarrow (\cdot, c_1) \cdots \rightarrow (\cdot, r_0) \rightarrow (r_0, c_{n+1}) \rightarrow (r_0, c_{n+2}) \cdots \rightarrow (r_0, r_1)$. So we define the Markov model $\mathcal{M}_s$ as having:

- A state space that is the set of tuples $(\mathbf{R}, S)$,
- A transition probability matrix P with the entry $p_{ij} = \frac{\#r_i c_1 \ldots c_n s_k s_j \in Seq}{\#r_i c_1 \ldots c_n s_k}$ as the normalized frequency with which the user is in state $i = (r_i, s_k)$ and then $j = (r_m, s_j)$, and
- Initial probability π with $\pi_i = \frac{\#Seg^j = (s_i \ldots)}{\#Seg}$.

However, the users belonging to q_2 have less regular movements and may change movement patterns based on events in the system. Therefore, we combine the Markov model with additional contextual knowledge about the states of the devices in the rooms. More specifically, we take each state that is associated with a room, and annotate it with a Boolean variable indicating whether a device failure has occurred within that room.

4.3 Phase 2: Online

The online phase involves determining, based on the behavior models derived from the offline phase, whether a user's access is an abnormality. We are advancing the state of the art of physical security systems by attempting to detect anomalous accesses as early as possible so that the appropriate response actions, such as blocking of access to a room, can be taken. The inputs to this phase are the tuple $(\mathcal{M}, \mathbf{Thresh})$ from the offline phase and the current state of the system $State_{t_{n+1}}(sys)$. The output of this phase is a real number in $\mathbb{R}$ that indicates the degree of abnormality of the access.

The current system state $State_{t_{n+1}}(sys)$ includes the location of a user u_i, $L_i^{t_{n+1}} = s_1 \in S$, and the physical access that is being made, $A = (u_i, s_1, s_2) \in Access_{t_{n+1}}$. In other words, the user is moving from s_1 to s_2. Using the knowledge of the building topology, the behavior model $\mathcal{M}$, and the current state, we can calculate the probability that the current access fits into the user's normal movement behavior. The calculated probability score is compared to the threshold value returned from the application of the $\mathbf{Thresh}$ function on the past accesses $Access_{Past} = Access_{t_0} \dots Access_{t_n}$. If the probability score falls below the threshold value, then the access is marked as anomalous.

Application: During the online phase, we keep track of certain properties of the past system state. For the Markov model $\mathcal{M}_r$, we keep track of the room that the user has last accessed $r_L \in \mathbf{R}$, where $Seq_{Past} = s_1 \dots r_L c_1 \dots c_m$. Then, based on the transition probability matrix, we can extract the set of probabilities associated with the next possible visited room $Next_R = \{r_i \in \mathbf{R} | p_{Li} > 0\}$. Since the Markov model $\mathcal{M}_r$ does not directly assign transition probabilities to the non-room spaces, we instead derive the probability score of an access by determining whether the user's movement is bringing him or her closer to the next possible room to be visited.

In other words, given the access A, we want to determine all the rooms that the user is likely to access. We first find all the rooms that are reachable from s_2, i.e., $P_T = \{r_i \in Next_R | \exists (s_2, s_i, \dots s_{i+m} r_i)\}$, where $(s_2, s_i, \dots s_{i+m} r_i)$ is a sequence of vertices on the path from s_2 to r_i. For all such vertices $r_i \in P_T$, we decide whether the user is likely to access r_i by moving to s_2 from s_1. If it's easier to access r_i from s_2, then we consider r_i as one of the likely rooms that u_i will move to next. To decide whether r_i is easily accessed, we calculate path lengths by using the weights on the edges, i.e., $d(s_i, s_j) = \min_{\forall (s_i, s_1 \dots s_m, s_j)} w(e(s_i, s_1)) + w(e(s_m, s_j)) + \sum_{k=1}^{m-1} w(e(s_k, s_{k+1}))$. We calculate the shortest path from s_1 to r_i, $d(s_1, r_i)$, and compare it to the shortest path from s_1 to r_i through s_2, $d(s_1, s_2) + d(s_2, r_i)$. If the shortest path through s_2 is similar in length to the shortest path $d(s_1, r_i) \approx d(s_1, s_2) + d(s_2, r_i)$, then we consider r_i to be a possible room that the user wants to access. In this paper, we choose a simple assignment of weights to edges. This assignment of weights can easily be changed depending on users' preferences for taking certain types of paths. We assign all edges a weight of 1, with the exception of edges that connect different levels of the building (i.e., staircases, elevators, and escalators). We assume that users prefer to take as few staircases as possible, so we assign a weight of 10 to those edges that connect different levels. We take the resulting shortlisted set of rooms and sum up their likelihoods $\sum_{r_i \in P_T} p_{Li}$ to obtain a final score.

Although we describe the construction of the $\mathbf{Thresh}$ function in this section for ease of understanding, the construction is actually performed in the offline stage alongside the construction of the movement models. The $\mathbf{Thresh}$ function maps the past accesses $Access_{Past}$, or, more specifically, a property of the past accesses that is kept track of in the online phase, to a threshold value; if the probability of a movement is below that value, the movement is considered anomalous. For the Markov model $\mathcal{M}_r$, $\mathbf{Thresh}$ takes in the room last accessed, r_L, and returns the xth percentile of the probability distribution p_L, i.e., $\mathbf{Thresh}(r_L) = percentile_x(p_L)$. The reasoning behind the threshold is that there are rooms that the user may seldom visit, so transitions to them may be deemed anomalous. Therefore, we take a certain percentile value, x, of the probability distribution associated with the room r_L, p_L, as the threshold. The percentile value can be changed by practitioners based on the system's requirements; a higher value reduces the false positives but potentially allows malicious movements to be missed, while a lower value catches more malicious movements but increases the false positives. We compare the probability score to the threshold value returned by the function $Z = \mathbf{Thresh}(r_L)$, and if the score is below Z, access A is deemed anomalous. The algorithm for this phase is given below as Algorithm 1.

Algorithm 1 ONLINEDETECTION algorithm for $\mathcal{M}_r$ Markov model

```
Require: $(L_i^{t+1}, A = s_1 \to s_2) \in State_{t+1}, r_L = Property(Access_{Past})$
  function ONLINEDETECTION($State_{t+1}, \mathcal{M}_r, Thresh, Property(Access_{Past})$)
      $score \leftarrow 0; Z \leftarrow Thresh(r_L)$
      $NextRoom \leftarrow \mathcal{M}_r(r_L)$
      for all $r_i \in NextRoom$ do
          $shortestlen \leftarrow GetShortestPath(s_1 \to r_i)$
          $len \leftarrow GetShortestPath(s_2 \to r_i) + w(e(s_1, s_2))$
          if $len < shortestlen \times k$ then $score \leftarrow score + p_{Li}$ end if
      end for
      if $s_2 \in \mathbf{R}$ then $r_L \leftarrow s_2$ end if
      if $score < Z$ then return Anomaly end if
  end function
```

For the Markov model $\mathcal{M}_s$, instead of calculating the probability of the current access, we calculate the probability that the sequence of movement steps is within normal user movement behavior. In other words, for a sequence of Markov states $b_0 \to b_1 \cdots \to b_n$, the probability of such a sequence is $\prod_{i=0}^{n} p_{b_i b_{i+1}}$. Since we are calculating that probability in an online fashion, we only need to keep track of the previous accumulated product, $\prod_{i=0}^{n-1} p_{b_i b_{i+1}}$, and the previous state b_{n-1}.

