(Smart)Watch Your Taps: Side-Channel Keystroke Inference Attacks using Smartwatches

Anindya Maiti, Murtuza Jadliwala, Jibo He
Wichita State University
{axmaiti, murtuza.jadliwala, jibo.he}@wichita.edu

Igor Bilogrevic
Google
ibilogrevic@google.com

Abstract
In this paper, we investigate the feasibility of keystroke inference attacks on handheld numeric touchpads by using smartwatch motion sensors as a side-channel. The proposed attack approach employs supervised learning techniques to accurately map the uniqueness in the captured wrist movements to each individual keystroke. Experimental evaluation shows that keystroke inference using smartwatch motion sensors is not only fairly accurate, but also better than similar attacks previously demonstrated using smartphone motion sensors.

Author Keywords
Smartwatch; keystroke; sensor; wearable; privacy.

ACM Classification Keywords
K.6.5 Management of Computing and Information Systems: Security and Protection

INTRODUCTION

The popularity of wearable devices such as smartwatches is soaring as they enable a plethora of novel context-based applications. However, the presence of a diverse set of on-board sensors also provides an additional attack surface to malicious applications on these devices. Security and privacy threats on handheld smartphones that take advantage of such sensors as side-channels have received significant attention in the literature. Notable examples include keystroke inference [3, 7, 9], activity identification [5] and location inference [4] attacks. As most modern mobile operating systems introduced stringent access controls on front end sensors, such as microphones, cameras and GPS, adversaries shifted attention to sensors which cannot be actively disengaged by users (e.g., accelerometer and gyroscope). Typically, handheld device usage is highly intermittent and such devices spend a majority of time in a constrained (e.g., in users' dress pocket) or activity-less (e.g., on a table) setting where most on-board sensors are partially or completely non-functional. Contrary to this, wearable device usage is much more persistent as they are constantly carried by the users on their body. This makes wearable devices a more desirable target for a variety of side-channel attacks. We hypothesize that, if access to wearable sensor data is not appropriately regulated, it can be used as a side-channel to infer sensitive user information.

Side-channel security vulnerabilities in smart wearables have not received much (if any) attention. We make one of the first contributions in this direction. Our first contribution is a comprehensive evaluation of the feasibility and effectiveness of keystroke inference attacks on handheld numeric touchpads by using smartwatch motion sensors as a side-channel. Numeric touchpads are typically targeted by adversaries for obtaining sensitive information such as security pins and credit card numbers. Our proposed attack comprises of first training (using supervised learning) appropriate classification models to learn the uniqueness in wrist motion caused during each individual keystroke, and then using the trained classifiers to infer unlabeled test keystrokes. During preliminary experiments, we observed that keystroke induced motion data captured by smartwatch and smartphone sensors differ significantly. Consequently, our second contribution is to thoroughly assess how significantly smartwatch motion sensors elevate the threat of keystroke inference, compared to similar attacks using only smartphone motion data [3, 7, 9].

RELATED WORK

Keystroke inference attacks using electromagnetic [8] and acoustic emanations [2] have already been investigated. While the requirement of sophisticated hardware prevents casual adversaries from carrying out electromagnetic emanation based attacks, the presence of microphones on most modern mobile devices makes acoustic attacks much more practical than previously thought. However, as touchscreen keypads emanate very weak acoustic signals, inference attacks using them is very difficult. Additionally, requirement of undisturbed eavesdropping is another major obstacle in using electromagnetic and acoustic emanations for such attacks. As a workaround to the above limitations, smartphone motion detection sensors have been used to recover keystroke events on the device. For instance, *TouchLogger* [3] utilizes change in orientation angles of the smartphone, as captured by its accelerometer, to extract appropriate features for keystroke inference. Similarly, *ACCessory* [7] also attempts to infer keystrokes using the smartphone accelerometer data by employing multiple supervised learning techniques. Alternatively, *TapLogger* [9] automates the training and logging phases and attempts to work stealthily on the smartphone.

