6-2017

Employing smartwatch for enhanced password authentication

Bing CHANG
Chinese Academy of Sciences

Ximing LIU
Singapore Management University, xmliu.2015@smu.edu.sg

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Pingjian WANG
Chinese Academy of Sciences

Wen-Tao ZHU
Chinese Academy of Sciences

See next page for additional authors

Citation

CHANG, Bing; LIU, Ximing; LI, Yingjiu; WANG, Pingjian; ZHU, Wen-Tao; and WANG, Zhan. Employing smartwatch for enhanced password authentication. (2017). *Wireless Algorithms, Systems, and Applications: WASA 2017: Guilin, China, June 19-21: Proceedings*. 10251, 691-703.
Available at: https://ink.library.smu.edu.sg/sis_research/3804

Author

Bing CHANG, Ximing LIU, Yingjiu LI, Pingjian WANG, Wen-Tao ZHU, and Zhan WANG

Employing Smartwatch for Enhanced Password Authentication

Bing Chang[1,2,3], Ximing Liu[4], Yingjiu Li[4], Pingjian Wang[1,2,3], Wen-Tao Zhu[1,2(✉)], and Zhan Wang[5]

[1] State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
wtzhu@ieee.org

[2] Data Assurance and Communication Security Research Center,
Chinese Academy of Sciences, Beijing, China

[3] School of Cyber Security, University of Chinese Academy of Sciences,
Beijing, China

[4] School of Information Systems, Singapore Management University,
Singapore, Singapore

[5] RealTime Invent, Inc, Beijing, China

Abstract. This paper presents an enhanced password authentication scheme by systematically exploiting the motion sensors in a smartwatch. We extract unique features from the sensor data when a smartwatch bearer types his/her password (or PIN), and train certain machine learning classifiers using these features. We then implement smartwatch-aided password authentication using the classifiers. Our scheme 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 on the developed prototype so as to evaluate its feasibility and performance. Experimental results show that the best classifier for our system is the Bagged Decision Trees, for which the 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.

Keywords: Wearable devices · User authentication · Sensor · Machine learning

1 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. Equipped with rich sensors, smartwatches can be used in many applications such as monitoring heart rate, steps taken and calories burned. In recent studies, smartwatch sensor data are exploited to conduct keystroke inference attacks. When a user types on a keyboard or PIN pad wearing a smartwatch, an attacker may access the user's smartwatch sensor data and infer what the user types from the sensor data, thus compromising user's security or privacy [7,8,15,16]. While prior studies reveal that smartwatch sensor data can be exploited for launching attacks, we further reveal that such data contain unique features of users' typing behaviors beyond what users type, and thus can be exploited to enhance password authentication against known password attacks and keystroke imitation attacks. An enhance password authentication system is still reliable even if an adversary knows a user's password and can imitate the user's keystroke dynamics.

In particular, we extract unique features from smartwatch sensor data when a smartwatch bearer types his/her password (or PIN), and train certain machine learning classifiers using these features. We then design a smartwatch-aided password authentication scheme using the trained classifiers. We show that our scheme can defend against the keystroke imitation attack proposed in [9]. Even if an adversary obtains users' passwords and imitates users' keystroke dynamics, our system can still differentiate imitators from legitimate users by analyzing smartwatch sensor data during password entry.

Our scheme is user-friendly since it does not require users to perform any additional actions when typing passwords or PINs other than wearing their smartwatches. The performance of our scheme is evaluated in an IRB-approved user study with 51 participants. Five widely used classification algorithms are evaluated in which the best performer turns out to be the Bagged Decision Trees. Rigorous experiments on the accuracy of our scheme are conducted in our user study, yielding 4.58% FRR and 0.12 FAR on the QWERTY keyboard, and 6.13% FRR and 0.16% FAR on the numeric keypad. It is also shown that the keystroke imitation attack has insignificant impact to the accuracy of our scheme.

2 Background

2.1 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 extensively used in user behavioral characterization, including sensor-based keystroke inference [7,8,15,16]. 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. Instead of using such sensor data for keystroke inference, we use them for user authentication.

