RESEARCH

Open Access

Making a good thing better: enhancing password/PIN-based user authentication with smartwatch

Bing Chang[1], Yingjiu Li[1*], Qiongxiao Wang[2,3], Wen-Tao Zhu[2] and Robert H. Deng[1]

Abstract

Wearing smartwatches becomes increasingly popular in people's lives. This paper shows that a smartwatch can help its bearer authenticate to a login system effectively and securely even if the bearer's password has already been revealed. This idea is motivated by our observation that a sensor-rich smartwatch is capable of tracking the wrist motions of its bearer typing a password or PIN, which can be used as an authentication factor. The major challenge in this research is that a sophisticated attacker may imitate a user's typing behavior as shown in previous research on keystroke dynamics based user authentication. We address this challenge by applying a set of machine learning and deep learning classifiers on the user's wrist motion data that are collected from a smartwatch worn by the user when inputting his/her password or PIN. Our solution is user-friendly since it does not require users to perform any additional actions when typing passwords or PINs other than wearing smartwatches. We conduct a user study involving 51 participants so as to evaluate the feasibility and performance of our solution. User study results show that the best classifier is the Bagged Decision Trees, which yields 4.58% FRR and 0.12% FAR on a QWERTY keyboard, and 6.13% FRR and 0.16% FAR on a numeric keypad.

Keywords: Wearable devices, User authentication, Sensor, Machine learning, Deep learning

Introduction

A smartwatch is a computerized wristwatch with functionalities beyond timekeeping. The use of smartwatch has become a rising trend in today's consumer electronics. According to a recent forecast (CCS Insight Forecast Predicts Apple Watch and Hearables to Fuel Growthin Wearables 2017), 71 million smartwatches will be sold in 2018 worldwide, and doubled to 140 million in 2022. Equipped with rich sensors, smartwatches can help enhance the security of password/PIN-based user authentication. This is based on an observation that smartwatch sensors can be used to track users' wrist movements when users type passwords or PINs, and thus authenticate users to a login system even if the users' passwords or PINs have already been revealed to attackers. Our solution requires that a machine learning or deep learning classifier be trained on a user's smartwatch sensor data, and be used to authenticate a user according to the user's smartwatch sensor data, where the data are collected from a smartwatch worn by the user when inputting his/her password or PIN.

Similar to this idea, keystroke dynamics has long been used for user authentication based on users' hand movements (Monrose and Rubin 1997; Monrose and Rubin 2000; Peacock et al. 2004), where keystroke dynamics refers to the timing patterns of a user who presses and releases keys on a keyboard. Keystroke dynamics had been considered to be a reliable user authentication factor until Meng et al. showed that keystroke dynamics may not be suitable for user authentication as it is vulnerable to user imitation attacks (Meng et al. 2013). In a user imitation attack, an attacker can imitate a user's keystroke dynamics and pass keystroke dynamics based user authentication after being trained with the victim's keystroke pattern.

Compared to keystroke dynamics, the sensor data obtained from a smartwatch worn by a user when the user types a password or PIN contain much more information

* Correspondence: yjli@smu.edu.sg
[1]School of Information Systems, Singapore Management University, Singapore, Singapore
Full list of author information is available at the end of the article

about the user's typing behavior, including acceleration and angular velocity data that are measured at a relatively high frequency (e.g., 50 Hz). It is difficult for an attacker to imitate a victim's wrist motions that are measured at a high frequency for password/PIN entry. We show that our solution can effectively thwart the imitation attack to users' keystrokes.

Interestingly, the sensor data measured by smartwatches have been exploited recently to conduct keystroke inference attacks (Liu et al. 2015; Maiti et al. 2016; Wang et al. 2016; Wang et al. 2015). While a user types on a QWERTY keyboard or numeric keypad, he/she moves his/her hand to reach the keys and this causes distinct motions of the user's wrist. The motion sensor data collected from the user's smartwatch can be used to track the user's wrist motions and thus infer the user's inputs such as PINs and passwords. In common, previous studies on keystroke inference attacks show that the motion sensors of smartwatches can be exploited to compromise user security and privacy. From another point of view, the motion sensor data collected from smartwatches contain unique features of users' typing behaviors, and can thus be exploited to enhance the security of password/PIN-based user authentication.

In this paper, we make good use of smartwatch sensor data to enhance password/PIN-based user authentication. Our solution is user-friendly since it does not require users to perform any additional actions when typing passwords or PINs other than wearing smartwatches. We show that our solution is secure against the keystroke imitation attack proposed in (Meng et al. 2013). Even if an attacker obtains a target user's password and is able to imitate the keystroke dynamics of the user, our solution can detect the imitation attack with a high probability by analyzing the sensor data collected from smartwatches. To evaluate the feasibility and performance of our solution, we conduct an IRB-approved user study with 51 participants. We test six popular machine learning algorithms for processing smartwatch data, and evaluate their performance in user authentication. We discover that the Bagged Decision Trees performs the best in our user study, yielding 4.58% false reject rate (FRR) and 0.12% false acceptance rate (FAR) on the QWERTY keyboard for password-based user authentication, and 6.13% FRR and 0.16% FAR on the numeric keypad for PIN-based user authentication.We also show that the keystroke imitation attack has insignificant impact on the accuracy of our scheme.

This paper extends a preliminary conference version "employing smartwatch for enhanced password authentication" published in WASA 2017 in the following aspects: (i) We apply a deep learning algorithm, long short-term memory (LSTM) network to process smartwatch sensor data and evaluate its performance for user authentication. We use TensorFlow to implement LSTM and compare its evaluation results with other machine learning algorithms. (ii) A comprehensive performance analysis shows that our solution is secure against the imitation attack presented in (Meng et al. 2013). Even if an attacker's keystroke timings are similar to a victim's, our solution can still differentiate imitators from legitimate users by analyzing smartwatch data. (iii) The presentation of this paper is improved with more details, analyses, and figures. Compared to the conference version, the length of this paper increases by about 50%.

Roadmap

The rest of this paper is organized as follows. Section "Background" presents some background information about smartwatch, sensor dynamics, and keystroke dynamics. Section "Assumptions" introduces the assumptions that are used in our solution. Section "Scheme Design" presents the details of our design. Section "Evaluation" evaluates our solution. Section "Discussion" provides discussions on motion leaks and limitations. Section "Related Work" summarizes the related work, and Section "Conclusion" concludes this paper.

