Implicit Authentication for Mobile Devices

Markus Jakobsson Elaine Shi Philippe Golle Richard Chow
Palo Alto Research Center
{*mjakobss, eshi, pgolle, rchow*}*@parc.com*

Abstract. We introduce the notion of *implicit authentication* – the ability to authenticate mobile users based on actions they would carry out anyway. We develop a model for how to perform implicit authentication, and describe experiments aimed at assessing the benefits of our techniques. Our preliminary findings support that this is a meaningful approach, whether used to increase usability or increase security.

1 Introduction

Mobile commerce (M-commerce) is experiencing rapid growth, and there is a trend toward hosting of applications and services on the web. This results in increased demand for authentication – whether of computers or users. In particular, higher-assurance authentication, such as through augmenting passwords with SecurID-like devices, has become standard in the enterprise and is beginning to penetrate high-value consumer markets, such as banking. The concern with these second-factor authentication devices has been their usability and cost. Another factor in the new authentication landscape is the greater market penetration of Mobile Internet Devices (MIDs), which complicate password entry due to the limitations of their input interfaces. We conducted a survey of 50 iPhone, BlackBerry, and gPhone users. The survey showed that 40% of users enter a password on a daily basis, and that 56% mistype a password at least one time in ten. Users find password entry on MIDs more annoying than lack of coverage, small screen size, or poor voice quality. Therefore, MIDs give rise to a need for authentication techniques without active user involvement, or with very limited user involvement.

In this paper we propose using observed user behavior to authenticate users, an approach we refer to as *implicit authentication*. Implicit authentication can be used to meet the following general authentication needs: 1) Used as a secondary factor for authentication, implicit authentication can augment passwords to achieve higher-assurance authentication in a cost-effective and user-friendly manner. 2) Used as a primary method of authentication, implicit authentication can replace passwords altogether, relieving users from the burden of entering passwords. 3) A third use of the technology is to provide additional assurance for credit card transactions, based on the security posture of the device owner. (We note that it is not necessary for the payment transactions in question to involve the device used to collect user data, or even to involve the user when collecting the data; they would also not need to be put on the critical path for performing the payment, but simply be used as an auxiliary fraud indicator.)

Implicit authentication can be implemented for any kind of computer, but is particularly suitable for portable computers – these are often characterized by a combination of text input constraints and access to rich information. For example, for MIDs the input data typically includes location, motion, phone activity, and other application activity. Medical devices often have input constraints similar to MIDs and are used to access and manipulate centrally maintained patient records. In spite of HIPAA requirements, password and account sharing is common for such devices. The ability to authenticate a person implicitly – and to log a person out due to a failed implicit authentication – can help maintain high security and privacy of patient records even under stressful conditions where account sharing would otherwise be very tempting. Implicit authentication finds applications to secure military equipment as well, where usability is as important as securing devices against enemy take-over. Finally, out-of-band transaction verification is another natural application. For example, when a consumer pays for an item with a credit card, his phone location history can be queried silently to see whether the recent use of the phone is consistent with the card use, including, of course, whether it is near the point-of-purchase or – for online transactions – at a location consistent with the observed IP address. For the sake of concreteness, we focus our study on MIDs herein.

We introduce and evaluate techniques to compute and

maintain an *authentication score* for each user, based on the recent activities of the user. This is based on identifying positive events, such as common habits, and boosting the score when a habitual event is observed; and on detecting negative events and degrading the score when these are observed. A negative event is one that is not commonly seen for a user, or which is associated with a common type of attack. *Time* is also a negative event in the sense that scores degrade over time. When the score falls below an event-specific threshold, that event can no longer be performed without the user first having to explicitly authenticate herself, e.g., by giving a password. Correct explicit authentication is a highly positive event, and a failed explicit authentication is a negative event. We describe an architecture that supports implicit authentication, and report on findings from preliminary experiments. Our findings support that this is an approach with great potential for use with devices with rich inputs.

Related work. Single Sign-On (SSO) addresses the problem of much-too-frequent authentication requirements, but – after a first password entry – only vouches for the identity of the *device*, and not its user. Therefore, SSO does not defend against theft and compromise of devices well, and does not address voluntary account sharing at all.

