Effective Mobile Web User Fingerprinting via Motion Sensors

Zhiju Yang
Department of Computer Science
Colorado School of Mines
zhijuyang@mines.edu

Rui Zhao
Colorado School of Mines and
University of Nebraska Omaha
ruizhao@unomaha.edu

Chuan Yue
Department of Computer Science
Colorado School of Mines
chuanyue@mines.edu

***Abstract*—Motion sensors are often equipped on smartphones to enable rich app functionality and interactivity. However, they can also be exploited by attackers as powerful side-channels to compromise users' security and privacy due to the unrestricted sensor data access on modern smartphone platforms. In this paper, we investigate motion sensor based user fingerprinting attacks that can be pervasively performed to severely compromise the privacy of mobile web users. We formulate our user fingerprinting attacks as a typical multi-class classification problem, and design a framework with unified classifiers for effectively performing the attacks. We implement our attacking framework and evaluate it using the motion sensor data collected from 20 volunteers. The evaluation results demonstrate that our attacks are indeed very effective. For example, the user fingerprinting accuracy is higher than 85% by using a classifier unified from seven randomly selected individual classifiers each trained with the motion sensor data of only 10 letter keystrokes.**

***Keywords*-Smartphone; Motion Sensor; Web User; Fingerprinting; Attacks**

I. Introduction

To enable rich functionality and interactivity, modern mobile devices such as smartphones are ubiquitously equipped with various motion sensors. For example, gyroscope sensors facilitate gesture-based interactions especially in game apps, and accelerometer sensors help users monitor their physical exercises. However, while those sensors can help improve user experience significantly, they can also be exploited by attackers to compromise users' security and privacy due to the unrestricted motion sensor data access on modern smartphone platforms [1].

In general, such attacks can be app-based or web-based. App-based attacks require the installation of a malicious app or compromise of a legitimate app on a user's smartphone. In contrast, web-based attacks do not require any app installation or compromise, thus can be pervasively performed to incur severe security and privacy risks to millions of users. Web-based attacks can occur because JavaScript code on webpages can access motion sensor data without requiring any permission or configuration from a user based on the W3C (World Wide Web Consortium) specification [2].

In this paper, we focus on investigating motion sensor based mobile web user fingerprinting attacks. Fingerprinting is the most challenging type of web tracking attacks as analyzed by Eckersley in the influential Panopticlick study [3]. Web tracking aims to compromise the privacy of web users by associating their identities with their browsing activities on different websites. Traditionally, stateful HTTP cookies are used as the dominant web tracking technique [4]. In recent years, more advanced stateful web tracking techniques such as supercookies as well as stateless browser fingerprinting and device fingerprinting techniques have also been widely used to track web and mobile users (Section II-B). Stateless techniques are more challenging than stateful techniques for users to protect against because they do not rely on any stateful information and cannot be easily avoided.

Our user fingerprinting attacks are further ***different*** from traditional browser or device fingerprinting attacks. Ours are direct user fingerprinting attacks that are based purely on users' behavioral biometrics derived from the motion sensor data associated with different user activities [1]. The intuition behind our attacks is that users' behavioral characteristics on web interactions, which are inherited to the motion sensor data, can be constructed as fingerprints for identifying individual users [1].

We formulate our user fingerprinting attacks as a typical multi-class classification problem, in which different users are different classes with unique fingerprints. We design an attacking framework consisting of data collection, data preprocessing, feature extraction, and model training components. More importantly, we design a unified classification mechanism based on joint probability calculation to effectively perform user fingerprinting attacks.

To evaluate the effectiveness of our user fingerprinting attacks, we recruited 20 volunteers to tap letters, digits, and special characters on webpages using smartphones for collecting the corresponding keystroke motion sensor data. Our evaluation results demonstrate that our attacks can be very effective. For example, when each individual classifier is trained with the motion sensor data of only 10 letter keystrokes, the user fingerprinting accuracy of a unified classifier is higher than 73% even with only three randomly selected individual classifiers (i.e., only 30 data samples in total are used), and is higher than 85% with seven randomly selected individual classifiers; when each individual classifier is trained with the motion sensor data of around 100 letter keystrokes, the user fingerprinting accuracy of a unified classifier is close to 90% even with only three randomly

selected individual classifiers, and is higher than 95% with seven randomly selected individual classifiers.

Our evaluation results have a few important implications for attackers. First, using a unified classifier based on joint probability calculation can significantly improve user fingerprinting accuracy upon using an individual classifier. Second, unified classifiers constructed from a small number of individual classifiers can already be very effective on user fingerprinting. Third, selecting best performing individual classifiers instead of random ones to construct a unified classifier can achieve better user fingerprinting accuracy.

Our paper makes two major contributions. First, we designed a framework with a unified classification mechanism for effectively performing motion sensor based mobile web user fingerprinting attacks (Section III). Second, we implemented the framework, and evaluated the effectiveness of our user fingerprinting attacks using the motion sensor data collected from a group of 20 users (Section IV).