The $\mathbf{Thresh}$ function, in this case, takes in the length of the movement sequence, in addition to the starting state $b_0 = (r_0, s_0)$. For each state $b = (r_i, s_j)$ in the transition probability matrix, we use the movement sequence from $Access_{Hist}$, obtained from the training data, to determine the typical probability distribution of a user who starts in state b and moves k steps within the building. More precisely, $\mathbf{Thresh}(b_0, k) = percentile_x(\{\prod_{i=0}^{k-1} p_{b_i b_{i+1}} | b_o \dots b_k \in Seq_{Hist}\})$. In the next section, we vary this percentile value to examine the trade-off between the false positive and true positive rates for a given Markov model.

Then we can compare the threshold value returned by the function $Z = \mathbf{Thresh}(b_0, k)$ with the accumulated probability score for the current access. If the probability score is below the threshold Z, then the access is deemed anomalous.

However, since this simple algorithm relies on the accumulation of probabilities, the presence of low probability values earlier in the movement sequence will directly affect the probability score of future accesses. In other words, the appearance of anomalies in the movement sequence would cause the probability of the movement chain to drop below the threshold even after the user's movement returns to normal, thus falsely marking future accesses as being equally anomalous.

To prevent that phenomenon from occurring, we divide the accumulated probability score (and threshold value) by the path length, i.e., $\frac{1}{k} \prod_{i=0}^{k} p_{b_i b_{i+1}}$. This division allows us to average out the effect of the low probability values.

However, even with that division, the probability of the movement chain after the anomalous movement cannot immediately bounce back to above the threshold value. Instead, it slowly increases in value. Therefore, to prevent the low probability values from influencing future accesses in the movement chain, we restart our calculations and start a new accumulation of a movement chain immediately after the anomalous accesses. To pinpoint the access immediately after the anomalous accesses, we compare the previous probability score $score_k$ with the current access's probability score $score_{k+1}$. If $score_k < score_{k+1} < Z$, that implies that the transition probability associated with the current access is high enough to cause the average to increase. That means that the current access is probably not anomalous. The algorithm using the Markov model $\mathcal{M}_s$ is given below as Algorithm 2. The algorithms for both M_r and M_s are of complexity $O(1)$.[3]

Algorithm 2 ONLINEDETECTION algorithm for the $\mathcal{M}_s$ Markov model.

```
Require: $(L_i^{t+1}, A = S_1 \to S_2) \in State_{t+1}$,$(r_L, score_{prev}, chainlen, state_{prev}, b_0) = Property(Access_{Past})$
  function ONLINEDETECTION($State_{t+1}, \mathcal{M}_s, Thresh, Property(Access_{Past})$)
      $state_{curr} \leftarrow (r_L, S_2)$
      $Z \leftarrow Thresh(b_0, chainlen + 1)$
      $score \leftarrow \frac{(score_{prev} \times chainlen) \times \mathcal{M}_s(prevState \to currState)}{chainlen+1}$
      if $score < Z, score \leq score_{prev}$ then return Anomaly end if
      if $score < Z, score > score_{prev}$ then
          $score_{prev} \leftarrow 0, chainlen \leftarrow 0$
          $r_L \leftarrow \phi, prevState \leftarrow \phi$
          $b_0 \leftarrow currState$
      else
          $score_{prev} \leftarrow score$
          $chainlen \leftarrow chainlen + 1$
          if $S_2 \in \mathbf{R}$ then $r_L \leftarrow S_2$ end if
          $prevState \leftarrow currState$
      end if
  end function
```

Finally, for the q_2 users, we include another step in the online detection algorithm that correlates the remaining suspicious accesses with logs about the device state after all the accesses have been vetted through the Markov model.

[3]For Algorithm 1, we can precompute the probability score for each space in the offline phase, thus removing the complexity of computing the shortest paths during the real-time analysis.

Intuitively, we can see that if a device in room R_d fails and then a physical access into R_d is logged, that physical access is considered non-malicious. When a device failure in a room is logged, we update the Boolean variable of the state in the Markov model that is associated with that room to indicate that a movement into that room may be impending. Then, when an access is made to that room, we run it through the ONLINEDETECTION algorithm. If it returns with a suspicion alert, then we check whether the Boolean variable associated with the room has been checked. If it has been, then we deem that access non-malicious.

5 Evaluation

In this section, we utilize real-world data traces to demonstrate the effectiveness of our framework in our railway transit station case study. First, by evaluating our usage of the entropy metric, we answer the question of whether the movement behavior of users can be characterized effectively in a complex system. Second, we determine the detection capability of our proposed behavior models. Finally, we examine the possibility of detecting malicious movement in an online manner.

5.1 Empirical Analysis of Data

We use a real-world data set containing information on physical card accesses to a railway station in a city. The accesses took place between June 2016 and August 2017. The station has 62 rooms, and a total of 141,357 accesses were made by 590 users. While we focus on one station in this work, the whole railway line consists of 33 stations, 12 of which are interchange stations. We estimate that the average number of accesses per hour over all the stations is approximately 450, whereas the highest number of accesses per hour is around 1,200. This poses a significant challenge if the associated logs need to be examined manually.

The data set contains the following information regarding physical accesses: (1) date and time, (2) door code, (3) user identification, and (4) type of access (legal entry, failed entry, clock-in, or clock-out). When an access is a failure, it implies either that the user's card has expired or that the user does not have permission to access the room. Those failed accesses serve as ground truth for known abnormal accesses. Clock-ins and clock-outs represent user swipes at a time punch clock inside a room in the station.

In motivating the design of our detection framework and user behavior model in Section 4, we make a number of claims about the spatial and temporal properties of user movement behavior. In the following section, we substantiate our claims by analyzing the physical movement data set and justify our modeling choices.