Background

Smartwatch and sensor dynamics

There are various sensors on smartwatches to collect information about users, including accelerometer, gyroscope, heart rate sensor, and microphone. We choose Moto 360 sport, which is powered by Android Wear OS, for our evaluation purpose. We collect data from accelerometer and gyroscope for the purpose of user authentication. The built-in motion sensor is an InvenSense MPU 6051 Six-Axis (Gyroscope + Accelerometer) MEMS motion tracking device, which can measure the accelerations and angular velocities of movement in x-, y- and z-axis regardless of the orientation of watch. Accelerometer and gyroscope in smartwatches have been

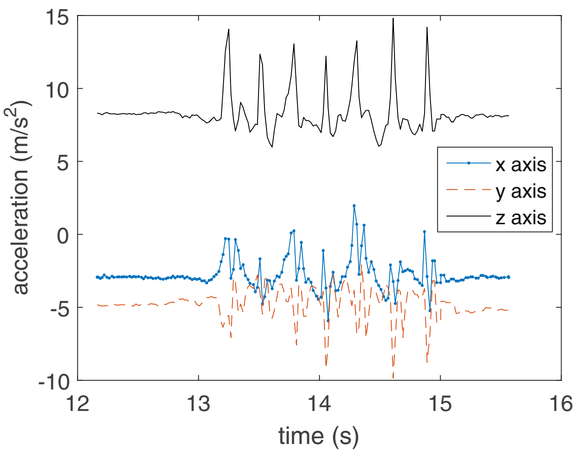

Fig. 1 The accelerometer data when a user types "924673" followed by "Enter"

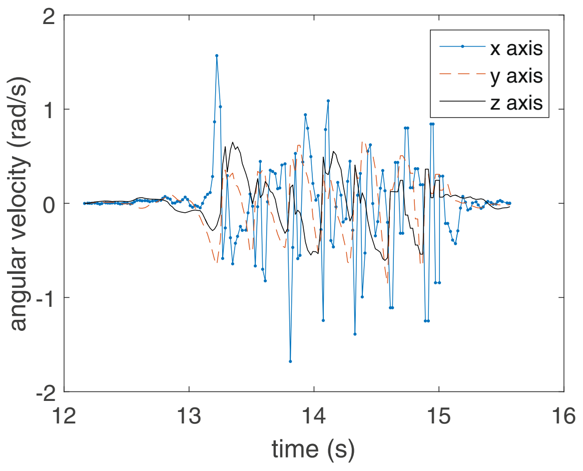

Fig. 2 The gyroscope data when a user types "924673" followed by "Enter"

extensively used in user behavioral characterization, including sensor-based keystroke inference (Liu et al. 2015; Maiti et al. 2016; Wang et al. 2016; Wang et al. 2015). The basic idea is that the sensor data provide necessary information which can be used to accurately recognize the hand movements performed by users wearing smartwatches. For instance, Figs. 1 and 2 show the accelerometer and gyroscope data that we collect when a user types "924,673" using the keypad of DELL SK-8115 keyboard. These data contain the motion information when the user types the PIN. Instead of using such sensor data for keystroke inference, we use them for user authentication.

Keystroke dynamics

Keystroke dynamics refers to the timing information associated with key-press events. Two types of key-press events are usually used in modeling keystroke dynamics, including (a) key-down event (*KD*): a user presses a key and (b) key-up event (*KU*): a user releases a key. One or more possible keystroke timings associated with consecutive key-press events, e.g., *KD-KU* time and *KD-KD* time, are considered as keystroke dynamics features in (Killourhy and Maxion 2010) and shown in Fig. 3. Keystroke dynamics features have been used to identify and authenticate users using both hardware keyboards (Clarke et al. 2003; Karatzouni and Clarke 2007; Zahid et al. 2009) and software keyboards (Tasia et al. 2014; Trojahn and Ortmeier 2012). However, Meng et al. (2013) revealed that a training interface can be set up to help attackers imitate users' keystroke dynamics, which makes it unsafe to employ keystroke dynamics for user authentication. Because keystroke dynamics contains only the timing information about users' keystroke, it is possible for an attacker to imitate a user's keystroke via a training interface. To address this problem, we model a user's typing behavior using both acceleration data and angular velocity data from the user's smartwatch. It is difficult for an attacker to imitate a user's typing behavior in our model without accessing the victims' smartwatch sensor data.

Assumptions

It is assumed that a user (the victim) wears a smartwatch such as Apple Watch or Moto 360 Sport, while he/she types passwords and PINs. The smartwatch is equipped with accelerometer and gyroscope which collect the motion information of the victim's wrist. If the victim uses one hand to type, the smartwatch is worn on the same hand. As smartwatches are widely used, it is not uncommon to make such assumption in daily life. We focus on two types of keyboards in this paper, including QWERTY keyboards and numeric keyboards, which can be used on PCs, mobile devices, Point of Sale (POS) terminals and Automatic Teller Machines (ATMs).

An attacker intends to login to a user/victim's account after the attacker obtains the victim's username and password/PIN. The attacker may observe or record the victim's entry of passwords or PINs. However, it is assumed that the attacker cannot obtain any sensor data about the victim's typing of passwords/PINs from the victim's smartwatch; instead, the attacker has the following capabilities. First, the attacker may obtain the victim's username and password (e.g., by shoulder-surfing attack or key logger). Second, the attacker may obtain the victim's keystroke timing data and imitate the

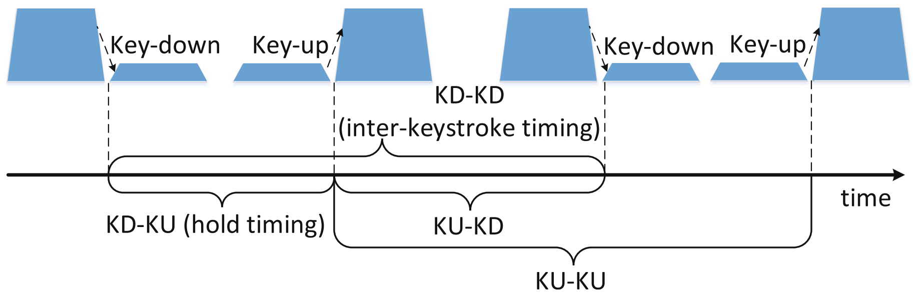

Fig. 3 Keystroke timings used in keystroke dynamics techniques

victim's keystroke as shown in (Meng et al. 2013). In such imitation attacks, the attacker may wear the same kind of smartwatch and access the same kind of keyboard as the victim's.

Scheme design

In this section, we present the design of our smartwatch enhanced password/PIN authentication scheme.

Overview

The main goal of our design is to demonstrate that using smartwatches can help enhance the security of password/PIN authentication systems. Password/PIN authentication systems suffer from password/PIN observation attacks such as shoulder surfing and key logging in which attackers may obtain users' passwords/PINs. We design and implement a system which can distinguish legitimate users from illegitimate users by processing the sensor data from the smartwatches worn on legitimate users' wrists. Even if an attacker types in the same password/PIN with the victim's, the attacker's hand motion is still different from the user's. The accelerometer and gyroscope in a smartwatch can be used to track its wearer's hand motion during password/PIN input. As smartwatches are widely used nowadays, our system does not require any additional actions when typing passwords/PINs other than wearing smartwatches, making our system user-friendly. Our system can be employed as long as a smartwatch is worn on the user's wrist when the user types a password/PIN on a keyboard, or keypad of any device such as PC, ATM, and mobile phone.