II. Background and Related Work

In this section, we first introduce some background information about smartphone motion sensors. We then review the related work on web tracking.

A. Smartphone Motion Sensors

1) Motion Sensors and Behavioral Biometrics: Accelerometer and gyroscope are two major types of motion sensors widely deployed on modern smartphones. A user's interactions with a smarthpone such as soft-keyboard tapping will make the device move along certain directions and rotate along certain axes, and the corresponding acceleration forces and rotation rates can be precisely captured by the accelerometer and gyroscope sensors.

An accelerometer measures the acceleration forces of a device in the unit of meter per second squared (m/s^2) along three directions: left and right (i.e., x-axis), forward and backward (i.e., y-axis), and up and down (i.e., z-axis) [2]. A gyroscope measures the rotation rates of a device in the unit of degrees per second along the three axes.

When a user performs the same interaction (such as tapping the same character on the soft-keyboard) with a smartphone multiple times, the corresponding motion sensor data are often consistent. However, such data are often different among users due to their biological and behavioral differences. For example, Alice might tap on a soft-keyboard in a gentle and quick manner, while Bob might do it slowly with more force. In essence, our fingerprinting attacks will exploit such behavioral biometrics of users derived from the motion sensor data to identify individual users.

2) Motion Sensor Data Collection using JavaScript: On smartphone platforms such as Android and iOS, mobile apps can use APIs to access motion sensor data without requiring any permission. Furthermore, webpages rendered on mobile browsers or WebView components of mobile apps can also access motion sensor data using JavaScript without requiring any permission as documented in the W3C DeviceOrientation Event Specification [2]. Indeed, collecting motion sensor data using JavaScript code on a webpage can be very easily performed by registering a handler for the "devicemotion" event on a webpage.

B. Related Work

HTTP cookies are still widely used on both first-party and third-party websites to track users [4]. However, in recent years, many advanced stateful or stateless tracking techniques (such as using supercookies, HTTP Etag, HTML5 local storage, and HTML canvas APIs [5], [6], [7], [8], [9]) have also become popular on the web.

Fingerprinting is one of the most challenging types of web tracking attacks. As analyzed by Eckersley in the influential Panopticlick study [3], for a user, avoiding being tracked (1) by basic stateful techniques such as HTTP cookies is tricky (e.g., need to configure the appropriate settings in browsers), (2) by advanced stateful techniques such as supercookies is harder (e.g., need to find ways to disable them), and (3) by stateless fingerprinting techniques will be most challenging. The Panopticlick study [3] is more about ***browser fingerprinting*** because the fingerprints are constructed based on the characteristics of the browsers.

Researchers have also investigated ***device fingerprinting*** by exploiting the manufacturing imperfections in smartphone hardware. For example, researchers showed that smartphones can be accurately fingerprinted by exploiting the manufacturing imperfections in speakers and microphones [10], [11], [12], or in accelerometer and gyroscope motion sensors [13], [14], [11].

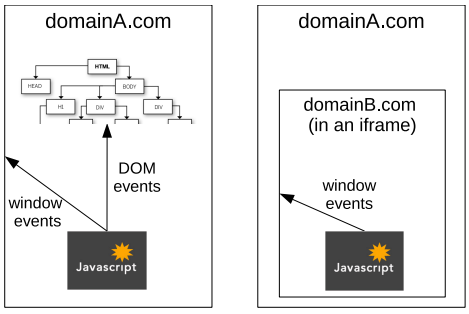
The fingerprinting attacks that we investigate in this paper are ***different*** from and ***complementary*** to those existing browser and device fingerprinting attacks as highlighted in Section I. Ours are direct ***user fingerprinting*** attacks that are based purely on users' behavioral biometrics derived from the motion sensor data associated with different user interactions [1]. Our techniques are valuable especially for accurately performing targeted attacks against a small group of users even if other web tracking techniques including browser and device fingerprinting become ineffective.

III. Design of the Attacks

In this section, we introduce the threat model of our user fingerprinting attacks, present the overview and details of our attacking framework, and discuss the strategies that attackers may take.

A. Threat Model

The basic threat model for our user fingerprinting attacks is that a first-party website (whose domain name is displayed in the browser's address bar) or a third-party website (whose documents are embedded, e.g., as *iframe* documents, in a



(a) First-party (b) Third-party

Figure 1: Motion sensor based user fingerprinting attacks.

first-party website) can exploit browsing behavioral biometrics obtained from motion sensors to track a smartphone user even if first-party and third-party persistent cookies are disabled, supercookies are removed, and browser or device fingerprinting risks are avoided [1].

We highlight two types of user fingerprinting attacks, first-party and third-party, as shown in Figure 1 [1]. Both of them can occur on Android and iOS platforms without requiring any app installation, app configuration, or user permission.