5.1.1 Movement Independence

First, we justify our decision to model user movement with first-order Markov chains. We make the following two claims in support of our decision: (1) a user's next movement (or access) can be well-predicted using only recent movement history (i.e., that user's movement behavior is Markovian), and (2) it is sufficient to use just the current state to determine a user's subsequent movement (i.e., the Markov models should be *first-order*). We substantiate both claims by using chi-square tests for Markov chain analysis, as described in [28]. We ran a multi-stage test in which each nth stage attempts to reject the null hypothesis that the data conform to a Markov chain of order higher than n. Accepting the null hypothesis for the nth test signifies that a user's subsequent movement depends only on the previous n states in the user's Markov chain (inclusive of the current state); rejecting the null hypothesis signifies that there is likely some higher-order dependence in the data. More precisely, we have three null hypotheses:

$$H0 : \chi^2 = \sum_{x_t, x_{t+1} \in \mathbf{ST}} \frac{(N(x_t, x_{t+1}) - \frac{N(x_t)N(x_{t+1})}{n-1})^2}{\frac{N(x_t)N(x_{t+1})}{n-1}} \tag{2}$$

$$H1 : \chi^2 = \sum_{x_t, x_{t+1}, x_{t+2} \in \mathbf{ST}} \frac{(N(x_t, x_{t+1}, x_{t+2}) - \frac{N(x_t,x_{t+1})N(x_{t+1},x_{t+2})}{N(x_{t+1})})^2}{\frac{N(x_t,x_{t+1})N(x_{t+1},x_{t+2})}{N(x_{t+1})}} \tag{3}$$

$$H2 : \chi^2 = \sum_{x_t, x_{t+1}, x_{t+2}, x_{t+3} \in \mathbf{ST}} \frac{(N(x_t, x_{t+1}, x_{t+2}, x_{t+3}) - \frac{N(x_t,x_{t+1},x_{t+2})N(x_{t+1},x_{t+2},x_{t+3})}{N(x_{t+1},x_{t+2})})^2}{\frac{N(x_t,x_{t+1},x_{t+2})N(x_{t+1},x_{t+2},x_{t+3})}{N(x_{t+1},x_{t+2})}} \tag{4}$$

where $\mathbf{ST}$ is the state space of the Markov chain, $N(x_i, \ldots, x_{i+m}) = |\{t | X_t = x_i, \ldots, X_{t+m} = x_{i+m}\}|$, and X_t represents the tth state of the user movement sequence Seq_u. The first null hypothesis $H0$ tests for independence between the Markov states. Null hypothesis $H1$ tests for first-order dependence, and null hypothesis $H2$ tests for second-order dependence. We expect that each user's movement sequence should reject $H0$, implying that there is dependence on recent movement history (Claim 1), but accept $H1$, implying that subsequent movements will best be predicted using only the current state (Claim 2).

For each user, we convert his or her historic movement sequence Seq_{Hist} into a sequence of Markov states, X_t. In the case of the Markov model $\mathcal{M}_r$, it would be a sequence of rooms that the user accesses. In the case of the Markov model $\mathcal{M}_s$, it would be a sequence of (room,space) tuples. Then, we calculate his/her chi-squared values for each of the three null hypotheses and, based on the rejection or acceptance of each hypothesis, determine the order of the Markov chain that statistically fits the user's movement sequence. More precisely, the order of the user's Markov chain is

$$n = \begin{cases} 0, & \text{if } p_0 > 0.15 \\ 1, & \text{if } p_0 \leq 0.15, p_1 > 0.01 \\ 2, & \text{if } p_0 \leq 0.15, p_1 \leq 0.01, p_2 > 0.01 \\ \geq 3 & \text{otherwise,} \end{cases}$$

where p_0, p_1, p_2 are the p-values obtained from the three chi-squared tests.

Table 1: The percentage of users who fit an nth-order Markov model $\mathcal{M}_r$ or $\mathcal{M}_s$.

		Order			
		0	**1**	**2**	**≥ 3**
Model	$\mathcal{M}_r$	19.0	64.6	10.9	5.5
	$\mathcal{M}_s$	7.5	89.1	1.4	2.0

The results in Table 1 show that when we examined users' room movements ($\mathcal{M}_r$ Markov models), we found that over 84% of users were best modeled by zero-order or first-order models, and when we examined (room,space) movements ($\mathcal{M}_s$ Markov models), over 95% of users were best modeled by zero-order or first-order models. Note that any user best modeled by a zero-order model (i.e., there is no dependence between movements) can be accurately modeled by a first-order model that is a complete graph with identical next-state transition probabilities from each state. Thus, both of our assumptions hold for the vast majority of users. We also find that the few users who exhibit second-order and higher dependence in their $\mathcal{M}_s$ models have a high entropy, with a minimum value of 1.04, which indicates that their movement behavior is relatively aperiodic and therefore unlikely to achieve good performance with our approach.

5.1.2 Consistency of Movement Paths

In this subsection, we justify our decision to model the sequence of spaces (or the path) taken by users in $\mathcal{M}_s$. We claim that users tend to follow a small set of distinct paths, and thus that those paths are a property or feature of their normal movement behavior; any deviation from those paths should be considered anomalous. We would like to show empirically that users habitually follow the same path when moving from a given room to another, only occasionally choosing a different path.

For each user, we group his or her historic physical accesses Seg_{Hist} by the sequence of doors visited between two rooms. We obtain tuples of $(u_i, r_j, r_k), u_i \in U, r_j, r_k \in \mathbf{R}$, and each tuple is associated with the collection of trips made between rooms r_j and r_k by u_i, $Trip(u_i, r_j, r_k) = T^i_{j,k} \subseteq P$. We define a trip as the sequence of doors that are accessed, $p = (d_1, d_2 \ldots d_n) \in P, d_i \in DoorCode$, and two trips are equal only if their sequences are exactly the same. We can then define the collection of distinct trips for each tuple $UniqTrip^i_{j,k} = \{p_1, p_2 \in T^i_{j,k} | \forall p_1, p_2 \in UniqTrip, p_1 \neq p_2\}$.

In total, there are 2,273 (u_i, r_j, r_k) tuples. For each tuple, we calculate the following:

- $Overlap = \frac{|\{d \in p_i \cap p_j | p_i, p_j \in UniqTrip, i \neq j\}|}{|\{d \in p_i | p_i \in UniqTrip\}|}$, the fraction of doors that are shared among the unique trips;
- $FrequentedTrip = \{p \in UniqTrip \, | \frac{|t \in T | t = p|}{|T|} > \frac{1}{|UniqTrip|}\}$, the unique trips that appear more than average, i.e., the unique trips that appear more than $\frac{1}{x}$ times, where x is the total number of unique trips; and
- $ConcentratedTrip = \frac{1}{|T|} \sum |\{t | t = p, p \in FrequentedTrip\}|$, the fraction of all trips that are in $FrequentedTrip$.

We divide the 2,273 (u_i, R_1, R_2) tuples into three categories:

- Tuples with $|UniqTrip| \leq 5$,
- Tuples with $|UniqTrip| > 5$ and $ConcentratedTrip > 0.8$, and

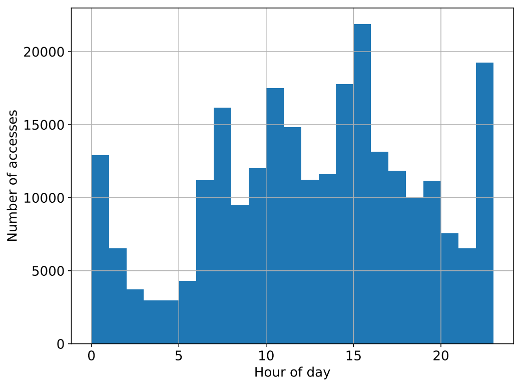

Figure 3: The distribution of all users' accesses over the time of day.

