PROTECTING PRIVATE DATA ON MOBILE SYSTEMS BASED ON SPATIO–TEMPORAL ANALYSIS

Sausan Yazji†, Robert P. Dick‡, Peter Scheuermann†, Goce Trajevski†
†EECS Dept., Northwestern University, Evanston, IL. 60208
‡EECS Dept., University of Michigan, Ann Arbor, MI. 48109
s-yazji@northwestern.edu, dickrp@eecs.umich.edu, (peters,goce)@eecs.northwestern.edu

Keywords: Mobile Security, Trajectory Analysis.

Abstract: Mobile devices such as smart phones and laptops are in common use and carry a vast amount of personal data. This paper presents an efficient behavior-based system for rapidly detecting the theft of mobile devices in order to protect the private data of their users. Our technique uses spatio-temporal information to construct models of user motion patterns. These models are used to detect theft, which may produce anomalous spatio-temporal patterns. We consider two types of user models, each of which builds on the relationship between location and time of day. Our evaluation, based on the Reality Mining dataset, shows that our system is capable of detecting an attack within 15 minutes with 81% accuracy.

1 INTRODUCTION

Mobile devices such as smart phones, iPhones, and laptops are used in a number of applications, including email, text messaging, gaming, web browsing, navigation, and recording pictures/videos [19]. Such devices are also used for financial transactions including Mobile Money [5], which is extensively used in China and Japan. Mobile computing devices store a lot of personal information and, if stolen, loss of control over these data may be even more important than loss of the mobile device.

Some prior work on mobile device security has focused on physical aspects and/or access control (e.g., strong passwords, voice recognition, or fingerprints). However, such approaches do not protect the private data on stolen devices in the post-authentication state. Many mobile devices (e.g., from Apple, Blackberry, Sony Ericsson, and Nokia) are equipped with location identification tools such as association with a cellphone tower ID, WiFi, Bluetooth, or Global Positioning System (GPS) receivers, which can be used to track location in case of theft. However, existing work that uses the GPS-feature for the purpose of protecting the users (e.g., GadgetTrak [1] and RecoveryCop [16]) depend on the owner to report the theft. It may take hours before the owner discovers the theft of a device, at which point private data may have already been violated. Even Laptop Cop [2], which has the goal of protecting data on stolen devices by remotely and manually deleting it, requires user intervention to initiate this process. In addition, these systems require cellular connections to protect the data, while our system is capable of detecting attacks and reacting without cellular access.

Our main goal is to develop efficient techniques for protecting data saved on mobile devices. Our approach is based on detecting the spatio-temporal behavior of intruders, which may be anomalous compared to the regular motion patterns of owners. In a previous study [23], we used network access patterns and file system activities to build a behavioral model that permitted attack detection with a latency of 5 minutes and an accuracy of 90%. We investigate the complementary approach of using spatio-temporal information and trajectory analysis to model user behavior and support anomaly detection.

There has been recent research [13, 18, 22] on mobility-based intrusion detection. To the best of our knowledge, ours is the first such technique to use spatio-temporal information and trajectory analysis to enable detection of an attack in 15 minutes and with 81% accuracy. The simple data structure used to model the users spatio-temporal behavior – 2-dimensional and 3-dimensional ma-
traces – enables efficient lookup-based attack detection.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 introduces the system architecture and detection techniques. Section 4 presents evaluation of our technique. Section 5 concludes the paper and indicates possible directions for future work.

2 RELATED WORK

Spatio-temporal data management and efficient query processing techniques have been the topics of intensive research in the field of Moving Objects Databases [11]. In particular, trajectory analysis and similarity detection have yielded numerous research results in the recent years [6, 9, 15]. Several results from this arena have goals similar to ours. For example, Mouza and Rigaux [7] propose regular expression based algorithms for detecting mobility patterns. However, those patterns do not explicitly model the temporal dimension of the motion, i.e., the focus is more on routes than trajectories. Hadjieleftheriou et al. [12] describe efficient indexing techniques and refinement algorithms for processing spatio-temporal pattern queries. The main distinction of our work is the use of probabilistic location-in-time patterns, which establish a threshold for detecting anomalous behavior.