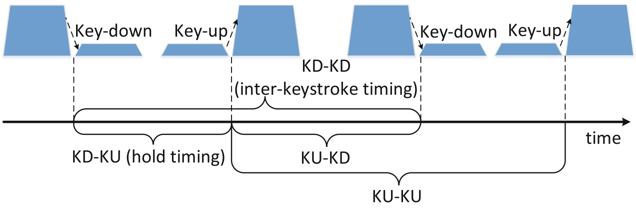


Fig. 1. Keystroke timings used in keystroke dynamics techniques.

2.2 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 [6] and shown in Fig. 1. Keystroke dynamics features have been used to identify and authenticate users on both hardware keyboards [1,5,17] and software keyboards [13,14]. However, Meng et al. [9] revealed that a training interface can be set up to help attackers imitate users' keystroke dynamics, which makes it unsafe to use 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.

3 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 victim's keystroke as shown in [9]. In attacks, the attacker may wear a same kind of smartwatch and access to a same kind of keyboard as the victim's.

4 Scheme Design

4.1 Overview

The main goal of our design is to demonstrate that using smartwatches can help enhance the security of password authentication systems. Password authentication systems suffer from password observation attacks such as shoulder surfing and key logger in which attackers may obtain users' passwords. 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 as a victim, the attacker's hand motion is different from the user's. The accelerometer and gyroscope in a smartwatch can be used to track its wearer's hand motion during password 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.

Our system takes as input the password and the raw sensor data (e.g., acceleration, angular velocity) from the smartwatch worn on a user's wrist. The password and the raw sensor data are sent to our server for verification. The password is for the conventional password authentication while the raw sensor data are processed to further verify the user. Our system consists of two phases, the training phase and the detection phase. During the training phase, the password is registered for the conventional password 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 password first. If the typed password 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 all the known user profiles to identify whether the password is typed by the legitimate user. A user is authenticated only if both the password is correct and the typing pattern matches the user's profile.

As the conventional password 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 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 help authenticate the user. We extract unique features from the sensor data and train certain classifiers using the features as user profiles. The classifiers are used to authenticate users.

4.2 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 from accelerometer and gyroscope are streams of timestamped real values along three axes. For a given timestamp, t, the accelerometer data are in the form of $\boldsymbol{a}(t) = (a_x, a_y, a_z)$ while the gyroscope data are in the form of $\boldsymbol{\omega}(t) = (\omega_x, \omega_y, \omega_z)$. 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 $9.8\,\mathrm{m/s^2}$ along the z-axis. We can 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 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 10 min to type in the password 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.

4.3 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 [3]. The detail of these features have been documented previously in report [2]. Since there are three axes for both sensors, we obtain a vector of 36 elements 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 the user according to the classifier.

4.4 Supervised Learning and Detection

In the training phase, after the system extracts all the features, it trains the classifier using the features. In Sect. 5, we evaluate five widely used classifica-

Table 1. Extracted features.

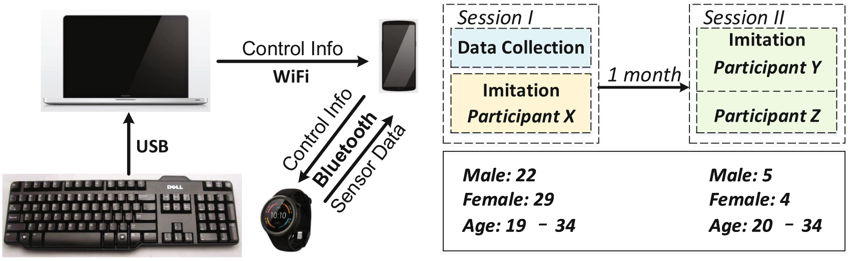
Feature	Description
Mean strength	Arithmetic mean of the signal strength
Standard deviation	Standard deviation of the signal strength
Average deviation	Average deviation from mean
Skewness	Measure of asymmetry about mean
Kurtosis	Measure of the flatness or spikiness of a distribution
RMS	Square root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

tion algorithms, including Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), Bagged Decision Trees (Matlab's Treebagger model), Naive Bayes classifier and Discriminant Analysis classifier. We discover that the Bagged Decision Trees outperforms the other classifiers in Sect. 5. In the detection phase, a feature vector is extracted from the sensor data of a user's smartwatch, and fed into a trained classifier which generates the authentication result: whether the user is legitimate or illegitimate.