- Tuples with $|UniqTrip| > 5$ and $ConcentratedTrip \leq 0.8$.

The first category is tuples that have fewer than 5 unique trips. We find that 96.9% of the tuples fall under this category. The average of $Overlap$ over those tuples is 0.42, which implies that the unique trips do share common doors that are accessed.

The second category is tuples in which there are a larger number of unique trips but in which more than 80% of the total trips are associated with a small subset of unique trips. There are 27 tuples in the second category. All the tuples have $|FrequentedTrip| \leq 5$, except for one tuple with $|FrequentedTrip| = 6$ and one with $|FrequentedTrip| = 7$. So fewer than 7 unique trips account for a majority of the trips. Those unique trips also have a higher average overlap, $Overlap = 0.74$. That implies that although there are more unique trips than in the first category of tuples, the paths are quite similar.

Finally, the third category is tuples whose trips are more varied. The third category has 44 tuples. For those tuples, $FrequentedTrip \leq 4$, and those trips that are not in $FrequentedTrip$ account for fewer than 16% of the total trips T. Even though there is a larger spread of unique trips, the average overlap between those trips is high: $Overlap = 0.72$. So the unique trips were very similar and differed only in terms of the sequence or repetition of some accessed doors. Of the 44 tuples, only 4 have a lower overlap, $Overlap \leq 0.5$. For those tuples, the less frequented trips $p \notin FrequentedTrip$ appeared only once or twice, so those unique trips were very rarely taken.

In conclusion, our analysis informs us that users tend to take a small set of unique paths, and if we delve further into the sequences of doors accessed, we find that the unique paths have high overlap. Thus, the paths that users take can be characterized as part of their normal movement behavior patterns, so any deviations from those paths would be empirically anomalous.

5.1.3 Temporal Behavior

Finally, we examine the temporal behavior of users' physical movements to determine whether such information can be used in tandem with spatial behavior to characterize users' movement sequences. First, we examine the time of day of users' accesses to test for the presence of time-of-day patterns in the users' movement behavior that could be used to detect anomalous movements. Figure 3 shows the distribution of all the users' accesses over the time of day. We see that the accesses are spread throughout the day, with multiple small surges of activity at certain times of the day. This implies that the users do not share a single, well-defined schedule, though users do seem to be more active during the normal workday (7 a.m.–8 p.m.).

Thus, we further examine the users' individual accesses. In our data set, we have accesses that represent users' clock-ins and clock-outs. Users with such accesses are more likely to have a regular schedule in the system, since their work hours are explicitly logged. We find that 29% of the users have clock-ins and/or clock-outs in their physical access logs. However, only 4 of those users consistently perform a clock-in and clock-out when they enter and leave the station. Further, when we examine those users' time-of-day accesses in Figure 4, we find that while some of the users have more well-defined shift timings, their shifts are not consistent and change frequently. For example, Figure 4a shows the time-of-day distribution of accesses for one of the users with clock-ins and clock-outs. There are three big spikes in the user's overall distribution, which implies that the user likely has three different shift schedules in the logs. However, we see in Figure 4c that the shifts do not have consistent lengths and that their timings can overlap (e.g., 6 a.m.–8 p.m. and 6 p.m.–8 a.m.). Furthermore, the changes in shifts do not follow a consistent periodic behavior. Users who do not have any clock-ins and clock-outs similarly have very sporadic behavior, as shown in Figures 4b

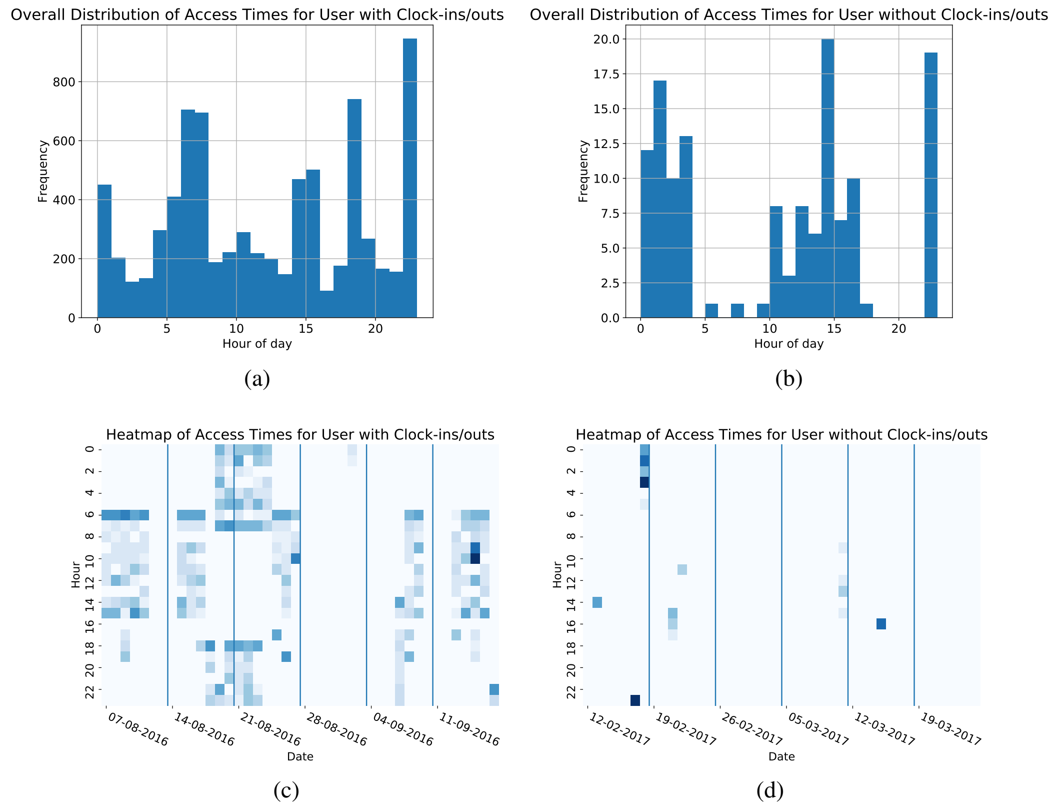

Figure 4: Distribution of access times for different users. (a) Overall distribution of the access times and (c) distribution of the access times over a few days of data for a user with clock-ins and clock-outs. (b) Overall distribution of the access times and (d) distribution of the access times over a few days of data for a user with no clock-ins or clock-outs.

and 4d. Without additional information about the actual assignment of shifts to users, it is nontrivial to predict timing information for users, and that limits the utility of using temporal behavior in detecting malicious accesses. Thus, while using time-of-day behavior might provide additional information, we decided not to use it in this work, opting instead to look only at spatial behavior. As we will show in the subsequent sections, our spatial-based detector performs very well on the users with clock-ins and/or clock-outs, so we believe that the utility of including temporal behavior in our model would be limited.