The importance of adding semantic information to trajectory data has been previously recognized. For example, in order to improve application awareness during trajectory data analysis, Alvares et al. [4] proposed adding semantic information during trajectory preprocessing. Hung, Chang, and Peng [14] proposed the complementary approach of using a probabilistic suffix tree to measure separation among users trajectories. Xie, Deng, and Zhou [21] addressed the problem of predicting social activities based on users trajectories. In addition, Trestian et al. [20] used association rule mining to investigate the relationships between geographic locations and use habits for mobile devices. In this work, we introduce two types of mobility models and combine them for efficient detection of anomalous use.

Some intrusion detection research has objectives similar to ours, but differs in approach. Sun et al. [18] proposed mobile intrusion detection based on the Lempel–Ziv compression algorithm and Markov Chains. The proposed technique used three-level Markov Chains, and did not consider the association between time of the day and the location. Their ability to detect attacks using the proposed technique is limited to the times at which the user is making phone calls and moving faster than 60 miles per hour. Yan et al. [22] improved on this work, yet the delay in detecting attack was 24 hours, since the traces were obtained once a day, with a sampling period of 30 minutes. Our technique has an attack detection latency of 15 minutes. Hall, Barbeau, and Kranakis [13] proposed an intrusion detection method based on mobility traces. Their focus was on public transportation traces in which the paths are pre-defined. Their results are inapplicable for detecting attacks based on individual motion patterns.

3 SYSTEM ARCHITECTURE

We now explain the main results of our work. First, we explain our detection system. We then describe techniques for data collection and feature extraction and present two user models for anomaly detection.

The main objectives of this work are to

1. develop efficient algorithms for deriving user models from spatio-temporal information and trajectory analysis;
2. determine the accuracy with which users can be distinguished using such models; and
3. ideally achieve a high detection accuracy with low latency and low energy cost.

The methodology proposed in this paper is based on the following observations:

• most mobile systems have location identification tools and can gather location traces;
• each individual typically has a small set of locations that are visited with high frequency, e.g., every day [10]; and
• individuals tend to take the same paths when moving among particular locations [10].

3.1 System Components

The system for automatic generation of mobility models and detection of spatio-temporal behavioral anomalies has the following main modules:

1. data collection,
2. feature extraction,
3. user profile building, and
4. anomaly detection.
Figure 1: System architecture.

Figure 1 illustrates the integration of these modules into the system architecture, which consists of the following sub-systems.

- (ICS) – the information capturing system, residing on the mobile device, which contains an application to track the device location, register it periodically, and save it in a new log file every \(T\) minutes. It also contains the feature extraction module.

- (IMS) – the information management system, which collects the log-files from the ICS and resides on a computer with higher performance and much looser power consumption constraints than the mobile device. It is responsible for building mobility models and performing anomaly detection. Upon building the user model, the IMS sends it to the mobile device, allowing the detection of attacks in the absence of wireless connection, at some computation power consumption penalty.

- (RMS) – the response management system, which resides on both the mobile device and the remote server that hosts the IMS. Upon receiving an alert, the RMS identifies the appropriate action to protect data on the mobile device, e.g., notifying the device owner, locking the device, or automatically deleting private data.

In this paper, we focus on the algorithms and implementation details for the ICS and the IMS modules, since the RMS consists of user-dependent actions that should be executed in case an attack is detected.

### 3.2 Data Collection and Feature Extraction

Motion traces are essential for model construction and anomaly detection. We considered human motion data which is

- **continuous**: collected for a long period of time continuously;
- **consistent**: collected at the same time every day; and
- **frequent**: collected at a high enough frequency to support fast anomaly detection.

The sampling frequency used by González et al. [10] was too low for our application. The openStreetMap [3] data, as well as the data used by Rhee et al. [17], were neither continuous nor consistent. Hence, we used the Reality Mining data set [8], which contains data for over 100 users during a nine-month period. It consists of phone calls logs, locations identified by tower IDs and area IDs, application usage logs, and device-specific data. The data collection interval ranged from a few seconds to 15 minutes, with an average of 2.5 minutes, except when the mobile device was off.