Figure 4 shows the flow of our system. Our system takes as input the password/PIN and the raw sensor data (e.g., acceleration, angular velocity) from the smartwatch worn on a user's wrist. The password/PIN and the raw sensor data are sent to our server for verification. The password/PIN is for the conventional password/PIN authentication while the raw sensor data are processed to further verify the user. Our system consists of two phases, a training phase and a detection phase. During the training phase, user ID and password/PIN are registered for the conventional password/PIN authentication and the raw sensor data are recorded. The raw sensor data are then processed according to our feature extraction method which translates all the recorded sensor data into features suitable for our classifier. After the features are extracted, we train the classifier with these features. During the detection phase, the system verifies the user ID and password/PIN first. If the typed password/PIN is correct, it extracts features from the sensor data and inputs the extracted features into the classifier so as to verify the user. The classifier matches the features extracted from the sensor data against the user's profile so as to identify whether the password/PIN is typed by the legitimate user. A user is authenticated only if both the password/PIN is correct and the typing pattern matches the user's profile.

As the conventional password/PIN authentication has been rigorously investigated, we focus on how to use machine learning techniques to process the sensor data of smartwatches and match users' profiles. We collect the sensor data when users type passwords on QWERTY keyboards or type PINs on numeric keypads. QWERTY keyboards and numeric keypads are mainstream devices for inputting passwords and PINs nowadays, respectively. As long as a user types passwords or PINs with the hand wearing the smartwatch, the sensor data can be collected and then help authenticate the user. We extract unique features from the sensor data and train several machine learning classifiers using the features as user profiles. The classifiers are used to authenticate users.

Data collection

Our system collects the accelerometer and gyroscope data within a time window from a smartwatch worn on a user's wrist. The time window begins when the user begins to type a password or PIN, and ends once the user presses "Enter" to finish the input. The data collected from accelerometer and gyroscope are streams of timestamped real values along three axes. For a given

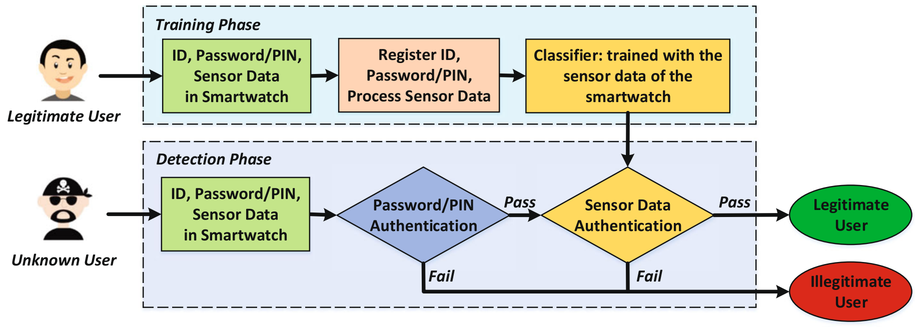
Training Phase
Legitimate User
ID, Password/PIN, Sensor Data in Smartwatch
Register ID, Password/PIN, Process Sensor Data
Classifier: trained with the sensor data of the smartwatch
Detection Phase
Unknown User
ID, Password/PIN, Sensor Data in Smartwatch
Password/PIN Authentication
Pass
Sensor Data Authentication
Pass
Legitimate User
Fail
Fail
Illegitimate User

Fig. 4 Overview of smartwatch enhanced authentication scheme

timestamp, t, the accelerometer data are in the form of $a(t) = (a_x, a_y, a_z)$ while the gyroscope data are in the form of $\omega(t) = (\omega_x, \omega_y, \omega_z)$. Note that the accelerometer data are affected by the earth gravity, so when the smartwatch is lying flat on the desk, the accelerometer data show that there is an acceleration of 9.8 m/s^2 along the z-axis.

We install an app in each smartwatch used in our experiment to collect the sensor data. The app is given the permission to access the accelerometer and gyroscope of the smartwatch. The app is also given the permission to communicate with the password/PIN input interface and obtain the timing information when the user begins typing and when the user finishes typing. According to the timing information, the app collects the sensor data and sends the data to our server which is used to authenticate users. We collect the sensor data in both the training phase and the detection phase. In the training phase, we collect enough data to train certain classifiers. Assuming it takes 6 s for a user to type in a password or PIN, it will take about 10 min to type in the password/PIN 100 times, which is enough for training. In the detection phase, the app collects the sensor data when the user types the password or PIN and send the data to our server to verify whether the user is legitimate.

Feature extraction

The raw data from accelerometer and gyroscope are streams of timestamped real values along three axes. We extract temporal features from these data for authentication purpose. We summarize the features that we extract from the sensor data streams in Table 1. These features have been previously used for sensor-based smartphone fingerprinting (Das et al. 2015) and tracking mobile web users. Since there are three axes for both sensors, we obtain a vector of 36 elements (6 features * 3 axes * 2 sensors) after extracting the features from a sensor data stream. Our server extracts the aforementioned features for certain classifier in both the training phase and the detection phase. In the training phase, all the extracted features are used to train the classifier, while in the detection phase, the features are used to authenticate users according to the classifier.

Supervised learning and detection

In the training phase, after the system extracts all the features from training data, it trains the classifier using the features. In Section "Evaluation", we evaluate six widely used classification algorithms, including Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), Bagged Decision Trees (Matlab's Treebagger model), Naive Bayes classifier, Discriminant Analysis classifier, and Long Short-Term Memory (LSTM) network. We discover that the Bagged Decision Trees outperforms the other classifiers in Section "Evaluation". In the detection phase, a feature vector is extracted from the sensor data of a user's smartwatch, and fed into a trained classifier for the user so as to determine whether the user is legitimate or not.

Evaluation

In this section, rigorous experiments are conducted to evaluate the performance and security of our proposed scheme.

Experimental setup

To collect the sensor data when a user wearing a smartwatch types in a password or PIN, we setup a data collection system which consists of four components, a keyboard/keypad, a laptop, a mobile phone and a smartwatch. Figure 5 illustrates our data collection system. A user needs to wear a smartwatch and type in passwords/PINs on a laptop using a keyboard. The sensor data are recorded automatically on the user's mobile phone.

Keyboard/keypad

We use a DELL SK-8115 keyboard for user input. Users type passwords on QWERTY keyboards and type PINs on numeric keypads.