5 Evaluation

5.1 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, a laptop, a mobile phone and a smartwatch. Figure 2(a) illustrates our data collection system. A user just needs to wear a smartwatch and type in passwords on the laptop using the keyboard. The sensor data will be recorded automatically on the mobile phone.



(a) Experimental setup.

(b) User study process.

Fig. 2. Overview of our experiment.

Keyboard. We use a DELL SK-8115 keyboard for data collection. Users type passwords on the QWERTY keyboard and type PINs on the numeric keypad.

Laptop. The laptop is a MacBook Pro with an Intel i7 2.7 GHz processor with 8 GiB RAM, running an Ubuntu 14.04 64-bit virtual machine. We obtain the source code of the data collection system from the authors of [9] and rebuild their system. We modify 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. The user interface is a web page for users to type in passwords or PINs according to a prompt. When the system shows the prompt, it sends out a "start" message to the mobile phone at the same time. Once 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 finishes its recording of the sensor data and sends the 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 contain the timestamps and the values of 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 the 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. Note that the sampling frequency (50 Hz) is the highest on Moto 360 sport and we specify the *SENSOR_DELAY_FASTEST* flag at the sensor listener registration time to accomplish this.

5.2 User Study

Figure 2(b) 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

[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.

years old). 26 of them are 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 password. The QWERTY keyboard password is used to simulate that a user types a password on a standard keyboard while the keypad password is used to simulate that a user types a PIN on a keypad of ATM or POS terminal. The participants are required to wear smartwatches on their right wrists, and type in QWERTY passwords with both hands while 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 standing. We choose the QWERTY keyboard password and the keypad password as "ths.ouR2" and "924673", respectively in our experiment. The password "ths.ouR2" is a strong password used in previous work [9] while "924673" is a randomly generated PIN. The participants are required to type each password 100 times.

Keystroke Imitation Attack. In order to find some participants who are good at keystroke imitation and test whether our system can resist the imitation attack proposed in [9], we arrange an imitation phase in both Session I and Session II. We rebuild the system proposed in [9] 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. 3) and a score to indicate the differences between this input and the target typing pattern. Note that in Fig. 3, the circles mean the hold timings and the bars mean the inter-keystroke timings. A participant can adjust his/her typing according to the interface. 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, the participants are required to imitate other two participants' typing

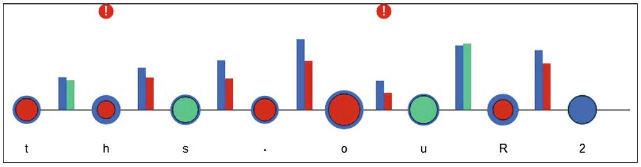


Fig. 3. The interface of the imitation system (Fig. 3 in [9]). The circles mean the hold timings and the bars mean the inter-keystroke timings. The blue circles and bars are the target's timing information. Imitators can adjust their typing according to the differences between their timing information and the target's. (Color figure online)

patterns of “ths.ouR2” and “924673”. Similar to the conclusion drawn in [9], we discover that it is unable to distinguish these imitators from the victims according to the keystroke dynamics only, but we aim to find out whether it is possible to distinguish them by analyzing the sensor data taken from smartwatches.

5.3 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 “924673”. The participants are required to type in the same password as we aim to find out whether the sensor data can help differentiate them. After deleting the invalid data caused by system error, we extract the features according to Sect. 4.3 and obtain 4,789 feature vectors for the QWERTY keyboard and 4,868 feature vectors for the numeric keyboard. For each participant, we have approximately 93 feature vectors, including the mean values of the three axis of the accelerometer. We delete some outliers based on 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 4,789 feature vectors for the QWERTY keyboard and 609 out of 4,868 feature vectors for the numeric keypad. To access the performance, we use FAR (false acceptance rate), which indicates the fraction of imposter access attempts identified as valid users, and FRR (false rejection rate), which indicates the fraction of valid user attempts identified as impostors.