5.2 Experiment Setup

We simulated malicious movement in order to conduct a more thorough assessment of our detection ability. For each user, we injected accesses into the testing data, simulating the case in which an adversary deviates from the normal movement data. We chose a few points within the sequence of movement accesses in the testing data to represent the time at which the adversary decided to move off-course. Then, for each point, S_i, we randomly selected a target room $R_T \in \mathbf{R}$ that the adversary would visit, because almost all rooms house critical assets and can be potential targets. For the sake of simplicity, we assume that the adversary will take the shortest path to reach the room R_T from the current point S_i, because the adversary would want to swipe through as few doors as possible to reduce the possibility of detection. Therefore, we calculated the shortest path from S_i to R_T as $S_i e_1 S_1 \ldots S_n e_n R_T$. For each edge $e_i, i \in [1, n]$ that has a door code, we added an injected access $A_i = S_i \rightarrow S_{i+1}$ that represents the logging of the adversary's movement as he or she uses the acquired access control device to move through the spaces protected by card readers. We can repeat this process to choose a series of rooms that the attacker visits (to consider the case in which an adversary wanders around a space to do a site survey and explore potential attack opportunities, or when an adversary needs to access a few rooms in sequence to achieve his or her goal). We can also restrict the choice of rooms to only those rooms that have been visited by the user in the historic physical access logs, to simulate a stealthier adversary. We set the number of rooms visited by the adversary as 1, and as described in Section 5.3.2, we experimented with increasing the number of rooms visited by the adversary. We set the rate at which we injected data (or chose points within the testing data) at 5%. In Section 5.3.2, we describe our experiments in which we varied the percentage of malicious accesses by changing the rate at which we injected data.

We split the data set into 80% training and 20% testing subsets and performed tenfold cross-validation. We conducted the experiments on a Windows 7 Home Premium machine with a 2.7 GHz CPU core and 4 GB of RAM.

5.3 Results

In this subsection, we present the evaluation results for our approach from Section 4 based on the physical card access data from the railway station. We also compare our approach to a baseline method that is a simple extension of building access controls. The baseline method keeps a list of spaces $Visited = \{s \in Seq_{Hist}(u_i)\}$ that have been visited by a user u_i, and any card access by u_i that leads to a space not contained in $Visited$ is marked as malicious. In other words, the baseline method is similar to the whitelisting in building access controls, but further limits the user's movement to only those spaces in the historic access logs that the user has visited before.

5.3.1 Implementation Performance

We evaluated the running times of both the offline and online phases. The average running times (over all users) of the construction of Markov models in the offline phase were 1.18 s and 12.4 s, respectively, for the $\mathcal{M}_r$ and $\mathcal{M}_s$ Markov models, whereas the average running times over all physical accesses of the ONLINEDETECTION function in the online phase were 0.2 ms and 0.07 ms, respectively, for the $\mathcal{M}_r$ and $\mathcal{M}_s$ Markov models. The running time of the offline phase for the $\mathcal{M}_s$ Markov model is longer than that of $\mathcal{M}_r$ due to the larger amount of information needed to construct the **Thresh** function. On the other hand, the running time of the online phase for the $\mathcal{M}_r$ Markov model is longer than that of $\mathcal{M}_s$ due to the calculation of shortest paths in the ONLINEDETECTION algorithm. The offline phase can be conducted sporadically during system downtime, whereas the online phase is fast enough to be executed in a real-time manner.

The average size over all the users of the constructed $\mathcal{M}_r$ Markov models is 3 vertices and 6 edges, with a maximum size of 28 vertices and 77 edges. On the other hand, the average size over all the users of the $\mathcal{M}_s$ Markov models is 16 vertices and 28 edges, with a maximum size of 162 vertices and 251 edges.

5.3.2 Detection Capability

We varied the percentile value used to select the threshold value in **Thresh** for the two different Markov models and plotted their *receiver operating characteristic* (ROC) curves. The ROC curve plots the true positive rate against the false positive rate obtained using different percentile values and is used to analyze the ability of a binary classifier. The closer the curve is to the top left-hand corner of the plot, the better the classifier (or, in this case, our intrusion detection solution) is. The ROC curves are shown in Figure 5. The *areas under the curve* (AUCs) of the $\mathcal{M}_r$ ROC curve and the $\mathcal{M}_s$ ROC curve are 0.80 and 0.85, respectively. Those ROC curves show that both models achieve relatively high accuracy rates, although at the slight cost of an increased false positive rate. Based on the higher-peaked curve and the AUC value, the $\mathcal{M}_s$ model does a better job at classification than the $\mathcal{M}_r$ model. For the results detailed in the following paragraphs, we picked a 95th percentile value as the threshold for both the $\mathcal{M}_r$ and $\mathcal{M}_s$ models, for fair comparison.

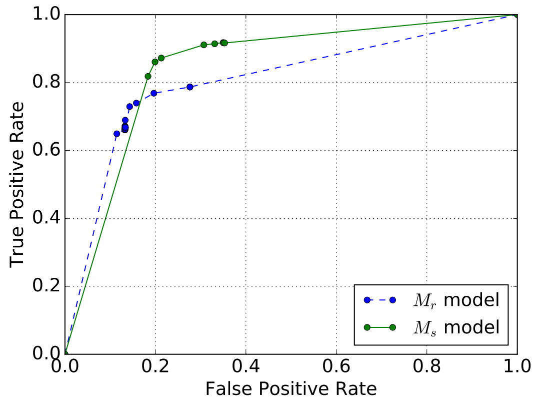

Figure 5: The ROC curves for the $\mathcal{M}_r$ and $\mathcal{M}_s$ Markov models.

Table 2: The number of physical accesses over time, with each row divided into three sections representing the malicious ground truth accesses, injected accesses, and valid (or non-malicious) accesses. The top row shows the total number of malicious ground truth accesses, injected accesses, and valid accesses. The second row shows the number of accesses deemed malicious by the $\mathcal{M}_r$ Markov model, and the third row shows the number of accesses deemed malicious by the $\mathcal{M}_s$ Markov model.