Laptop

The laptop is a MacBook Pro with an Intel i7 2.7GHz processor and 8GB RAM, running an Ubuntu 14.04 64-bit virtual machine. We obtained the source code of the data collection system from the authors of (Meng et al. 2013) and rebuilt their system. We then modified their system for our experiments. The main functions of the modified system include providing tasks for users to type, judging whether users' inputs are correct and sending control information to the mobile phone via WiFi connection. A user interface is provided as a web page for users to type in passwords or PINs according to a prompt. When the system

Table 1 Extracted features

Feature	Description
Mean Strength	Arithmetic mean of the signal strength
Standard Deviation	Standard deviation of the signal strength
Average Deviation	Average deviation from mean
Skewness	Measure of asymmetry about mean
Kurtosis	Measure of the flatness or spikiness of a distribution
RMS	Square root of arithmetic mean of squares of the signal strength

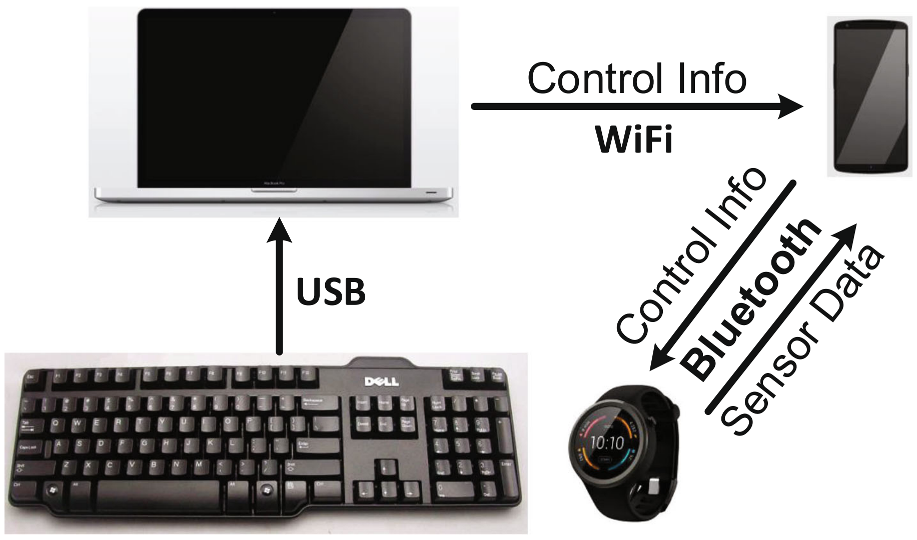

Fig. 5 Experimental setup

shows the prompt, it sends out a "start" message to the mobile phone at the same time. Upon receiving the message, the mobile phone also sends a "start" message to the smartwatch, which begins to record the sensor data. When the user presses "Enter" to finish the input, the system sends a "finish" message to the mobile phone and triggers it to send a "finish" message immediately to the smartwatch. The smartwatch terminates its recording of the sensor data and sends the recorded data to the phone. If the input password is incorrect or the user presses "backspace", the user's input is erased and the system sends a "restart" message to the phone and in turn to the smartwatch which restarts the recording of the sensor data.

Mobile phone

The mobile phone is a Nexus 6 powered by Android 6.0. We install an app in this phone to communicate with the laptop and the smartwatch, as well as store the sensor data obtained from the smartwatch. The app receives the control information from the laptop through WiFi connection and communicates with the smartwatch through Bluetooth connection. After the user finishes typing each password or PIN, the accelerometer data and gyroscope data from the smartwatch are stored in two files respectively. Each file is a list of the sensor data entries which consist of timestamps and data values in three axes.

Smartwatch

The smartwatch is a Moto 360 Sport, which runs on the Android Wear platform. We install an app in this smartwatch to collect its sensor data. When the app receives a "start" message from the phone, the app starts recording accelerometer and gyroscope readings. During data collection, the sensor data are stored locally. When the app receives a "finish" message, the sensor data are transferred to the phone via Bluetooth. The highest sampling frequency that Moto 360 sport supports is 50 Hz and we specify the *SENSOR DELAY FASTEST* flag at the sensor listener registration time to achieve this.

User study

Figure 6 shows the process of our user study[1]. We collect testing data from 51 participants in our university (students and staff), including 22 males and 29 females with ages between 19 and 34 (45 of them are between 20 and 27 years old). Twenty-six of them major in computer science and all of them are skilled keyboard users. Our user study involves two sessions, and each of them takes about 60 min. Every participant takes part in Session I and we choose 9 of them (5 males and 4 females) to take part in Session II. Each participant is paid with 10 dollars after completing each session.

Data collection

In the data collection phase of Session I, we collect the sensor data when each participant types a predefined QWERTY keyboard password and a predefined keypad PIN. The QWERTY keyboard password is used to simulate that a user types a password on a standard keyboard while the keypad PIN is used to simulate that a user types a PIN on a keypad of ATM or POS terminal. The layouts of POS terminal keypad, ATM keypad and the keypad on a keyboard are shown in Fig. 7. There is no fundamental difference to use a smartwatch to track a user's wrist motion when the user inputs PINs on them. Therefore, we choose the keypad on DELL SK-8115 keyboard in our user study. The participants are required

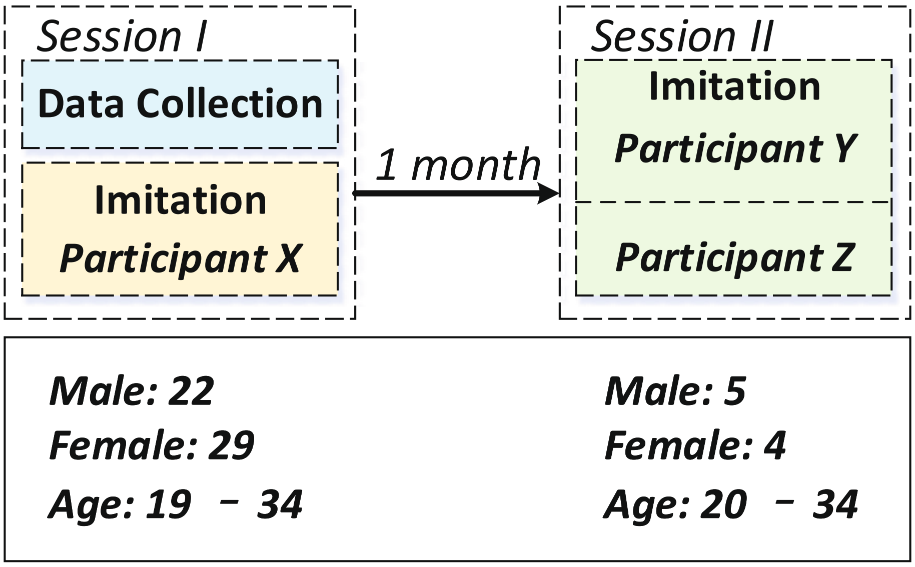

Fig. 6 User study process

to wear smartwatches on their right wrists, type in QWERTY passwords with both hands, and type in PINs with the right hands. The participants are also required to keep standing when they type PINs, since people usually type PINs on ATMs or POS terminals while standing. We choose the QWERTY keyboard password and the keypad PIN as "ths.ouR2" and "924,673", respectively in our experiment. The password "ths.ouR2" is a strong password used in previous work (Meng et al. 2013) while "924,673" is a randomly generated PIN. The participants are required to type each password/PIN 100 times.