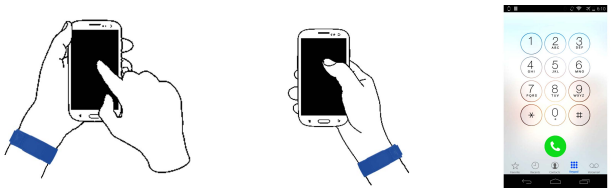
ATTACK DESCRIPTION

In this paper, we focus on two popular typing (or tapping) scenarios for our inference attack. We consider a user typing

ISWC '15, September 7– 11, 2015, Osaka, Japan.

http://dx.doi.org/10.1145/2802083.2808397

on a smartphone's numeric touchscreen keypad while wearing a smartwatch on one of his/her hand. In the first case, smartwatch and smartphone are on the same hand and the user types with the hand not holding the smartphone (see Figure 1(a)). In the second case, smartwatch and smartphone are again on the same hand and the user types with a finger (generally, thumb) of the smartphone holding hand (see Figure 1(b)). In the chosen scenarios, the action of tapping a key results in a unique motion of the wrist (on the smartphone holding hand) for each keystroke, which can be captured by the motion sensors (e.g., accelerometer and gyroscope) of the smartwatch on the user's wrist. While there exist several other typing or tapping scenarios [1], the chosen cases allow us to make an equitable comparison when the attack uses data from the smartphone's on-board motion sensors.



(a) (b) (c)

Figure 1: Smartwatch and smartphone on same hand and (a) **N**on-**H**olding **H**and **T**yping (NHHT), or (b) **H**olding **H**and **T**yping (HHT) (c) numeric keypad used in our experiments.

We assume an adversary whose goal is to infer a target's keystrokes on a generic smartphone numeric keypad or touchpad (as shown in Figure 1(c)), based on the wrist movements perceptible by the target's smartwatch motion sensors. The adversary may gain access to the target's smartwatch by installing a malicious application on it which records the activity of the on-board accelerometer and gyroscope sensors. This step can be achieved by exploiting known software vulnerabilities or by tricking the victim into installing malicious code, e.g., using a Trojan horse. Based on the fact that most common smartwatch operating systems (e.g., Google's Android Wear, Apple's watchOS, etc.) do not implement access control and/or user notification for motion sensor usage, the malicious application may have unrestricted and undetected access to the on-board accelerometer and gyroscope. As a result, the infected smartwatch can act as an eavesdropping device that the targets themselves may place on their wrist, and unsuspectingly have it on their wrist while typing on a smartphone. The malicious application also maintains a covert communication channel with the adversary, and uploads the collected wrist motion data using this channel. We assume that the adversary also has sufficient off-site storage and computational resources to download the raw sensor data, extract keystroke events, and classify the keystrokes using trained classifiers (as explained next). For comparison with the smartphone data based attacks, we assume similar adversarial capabilities and actions for the smartphone.

ATTACK EXPERIMENT

Our attack experiment consists of a learning phase followed by an attack phase. Both phases go through similar steps, as outlined below (Figure 2), with the learning phase culminating in training while the attack phase in classification.

Data Collection: We begin our experiments by collecting keystroke associated motion data from 12 voluntary participants[1]. Participants in our experiments were instructed to type on a numeric keypad (Figure 1(c)) of a smartphone while wearing a smartwatch. Each participant typed 400 keys in NHHT and 400 keys in the HHT setting. An audio stream of uniformly distributed random numbers between 0 to 9 guided the participants in typing. Participants were also given optional breaks, during which they were allowed to set down the phone on the table and some participants even went out of the room. However, they returned to approximately the same holding position after the break. A custom data collection application that continuously samples linear accelerometer measurements is installed on both the smartwatch and the smartphone, and is running in the background during the experiments. In the learning phase, the data collection application also records the keystroke ground truth for labeling purposes.