		Date (month/day)									
		15/7	**31/8**	**15/10**	**30/11**	**15/1**	**28/2**	**15/4**	**31/5**	**15/7**	**31/8**
Total	**Invalid**	57	67	63	21	27	35	35	39	60	45
	Injected	1067	1044	1134	1036	1215	1237	1292	1250	1159	1180
	Valid	9963	9122	9542	9138	10428	11174	11017	9687	9162	9136
M_r	**Invalid**	47	44	36	17	24	21	28	29	40	34
	Injected	756	762	832	749	884	923	902	875	830	858
	Valid	1145	642	881	742	985	1214	961	781	905	931
M_s	**Invalid**	49	49	38	18	25	27	30	29	38	37
	Injected	874	844	888	816	996	1011	1004	991	948	949
	Valid	1693	1066	1155	1188	1204	1659	1355	1096	1205	1308

Out of the 98,818 accesses, our $\mathcal{M}_r$ Markov model approach marked 9,187 as suspicious, whereas the $\mathcal{M}_s$ Markov model approach marked 12,929 as suspicious. Hence, either way, the practitioner's effort would have been reduced by over 85%. Table 2 shows the number of physical accesses over the ten testing subsets. We can see that a large fraction of the injected accesses and malicious ground truth data were detected as malicious. The numbers of false positives are low and fairly constant over the ten subsets.

On average, over the ten testing subsets, the $\mathcal{M}_r$ approach gives a false positive rate of 0.09 and a false negative rate of 0.28, while the $\mathcal{M}_s$ approach gives a false positive rate of 0.13 and a false negative rate of 0.20. In comparison, the baseline method had a false positive rate of 0.03 and a false negative rate of 0.35. Since the $\mathcal{M}_s$ Markov model incorporates more information about the user's normal movement behavior, its false negative rate is lower, since we can more easily detect deviations from that normal behavior. However, that drop comes at the cost of a slightly increased false positive rate. On average, about 30% of the users had a decreased false positive rate, with an average decrease of 0.35 and a maximum decrease of 1.0.

On the other hand, the baseline method has the lowest false positive rate, but at the cost of a higher false negative rate. In particular, we find that if an attacker carefully selects his or her path by moving only to previously visited rooms, the baseline method will not be able to identify any of those paths (i.e., its false negative rate will be 100%). In comparison, our solution can still raise an alarm if the path covers any unusual transitions among previously visited rooms. When the number of rooms visited by the attacker is 1, our false negative rate for the $\mathcal{M}_r$ Markov model is relatively high (i.e., 0.28) because a substantial fraction of the generated malicious paths are indistinguishable from legitimate paths that a user actually traveled before. In other words, since most of the generated paths are short, it becomes impossible in a certain fraction of cases to differentiate anomalous and normal movement behavior.

To study how the length of the attacker's path affects the performance of our approach, we varied the number of visited rooms in the path; the results are presented in Figure 6. The baseline method is still unable to detect any of these malicious paths, regardless of their lengths. In contrast, the probability that both of our Markov model approaches will detect the path increases as the path grows longer. As seen in Figure 6, the $\mathcal{M}_s$ Markov model performs better than the $\mathcal{M}_r$ Markov model.

We also varied the rate at which we injected malicious accesses into the testing data, and plotted the resulting false positive and false negative rates obtained from using the $\mathcal{M}_s$ Markov model. The plot is shown in Figure 7. We can see that the false positive rate increased almost linearly with the percentage of malicious accesses that were injected into the testing data.

For every point of injection into the data, A_iA_{i+1}, where A_i and A_{i+1} are accesses and the point of injection is between the two of them, the resulting sequence of physical movement accesses would be $A_iM_1 \ldots M_rA_{i+1}$, where $M_i, i \in \{1, \ldots, r\}$ are the injected malicious accesses ending in a room, $M_r = S_i \to R_m$. Now, if we consider the access $A_{i+1} = S_j \to S_{j+1}$, the previously visited room is now R_m, which is a random room in the station, according to our simulation strategy. Thus, the current state would be (R_m, S_{j+1}). The likelihood is low that this state belongs in the Markov model $\mathcal{M}_s$ and has a high transition probability from the previous state. Therefore, our detection algorithm will mark it as a malicious access. Then, for every point of injection into A_iA_{i+1}, the originally non-malicious access A_{i+1} would be marked as malicious instead of normal. Thus, the false positive rate would be correlated with the rate of injection of data.

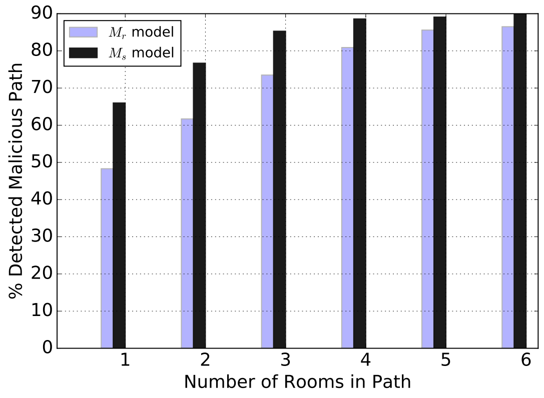

Figure 6: The percentage of detected malicious paths vs. number of rooms in the path for the $\mathcal{M}_r$ Markov model and the $\mathcal{M}_s$ Markov model.

On the other hand, the false negative rate decreases slightly as more malicious data are injected. In particular, the false negative rate for the first malicious access on the path, M_1, decreases, while the false negative rates for the second, third, and fourth malicious accesses, M_2, M_3, and M_4 (respectively), remain relatively steady. The more active an adversary is in moving around the station (i.e., the higher the injection rate of malicious accesses), the higher our chances of detecting the malicious movement early on.

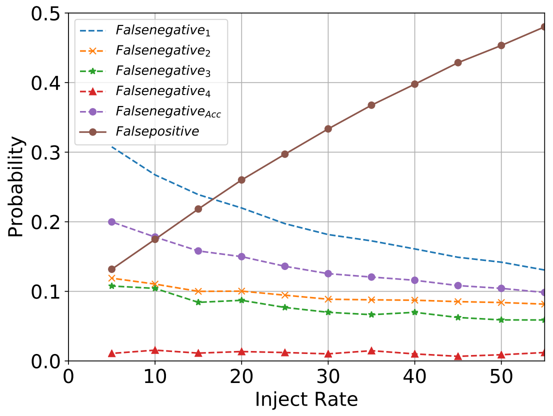

Figure 7: The false positive and false negative rates for the $\mathcal{M}_s$ model vs. the rate of injection of malicious accesses. The solid line represents the false positive rate, and the dashed lines are the false negative rates for the malicious accesses on the path and the total false negative rate.

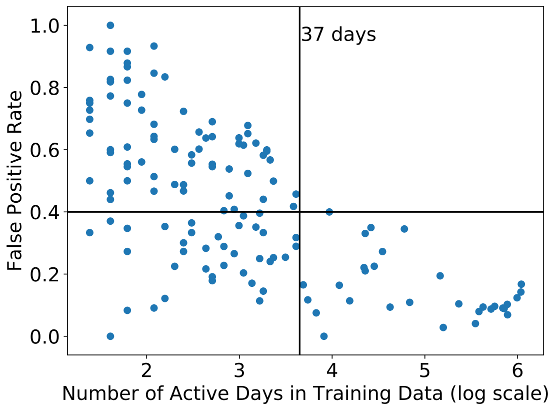

Figure 8: The false positive rates vs. the number of days that a user is active in the training data.