In a study of users' perceptions of authentication on mobile devices, Furnell et al. [5] showed that users want a solution to authentication that increases security, provides transparent authentication and "authenticates the user continuously/periodically throughout the day in order to maintain confidence in the identity of the user". The study found users receptive to the use of biometrics and other behavioral indicators, but not receptive to security tokens.

Some biometric authentication methods, notably keystroke dynamics and typing patterns (e.g., [10–12]) are *implicit* in the sense that they continually observe the user behavior and make authentication decisions based on the observations. Recently, Chang et al. used accelerometers in television remote controls to identify individuals [3]. Kale et al. [9] and Gafurov et al. [6] used gait recognition to detect whether a device is being used by the owner. There has also been a lot of work in combining several biometric inputs to produce an aggregate authentication score [1, 2]. In location-based authentication and access control [4, 14], the subject's location is used to decide whether the subject should be allowed to access a certain resource. Greendstadt and Beale [7] noted the need for "cognitive security" for personal devices. Specifically, they proposed a multi-modal approach "in which many different low-fidelity streams of biometric information are combined to produce an ongoing positive recognition of a user". Our efforts are a step in the direction of realizing the vision laid out in [7].

Centralized analysis of data collected on resource-constrained devices is also beneficial in the context of defending against malware [8, 13], and it is likely that there will be functional overlap between an implicit authentication engine and a centralized anti-virus component. Whereas both security technologies benefit from collection and centralized analysis of data from mobile devices, and some of this data could be obtained from carriers and service providers, it is important to note that they do not in any sense require a departure from net neutrality. In other words, the technologies can be implemented in a way that makes them independent of the selection of provider of connectivity, software, and hardware.

2 Adversarial Model

We consider adversaries that acquire physical access to the device, such as a family member or co-worker attempting to access the device or associated resources, or a stranger who steals or finds the device and tries to monetize it and its information.

Malware can also pose a threat. One potential attack by malware is the cloning attack: the malware sends all information on the infected device to a colluding host on the network, and then – remotely, and with no further access to recent information – attempts to create an event to be accepted. We can defend against cloning attacks by having packets signed by a SIM card which is hard to clone. Persistent malware on a device can perform more powerful attacks, as it is able to control and observe all events, and is able to mimick the user's behavioral patterns. By logging keystrokes, malware can also obtain a user's password to `bank.com` and thus acquire access to the user's online banking account. Malware defense is outside the scope of this paper, and should be seen as an orthogonal issue. We refer readers to [8] for a treatment of mobile malware defenses.

3 Data and Architecture

The data sources used to make authentication decisions can be grouped into three classes: *data primarily available on the device*, *data available to the carrier*, and *data available from other providers.* Some data may belong to more than one class; and some data are device specific, depending on the type of hardware and the type of use.

Device data. Modern mobile devices provide rich sources of data for implicit authentication: 1) Location and co-location data from GPS coordinates possibly augmented by accelerometer measurements. WiFi/Bluetooth connections and USB connections indicate co-location. Moreover, successful authentication to a known access point or sync with a known PC is a strong positive indicator. 2) Application usage, such as brows-

ing patterns and software installations. 3) Biometric-style measurements, such as keyboard typing patterns and voice data. Touch-screen devices might also be able to fingerprint users, or at least measure the size and shape of any portion of fingers in contact with the screen. Future devices may also have user-configured auxiliary sensors, e.g., to monitor the user's pulse, temperature and blood pressure. 4) Contextual data, such as the contents of calendar entries, the current time of day, day of week, etc.

Carrier data. Carriers know users' approximate location, as identified by the selection of cellular tower. Carriers also know the users' phone call patterns and may also have voice data. For users who access the Internet through the cellular network, the carrier may also know their Internet access patterns, for example, by examining their DNS requests and network packets in general.

Cloud data. An increasing number of applications are hosted on the network, and have usage information of relevance – whether this amounts simply to what applications were used and when, or the data content, such as calendar entries.

System architecture. We now explain possible architecture choices and discuss their pros and cons. The authentication decision may be made by the mobile device, by the carrier or by another trusted third party. Consumers of the authentication decision (or score) include 1) the mobile device (e.g., to decide whether a password is necessary to unlock the device or use a certain application); 2) a service provider that wishes to authenticate the user (e.g., an online banking website or a cashier's register if the device is used as an electronic credit card.)