Specifically, in the first-party user fingerprinting attacks (Figure 1a), JavaScript code included or embedded in a first-party website can register to receive the window associated events for obtaining device motion data [1]; although it can also exploit behavioral biometrics by monitoring DOM (Document Object Model) events, motion sensor data are more generally accessible and more representative of the characteristics of individual users. ***Even first-party user fingerprinting attacks can raise severe privacy concerns*** because a first-party website may either purposefully authorize a third-party website to learn about its users or accidentally allow a third-party website to do so due to insecure JavaScript inclusion practices [15]; meanwhile, users may not want to be tracked by a first-party website in the first place. Those are also the reasons why all the popular web browsers provide the privacy configuration features such as disabling first-party cookies and sending the "Do Not Track" requests to websites [16].

In the third-party user fingerprinting attacks (Figure 1b), JavaScript code in an iframe child document can register to receive the window events (from the window object of the iframe document) for obtaining device motion data [1], although it cannot access the DOM events of the first-party webpage due to the same-origin policy [17]. ***Such third-party user fingerprinting attacks can directly and severely compromise the privacy of mobile web users, and they can indeed be pervasively performed.*** For example, third-party advertisements are often included in iframes on millions of first-party websites. Malicious or compromised advertising websites definitely have the strong motivations to perform such attacks; legitimate behavioral advertising websites that

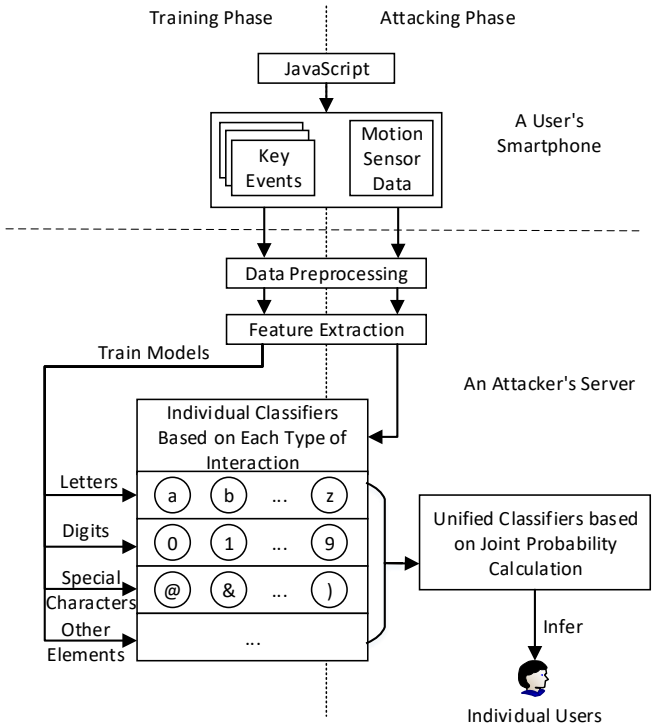


Figure 2: The framework for our user fingerprinting attacks.

infer user privacy for profit also have the strong motivations to do so.

B. Framework Overview

We formulate our user fingerprinting attacks as a typical multi-class classification problem, in which different users are different classes with unique fingerprints. We design an attacking framework consisting of data collection, data preprocessing, feature extraction, and model training components as shown in Figure 2. More importantly, we design a unified classification mechanism based on joint probability calculation to effectively perform user fingerprinting attacks.

In more details, our framework (1) collects acceleration force and rotation rate motion sensor data when a user is interacting with a webpage, (2) preprocesses the motion sensor data for each individual user interaction, (3) extracts time and frequency domain statistical features from those data, (4) trains individual classifiers or models based on the data of each type of interaction, and (5) applies our unified classification mechanism to perform the final user fingerprinting. Step (1) occurs on a user's smartphone, while steps (2), (3), (4), and (5) occur on an attacker's server.

Such a framework will allow attackers to accurately perform targeted attacks against a small group of users. In the training phase, an attacker (who controls either a first-party or a third-party website) collects smartphone motion sensor data when each user of the targeted group performs certain web interactions. The attacker is able to label each individual user with the corresponding motion sensor data in an initial

web session simply using traditional techniques such as a session identifier or a session cookie. The labeled data will be then leveraged to train the classifiers corresponding to each type of user interaction using some supervised machine learning algorithms.

In the attacking phase, when users in the targeted group perform certain web interactions in new web sessions with different session identifiers or session cookies, even if they disable persistent cookies, remove supercookies, and use different browsers or devices, the machine learning classifiers and fingerprints constructed in the training phase can still be used by any attackers to identify each individual user based on the newly collected motion sensor data, and associate the user's activities in new and old web sessions on different websites.

Web interactions can be input field filling through soft-keyboard, link or button clicking, page swiping or scrolling, and so on. Their corresponding motion sensor data can all be exploited by attackers in the training and attacking phases. For example, in input field filling, soft-keyboard tapping on each specific letter, digit, or special character is a specific type of interaction, for which an individual classifier will be trained and used.

C. Data Preprocessing and Feature Extraction

The motion sensor data collected from a user's smartphone by JavaScript will be preprocessed before performing feature extraction and training classifiers.