Our spatio-temporal analysis techniques depend on extracting the following features from the Reality Mining log: (1) User ID \(u_i\); (2) Location information \(l_j\), represented by the area ID in the traces; and (3) Timestamps \(t_k\) of the data records in the trace. Thus, our input data records are tuples of the form \((u_i, l_j, t_k)\).

We developed and evaluated two modeling techniques for anomaly detection: Model #1 considers time–location relationships and Model #2 considers time–location sequences of recently visited locations. We relate the anomaly detection rate to the total number of distinct locations for each user, based on which we propose a method to adaptively select the best model.

In the next section, we describe each of the models in greater detail.

### 3.3 Model #1: Spatio-Temporal Information

In Model #1, for each user \(u_i\), we extract the location \(l_j\) and timestamp \(t_k\). For conciseness,
we will sometimes neglect notation for user ID when it is clear from the context.

### 3.3.1 Building User Profile

Our goal is to protect private data on mobile devices by detecting attacks based on identification of unacceptable deviation from the user’s normal behavior. Our first step is to behaviorally model each user’s normal behavior. To build the user profile for the 100 users in the reality mining data set, we divided the data evenly into two consecutive series: model data (used for model construction) and test data (used for evaluation).

Utilizing the model data, user profile was constructed as follows.

1. **Build a list of the user’s distinct locations** \((L_i)\).
2. **Extract from the distinct location list the user’s common locations set** \((UCL_i)\), which consists of locations the user visited more than 1% of the time during the data collection period.
3. **Construct the LOC-IN-TIME\(_i\) table for a 24-hour time period using one-minute intervals.** Each entry \(LOC-IN-TIME\(_i\)(j,k)\) is the weighted probability value \(\text{Prob}_i(l_j,t_k)\), which represents the fraction of time in the model data the user \(u_i\) was at location \(l_j\) at time \(t_k\), where \(1 \leq j \leq |UCL_i|\), and \(1 \leq k \leq NT\).

As explained above, \(UCL_i\) denotes the set of locations visited by \(u_i\) more than 1% of the time during the data collection period, and \(NT\) denotes the number of one-minute intervals.

At any given time \(t_k\), the user \(u_i\) should be at only one location \(l_j\) from the location list \(L_i\). Therefore the total probability value calculated for that time of the day should always be equal to one. The weighted probability value of \((\text{Prob}_i(l_j,t_k))\) is the probability of user \(u_i\) being at location \(l_j\) at time \(t_k\), divided by the number of records in the model data set that represent the locations in the \(UCL_i\).

The profile construction process is formally described in [Algorithm 1]. This process is repeated for each user, as shown in Line 3. The first step is to construct a list of all locations visited by user \(u_i\), as shown in Line 4. In Line 8, we calculate the weighted probability value. All locations that have been visited less than 1% of the time are excluded as explained in Lines 9–12. In Line 13, \(P_{trust}\) is calculated as described in Section 3.3.2.

### Algorithm 1: Build User Profile Based on Spatio-Temporal Information

1. **INPUT:** model data log
2. **OUTPUT:** user Profile LOC-IN-TIME
3. **for all** users \(u_i\) **do**
4. **Read each record in the model data log**
5. **Identify the list of distinct locations \((L_i)\) visited by the user**
6. **Allocate space for table LOC-IN-TIME\(_i\) with \(UCL_i\) columns and \(NT\) rows**
7. **Calculate the weighted probability value**
8. **Calculate the \(P_{trust}\) value for each user**

![Figure 2: User profile for Model #1.](image)

Figure 2 shows the profile for user \(u_i\). The user profile is a two-dimensional matrix with \((|UCL| \times NT)\) elements. Rows correspond to minutes of the day and columns correspond to locations.

### 3.3.2 Anomaly Detection

Attacks are detected via mismatches between limited-duration spatio-temporal traces and the model of normal user behavior, yielding an attack detection latency \(\leq T\). When the probability of a specific trace being generated by the user model drops below the \(Trust\ value (P_{trust})\), our system concludes that the mobile device is used by someone other than its owner.