Keystroke imitation attack

In order to find some participants who are potentially good at keystroke imitation and test whether our system can resist the imitation attack proposed in (Meng et al. 2013), we arrange an imitation phase in both Session I and Session II. We have re-implemented the system proposed in (Meng et al. 2013) and require that each participant uses this system to imitate a previous participant's keystroke dynamics. After the participant finishes each input, the system shows an interface (Fig. 8) and a score to indicate the differences between this input and the target typing pattern. Note that in Fig. 8, the circles mean the hold timings and the bars mean the inter-keystroke timings. The blue circles and bars are the target's timing information. Imitators can adjust their typing according to the differences between their timing information and the target's. In the imitation phase of Session I, we aim to find some participants who are good at imitation, so each participant is required to imitate a previous participant's typing pattern of "ths.ouR2". We find 9 best imitators according to the imitation performance and they are invited to take part in Session II. In Session II, each participant is required to imitate other two participants typing "ths.ouR2" and "924,673", respectively. We discover that it is unable to distinguish these imitators from the corresponding victims according to the keystroke dynamics only, which is similar to the conclusion drawn in (Meng et al. 2013). In section "Defending against Keystroke Imitation Attack", we further investigate

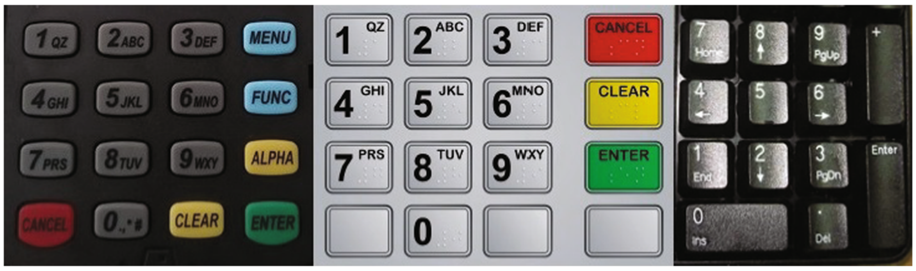

Fig. 7 Three kinds of keypad: keypad of POS terminal (left), keypad of ATM (middle), keypad on keyboard (right)

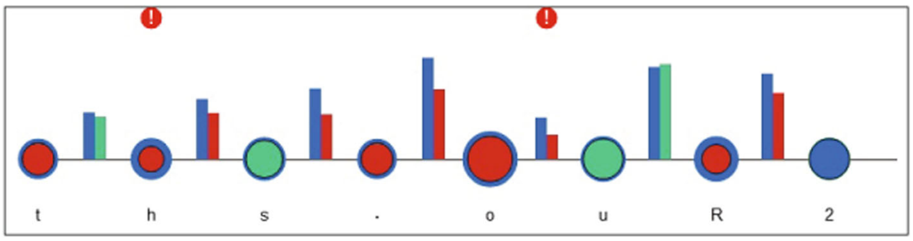

Fig. 8 The interface of the imitation system (Meng et al. 2013)

whether it is possible to distinguish them by analyzing the sensor data taken from smartwatches.

Performance analysis

Data processing

To show the performance of our system on both QWERTY keyboard and numeric keypad, we process the sensor data collected when the 51 participants type "ths.ouR2" and "924,673", respectively. Different participants are required to type in the same password/PIN as we aim to find out whether and to what extent the sensor data can help differentiate them. After deleting invalid data caused by system error, we extract the features according to Section "Feature Extraction" and obtain 4789 feature vectors for the QWERTY keyboard and 4868 feature vectors for the numeric keypad. For each participant, we have approximately 93 feature vectors. We delete some outliers from the accelerometer data as follows. We first calculate the mean value M and the standard deviation D of the mean strengths, and then calculate the difference between M and each mean strength. If the difference is larger than three times of D, we delete the corresponding feature vector. In addition, if the D values of some participants are three times higher than others, we also delete these data to improve the quality of the collected data. In total, we delete 759 out of 4789 feature vectors for the QWERTY keyboard and 609 out of 4868 feature vectors for the numeric keypad. To evaluate the performance of our scheme, we adopt false acceptance rate (FAR), which indicates the fraction of imposter access attempts identified as valid users, and false rejection rate (FRR), which indicates the fraction of valid user attempts identified as impostors.

Performance of different classifiers

We evaluate the performance of five machine learning classifiers, including Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), Bagged Decision Trees (Matlab's Treebagger model), Naive.

Bayes classifier, and Discriminant Analysis classifier. For training and testing of these classifiers, we *randomly* select 50% of the feature vectors for each participant as a training set while the remaining 50% as a testing set. To prevent any bias in our experiments, we randomize the training and testing sets 10 times and compute the average accuracy. Our experimental results are shown in Table 2 and Table 3. In both tables, "keyboard (improved)" and "keypad (improved)" mean the improved data set derived by removing outliers from the original data set. The results show that the Bagged Decision Trees outperforms the other classifiers and its accuracy is 4.58% FRR and 0.12% FAR on the QWERTY keyboard, and 6.13% FRR and 0.16% FAR on the numeric keypad.

Deep learning method

As deep learning methods are increasingly used in user authentication systems, we evaluate the performance of a widely used deep learning algorithm, LSTM (Hochreiter and Schmidhuber 1997). We use the basic LSTM cell in TensorFlow (Abadi et al. 2016) to conduct our experiment. Instead of using the extracted features, we use the raw sensor data as the input to this algorithm. The sensor data is preprocessed to obtain the last 200 data points for each sample, and pad with 0 if the sample contains less than 200 data points. A challenge in training is that LSTM needs to be trained with a large amount of training data (i.e., data hungry); however, the

Table 2 FRR in different scenarios

	keyboard (improved)	keypad (improved)	imitation I (keyboard)	imitation I (keypad)	imitation II (keyboard)	imitation II (keypad)
SVM	18.15%	11.79%	14.81%	5.46%	14.00%	6.64%
k-NN	28.03%	20.02%	22.10%	9.23%	20.99%	8.80%
BDT	4.58%	6.13%	1.93%	1.51%	2.03%	3.41%
Naive Bayes	8.79%	11.03%	12.02%	6.97%	11.42%	9.34%
DAC	6.08%	6.09%	1.72%	1.51%	1.47%	3.95%
LSTM	6.77%	8.04%	3.64%	2.79%	4.12%	4.75%

Table 3 FAR in different scenarios

	keyboard (improved)	keypad (improved)	imitation I (keyboard)	imitation I (keypad)	imitation II (keyboard)	imitation II (keypad)
SVM	0.43%	0.28%	1.5%	0.47%	1.3%	0.63%
k-NN	0.67%	0.48%	2.2%	0.83%	1.9%	0.80%
BDT	0.12%	0.16%	0.21%	0.15%	0.24%	0.47%
Naive Bayes	0.21%	0.26%	1.2%	0.78%	0.78%	1.0%
DAC	0.14%	0.14%	0.17%	0.15%	0.08%	0.04%
LSTM	0.17%	0.19%	0.58%	0.49%	0.57%	0.73%

collected data set is relatively small. To address this challenge, we add ±1% perturbation to the original data so as to generate more training data. In total, we use about 48,000 samples for training LSTM model for each user in our experiment. We conduct the experiments 10 times and compute the average accuracy. The results are also shown in Table 2 and Table 3. The results show that the accuracy of LSTM is 6.77% FRR and 0.17% FAR on the QWERTY keyboard, and 8.04% FRR and 0.19% FAR on the numeric keypad.