The major preprocessing task is data segmentation, in which motion sensor data for each individual user interaction (e.g., soft-keyboard tapping) will be segmented based on the timestamp of the corresponding event (e.g., *keydown*) to represent this interaction. This segmentation is similar to that in [18], [19]. Meanwhile, very noisy data samples will also be discarded in this preprocessing step.

Based on the existing research such as [18], [20], [21], [19] on processing segmented motion sensor data, we extract 302 statistical features to characterize the acceleration forces and rotation rates along the three axes (x, y, and z) in both the time domain and the frequency domain. Those statistical features include the *maximum value*, *minimum value*, *mean value*, *variance*, *standard derivation*, *root mean square*, *skewness*, *kurtosis*, *area under curve*, *cross correlation between pairs of individual values*, and so on, as detailed in [22].

D. Classifier Training and Unification

Based on the extracted features, our framework will first train individual classifiers and then construct unified classifiers.

1) Individual Classifier Training: Each individual classifier is a multi-class classifier. It is trained using some machine learning algorithm to identify multiple users (i.e., multiple classes) in a targeted group, and its training is based purely on the motion sensor data of a particular type of interaction. Fox example, based on the motion sensor data of tapping the letter 'a' on the soft-keyboard by a group of users, an attacker can train a "letter-a" classifier; similarly, the attacker can train other individual classifiers such as "letter-b" and "digit-8" classifiers for the same group of users based on their corresponding interactions performed in some initial web sessions.

Here we require an individual classifier to be a probabilistic classifier. That is, it can output a probability distribution of identification for a set of ***m*** users or classes as shown in Sequence 1.

$$(\ Prob(C_1^j), \ \ldots \ , \ Prob(C_m^j) \ ) \tag{1}$$

Here $Prob(C_i^j)$ represents the probability of identifying a user as the user i by an individual classifier C^j based on the newly collected motion sensor data for the type-j interaction. The output of the individual classifier C^j is the most probable class or user $\hat{i}$ with the highest probability as shown in Formula 2.

$$\hat{i} = \arg \max_{i \in \{1,m\}} \{Prob(C_1^j), \ \ldots \ , Prob(C_i^j), \ \ldots \ , \ Prob(C_m^j)\} \tag{2}$$

For example, the "letter-a" classifier may output a probability distribution of identification, such as (15%, 5%, 40%, 30%, 10%), for a targeted group of five users from the user 1 to the user 5, and its identification result is the user 3 (i.e., $\hat{i} = 3$); similarly, the output of the "letter-b" classifier might be (20%, 10%, 20%, 30%, 20%) for the five users, and its identification result is the user 4.

2) Unified Classifier Construction: Although each individual classifier can already be used to independently identify the users in a targeted group, we ***hypothesize*** that making an identification decision jointly by multiple individual classifiers has the potential to further increase the user fingerprinting accuracy. This hypothesis is ***intuitive*** as the old proverb says that "two heads are better than one."

To test the hypothesis, we design ***a unified classification mechanism*** and construct ***unified classifiers*** based on joint probability calculation for performing user fingerprinting attacks. A unified classifier combines the classification results of two or more individual classifiers.

We formalize the ***joint probability calculation*** and the ***unified classification*** using the following Formula 3 and Formula 4, respectively.

$$Joint\_Prob(U_i^k) = \prod_{j=1}^{k} Prob(C_i^j). \tag{3}$$

Here in Formula 3, $Joint\_Prob(U_i^k)$ is the probability of identifying a user as the user i by a unified classifier U^k; it is the multiplication from $Prob(C_i^1)$ to $Prob(C_i^k)$,

representing the joint probability of identifying a user as the user i by certain k individual classifiers.

$$\hat{i} = \arg \max_{i \in \{1,m\}} \{Joint\_Prob(U_1^k),\ \ldots\ , \\ Joint\_Prob(U_i^k),\ \ldots\ ,\ Joint\_Prob(U_m^k)\} \tag{4}$$

Here in Formula 4, $\hat{i}$ is the optimal value of the argument i that maximizes the value of the objective function $Joint\_Prob(U_i^k)$ for m users from the user 1 to the user m in the targeted group. In other words, the user $\hat{i}$ is the final user fingerprinting result jointly derived from certain k individual classifiers.

Continuing the aforementioned example and assuming that an attacker has trained a set of two individual classifiers {"letter-a", "letter-b"} for identifying the five users in the same targeted group, a unified "letter-ab" classifier calculates the joint probability from the results of these two individual classifiers to identify each of the five users. The values of $Joint\_Prob(U_i^2)$ for i from 1 to 5 are (0.03, 0.005, 0.08, **0.09**, 0.02); therefore, the value of $\hat{i}$ is 4 and the user 4 is the final user fingerprinting result jointly derived from the two individual classifiers "letter-a" and "letter-b".