The mobile device can make authentication decisions locally to decide whether a password is necessary to unlock the device or use a certain application. The device's SIM card can also sign the authentication decision (or score) and send it to a service provider. This approach protects the user's privacy This approach protects the user's privacy against cheating servers, but not theft and corruption of devices. Namely, if the device is captured, an attacker may be able to obtain the data stored in the memory and learn the user's behavioral patterns. As mobile devices are battery-constrained, we also need to ensure that the authentication score is fast to compute.

Carriers are well-suited to be the trusted third party in charge of making authentication inferences and communicating trust statements to qualified service providers. However, it is also possible for carriers simply to provide data to third parties entrusted with the analysis of data and the making of authentication decisions. In either case, participation of carriers is simplified by the fact that they already have established a trust relationship with the consumer, and because of their natural ability to communicate with the consumer devices. It is possible to enroll devices that cannot be used to sense or report events by relying on carrier data, and – based on consumer opt-in – on data from network service providers.

Network service providers are not only producers of data, but also consumers of trust statements. They would make decisions on whether to require an explicit authentication or not based on such statements, and – in the case when an explicit authentication session is required – feed back the outcome of this step to the trusted third party. It is possible for a network service provider to play the role of the trusted third party, given client-side software that reports information, using the carrier simply as a conduit of information.

Privacy. If we adopt an approach where data is reported to a trusted-third party, users' privacy will be a concern. The following approaches are possible for enhancing user privacy: 1) removing identifying information (such as names or phone numbers) from the data being reported; 2) using a pseudonym approach, e.g., "phone number A, location B, area code D"; 3) using coarse-grained or aggregate data, e.g., reporting a rough geographic location rather than precise GPS coordinates, and reporting aggregate statistics rather than full traces.

4 Learning Framework

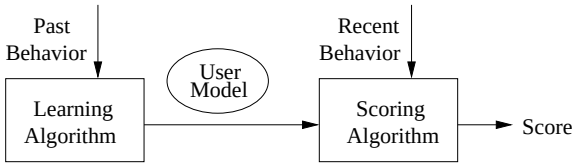

Figure 1: Architecture.

Figure 1 outlines the framework of the machine learning algorithm. We first learn a *user model* from a user's past behavior which characterizes an individual's behavioral patterns. To make an authentication decision in real-time, a *scoring algorithm* examines the user model and the user's recent behavior, and outputs a score indicating the likelihood that the correct user is using the device. The score is used to make an authentication decision: typically, we can use a threshold to decide whether to accept or reject the user, and the threshold can vary for different applications, depending on whether the application is security sensitive. The score may also be used as a second-factor indicator to augment traditional password-based authentication.

4.1 Modelling the User

The user model should characterize the user's behavioral patterns. For example, how frequently the user typically makes phone calls to numbers in the phone book, where the user typically spends time, etc. The user model may

also consider combinations of different indicators. For example, given that the user is in her office and has received a call from number A, then with 90% probability, she will send an email to address B within the next 10 minutes.

First step: an independent feature model. We now describe a naive model where we assume independence between different categories of activities. In other words, we assume that the user's phone call pattern is independent from her location, browser usage, and other activities. We assume that the user's behavior depends on the time of day and day of week. For example, one user might place and receive frequent phone calls in the afternoon, but might not place any phone calls between 6 pm and 10 pm, and only receive phone calls for the first half of that interval.

Let $V_1, V_2, \ldots, V_k$ denote k independent random variables, also referred to as *features*. Example of features include: V_1 = *time elapsed since last good call*, V_2 = *inter-arrival time between bad calls*, V_3 = *GPS coordinates*, etc. In the above, a good call is one made to a number in the phone book or a number that has been called in the past, and a bad call is one made to a number that has never been seen before.

A user model is the product of k probability density functions conditioned on the variable T = (*time of day, day of week*):

$$\text{user model} := \big[p(V_1|T), p(V_2|T), \ldots, p(V_k|T)\big]$$

The learning algorithm in Figure 1 basically estimates these density functions, thereby forming a user model.

4.2 Scoring Algorithm

Given a user model and reported recent behavior, the scoring algorithm outputs a score indicating the likelihood that the device is in the hands of the rightful owner.

Scoring independent features. We now describe a potential way to design the scoring function under the independent feature model.