Performance of Different Classifiers. We evaluate the performance of five 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 Tables 2 and 3. In the 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.

Impact of Different Sensors. To understand the impact of different sensors, we also test our system using the data from one sensor only. Figure 4 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

Table 2. FRR in different scenarios (BDT: Bagged Decision Trees; DAC: Discriminant Analysis Classifier).

	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%

Table 3. FAR in different scenarios (BDT: Bagged Decision Trees; DAC: Discriminant Analysis Classifier).

	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%

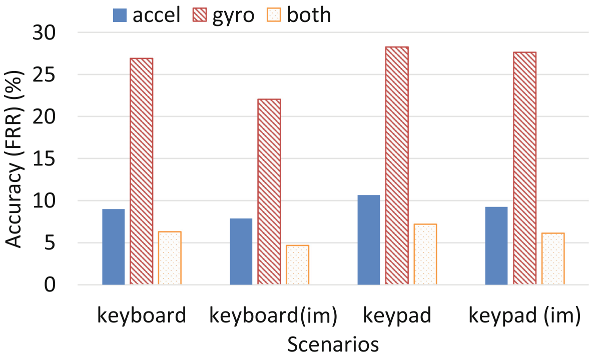


Fig. 4. The accuracy (FRR) when using only one sensor.

gyroscope only results in 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 system.

5.4 Defending Against Keystroke Imitation Attack

To test whether our system can defend against the keystroke imitation attack proposed in [9], we process the sensor data when nine selected participants imitate others. 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 patterns on QWERTY keyboard and numeric keypad. We have reproduced the results of [9] with these nine participants. After trained with the system proposed in [9], the selected participants can imitate the target typing patterns in a success rate higher than 90%. To test whether our system can differentiate original users from imitators, we first extract the features from the sensor data collected from the original users and from the imitators, respectively. We then *randomly* select 50% of the feature vectors from each person to train the classifiers. The other 50% of the feature vectors are used as the testing set. The results are shown in Tables 2 and 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 keystroke imitation attack has little impact on our system.

6 Related Work

Sensor Information Leaks on Smartwatches. Previous research has studied sensor information leaks on smartwatches [7,8,15,16]. Wang et al. [16] propose a linguistic model based system to infer user typed words on a standard keyboard using accelerometer and gyroscope data in smartwatches. Their system is unable to deal with non-contextual inputs, such as passwords and PIN sequences, since the system relies on a linguistic model. Liu et al. [7] make use of the sensors in smartwatches, including accelerometer and microphone, to infer users' inputs on keyboards or POS terminals. Their approach is based on machine-learning techniques and training of hand movements between keystrokes. Maiti et al. [8] also make use of the sensors in smartwatches to infer users' input, and present a protection framework to regulate sensor access. Wang et al. [15] propose a training-free and contextual-free system to infer users' input by exploiting the sensors in wearable devices, including accelerometers, gyroscopes and magnetometers. Their system does not require any training or contextual information.

Keystroke Dynamics. Tremendous efforts have been made on using keystroke dynamics as biometrics (e.g., [10–12]). However, Meng et al. [9] propose a feedback and training interface, called *Mimesis*, which can help one person imitate another through incremental adjustment of typing patterns. If an attacker can obtain the information of a victim's typing pattern, the attacker can imitate the victim with the help of *Mimesis*. This makes keystroke dynamics based authentication systems insecure. Giuffrida et al. [4] propose sensor-enhanced keystroke dynamics to authenticate users typing on mobile devices. They use motion sensor data to characterize users typing behavior and use machine learning techniques to perform user authentication. However, their system works on mobile devices only. When users type passwords on standard keyboards or PINs on keypads, their system does not work. In comparison, our solution is more generic, since the smartwatch is worn on the user's wrist. Wherever the user types, our system can obtain the sensor data which reflect the motions of the user's wrist, and thus authenticate the user by analyzing the sensor data.

7 Conclusion

In this paper, we propose to use smartwatches to track the motion of users' wrists when they type passwords on standard keyboards or numeric keypads. In particular, we present a novel enhanced password authentication scheme by systematically exploiting the motion sensors in the users' smartwatches. The experimental results show that the best classifier for our system 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. Our work paves the way for authenticating users using smartwatch sensor data and machine learning techniques.