3) Unification Strategy: From the attackers' perspective, the attacking accuracy and efficiency are their major concerns. Assuming that in total, attackers can collect and use the motion sensor data of ***n*** types of interactions to train n individual classifiers, they can use any k out of these n classifiers to construct a unified classifier. There is a trade-off between selecting a large value of k and selecting a small value. The larger the value of k, the more accurate a unified classifier can potentially be. However, because a larger value of k also requires more data collection effort, it is desirable for attackers to select a small value for k. Meanwhile, individual classifiers may often have different identification accuracy; therefore, for a given small value of k, it is further desirable to select certain k individual classifiers with good (instead of poor) performance to construct a unified classifier.

IV. Evaluation of the Attacks

We have implemented a complete framework for performing our user fingerprinting attacks. At the client side, we wrote JavaScript code to collect the motion sensor data. At the server side, we wrote Java code, and used the R statistical computing and the Weka machine learning packages to preprocess the collected data, extract features, train individual classifiers, and construct unified classifiers.

In the following subsections, we first describe the user fingerprinting accuracy metrics and the datasets used in our evaluation; we then evaluate the accuracy of different individual classifiers and unified classifiers.

A. User Fingerprinting Accuracy Metrics

F-measure is the harmonic mean of precision and recall, and is often chosen by researchers as a major metric to represent the overall accuracy of a classifier. Specifically, we use the balanced F-measure score (also called the F_1 score) shown in Formula 5 as the user fingerprinting accuracy metric in our evaluation for both individual and unified classifiers.

$$\text{F-measure} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{5}$$

Precision is the ratio of true positives (TPs) over the sum of true positives (TPs) and false positives (FPs). Recall is the ratio of true positives (TPs) over the sum of true positives (TPs) and false negatives (FNs). Therefore, we first count the three metrics: TPs, FPs, and FNs, and then calculate the F-measure score for each classifier. In this work, each classifier is a multi-class classifier, and these three metrics have the following meanings:

- TP: An instance predicted as belonging to a class indeed belongs to that class (e.g., the predicted user Alice is indeed Alice).
- FP: An instance predicted as belonging to a class indeed does not belong to that class (e.g., the predicted user Alice is indeed not Alice but another user such as Bob).
- FN: An instance of a given class is incorrectly predicted as not belonging to that class (e.g., user Alice is incorrectly predicted as not Alice but another user such as Bob).

B. Data Collection and Datasets

We focus on collecting the motion sensor data of the participants' soft-keyboard tapping on letters, digits, and special characters to evaluate the effectiveness of our attacks.

With the IRB (institutional review board) approval, we recruited 20 volunteers (14 male and 6 female students and faculty members) to participate in our data collection. In the recruitment process, potential participants were administered the informed consent, in which they were told that they do not need to tap any sensitive information. We asked the participants to use their own or our provided Android smartphones, and use the Google Chrome web browser with the default Google Keyboard to perform tapping tasks.

We created four webpages that are capable of collecting motion sensor data using JavaScript. Each webpage displays one different letter pangram and one different digit pangram, and provides two input fields for participants to tap the two displayed pangrams, respectively. A letter pangram on our webpage is a sentence using every lower-case letter of the alphabet exactly once, and a digit pangram contains 10 unique digits and 3 special characters ('@', '&', and ')') located at the left, middle, and right part of the soft-keyboard. We refer to the 26 letters as the ***letter charset***, and

refer to the 10 digits and 3 special characters as the ***digit charset***. Because the characters in these two charsets are displayed on two separate layouts of the Google Keyboard, we will present all our results separately for these two charsets in this paper.

We asked each participant to visit the four webpages and tap the displayed pangrams in each session. We asked each participant to complete at least 25 sessions, but allowed them to do so in two weeks. Therefore, we obtained from each participant over 100 keystroke samples of motion sensor data for each of the 39 types of interactions (i.e., corresponding to the 26 lower-case letters, 10 digits, and 3 special characters). We use the ***Complete Dataset*** to refer to this original dataset with over 78,000 keystroke samples of motion sensor data for all the 20 participants.

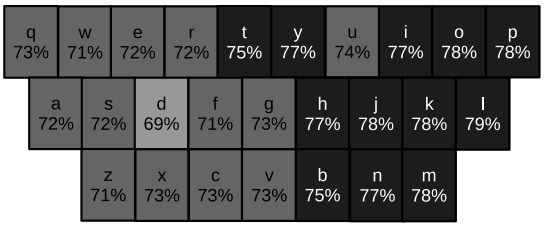
In real attacking scenarios, an attacker might not be able to collect over 100 keystroke samples of motion sensor data for a given type of interaction to train their classifiers; therefore, to evaluate the accuracy of our attacks with less amount of data, we further randomly selected 10%, 20%, ..., until 90% of data samples of each interaction type for each participant from the *Complete Dataset*, and created nine new datasets referred to as ***10% Dataset***, ***20% Dataset***, ..., and ***90% Dataset***. Note that we repeated this random selection process five times, and our evaluation results for these datasets presented in this paper are averaged results over five times.