A user's recent behavior may be described by a tuple $(t, v_1, v_2, \ldots, v_k)$, where t denotes the current time, and $v_1, \ldots v_k$ denote the values of variables $(V_1, \ldots, V_k)$ at time t. The idea is to compute a separate score for each feature, and then use a function f to combine these separate scores into a final score. Basically, we will have k scoring functions denoted $S_1, S_2, \ldots, S_k$. Let $1 \leq i \leq k$. Given the probability density distribution $p(V_i|T)$ and an observed value v_i, the i-th scoring function S_i outputs a score s_i for this feature.

As an example, consider how one might design a scoring function for V_1 = *time elapsed since last good call*. The idea is that the score should decay over time during periods of inactivity. However, the rate of decay depends on the time of day and day of week. If a user typically makes frequent phone calls in the afternoon, then the score should decrease faster in the afternoon during periods of inactivity. By contrast, if the user typically makes no phone calls between 12am and 8am, then the score should decrease more slowly over this period of time. One potential candidate for the scoring function is as follows. Let v_1 denote the lapse since the last call at time t. Let $F(x|T = t) = Pr(V_1 \leq x|T = t)$ denote the cumulative distribution of variable V_1 for time t. Define the score to be the probability that a lapse of v_1 or longer is seen at time t:

$$S_1(v_1) = 1 - F(v_1|T = t) \tag{1}$$

The scoring function for user location may assign to a location visited at a certain time of day a score which is inversely proportional to the distance to the nearest location cluster associated with this time of day. For example, a user who is typically at a "work" cluster during working hours and at a "home" cluster at night would receive the highest score for being located at the expected cluster at the expected time. Locations near expected clusters would receive partial credit that decreases to zero as the distance to the cluster increases.

Learning the scoring function. Given the separate scores for k different features, we call $f(S_1(v_1), \ldots, S_k(v_k))$ to compute the final score. For example, suppose that each score $S_i(v_i)$ $(1 \leq i \leq k)$ is the probability of v_i, i.e., $S_i(v_i) = \Pr[V_i = v_i]$ or $S_i(v_i) = \Pr[V_i \geq v_i]$ as in Eq(1). Then a natural way to combine the scores is to compute the joint probability of $(v_1, \ldots, v_k)$. As we assume independence between features, the final score would be the product of these probabilities: $f(s_1, \ldots, s_k) = s_1 \cdot s_2 \cdot \ldots \cdot s_k$.

Another potential design for f is a weighted sum:

$$f(s_1, \ldots, s_k) := w_1 s_1 + w_2 s_2 \ldots + w_k s_k$$

The weights $w_1, \ldots w_k$ should be determined through a training process.

To learn the scoring function, assume that we collect activity data for m individuals. This set of data is divided into a training set and a test set. The training set will be used as postive examples in the training process. We synthetically generate negative examples (attack data) for training. In particular, we use a splicing method to create negative examples. If person A and person B appear in the vicinity of each other at time t, then we splice the data for A before t and the data for B after time t. This models an attack where B picks up or steals A's mobile phone and starts using it on her own. In reality, B could be a friend, colleague, acquaintance or a stranger.

Now, training the weights $w_1, \ldots, w_k$ can be expressed as a minimization problem: suppose we fix the false negative rate, e.g., we say that a legitimate user is

denied access and has to enter her password at most once a day on average. Our goal is then to minimize the false positive rate (failure to detect an attack) and the time till detection in the presence of an attack.

5 Data Collection and Initial Experiments

Data collection. There are three mainstream mobile platforms to consider: the iPhone, Android devices, and Symbian devices (e.g., BlackBerry). We chose to perform experiments to validate our approach using the BlackBerry platform, which supports multi-threading.

We ran an experiment in which we recorded a list of actions and events for BlackBerry equipped subjects. Information relating to the following types of events was recorded: emails, calls, SMSs, location, contacts, calendar, tasks, memos, alerts, battery level, (un)holstering, USB connections, power on/off, SD card removal/insertions.

In more detail, we recorded the following email activity: New email detection, opening/closing email messages, adding/removing email messages to/from folders, creation and sending of new email messages, types and file extensions of email attachments. For call activity, we recorded the date, duration, notes, and the status of calls. We recorded sending and receiving of SMSs. The location was saved with 5-minute intervals. We recorded the addition, removal and editing of contacts, calendar events, task list items and memo pad entries. We also recorded alerts started and stopped – e.g., calendar appointment warnings – and battery level changes (high, medium, low).