Pre-processing: A careful analysis of the sampled linear accelerometer readings reveals that the data samples corresponding to the keystroke events are clearly separated from one another with small but clear inactive time regions, in both typing cases. It is observed that movement due to a keystroke subsides after approximately 350 $msecs$, thus eighteen samples (at 50 Hz sampling frequency) sufficiently captures all motion features related to a keystroke. During preprocessing, the collected sequence of linear acceleration samples from both the smartphone and the smartwatch are dissected into individual keystrokes, based on peaks in linear acceleration on each axis individually, and then the sample with highest magnitude is mapped as the fourth time sample in each keystroke segment. After removing erroneous[2] and double-tapped[3] keystrokes by using an automated script, an equal number (30) of keystroke data per key were used in the experiments, with each keystroke consisting of 18 linear accelerometer samples. It should be noted that we do not always have 100 erroneous keystrokes; after removing erroneous data, we equalize the remaining data to select exactly 30 keystrokes per key (i.e., a total of 300 for 10 numeric keys). Out of a total of 300 keystrokes, 67% (200, 20 per key) are randomly chosen as the training set, and the remaining 33% (100, 10 per key) as the test set.

Feature Extraction: Our attack infers keystrokes based on features captured from the wrist motion (and palm for smartphone). A good feature vector should be similar with other feature vectors of the same key, simultaneously being distinguishable between feature vectors of other keys. Based on the location of a key on the screen, the degree of movement caused by a tap varies on each of the X, Y, and Z axis of the linear accelerometer. However, this movement remains

[1]Our data collection experiments have been approved by the Institutional Review Board (IRB) at Wichita State University.

[2]Mismatch between the ground truth logged on the phone and the audio stream logged on the PC.

[3]Tap very close to an erroneous tap.

consistent for the same key tap. Thus, in this initial work we only use the eighteen linear accelerometer samples of each keystroke event to build a 54-dimensional (18 accelerometer magnitudes over 3 axes) feature vector. Additionally, in the learning phase, the feature vectors are also labeled, using the ground truth recorded by the data collection application.

Training and Classification: A total of 2400 labeled keystroke feature vectors (200 per participant) were used as the training set, and 1200 unlabeled keystroke feature vectors (100 per participant) as the test set, each for smartwatch and smartphone. We model the keystroke inference problem as a multi-class classification problem. Labeled feature vectors are used to train classifiers in the learning phase, whereas unlabeled feature vectors are mapped to the "closest" matching class by the already trained classifiers in the attack phase. To train our classifiers, we employ three different classification algorithms that are appropriate given the properties of our features: (i) simple linear regression (SLR), (ii) random forest (RF) and (iii) k-nearest neighbors (k-NN). We observe the presence of such interactions in our dataset that makes it highly likely that each of the feature (in the vector) may contribute independently to the output. As a result, we feel that linear or distance-based classification algorithms (such as, SLR, RF and k-NN) may perform better in our case. However, other classification algorithms can also be used by taking precautions to avoid over-fitting. The SLR training process is optimized using the LogitBoost algorithm. For k-NN, we set $k = 1$ (based on the observed distribution of features) and we use a linear search algorithm with distance and weight inversely proportional to each other. In RF, the depth of the trees is left unrestricted, number of random trees is set to 100, and number of features in each tree is chosen to be 6.

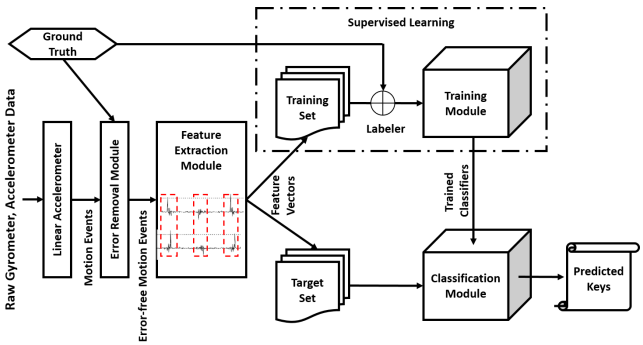


Figure 2: Overview of our attack framework.