C. Accuracy of the Individual Classifiers

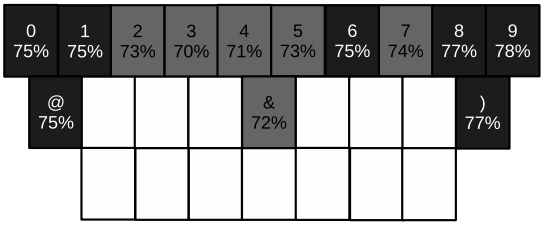
Based on these datasets, we experiment with a variety of machine learning algorithms using the Weka package. These algorithms include Bayes Network, Naive Bayes, Logistic Regression, Support Vector Machines (SVM), K-Nearest Neighbors, Decision Tree, and Random Forests. Our experimental results show that the default Sequential Minimal Optimization (SMO) for training SVM classifiers (with default parameters and the default linear kernel) outperforms other algorithms with their default settings in user fingerprinting accuracy. Therefore, we only present and analyze the results from SVM classifiers in this paper.

Each of the aforementioned 10 datasets consists of 39 types of interactions for all the 20 participants; therefore, from each dataset, 39 individual SVM classifiers are trained. More specifically, each classifier is trained and assessed using 10-fold cross validation, and the F-measure score (Formula 5) is calculated to represent the user fingerprinting accuracy. Since we have 20 users in our evaluation, the value of m is 20 in Sequence 1 and Formula 2.

Figure 3 illustrates the user fingerprinting accuracy of the 39 individual classifiers trained and assessed using the *Complete Dataset*, in which Figure 3a is for the 26 letters while Figure 3b is for the 10 digits and the 3 special characters. Overall, all these individual classifiers perform reasonably well. The darker the background of a character



(a) Accuracy of the 26 individual classifiers from the letter charset



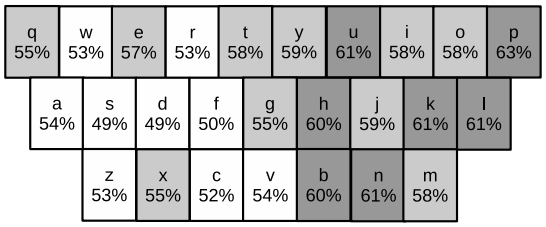
(b) Accuracy of the 13 individual classifiers from the digit charset

Figure 3: User fingerprinting accuracy of the 39 individual classifiers trained and assessed using the *Complete Dataset*.

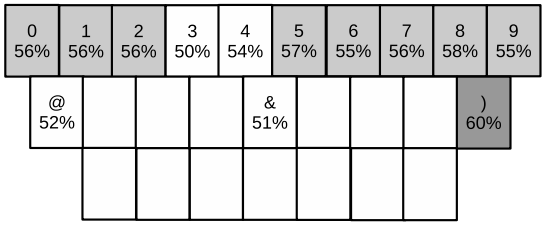
in the figure, the higher the corresponding user fingerprinting accuracy. For example, in Figure 3a, the highest and lowest F-measure scores are 79% and 69% for the "letter-l" and "letter-d" classifiers, respectively; in Figure 3b, the highest and lowest F-measure scores are 78% and 70% for the "digit-9" and "digit-3" classifiers, respectively. In addition, the F-measure scores do not differ significantly between these two subfigures.

Similarly, Figure 4 illustrates the user fingerprinting accuracy of the 39 individual classifiers trained and assessed using the *10% Dataset*. All these individual classifiers still perform reasonably well, but are inferior to the corresponding classifiers shown in Figure 3 (and to the classifiers for other datasets such as *20% Dataset* that are not presented here). For example, in Figure 4a, the highest and lowest F-measure scores are 63% and 49% for the "letter-p" and "letter-s" classifiers, respectively; in Figure 4b, the highest and lowest F-measure scores are 60% and 50% for the "character-)" and "digit-3" classifiers, respectively. Note that the accuracy of the "letter-d" classifier is a little bit higher than that of the "letter-s" classifier on the fractional part. The F-measure scores do not differ significantly between these two subfigures either.

An interesting observation is that the classifiers of the keys located on the right side of the keyboard often achieve a higher user fingerprinting accuracy than the classifiers of those on the left side. One possible explanation is that the majority of the participants used their left hands to hold the smartphone while tapping using their right hands, thus the characteristics of tapping on the right side keys were better



(a) Accuracy of the 26 individual classifiers from the letter charset



(b) Accuracy of the 13 individual classifiers from the digit charset

Figure 4: User fingerprinting accuracy of the 39 individual classifiers trained and assessed using the *10% Dataset*.

captured.

Table I further lists the accuracy ranking of the individual classifiers shown in Figures 3a, 3b, 4a, and 4b, respectively. This ranking will be leveraged in the construction of the unified classifiers in Section IV-D.

Table I: Accuracy ranking of the individual classifiers from the most accurate one (left) to the least accurate one (right).