Finally, we determine the amount of training data that is required for the algorithm to construct an accurate model of normal movement behavior. We see in Figure 8 that the larger the number of days that a user is active in the training data, the lower the false positive rates obtained. More precisely, the upper bound on the false positive rate is seen to decrease linearly with an exponential increase in the number of distinct (but not necessarily consecutive) days of activity. In particular, the vertical line in the figure shows that 37 active days of data are needed to achieve at worst a false positive rate of 0.4. Thus, our detection system requires around a month's worth of active accesses for training, which, in practice, is a reasonable amount of data to obtain.

5.3.3 User Characterization

Next, we evaluate whether the entropy metric defined in Section 4 is suitable for characterizing user behaviors. If the entropy metric can differentiate among user behaviors, then the Markov models constructed for users with lower entropy (q_1) will have a better detection capability (lower false positive and negative rates) than those constructed for users with higher entropy. That would substantiate our decision to use Markov models for q_1 users, and in the next subsection, we evaluate the augmented Markov models for q_2 users and study how they improve on the base Markov model. We plot the entropy vs. false positive rates for the $\mathcal{M}_r$ Markov model in Figure 9.[4] Each point in Figure 9 represents a single user in one subset. One user should map to a maximum of 10 points in the plot.

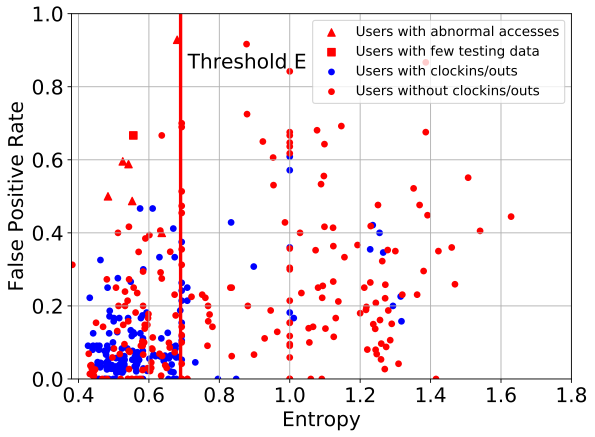

Figure 9: The distributions of false positive rates according to the entropy of users' movements. Blue points represent users with clock-ins and/or clock-outs, and red points represent users without clock-ins or clock-outs.

We can see that users whose entropy is below 0.7 have a lower false positive rate, on average, than users whose entropy is above 0.7. We also note that users with clock-ins and/or clock-outs have a lower entropy value in general and a lower false positive rate. The implication is that there is some correlation between the spatial and temporal regularity of movement, and thus we have corroborated our claims that our approach works better with users who have more periodic behavior, and that including temporal behavior in the model would likely yield limited improvement in the detection capability, as mentioned in Section 5.1.3. Thus, we can set our entropy threshold $\mathbf{E}$ to 0.7 in order to distinguish between the user types. Then, 29% of the users would belong to q_1 and 71% would belong to q_2. Although fewer users are categorized as q_1, these users account for 63% of the accesses.

However, several outliers show a high false positive rate for q_1-type users. We studied each of them individually and found that there were reasons why the accesses were marked as suspicious. The users represented by triangles in Figure 9 accessed rooms that they had not previously visited, and the users represented by squares had small testing sets (with fewer than 10 accesses), so their false positive rates are disproportionately high. In actual operation, the training data set and set of real-time accesses would be much larger, so there wouldn't be outliers.

5.3.4 Integration of Device State

In Section 4, we proposed to correlate logs documenting device state with physical access logs in order to decrease the false positives for the q_2-type users. Since the device-state logs in our case study were collected separately from the physical access logs, we have to assume that the timing information in the device-state logs and physical access logs is synchronized. The logs collected for different types of device (e.g., breakers or lights) are different, and thus the amount of information about the device's state that can be extracted varies. We extracted the textual description

[4]The results for the $\mathcal{M}_s$ Markov model are similar, and we therefore do not show them here.

and alarm values that indicated device failure and searched the device logs for failure incidents. In particular, there are four rooms in the station for which we could identify the failure of a device from the logs with particularly strong confidence. These rooms contain devices that control the environment in the station (e.g., air chiller and water pumps).

With the additional step of correlating device failures with physical accesses, we reduced the false positives for a subset of the users by an average of 0.45 for the four rooms. This preliminary result shows that we can use additional logs regarding the system environment to determine whether an access was malicious.

5.3.5 Online Detection

In Section 5.3.1, the running time of our algorithm is shown to be on the order of milliseconds. Since the complexity of the algorithm is constant, the algorithm is also scalable. Thus, our approach can work fast enough to process physical accesses in real time. In addition, we want to study how early on a malicious path of a certain length can be detected, because we want to analyze how effective our approach is at raising an alert in real time before an attacker reaches his or her destination room. We considered injected paths with a length of 4, and the false negative rates for the first, second, third, and fourth injected accesses in the path were 0.49, 0.3, 0.12, and 0.04, respectively, for the $\mathcal{M}_r$ Markov model and 0.32, 0.12, 0.09, and 0.02, respectively, for the $\mathcal{M}_s$ Markov model. This shows that our approach is able to detect malicious paths (with a certain minimum length) with high confidence, and even before an attacker has reached the destination room.

6 Discussion and Future Work

In this paper, we present the first step towards understanding how physical access logs can be used to enhance the detection capability of a system. In our case study of railway transit stations, we characterize two different types of user movement behavior. The first user type, q_1, performs well in terms of false positive rates. We also found some correlation between temporal and spatial behavior. We aim to explore this correlation in further detail. For the second user type, q_2, the false positive rates are much higher than q_1's. We have shown that using knowledge of the device states improves the false positive rates. However, many issues need to be resolved, such as time drifts between the device logs and physical access logs, differences in contextual understanding of diverse device logs, and missing data regarding device state. We intend to address these issues and pursue this line of thought in future work.

For this paper, we created movement models only for each user in isolation. Thus, we do not handle colluding insiders who may tailor their movements such that both parties remain within their movement patterns, but they are able to achieve their malicious goal together. We need to have a more comprehensive view of the system and user movements as a whole in order to tackle such adversaries. However, the data trace that we have consists of information on the card swipes made by users at doors, and information on whether each card swipe was legal (made with a valid card with the appropriate permissions) or an illegal access. That information tells us only that the user is likely to have entered a space, and it is not conclusive, since the user may have swiped at the door but decided not to enter the space. Since the building access control system offers only a limited view of a user's movement trajectory, it is hard to make claims about the relative location of a user with respect to other users, given the limitations of our data trace. We intend to tackle this issue in future work.

This paper also shows favorable results in using online-based detection in a real-world system. Our algorithm has constant time complexity, and the running times we obtained are on the order of milliseconds. That shows that our algorithm can be applied in real time to physical accesses. We also explore the ability of our algorithm to detect malicious behavior preemptively before an adversary reaches his or her destination. If we can detect a malicious physical access early on in a user's movement, we can make suitable responses to prevent a potential breach. For example, an administrator can temporarily remove a user's permissions to certain critical rooms, or place the user under further observation.