Data analysis. Whereas a full-blown system would use all detectable events to make authentication decisions, we started by examining a subset of them for the sake of simplicity. Below we report some initial findings of the analysis we performed on phone data and location data.

We collected three months of phone data for each participant in the study, and studied their call patterns. We find that their call history exhibit distinct patterns depending on the time of the day. For example, between 12 am and 4 am over a period of 3 months, one participant made only a small number of calls to 3 different numbers. On the other hand, between 12 pm and 4 pm, the same participant made a total of 267 calls to 44 different numbers, averaging to about 3 calls per day during this time period. Among the 267 calls, about a half of them were made to a family member of the participant. The second most frequent number which turned out to be a collaborator of the participant was called 17 times. Both the numbers called and the call patterns authenticate users – in Figure 2, we plot the cumulative distribution of the time elapsed since the previous call for the individual described above. The two curves in the figure represent 3 pm and 5 am respectively. The figure indicates that around 3 pm, the typical (median) lapse since last call is 108 minutes. By contrast, the typical (median) lapse at 5 am is around 521 minutes. The patterns vary for different individuals. For example, another individual in the study made only 60 calls between 12 pm and 4pm over the course of 3 months, averaging about 0.67 calls per day. The score for the latter individual should therefore decay more slowly than the former individual during this period of time.

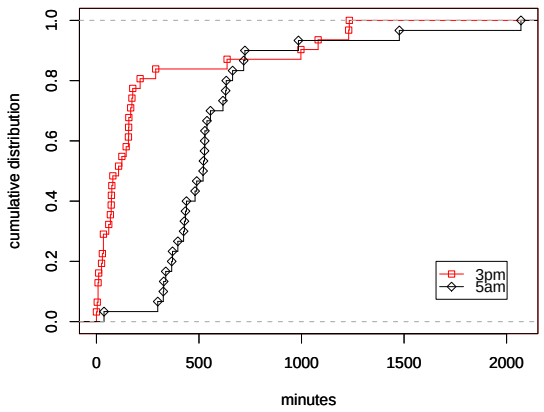

Figure 2: CDF of lapse since previous call.

We also analyzed the location traces for participants in the study. We used the interactive clustering algorithm of [15] to compute clusters of the most frequently visited locations. The clusters are parametrized by the maximum distance between points within a cluster, and by the minimum number of points within a cluster. We experimented with different values for these parameters. For a trace representing a few days' worth of activities with locations recorded every 5 minutes, we found that a maximal distance of 1,000 meters and a minimum of 20 points per cluster successfully produced a small number of clusters corresponding to where the user lives, works and shops (see Figure 3).

Scoring algorithm. In a full-blown system the score will combine all features collected. In our preliminary analysis, we design a scoring algorithm for phone data combining two features: V_1 = *time elapsed since last good call*, and V_2 = *number of consecutive bad calls*. For example, if the most recent call is a good call, then $V_2 = 0$. If the most recent three calls are good, bad and bad respectively, then $V_2 = 2$.

Figure 4 plots the authentication score for one individual over a period of one day. The x-axis represents the time of day expressed in hours 0-23. The y-axis represents the score, which is a value between 0 and 1. Every time the individual makes a call to a known good num-

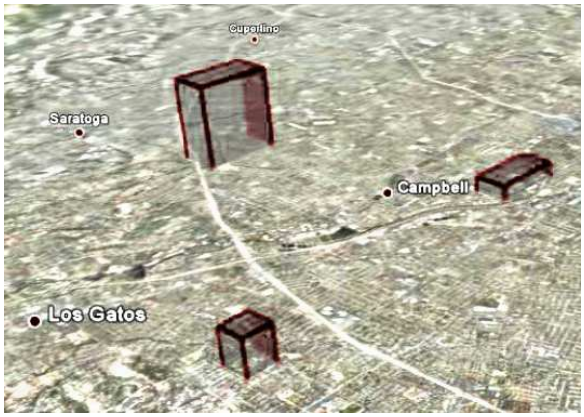

Figure 3: Locations clusters computed from a user's GPS trace. The height of the cluster is proportional to the time spent within the cluster.

ber, the score goes up to 1. As marked by red "X" in the figure, the score is decreased whenever a bad call is made (i.e., a call to an unknown number). During silent periods, the score decays over time. Specifically, the figure shows that the score decays faster in the afternoons than at night. For example, during a silent period between 7 pm and 9 pm, the score decays fast. By contrast, during a silent period between 12 am and 5 am, the score decays very little. This is because this user typically makes more phone calls between 7 pm and 9 pm, but typically does not make any calls between 12 am and 5 am. The small local oscillations are due to an artifact in the scoring function defined in Eq(1). Eq(1) does not guarantee that the score is strictly monotonically decreasing. It is possible to have small local oscillations while the score will decay over longer periods of time.