Acknowledgment. This work is supported by the Singapore National Research Foundation under NCR Award Number NRF2014NCR-NCR001-012. This work is partially supported by Youth Innovation Promotion Association, CAS.

References

1. Clarke, N.L., Furnell, S., Lines, B., Reynolds, P.L.: Keystroke dynamics on a mobile handset: a feasibility study. Inf. Manag. Comput. Secur. **11**(4), 161–166 (2003)
2. Das, A., Borisov, N., Caesar, M.: Exploring ways to mitigate sensor-based smartphone fingerprinting. CoRR, abs/1503.01874 (2015)
3. Das, A., Borisov, N., Caesar, M.: Tracking mobile web users through motion sensors: attacks and defenses. In: Proceedings of the 23rd NDSS (2016)
4. Giuffrida, C., Majdanik, K., Conti, M., Bos, H.: I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics. In: Dietrich, S. (ed.) DIMVA 2014. LNCS, vol. 8550, pp. 92–111. Springer, Cham (2014). doi:10.1007/978-3-319-08509-8_6
5. Karatzouni, S., Clarke, N.: Keystroke analysis for thumb-based keyboards on mobile devices. In: Venter, H., Eloff, M., Labuschagne, L., Eloff, J., Solms, R. (eds.) SEC 2007. IIFIP, vol. 232, pp. 253–263. Springer, Boston, MA (2007). doi:10.1007/978-0-387-72367-9_22
6. Killourhy, K., Maxion, R.: Why did my detector do *That*?!. In: Jha, S., Sommer, R., Kreibich, C. (eds.) RAID 2010. LNCS, vol. 6307, pp. 256–276. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15512-3_14
7. Liu, X., Zhou, Z., Diao, W., Li, Z., Zhang, K.: When good becomes evil: keystroke inference with smartwatch. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 1273–1285. ACM (2015)
8. Maiti, A., Armbruster, O., Jadliwala, M., He, J.: Smartwatch-based keystroke inference attacks and context-aware protection mechanisms. In: Proceedings of the 11th ACM Asia Conference on Computer and Communications Security (2016)

9. Meng, T.C., Gupta, P., Gao, D.: I can be you: questioning the use of keystroke dynamics as biometrics. In: Proceedings of the 20th NDSS (2013)
10. Monrose, F., Rubin, A.: Authentication via keystroke dynamics. In: Proceedings of the 4th ACM Conference on Computer and Communications Security (1997)
11. Monrose, F., Rubin, A.D.: Keystroke dynamics as a biometric for authentication. Future Gen. Comput. Syst. **16**(4), 351–359 (2000)
12. Peacock, A., Ke, X., Wilkerson, M.: Typing patterns: a key to user identification. IEEE Secur. Privacy **2**(5), 40–47 (2004)
13. Tasia, C.-J., Chang, T.-Y., Cheng, P.-C., Lin, J.-H.: Two novel biometric features in keystroke dynamics authentication systems for touch screen devices. Secur. Commun. Netw. **7**(4), 750–758 (2014)
14. Trojahn, M., Ortmeier, F.: Biometric authentication through a virtual keyboard for smartphones. Int. J. Comput. Sci. Inf. Technol. **4**(5), 1 (2012)
15. Wang, C., Guo, X., Wang, Y., Chen, Y., Liu, B.: Friend or foe?: your wearable devices reveal your personal pin. In: Proceedings of the 11th ACM Asia Conference on Computer and Communications Security, pp. 189–200. ACM (2016)
16. Wang, H., Lai, T.T.T., Roy Choudhury, R.: Mole: motion leaks through smartwatch sensors. In: Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, pp. 155–166. ACM (2015)
17. Zahid, S., Shahzad, M., Khayam, S.A., Farooq, M.: Keystroke-based user identification on smart phones. In: Kirda, E., Jha, S., Balzarotti, D. (eds.) RAID 2009. LNCS, vol. 5758, pp. 224–243. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04342-0_12