To calculate the \(P_{trust}\) associated with a given user profile we used the test data set. We ran-
Algorithm 2: Detect Mobile Theft Based on Location Information

1: INPUT: LOC-IN-TIME
2: INPUT: User trajectory every T minutes
3: OUTPUT: Alarm in case of attack
4: Initialize the Trajectory Probability value \( TP \)
5: \( TP_i = 0 \)
6: for all \( t_j \) in the obtained trace do
7: if \( t \in UCL_i > 0 \) then
8: Get the probability value \( LOC-IN-TIME_i(t_j, t_k) \)
9: Calculate the cumulative probability value for the trace \( TP_i = TP_i + LOC-IN-TIME_i(t_j, t_k) \)
10: end if
11: end for
12: if \( TP_i \leq P_{trust,i} \) then
13: Trigger an alarm
14: end if

3.4 Model #2: Trajectory Analysis

The main feature of Model #2 is that it considers the probabilities of moves implicitly contained in the sequence of \((\text{time}, \text{location})\) points visited by the user in the model data. Conceptually, the user’s location–duration trace is divided into sequences, i.e., trajectories. Each trajectory consists of a start point, a number of intermediate points, and an end point, and may differ semantically due to the notion of stopping time \( STP \).

- **Stopping point (STP)** is the time interval for which the user is stationary. Based on observations from other researchers [21], we use \( STP = 30 \) minutes for all users.
- **Start point (SSP)** = \((u_i, l_j, t_k)\) is the first location identified in the sequence where \((t_k - t_{k-1}) \geq STP\).
- **Intermediate point (SIP)** = \((u_i, l_j, t_k)\) is a point in the sequence where \( t_{k-1} > t_k \) and \((t_k - t_{k-1}) \leq T\).
cumulative probability value as follows: the probability value for each trace rather than the however, for the Model #2 we calculate the joint user is similar to that described in Section 3.3; The computation of trust values (3.4.2 Anomaly Detection

3.4.1 Building User Profile
During user profile construction, the Model #1 feature extraction technique is used (see Section 3.3) and the list $UCL_i$ is constructed as described in Algorithm 1. However, for Model #2, the user profile is a three-dimensional table $LOC\_TIME\_MOVE$ of size $(|UCL| \times |UCL| \times NT)$. Each entry in this table, $LOC\_TIME\_MOVE_i(j, j_1, k, k_1)$, represents the probability of the user $u_i$ moving from location $l_j$ at time $t_k$ to location $l_{j_1}$ at time $t_{k_1}$.

Similarly to the corresponding structure used in Section 3.3, each entry $LOC\_TIME\_MOVE_i(j, j_1, k, k_1)$ represents the weighted probability of $Prob(l_j \rightarrow l_{j_1}, t_k \rightarrow t_{k_1})$. Figure 4 presents an example of a trace of sequence information. Figure 5 shows the user profile data structure.

3.4.2 Anomaly Detection

The computation of trust values ($P_{trust}$) for each user is similar to that described in Section 3.3 however, for the Model #2 we calculate the joint probability value for each trace rather than the cumulative probability value as follows:

$$SP_m = \prod_{(j, j_1, k, k_1) \in m} LOC\_TIME\_MOVE_i(j, j_1, k, k_1).$$

(2)

The joint probability value is the product of the probabilities of all records in the trace, as indicated in the $LOC\_TIME\_MOVE$ table. Equation 2 indicates that if any record in the sequence has a probability of zero, which indicates that the user has never been at that location at that time, the trace will be considered an attack. To reduce the penalty of deviation from the normal path, we introduce the concept of Trace Threat Level (TL), which represents the percentage of the sequence that has no representation in the user profile. Thus, if $LOC\_TIME\_MOVE_i(j, j_1, k, k_1) = 0$, we eliminate this value from the calculation of the trace joint probability value, and increase the threat level value by one. We use a threat level threshold of $TL_{trust} = 10\%$ of the total records in the trace, based on empirical analysis.