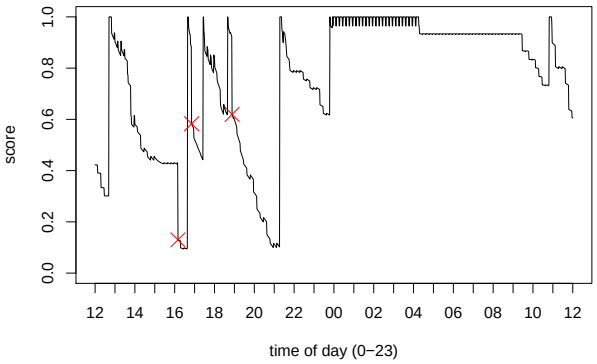

Figure 4: Score over time. Each red "X" marks a call to an unknown number, where the score is decreased.

We also ran the scoring algorithm on an adversarial trace obtained by splicing two individuals' traces. Not surprisingly, the adversary's score quickly decreases to 0 as the adversary calls a disjoint set of numbers from the owner of the device.

6 Future Work

As future work, we plan to investigate the following: 1) Make use of all features for the scoring, and report results on false positive and false negative rates. 2) Research methods to model the dependence between different features (i.e., activities). 3) Research methods to model adversarial behavior.

Acknowledgments

We gratefully acknowledge insightful feedback and invaluable suggestions from Oliver Brdiczk, David Goldberg, Mark Grandcolas, Jessica Staddon and Arvind Narayanan. We thank Yuan Niu and Alan Walendowski for their help with data collection. We also thank the anonymous reviewers for their helpful comments.

References

[1] J. Bigun, J. Fierrez-Aguilar, J. Ortega-Garcia, and J. Gonzalez-Rodriguez. Combining biometric evidence for person authentication. In *Advanced Studies in Biometrics*, 2005.

[2] R. Brunelli and D. Falavigna. Person identification using multiple cues. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1995.

[3] K. Chang, J. Hightower, and B. Kveton. Inferring identity using accelerometers in television remote controls. In *Proceedings of the International Conference on Pervasive Computing*, 2009.

[4] M. L. Damiani and C. Silvestri. Towards movement-aware access control. In *Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*, 2008.

[5] S. Furnell, N. Clarke, and S. Karatzouni. Beyond the pin: Enhancing user authentication for mobile devices. *Computer Fraud and Security*, 2008.

[6] D. Gafurov, K. Helkala, and T. Søndrol. Biometric gait authentication using accelerometer sensor. *JCP*, 1(7):51–59, 2006.

[7] R. Greenstadt and J. Beal. Cognitive security for personal devices. In *The First ACM Workshop on AISec*, 2008.

[8] M. Jakobsson and A. Juels. Server-side detection of malware infection. In *NSPW*, 2009.

[9] A. A. Kale, N. Cuntoor, and V. Krüger. Gait-based recognition of humans using continuous hmms. In *FGR '02: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, 2002.

[10] G. Leggett, J. Williams, and M. Usnick. Dynamic identity verification via keystroke characteristics. In *International Journal of Man-Machine Studies*, 1998.

[11] F. Monrose and A. Rubin. Authentication via keystroke dynamics. In *4th ACM conference on Computer and communications security*, pages 48–56, 1997.

[12] M. Nisenson, I. Yariv, R. El-Yaniv, and R. Meir. Towards behaviometric security systems: Learning to identify a typist. In *PKDD*, 2003.

[13] J. Oberheide, E. Cooke, and F. Jahanian. Cloudav: N-version antivirus in the network cloud. In *Proceedings of the 17th USENIX Security Symposium (Security)*, 2008.

[14] N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *WiSe '03: Proceedings of the 2nd ACM workshop on Wireless security*, 2003.

[15] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen. Discovering personally meaningful places: An interactive clustering approach. In *ACM Transactions on Information Systems*, volume 25, page 12. ACM, 2007.