	Letter charset	Digit charset
Complete Dataset	lpojmkiynhbtugqcvxeasrfzwd	9)8106@752&43
10% Dataset	pulknbhyjomitexgqvarzwcfds	)851702694@&3

D. Accuracy of the Unified Classifiers

Assuming that in total, n individual classifiers are available for us to construct a unified classifier U^k as shown in Formulas 3 and 4, we need to first determine the number of individual classifiers k, where $1 \leq k \leq n$, to be used. For a given value of k, we need to further determine which one of the $C(n,k) = \frac{n!}{k!(n-k)!}$ possible unique sets of k individual classifiers will be selected to use.

Leveraging the accuracy ranking of the individual classifiers shown in Table I, we construct and evaluate unified classifiers for a given k value in ***three ways***: (1) selecting the set of the k best performing individual classifiers, (2) selecting the set of the k worst performing individual classifiers, and (3) randomly selecting a set of k individual classifiers multiple times to obtain the averaged accuracy. For example, with $k = 3$ and based on the ranking of the 26 individual classifiers from the letter charset using the

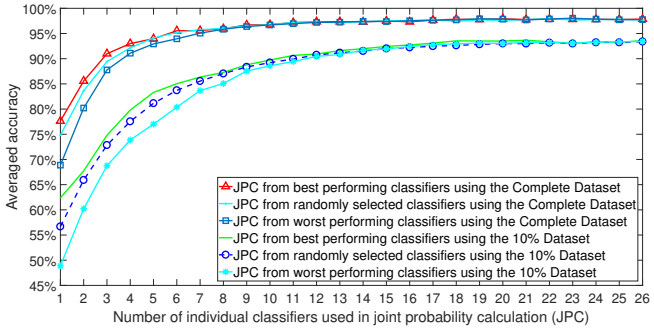


Figure 5: User fingerprinting accuracy of different unified classifiers constructed from k out of 26 individual classifiers based on the letter charset.

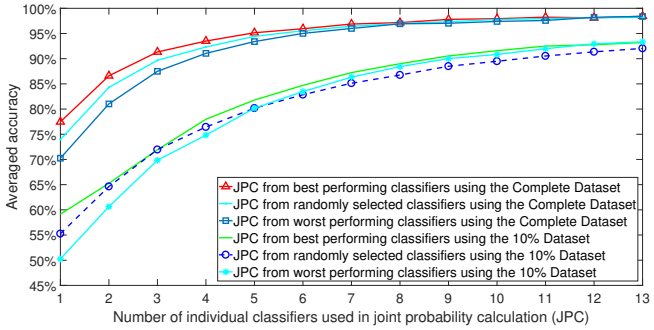


Figure 6: User fingerprinting accuracy of different unified classifiers constructed from k out of 13 individual classifiers based on the digit charset.

Complete Dataset, the set of the 3 best performing individual classifiers is {"letter-l", "letter-p", "letter-o"}, the set of the 3 worst performing individual classifiers is {"letter-z", "letter-w", "letter-d"}, and the random selections are from the $C(26,3)$ possible unique sets.

In the training and assessing of each individual classifier using 10-fold cross validation, we recorded the probability distribution of identification as shown in Sequence 1 and recorded the corresponding data samples used in the classifier testing. This recording is beneficial for three reasons. First, the joint probability calculation for any unified classifier can be directly performed based on the recorded information without the need to retrain any individual classifier. Second, if different unified classifiers share some common individual classifiers, this recording ensures that the identical individual classifiers are used so that the accuracy comparison can be fairly performed. Third, it allows us to easily select many random combinations of the data samples used in testing individual classifiers to assess each unified classifier and report its averaged accuracy.

Figure 5 illustrates the user fingerprinting accuracy of different unified classifiers constructed from k out of 26 individual classifiers based on the letter charset, where k

is from 1 to 26. Figure 6 illustrates the user fingerprinting accuracy of different unified classifiers constructed from k out of 13 individual classifiers based on the digit charset, where k is from 1 to 13. In both figures, the results for $k = 1$ are equivalent to those for without unifying the individual classifiers, and it is evident that unified classifiers can significantly improve user fingerprinting accuracy upon individual classifiers. The more individual classifiers being used to construct a unified classifier, the larger the improvement is. The improvement is dramatic when the value of k is increasing from 1 to 7, and becomes moderate when the value of k continues to increase.

Unified classifiers with small k values can already be very effective on user fingerprinting as shown in both Figure 5 and Figure 6. For example, from the curves corresponding to the *10% Dataset* (i.e., each individual classifier is trained with the motion sensor data of only 10 keystrokes) shown in Figure 5, the user fingerprinting accuracy of a unified classifier is higher than 73% even with only three (i.e., $k = 3$) randomly selected individual classifiers (i.e., only 30 data samples in total are used with 9 of them for training each individual classifier in the 10-fold cross validation), and is higher than 85% with seven (i.e., $k = 7$) randomly selected individual classifiers. From the curves corresponding to the *Complete Dataset* shown in Figure 5, the user fingerprinting accuracy of a unified classifier is close to 90% even with only three randomly selected individual classifiers, and is higher than 95% with seven randomly selected individual classifiers.