Impact of different sensors

To understand the impact of different sensors, we test our scheme using the data collected from one sensor only, instead of two sensors. Figure 9 shows the evaluation results with the Bagged Decision Trees. In all cases, using accelerometer only can reach almost the same accuracy as using both sensors, while using gyroscope only results in much lower accuracy. Nonetheless, using both sensors can improve the accuracy by about 3% compared to using accelerometer only. As a result, we use both sensors in our scheme.

Defending against keystroke imitation attack

To test whether our scheme can defend against the keystroke imitation attack proposed in (Meng et al. 2013), we process the sensor data collected from nine selected participants imitating others in session II of our experiments. Note that the selected participants are the best imitators among the 51 participants selected in Session I. In Session II, they are requested to imitate other two participants typing on QWERTY keyboard and numeric keypad, respectively. We have reproduced the results of (Meng et al. 2013) with these nine participants. After trained with the system proposed in (Meng et al. 2013), the selected participants can imitate their targets' typing patterns in a success rate higher than 90%.

To investigate whether our scheme can differentiate original users from imitators, we extract the features from the sensor data collected from the original users and from their imitators, respectively. We then *randomly* select 50% of the feature vectors collected from each original user to train all classifiers. The testing set consists of (i) other 50% of the feature vectors derived from the sensor data collected from the same user, and (ii) all feature vectors derived from the sensor data collected from imitators. Our evaluation results are shown in Table 2 and Table 3. In the first round of imitation, the results show that the accuracy of the Bagged Decision Trees is 1.93% FRR and 0.21 FAR on the standard keyboard, and 1.51% FRR and 0.15% FAR on the numeric keypad. In the second round of imitation, the accuracy of the Bagged Decision Trees is 2.03% FRR and 0.24 FAR on the standard keyboard, and 3.41% FRR and 0.47% FAR on the numeric keypad. The results show that the keystroke imitation attack has little impact on our scheme.

To show the differences in the typing pattern between an original user and an imitator of the user, we analyze the keystroke timing data and the sensor data collected from them. Figures 10 and 11 show the keystroke timings of the original

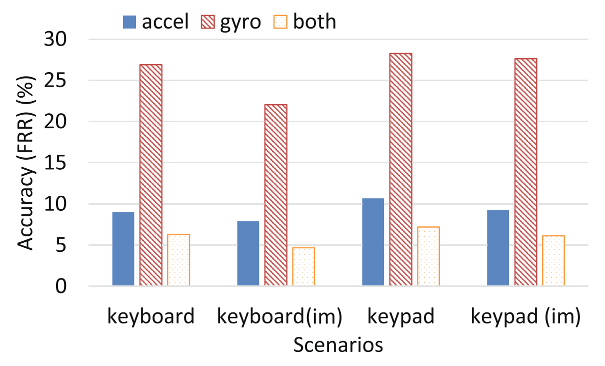

Fig. 9 The accuracy (FRR) when using only one sensor

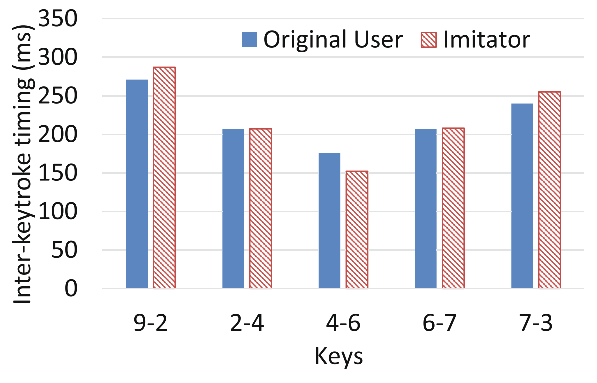

Fig. 10 Inter-keystroke timings of an original user and the imitator

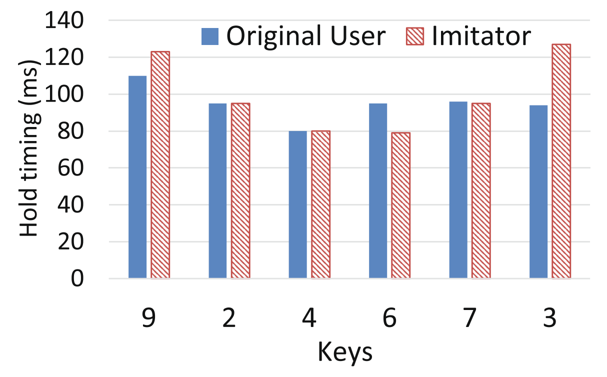

Fig. 11 Hold timings of an original user and the imitator

user and the user's imitators. We can see that the inter-keystroke timings (Fig. 10) and the hold timings (Fig. 11) of them are rather similar. However, their sensor data are clearly different. Figures 12 and 13 shows the sensor data of the original user and the imitator. The acceleration data of the original user are shown in Fig. 12 while the acceleration data of the imitator are shown in Fig. 13. We can see the mean values of the original user's sensor data are around (6,-4,5), while the mean values of the imitator's sensor data are around (3,-10,0). Although their keystroke timings are very similar, their sensor data can be used by our scheme to differentiate imitators from legitimate users.

Discussion

Motion leaks vs. sensor enhanced authentication

Previous research shows that sensor data collected from a user's smartwatch can help an attacker infer the user's keystroke (Liu et al. 2015; Maiti et al. 2016; Wang et al. 2016; Wang et al. 2015). We show that the sensor data can also help authenticate users. In both cases, the sensor data in smartwatches are closely related to users' privacy and thus should be protected properly. In particular, the smartwatch

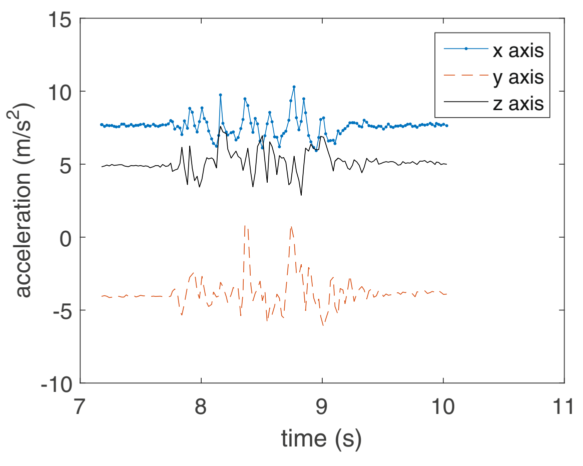

Fig. 12 The original user's acceleration data

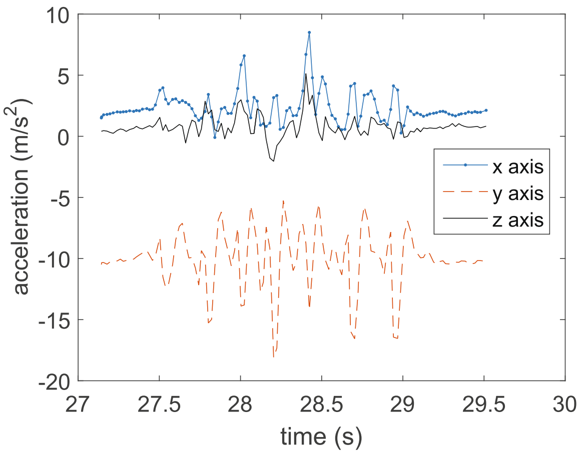

Fig. 13 The imitator's acceleration data

sensor data contain vital information about the motions of users' wrists. By analyzing the common features across different users typing the same PINs or texts, attackers can infer what users typed or narrow down their search scope significantly. On the other hand, by analyzing the unique features of different users typing the same passwords or PINs, authentication servers can differentiate legitimate users from imposters. It is worth to note that the security of sensor data has not been addressed rigorously., We recommend using more strict access control to protect sensor data in smartwatches (e.g., (Xu et al. 2012; Xu and Zhu 2015)).