7 Conclusion

One way in which organizations address insider threats is through physical security. However, the state of the art in building access control is lacking. In this paper, we study the use of physical access logs for detecting malicious movement within a building. We propose a systematic framework that uses knowledge of the system and its users in order to analyze physical access logs, and we apply the framework to a real-world data trace of physical accesses in railway stations. First, we characterize users by using a set of metrics that take historical physical access data as input. In particular, we define a metric based on the approximate entropy of a time series that measures the regularity of user movement. Then, each user type is mapped to a behavior model, and the details of the model are learned through use of the users' past physical accesses. More specifically, we define two different Markov models, which represent different amounts of information about the users' movement sequences. Finally, we develop an online detection algorithm that takes the behavior model and the building topology as input, and returns a score indicating the likelihood that the user's access is anomalous. The results show that our proposed Markov models perform well in representing user movement behavior. In particular, inclusion of information about the path taken between rooms in the model helps the detection algorithm flag more malicious movement at the cost of a slightly increased false positive rate. The results also show that our framework can be applied in an online fashion to monitor and flag suspicious movement before an insider reaches the final destination room.

References

[1] M. B. Salem, S. Hershkop, and S. J. Stolfo. A survey of insider attack detection research. In Salvatore J. Stolfo, Steven M. Bellovin, Angelos D. Keromytis, Shlomo Hershkop, Sean W. Smith, and Sara Sinclair, editors, Insider Attack and Cyber Security: Beyond the Hacker, pages 69–90. Springer, 2008.

[2] Alien Vault. Insider threat detection software. https://www.alienvault.com/, 2016.

[3] Tripwire. Insider threat security & detection. http://www.tripwire.com/, 2016.

[4] CERT Insider Threat Center. Insider threat and physical security of organizations. https://insights.sei.cmu.edu/insider-threat/2011/05/insider-threat-and-physical-security-of-organizations.html, 2011.

[5] Matthew E. Luallen. Managing insiders in utility control environments. Technical report, SANS Institute, 2011.

[6] Lujo Bauer, Lorrie Faith Cranor, Robert W. Reeder, Michael K. Reiter, and Kami Vaniea. Real life challenges in access-control management. In Proc. ACM SIGCHI Conference on Human Factors in Computing Systems, pages 899–908, 2009.

[7] Alexander D. Kent, Lorie M. Liebrock, and Joshua C. Neil. Authentication graphs: Analyzing user behavior within an enterprise network. Computers & Security, 48:150–166, 2015.

[8] G. Pallotta and A. L. Jousselme. Data-driven detection and context-based classification of maritime anomalies. In Proc. 18th International Conference on Information Fusion, pages 1152–1159, 2015.

[9] A. N. Radon, K. Wang, U. Glasser, H. Wehn, and A. Westwell-Roper. Contextual verification for false alarm reduction in maritime anomaly detection. In Proc. IEEE International Conference on Big Data, pages 1123–1133, 2015.

[10] M. Dash, K. K. Koo, J. B. Gomes, S. P. Krishnaswamy, D. Rugeles, and A. Shi-Nash. Next place prediction by understanding mobility patterns. In Proc. IEEE International Conference on Pervasive Computing and Communication Workshops, pages 469–474, 2015.

[11] Miao Lin, Hong Cao, Vincent Zheng, Kevin Chen-Chuan Chang, and Shonali Krishnaswamy. Mobility profiling for user verification with anonymized location data. In Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15, pages 960–966. AAAI Press, 2015.

[12] Arpad Gellert and Lucian Vintan. Person movement prediction using hidden Markov models. Studies in Informatics and Control, 15(1):17–30, 2006.

[13] Christian Koehler, Nikola Banovic, Ian Oakley, Jennifer Mankoff, and Anind K. Dey. Indoor-ALPS: An adaptive indoor location prediction system. In Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing, pages 171–181, 2014.

[14] William Eberle and Lawrence Holder. Anomaly detection in data represented as graphs. Intelligent Data Analysis: An International Journal, 11(6):663–689, 2007.

[15] Michael Davis, Weiru Liu, Paul Miller, and George Redpath. Detecting anomalies in graphs with numeric labels. In Proc. 29th ACM Conf. on Information and Knowledge Management, pages 1197–1202, 2011.

[16] William Eberle, Lawrence Holder, and Jeffrey Graves. Detecting employee leaks using badge and network IP traffic. In IEEE Symposium on Visual Analytics Science and Technology, October 2009.

[17] Chuanren Liu, Hui Xiong, Yong Ge, Wei Geng, and Matt Perkins. A stochastic model for context-aware anomaly detection in indoor location traces. In Proc. IEEE 12th International Conference on Data Mining, pages 449–458, 2012.

[18] Robert P. Biuk-Aghai, Yain-Whar Si, Simon Fong, and Peng-Fan Yan. Individual movement behaviour in secure physical environments: Modeling and detection of suspicious activity. In Longbing Cao and Philip S. Yu, editors, Behavior Computing, pages 241–253. Springer, 2012.

[19] Mark J. Hoesl. Integrated physical access control and information technology security, 2014. U.S. Patent No. 6641090 B2, granted on June 17, 2014.

[20] Himanshu Khurana, Valerie Guralnik, and Robert Shanley. System and method for insider threat detection, 2014. U.S. Patent No. 8793790 B2, granted on July 29, 2014.

[21] Carmen Cheh, Binbin Chen, William G. Temple, and William H. Sanders. Data-driven model-based detection of malicious insiders via physical access logs. In Nathalie Bertrand and Luca Bortolussi, editors, Quantitative Evaluation of Systems, pages 275–291, Cham, 2017. Springer International Publishing.

[22] Graeme Baker. Schoolboy hacks into city's tram system. The Telegraph, January 11 2008.

[23] Shelby Grad. Engineers who hacked into L.A. traffic signal computer, jamming streets, sentenced. Los Angeles Times, December 1 2009.

[24] Carmen Cheh, Ken Keefe, Brett Feddersen, Binbin Chen, William G. Temple, and William H. Sanders. Developing models for physical attacks in cyber-physical systems. In Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy, CPS '17, pages 49–55, New York, NY, USA, 2017. ACM.

[25] Steven M. Pincus. Approximate entropy as a measure of system complexity. Proceedings of the National Academy of Sciences, 88(6):2297–2301, 1991.

[26] Xun Li. Using complexity measures of movement for automatically detecting movement types of unknown GPS trajectories. American Journal of Geographic Information System, 3(2):63–74, 2014.

[27] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-László Barabási. Limits of predictability in human mobility. Science, 327(5968):1018–1021, 2010.

[28] Patrick Billingsley. Statistical methods in markov chains. The Annals of Mathematical Statistics, 32(1):12–40, 1961.