Experimental Setup

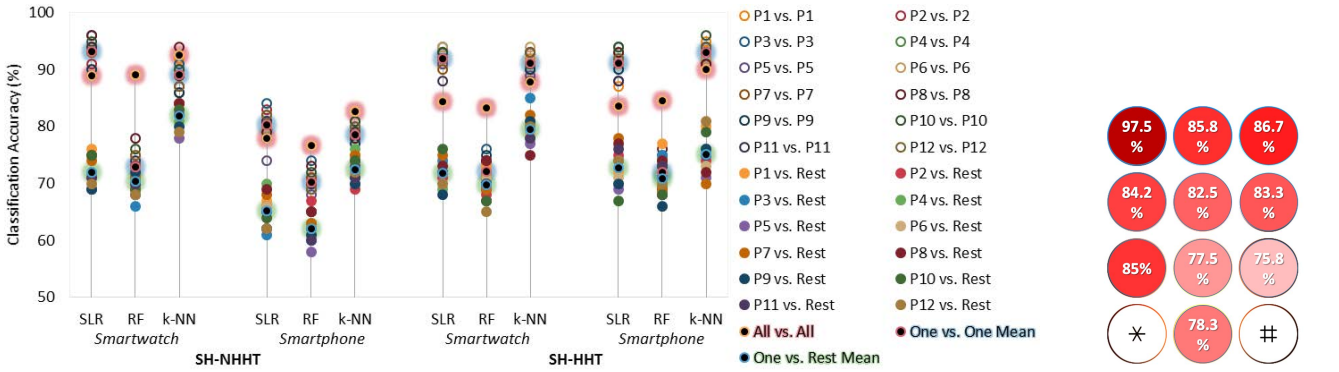
Our data collection experiments involve 12 participants, aged between 19-32 years. The identity of these participants are anonymized as $P_1, P_2, \ldots, P_{12}$. We employ a Samsung Gear Live smartwatch equipped with an InvenSense MP92M 9-axis Gyro + Accelerometer + Compass sensor. Smartwatch was worn on left hand for NHHT (Figure 1(a)) and on right hand for HHT (Figure 1(b)). For the smartphone, we use a Motorola XT1028. Linear accelerometer of both the watch and phone was sampled at 50 Hz and the smartphone was held in the same hand on which smartwatch was worn. We used the Weka 3.7.12 [6] libraries for both training and testing the classifiers. As our goal in this work is to show the feasibility of the proposed attacks, we employ only a single state-of-the-art smartwatch and smartphone in our experiments. However, different hardware may have sensors operating at different sensitivities, which may affect the accuracy of the sampled data slightly. This will obviously positively or negatively affect the accuracy of the proposed attacks, but we do not anticipate its impact to be too significant. We plan to investigate our attacks on additional hardware in the future.

EVALUATION AND RESULTS

We comparatively evaluate the classification accuracy of our classifiers for the following three types of training datasets:

- ***One vs. One***: Here we measure the percentage of successful inferences on an individual participant, with classifiers trained from the training set of the same participant. Target set size is 100 (10 per key) and training set size is 200 (20 per key), each for smartphone and smartwatch. *One vs. One* is not only a best case scenario, but also represents how the attack will perform if the adversary is able to collect target-specific training data.
- ***One vs. Rest***: Here we measure the percentage of successful inferences on an individual participant, with classifiers trained from the training set of the rest of the participants. Target set size is 100 (10 per key) and training set size is 2200 (220 per key), each for smartphone and smartwatch. *One vs. Rest* is a typical scenario where the adversary has a target, but is unable to obtain labeled training data from the target.
- ***All vs. All***: Here we measure the percentage of successful inferences on all participants, with classifiers trained from training set of all participants. Target set size is 1200 (120 per key) and training set size is 2400 (240 per key), each for smartphone and smartwatch. *All vs. All* is helpful in understanding how our attack framework will perform if the adversary constructs a heterogeneous training data set to infer keystrokes from multiple non-specific targets.