Limitations

Our approach has two major limitations. First, a user is supposed to wear a smartwatch on the hand which is used to type in keypad PINs.[2] It is a problem if a user wears a smartwatch on one hand while types in a PIN using the other hand. Since smartwatches are commonly designed to be worn comfortably on either wrist, a user can wear a smartwatch on the hand which he/she uses to type in PINs. Given the increasingly cheaper price of wearable devices, people may wear both smartwatch and fitness tracker[3] on different hands, so that users can type in PINs using any hands. The second limitation is that a user is supposed to maintain his/her typing pattern for successful login. This is a common issue in keystroke dynamics. If the user's typing pattern changes or the user's hand is injured, the user should update his/her typing pattern or switch to other two factor authentication options. The third limitation is that our scheme is keyboard specific. (Since all the QWERTY keyboard are designed based on a basic layout for convenient use, our scheme should be able to work on similar QWERTY keyboards.) Another problem is that our solution is not scalable if a user registers different passwords to multiple websites. This problem can be mitigated to a certain degree by using single sign-on (Pashalidis and Mitchell 2003) services.

Related work

Typing inference from sensor data

Previous research has shown that attackers can infer what users type on smart devices such as smartphones and tablets via various side channel attacks (Aviv et al. 2012; Cai and Chen 2011; Cai and Chen 2012; Miluzzo et al. 2012; Owusu et al. 2012; Xu et al. 2012). When users type on different locations on the virtual keyboard of a smart device, the users' keystrokes may cause distinct motions of the smart device, and the motion sensor data generated by the smart device can be used to infer the tapped locations and pressed keys.

In particular, typing inference can be made from the sensor data that are collected from smartwatches. For example, Wang et al. proposed using a linguistic model to infer which word a user types on a standard keyboard according to the accelerometer and gyroscope data collected from the user's smartwatch (Wang et al. 2015). A limitation of this solution is that it cannot deal with non-contextual inputs such as passwords and PINs, which are not covered by a linguistic model. Liu et al. proposed another solution that makes use of the sensors in smartwatches, including accelerometer and microphone, to infer users' inputs on keyboards or POS terminals (Liu et al. 2015). Their solution is based on a machine-learning classifier which should be trained from smartwatch sensor data measuring hand movements between keystrokes. Maiti et al. also proposed a solution that makes use of smartwatch sensors to infer users' input; in addition, they proposed a protection framework that regulates sensor data access (Maiti et al. 2016). Different from these research efforts, Wang et al. proposed a contextual-free and training-free solution to infer users' PINs by exploiting the sensors in wearable devices, including accelerometers, gyroscopes, and magnetometers (Wang et al. 2016).

These studies have shown that it is possible to infer users' typing on keyboards using the sensor data collected from smartwatches. In comparison, our study aims to authenticate users according to the sensor data collected from users' smartwatches. To the best of our knowledge, we are among the first to exploit smartwatch sensors for user authentication purpose.

Keystroke dynamics

Tremendous work has been made in previous research on using keystroke dynamics as biometrics (e.g., (Monrose and Rubin 1997; Monrose and Rubin 2000; Peacock et al. 2004)). The majority of research in this line focuses on how to design a solution that can best distinguishes legitimate users from imposters. The study of keystroke dynamics was initially conducted on PCs and hardware keyboard (Bergadano et al. 2002; Joyce and Gupta 1990; Kang et al. 2007; Killourhy and Maxion 2010; Killourhy and Maxion 2009; Kotani and Horii 2005; Monrose and Rubin 1997; Monrose and Rubin 2000; Obaidat and Sadoun 1997; Peacock et al. 2004). As mobile devices became increasingly popular, the research of keystroke dynamics switched to mobile devices (Campisi et al. 2009; Clarke et al. 2003; Hwang et al. 2009; Karatzouni and Clarke 2007; Zahid et al. 2009), and software keyboards (Huang et al. 2012; Saevanee and Bhatarakosol 2008; Saevanee and Bhattarakosol 2009; Tasia et al. 2014; Trojahn and Ortmeier 2012). Commercial products based on keystroke biometrics have been developed over the years (Id control 2018; Intensity analytics 2018; Keyboard biometrics - KeyTrac 2018; Plurilock Security Solutions Inc 2018).

To challenge the use of keystroke dynamics as biometrics, Meng et al. (Meng et al. 2013) proposed a feedback and training interface, called *Mimesis*. Mimesis can help one user imitate another user's keystroke dynamics through incremental adjustments of typing patterns. This imitation attack poses an threat to keystroke dynamics based user authentication. Giuffrida et al. (2014) proposed using sensor-enhanced keystroke dynamics to authenticate users typing on mobile devices. A limitation of their solution is that it works on mobile devices only. It does not apply to users typing passwords/PINs on hardware keyboards/keypads. In comparison, our solution is more generic; it can be applied in all cases no matter what kinds of devices and keyboards are used for password/PIN entry as long as users wear smartwatches when they input their passwords/PINs.

Conclusion

In this paper, we proposed a novel solution for enhancing password/PIN-based user authentication with smartwatches. Our solution relies on machine learning classifiers to distinguish legitimate users from imposters according to the sensor data collected from the user's smartwatch during password/PIN entry. Our solution is reliable even if users' passwords/PINs are revealed to attacks. Our experimental results show that our solution is highly accurate. The best classifier for our solution achieves an accuracy of 4.58% FRR and 0.12% FAR on the QWERTY keyboard, and 6.13% FRR and 0.16% FAR on the numeric keypad. While previous keystroke dynamics based user authentication is subject to keystroke imitation attacks, we have shown that our solution is immune to such attacks.

Endnotes

[1]The user study was approved by the Institutional Review Board of our university. Data collected from the participants were anonymized and protected according to the corresponding IRB submission documents

[2]There is no restriction on which hand a smartwatch is worn when a user types in passwords on the standard keyboard with two hands.

[3]Fitness trackers are usually equipped with motion sensors such as accelerometer and gyroscope.

Funding
This work is partially supported by the Singapore National Research Foundation under NCR Award Number NRF2015NCR-NCR003–002. The role of the funding body is in all the parts of the paper. This work is also supported by the National Key Research and Development Program of China under Grant 2016YFB0800500. This research is also supported by AXA Research Fund.

Availability of data and materials
The data used in this paper will not be shared, because the data is collected from a user study and is protected according to the corresponding Institutional Review Board submission documents.