As an example, Figure 6 shows two paths. The solid curve represents the normal path in the user’s profile and the dashed curve represents the currently detected trajectory. In this example, the user profile indicates that when the starting point at time $t$ is location $B$, the normal path of duration $T$ is $B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$. In contrast, the captured user trajectory that starts at location $B$ at time $t$ consists of the sequence $B \rightarrow A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$. To determine whether this is an expected or anomalous user behavior, we compare the calculated probability of this path with the profile of the particular user. The calculated value should be equal to or greater than the trust value for that user.

To calculate the captured trace joint probability $TP$, we first identify the starting point $SSP = l_j$ and the time $t_k$. Then we check whether $l_j \in UCL_i$ or not. If not, we increase the value of the threat level TL. Otherwise, we identify the next location $l_{j_1}$ at time $t_{k_1}$. If $l_{j_1} \in UCL_i$, we obtain the joint probability value $LOC\_TIME\_MOVE_i(j, j_1, k, k_1)$. If not, we increase the TL value again. This process is repeated for the entire sequence and, upon completion, if $TL \geq TL_{trust}$, this sequence is judged to
Algorithm 3: Detect Mobile Device Attack Based on User Trajectory

1: INPUT: LOC-TIME-MOVE_i
2: OUTPUT: User trajectory every T minutes
3: Initialize the Trace Probability ($TP_i$) and Trace Threat Level ($TL_i$) values
4: $TP_i = 1$, $TL_i = 0$
5: for all n records in the sequence do
6: Read $l_j$ at time $t_k$ and $l_{j1}$ at time $t_{k1}$
7: if ($l_j$ and $l_{j1}$) $\in$ UCL_i > 0 then
8: calculate the joint probability value $TP_i = TP_i \times LOC$-TIME-MOVE_i($j,j1,k,k1$)
9: else
10: $TL_i = TL_i + 1$
11: end if
12: end for
13: Check for anomaly
14: if ($TL_i \leq TL_{trust,i}$) and ($TP_i \geq P_{trust,i}$) then
15: Continue
16: else
17: Trigger an alarm
18: end if

4 EXPERIMENTAL RESULTS

We now describe the experimental setup and present the results from the evaluation of our techniques.

As discussed in Section 3, we used the Reality Mining mobility traces of students and staff at a major university. The traces had the following sources: 60% graduate students, 27% incoming students at the university’s business school, and 8% staff. The number of distinct locations per user ranges from 1–100, with an average of 28. We eliminated single-location users and those with fewer than 1,000 records (i.e., 3.5 records per day) because it was not possible to build models for users with very few records, leaving 93 users.

Each user log was divided into training (model_data) and testing (test_data) portions as described in Section 3.3. For each user, we randomly selected 100 duration T samples from the test_data log. We repeated each test for four different T values (5 min, 15 min, 30 min, and 60 min). The T value is the attack detection latency.

4.1 Results for Spatio-Temporal Model (Model #1)

For each of the 93 users, we constructed models and calculated trust values $P_{trust,i}$ following the steps described in Section 3.3. Attacker behavior traces are not presently available. However, traces for different users are available. We evaluated the probability of detecting the anomalous mobility patterns of other “Reality Mining” study participants.

The limited number of locations, and the fact that around 68% of the study participants worked in the same set of locations (buildings), but different traces for different users are available. We evaluated the probability of detecting the anomalous mobility patterns of other “Reality Mining” study participants.
different rooms and floors (lab, library, office, etc.) made this a challenging dataset for motion-based anomaly detection. Using the area ID rather than the cellphone tower ID during feature extraction was necessary to enable this study. Figure 7 shows that users sharing the same locations in their profiles can have very different probability distributions over a 24-hour period. Figure 7(a) shows the probability distribution for user \(u_1\) and location ID = 1, while Figure 7(b) shows the probability distribution of the same location ID = 1 over 24 hours for the user \(u_{74}\).