Results: Consolidated classification results are shown in Figure 3(a). For inference using smartwatch data, SLR has high *One vs. One* classification accuracy of more than 90%, but was relatively less accurate in *One vs. Rest* (still with a classification accuracy of around 70%). Similarly, classification accuracy of RF in *One vs. One* and *One vs. Rest* was around 70%, but achieved higher classification accuracy in *All vs. All* (more than 80%). *One vs. One* and *All vs. All* accuracy of k-NN is high (close to 90%), while its *One vs. Rest* classification accuracy is moderate (close to 80%). It is also observed that the keystroke inference attacks in NHHT resulted in much better classification accuracy on the smartwatch consistently, while in HHT classification accuracy results are mixed, and nearly equal, on both smartwatch and smartphone. *In summary, these results:* ***(i)*** *validate our hypothesis that smartwatch motion sensors can be employed as effective side-channels to infer private information, and* ***(ii)*** *the threat of motion-based keystroke inference is moderately amplified due to smartwatches.*



(a) (b)

Figure 3: (a) Classification accuracy for *One vs. One*, *One vs. Rest*, and *All vs. All*. (b) *All vs. All* SLR classification accuracy for individual keys in HHT using smartphone data.

An interesting pattern for classification accuracy can be observed for inference using only smartphone data in HHT. We observe that the classification accuracy of certain keys is distinctly higher than others (see Fig. 3(b)). Interestingly, this occurrence is not recognizable for the smartwatch dataset. This may be due to the fact that keys farther away from the thumb impels the user to bend the phone towards the thumb. As a result, significantly greater movement of the phone occurs, compared to keys that are near the thumb.

Reduced Sampling Frequency: We also briefly investigate how our attack will perform with half the sampling rate (25 Hz), a more realistic scenario for low-cost wearables, equipped with less precise sensors. We repeat the experiments with smartwatch data sampled at a reduced frequency, and Table 1 shows the accuracy of our attacks for both the NHHT and HHT scenarios. Results indicate that classification accuracy drops with reduction in sampling frequency, but percentage of successful classification is fairly substantial even at a sampling frequency of 25 Hz.

	NHHT			HHT		
	SLR	RF	k-NN	SLR	RF	k-NN
P_i vs. P_i	78%	66%	81%	81%	66%	83%
P_i vs. Rest	62%	52%	71%	63%	59%	77%
All vs. All	80%	78%	81%	79%	75%	82%

Table 1: Classification accuracy when sampling rate was halved to 25 Hz, results averaged over all 12 participants.

CONCLUSION AND FUTURE WORK

Our results indicate that keystroke inference attacks using typing-induced motion data captured by smartwatch sensors is highly effective. As part of future work, we will investigate the effects of posture, fatigue and other disturbances on the effectiveness of our attack. We will also study other typing styles (e.g., two-handed), which may require completely different attack strategies. Our current attack approach can also be easily extended to an alphabetic or qwerty soft keyboard. However, due to the size and close positioning of keys in such keyboards, we expect a much larger classification error in this case. As in this preliminary work we focus only on side-channel attacks due to malicious eavesdropping applications installed on smartwatches, we assume that the adversary will not have access to the sensors on the smartphone (on which the user is typing). Thus, we do not train our classifiers by fusing data from both the watch and the phone. Intuitively, the fusion of watch and phone data will likely lead to better inference results for the adversary. We will investigate this as part of future work.

REFERENCES

1. How do users really hold mobile devices? (2013). `http://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php`.
2. Asonov, D., and Agrawal, R. Keyboard acoustic emanations. In *IEEE S & P* (2004).
3. Cai, L., and Chen, H. Touchlogger: Inferring keystrokes on touch screen from smartphone motion. In *HotSec* (2011).
4. Gao, X., Firner, B., Sugrim, S., Kaiser-Pendergrast, V., Yang, Y., and Lindqvist, J. Elastic pathing: You speed is enough to track you. In *ACM UbiComp* (2014).
5. Ghosh, A., and Riccardi, G. Recognizing human activities from smartphone sensor signals. In *ACM Multimedia* (2014).
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The weka data mining software: an update. *In ACM SigKDD Explorations Newsletter 11*, 1 (2009), 10–18.
7. Owusu, E., Han, J., Das, S., Perrig, A., and Zhang, J. ACCessory: Password Inference Using Accelerometers on Smartphones. In *ACM HotMobile* (2012).
8. Vuagnoux, M., and Pasini, S. Compromising electromagnetic emanations of wired and wireless keyboards. In *USENIX Security* (2009).
9. Xu, Z., Bai, K., and Zhu, S. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *ACM WiSec* (2012).