Authors' contributions
BC contributions include scheme design, data collection, performance analysis and paper writing. YL contributions include research ideas, scheme design, discussions and paper writing. Other authors' contributions include research ideas, discussions and paper writing. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]School of Information Systems, Singapore Management University, Singapore, Singapore. [2]Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, Beijing, China. [3]State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China.

Received: 8 April 2018 Accepted: 2 August 2018
Published online: 20 August 2018

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado,A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2016

A. J. Aviv, B. Sapp, M. Blaze, and J. M. Smith. Practicality of accelerometer sidechannels on smartphones. In Proceedings of the 28th Annual Computer Security Applications Conference, pages 41–50. ACM, Orlando, 2012

Bergadano F, Gunetti D, Picardi C (2002). User authentication through keystroke dynamics. ACM Transactions on Information and System Security (TISSEC), 5(4):367–397.

Cai L, Chen H (2011) Touchlogger: Inferring keystrokes on touch screen from smartphone motion. In: 6th USENIX Workshop on Hot Topics in Security (HotSec)

L. Cai and H. Chen. On the practicality of motion based keystroke inference attack. In International Conference on Trust and Trustworthy Computing, pages 273–290. Springer, Vienna, 2012

Campisi P, Maiorana E, Bosco ML, Neri A (2009) User authentication usingkeystroke dynamics for cellular phones. IET Signal Proc 3(4):333–341

CCS Insight Forecast Predicts Apple Watch and Hearables to Fuel Growthin Wearables. https://www.ccsinsight.com/press/company-news/3375-ccsinsight-forecast-predicts-apple-watch-and-hearables-to-fuel-growthin-wearables, 2017

Clarke NL, Furnell SM, Lines BM, Reynolds PL (2003). Keystroke dynamics on a mobile handset: a feasibility study. Inf Manag Comput Secur 11(4):161–166

A. Das, N. Borisov, and M. Caesar. Exploring ways to mitigate sensor-based smartphone fingerprinting. CoRR, abs/1503.01874, 2015

Giuffrida C, Majdanik K, Conti M, Bos H (2014, July). I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (pp. 92–111). Springer, Cham

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Huang X, Lund G, Sapeluk A (2012). Development of a typing behaviour recognition mechanism on android. In IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 1342-1347). IEEE

Hwang SS, Cho S, Park S (2009) Keystroke dynamics-based authentication for mobile devices. Comput Secur 28(1-2):85–93

Id control. http://www.idcontrol.com, 2018

Intensity analytics. http://www.intensityanalytics.com, 2018

Joyce R, Gupta G (1990) Identity authentication based on keystroke latencies. Commun ACM 33(2):168–176

P. Kang, S.-s. Hwang, and S. Cho. Continual retraining of keystroke dynamicsbased authenticator. In International Conference on Biometrics, pages 1203–1211. Springer, Seoul, 2007

Karatzouni S, Clarke N (2007) Keystroke dynamics-based authentication for mobile devices. In IFIP International Information Security Conference (pp. 253–263). Springer, Boston

Keyboard biometrics - KeyTrac. https://www.keytrac.net, 2018

K. S. Killourhy and R. A. Maxion. Comparing anomaly-detection algorithms forkeystroke dynamics. In 2009 IEEE/IFIP International Conference on Dependable Systems & Networks, pages 125–134. IEEE, Estoril, 2009

Killourhy K, Maxion R (2010) Why did my detector do that?!. In International Workshop on Recent Advances in Intrusion Detection (pp. 256-276). Springer, Berlin, Heidelberg.

Kotani K, Horii K (2005) Evaluation on a keystroke authentication system by keyingforce incorporated with temporal characteristics of keystroke dynamics. Behav Inform Technol 24(4):289–302

X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang. When good becomes evil: Keystrokeinference with smartwatch. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pages 1273–1285. ACM, Denver, 2015

Maiti A, Armbruster O, Jadliwala M, He J (2016) Smartwatch-based keystrokeinference attacks and context-aware protection mechanisms. In: Proceedings of the 11th ACM Asia Conference on Computer and Communications Security

Meng TC, Gupta P, Gao D (2013) I can be you: Questioning the use of keystroke dynamics as biometrics. In: Proceedings of the 20th Network and Distributed System Security Symp

E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury. Tapprints: yourfinger taps have fingerprints. In Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services, pages 323–336. ACM, Low Wood Bay, 2012

Monrose F, Rubin A (1997) Authentication via keystroke dynamics. In: Proceedings of the 4th ACM Conference on Computer and Communications Security

Monrose F, Rubin AD (2000) Keystroke dynamics as a biometric for authentication. Futur Gener Comput Syst 16(4):351–359

Obaidat MS, Sadoun B (1997) Verification of computer users using keystroke dynamics. IEEE Trans Syst Man Cybern B Cybern 27(2):261–269

E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang. Accessory: password inferenceusing accelerometers on smartphones. In Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications, page 9. ACM, Phoenix, 2012

A. Pashalidis and C. J. Mitchell. A taxonomy of single sign-on systems. In Australasian Conference on Information Security and Privacy, pages 249–264. Springer, Wollongong 2003

Peacock A, Ke X, Wilkerson M (2004) Typing patterns: A key to user identification. IEEE Secur Priv 2(5):40–47

Plurilock Security Solutions Inc. http://www.plurilock.com, 2018

H. Saevanee and P. Bhatarakosol. User authentication using combination of behavioral biometrics over the touchpad acting like touch screen of mobile device. In International Conference on Computer and Electrical Engineering, pages 82–86. IEEE, Phuket, 2008

H. Saevanee and P. Bhattarakosol. Authenticating user using keystroke dynamicsand finger pressure. In 6th IEEE Consumer Communications and Networking Conference, pages 1–2. IEEE, Las Vegas, 2009

Tasia C-J, Chang T-Y, Cheng P-C, Lin J-H (2014) Two novel biometric features in keystroke dynamics authentication systems for touch screen devices. Secur Commun Netw 7(4):750–758

Trojahn M, Ortmeier F (2012) Biometric authentication through a virtual keyboard for smartphones. Int J Comput Sci Inf Technol 4(5):1

H. Wang, T. T.-T. Lai, and R. Roy Choudhury. Mole: Motion leaks through smartwatch sensors. In Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, pages 155–166. ACM, Tucson, 2015

C. Wang, X. Guo, Y. Wang, Y. Chen, and B. Liu. Friend or foe?: Your wearabledevices reveal your personal pin. In Proceedings of the 11th ACM Asia Conference on Computer and Communications Security, pages 189–200. ACM, Xi'an, 2016

Z. Xu and S. Zhu. Semadroid: A privacy-aware sensor management framework forsmartphones. In Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, pages 61–72. ACM, San Antonio, 2015

Z. Xu, K. Bai, and S. Zhu. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks, pages 113–124. ACM, 2012

Zahid S, Shahzad M, Khayam SA, Farooq M (2009) Keystroke-based user identification on smart phones. In International Workshop on Recent Advances in Intrusion Detection (pp. 224–243). Springer, Berlin