We calculated the \(TP_{m,y}\) value for all 100 test samples for each user \(u_i\), where \(x \neq i\) and \(1 \leq m \leq 100\). Subsequently, we calculated the \(FAR_y\) value that represents the percentage of the test samples for which the total probability value is \(TP_{m,y} \geq P_{\text{trust},i}\). Figure 8 illustrates the \(TP_{m,y}\) results for a randomly selected user \(u_{30}\). Specifically, Figure 8(a) shows the probability distribution of \(TP_{m,30}\) where \(y = i = 30\). Figure 8(b) shows the probability distribution of \(TP_{m,y}\), where \(y \neq i\) and \(i = 30\). We observed that for user \(u_{30}\), only 5% of the samples have \(TP_{m,30} \leq 0.02\), while more than 80% of the samples for each of the other 92 users have \(TP_{m,30} \leq 0.02\).

Figure 9 illustrates the ability to distinguish the behavior of a given user \(u_i\) from that of the other 92 users, given 100 samples each. The accuracy is \((100 - FAR_y)\) when \(T=5\) minutes.

For example, Figure 11(a) shows that the average accuracy of Model #1 is in the same range (76.08%–76.6%) for all sample sizes. Therefore, we conclude that the sample size does not have a large impact on attack detection accuracy for Model #1. Figure 11(b) shows a standard deviation above 20%, which is also clear from Figure 9, in which detection accuracy for some users was 100% (e.g., users \(u_{22}, u_{27}, u_{75}\), and \(u_{84}\) ) and in which others have detection accuracies ranging from 9%–47% (e.g., \(u_{10}, u_{20}, u_{47}, \) and \(u_{88}\) ). High accuracy is possible for users with few distinct locations (3–8). Accuracy is low for users with many distinct locations (69–100). Section 4.3 provides more details.

4.2 Results Based on Trajectory Analysis (Model #2)

We followed the same steps described in the previous section to calculate \(FAR_y\) values. Figure 10 shows the results of the trajectory analysis for different test sample lengths. In Model #2, detection accuracy is affected by test sequence length, with \(T = 15\) minutes yielding the highest accuracy and \(T = 60\) minutes yielding the lowest accuracy (see Figure 11(a)). Lower \(P_{\text{trust}}\) values are associated with the longer traces, which indicates that it is uncommon for normal users to make large day-to-day changes in motion patterns affecting short intervals within a trace. However, longer intervals are more likely to change from day to day.

4.3 Model Comparison

As illustrated in Figure 11(a), the average accuracy is slightly better for Model #2 than for Model #1 for small sample intervals (less than 30 min). However, the standard deviation is significantly better, as shown in Figure 11(b). It can be observed that there is improvement in accuracy for users with many distinct location and degradation in accuracy for users with few dis-
distinct locations. Hence, Model #1 is more accurate in the cases when the users have few distinct locations and Model #2 is more accurate for users with many distinct locations. Thus, a combined approach might be useful.

$NL$ is the discrete location count threshold at which Models #1 and #2 have equal accuracies. If the size of $UCL_i \leq NL$, then Model #1 should be used. Otherwise, Model #2 should be used. To determine $NL$, we tested a combined approach with several values $(5, 6, 7, \cdots, 30)$, where 28 is the average number of distinct locations in the data set. Figure 12 illustrates the average accuracies for each $NL$ value depending on time $T$. $NL = 10$ allowed the highest accuracy: 80.59% when $T = 15$ minutes. Therefore our recommendation is to use a combined approach to permit

- faster detection if there are few distinct locations and
- lower energy consumption due to decreased calculations.

5 CONCLUDING REMARKS

We presented an approach for detecting anomalous use of mobile devices. Our system uses spatio-temporal mobility data to build models that have high anomaly detection accuracy. Combining the spatio-temporal model (for users with few locations) and trajectory-based model (for users with many locations) allowed an average attack detection rate of 81%, with a latency of 15 minutes. The simplicity of the resulting user models resulted in an efficient anomaly detection process supporting an average detection time 0.02 seconds, as shown in Figure 13. A comparison between our results and those of existing systems is given in Table 2.

In the future, we plan to expand this study to cover additional mobile computing data sources such as phone and application logs in order to determine the change in detection accuracy when more user-specific data are acquired.

Acknowledgments: We thank Prof. Ruoming Jin for his insightful comments. This work was supported in part by the National Science Foun-
Anomaly detection elapsed time according to sample interval.

REFERENCES