V. Conclusion

In this paper, we highlighted and investigated motion sensor based user fingerprinting attacks that can be pervasively performed to severely compromise the privacy of mobile web users. Complementary to the existing browser and device fingerprinting attacks, our attacks are purely based on users' behavioral biometrics derived from the motion sensor data associated with different web interactions. We formulated our user fingerprinting attacks as a typical multi-class classification problem, and built a complete framework for performing the attacks. We designed a unified classification mechanism based on joint probability calculation to effectively perform user fingerprinting attacks. We evaluated our framework using the motion sensor data collected from 20 participants, and the results show that our attacks are indeed very effective. We found that using a unified classifier based on joint probability calculation can significantly improve user fingerprinting accuracy upon using an individual classifier, and unification from a small number of individual classifiers can already be very effective.

Acknowledgment

This research was supported in part by the NSF grant DGE-1619841 and NSA grant H98230-17-1-0403.

References

[1] C. Yue, "Sensor-based mobile web fingerprinting and cross-site input inference attacks," in *IEEE Workshop on Mobile Security Technologies (MoST)*, 2016.

[2] 2018, https://w3c.github.io/deviceorientation/spec-source-orientation.html.

[3] P. Eckersley, "How Unique is Your Web Browser?" in *International Conference on Privacy Enhancing Technologies (PETS)*, 2010.

[4] F. Roesner, T. Kohno, and D. Wetherall, "Detecting and Defending Against Third-party Tracking on the Web," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2012.

[5] A. Soltani, S. Canty, Q. Mayo, L. Thomas, and C. J. Hoofnagle, "Flash Cookies and Privacy," in *AAAI Spring Symposium: Intelligent Information Privacy Management*, 2010.

[6] M. Ayenson, D. Wambach, A. Soltani, N. Good, and C. Hoofnagle, 2011, http://dx.doi.org/10.2139/ssrn.1898390.

[7] K. Mowery and H. Shacham, "Pixel Perfect: Fingerprinting Canvas in HTML5," in *Web 2.0 Security & Privacy (W2SP) workshop*, 2012.

[8] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The Web Never Forgets: Persistent Tracking Mechanisms in the Wild," in *ACM Conference on Computer and Communications Security (CCS)*, 2014, pp. 674–689.

[9] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting," in *IEEE Symposium on Security and Privacy*, 2013.

[10] A. Das, N. Borisov, and M. Caesar, "Do You Hear What I Hear?: Fingerprinting Smart Devices Through Embedded Acoustic Components," in *ACM Conference on Computer and Communications Security (CCS)*, 2014, pp. 441–452.

[11] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile Device Identification via Sensor Fingerprinting," *CoRR*, vol. abs/1408.1416, 2014. [Online]. Available: http://arxiv.org/abs/1408.1416

[12] Z. Zhou, W. Diao, X. Liu, and K. Zhang, "Acoustic fingerprinting revisited: Generate stable device id stealthily with inaudible sound," in *ACM Conference on Computer and Communications Security (CCS)*, 2014, pp. 429–440.

[13] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable," in *Network and Distributed System Security Symposium (NDSS)*, 2014.

[14] A. Das, N. Borisov, and M. Caesar, "Tracking Mobile Web Users Through Motion Sensors: Attacks and Defenses," in *Network and Distributed System Security Symposium*, 2016.

[15] C. Yue and H. Wang, "A Measurement Study of Insecure Javascript Practices on the Web," *ACM Transactions on the Web (TWEB)*, vol. 7, no. 2, pp. 7:1–7:39, 2013.

[16] 2018, http://www.w3.org/2011/tracking-protection/.

[17] (2018) https://www.w3.org/Security/wiki/Same_Origin_Policy.

[18] A. J. Aviv, B. Sapp, M. Blaze, and J. M. Smith, "Practicality of Accelerometer Side Channels on Smartphones," in *Annual Computer Security Applications Conference (ACSAC)*, 2012.

[19] Z. Xu, K. Bai, and S. Zhu, "TapLogger: Inferring User Inputs on Smartphone Touchscreens Using On-board Motion Sensors," in *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, 2012, pp. 113–124.

[20] L. Cai and H. Chen, "On the Practicality of Motion Based Keystroke Inference Attack," in *International Conference on Trust and Trustworthy Computing (TRUST)*, 2012.

[21] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury, "Tapprints: Your Finger Taps Have Fingerprints," in *International Conference on Mobile Systems, Applications, & Services (MobiSys)*, 2012.

[22] R. Zhao, C. Yue, and Q. Han, "Cross-site Input Inference Attacks on Mobile Web Users," in *Proceedings of International Conference on Security and Privacy in Communication Networks (SecureComm)*, 2017.