Beat-PIN: A User Authentication Mechanism for Wearable Devices Through Secret Beats

Ben Hutchins
CSE Department,
University of Nevada, Reno
benrhutchins@gmail.com

Anudeep Reddy
CSE Department,
University of Nevada, Reno
ganudeep4@gmail.com

Wenqiang Jin
CSE Department,
University of Nevada, Reno
wenqjin@nevada.unr.edu

Michael Zhou
CSE Department,
University of Nevada, Reno
michaelzhou@nevada.unr.edu

Ming Li
CSE Department,
University of Nevada, Reno
mingli@unr.edu

Lei Yang
CSE Department,
University of Nevada, Reno
leiy@unr.edu

ABSTRACT

Wearable devices that capture users' rich information regarding their daily activities have unmet authentication needs. Today's solutions, which primarily rely on indirect authentication mechanisms via users' smartphones, thus cumbersome and susceptible to adversary intrusions. Even though there have been some efforts trying to fill this gap, they either rely on some superior sensors, such as cameras and electrocardiogram (ECG) pads, or are awkward to use, e.g., users are asked to perform some pre-defined movement/gesture for authentication. Therefore, an authentication mechanism for wearable devices that is accurate, robust, light-weight and convenient is in dire need.

In this paper, we present the design, implementation and evaluation of a user authentication mechanism, *Beat-PIN*, for wearable devices that are equipped with touch sensors. A user's password is a set of recorded beats when he/she taps the device. We call this rhythm-based password as a beat-PIN, which is represented by the timing of its beats. To achieve high authentication accuracy with short training overhead, we propose a novel classification method. Through extensive experimental evaluation with 124 participants, we show that our mechanism can achieve the average EER of 7.2% with only 7 training samples. Besides, its login time is as low as 1.7s. We also show that its average power consumption for training and login is 337.2mW and 181.4mW, separately, which is lower than that for most common operations on smartwatches. More importantly, we provide a theoretical analysis over the beat-PIN's raw space size and show that it is much larger than that of digit-PINs and traditional passwords.

CCS CONCEPTS

• **Security and privacy → Authentication**;

KEYWORDS

User authentication; wearable devices; rhythm-based passwords

ACM Reference Format:
Ben Hutchins, Anudeep Reddy, Wenqiang Jin, Michael Zhou, Ming Li, and Lei Yang. 2018. *Beat-PIN*: A User Authentication Mechanism for Wearable Devices Through Secret Beats. In *ASIA CCS '18: 2018 ACM Asia Conference on Computer and Communications Security, June 4–8, 2018, Incheon, Republic of Korea.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3196494.3196543

ASIA CCS '18, June 4–8, 2018, Incheon, Republic of Korea

ACM ISBN 978-1-4503-5576-6/18/06…$15.00
https://doi.org/10.1145/3196494.3196543

1 INTRODUCTION

Recently, we are witnessing a new trend in the mobile device market. Users are showing an increasing interest in wearing mobile devices to enhance the quality of life in a way that smartphones alone cannot deliver. These devices, which include smartwatches, wrist bands, smartglasses, and so on, can sense, collect, and upload physiological data in a 24×7 manner. Besides, they can also help users perform many tasks, such as checking incoming text messages and viewing urgent information, in a much more convenient way. According to recent market reports [2, 5], it is forecasted that the yearly shipment of wearable devices will reach 200 million in the year 2019. Moreover, the wearable technology market is expected to reach a value of $57,653 million by 2022, which is almost 3 times of that in 2016 ($19,633 million) [10].

With the high penetration to people's daily life, wearable devices read and store rich information regarding their owners. For example, in the domain of smart health, the current key application of wearable devices focuses on tracking activities or vital signals from the wearers, whose sensitive data, such as heartbeats, weight, blood pressure, are collected by wearable devices. Similarly, another wearable application in the area of tracking is child monitoring. A simple device, e.g., smartwatch, is worn by a kid, broadcasting his/her location to a parent or guardian. While the objective is clearly to increase the safety of the child, the data itself represents highly sensitive information that attackers could seek to compromise. In either case above, wearers are unwilling to disclose the information stored in their wearables to others without permission. Therefore, user authentication are of critical needs for wearable devices.

Before we explore existing authentication methods for wearable devices, we'd like to first briefly cover existing solutions for general mobile devices, especially smartphones and tablets. Generally,

the most commonly used authentication methods on mobile devices are password/PIN/pattern-based methods and biometric-based methods. However, none of them is really suitable for wearables. Typing passwords or drawing patterns on wearable devices can be rather cumbersome due to their small input/output units. Collecting and recognizing physiological biometrics, such as fingerprint, facial characteristics, hand/finger geometry, iris and retina, requires specialized sensing hardware and dedicated processing resources which are always missing in wearables. Due to the fact that many of these sensors are even larger than the size of wearables themselves, it is also impractical to equip them in wearables.

Due to the above reasons, authentication on current wearable devices relies on a so-called "indirect mechanism", where users can log in to their wearables through smartphones. For this purpose, the wearable device has to be registered and paired to the mobile device. Besides, both devices should be carried by the user, which can be highly inconvenient in practice. The security of this approach is also in question. Password is only required for the first time during the paring process and the device will automatically be paired for later use without checking the user's identity. Such one-time authentication is over optimistic by assuming that there is no misbonding between the device and the user. Some devices including Google Glass [4] and FitBit's health tracker [1] allow linking the device to online accounts instead of a mobile device, which, however, bears security vulnerabilities as well; once the account is compromised, so is the authentication. Nonetheless, indirect authentication remains a dominant paradigm for wearables despite these fundamental shortcomings because such devices are extremely resource-constrained.

In this work, we develop a user authentication scheme, called *Beat-PIN*, for wearable devices that are equipped with touch sensors, e.g., a touch screen, a sensed surface, or a single button whose output signals can be stamped. It is a new passcode-style authentication. However, rather than numbers, letters, or characters, users choose different beats/rhythms when tapping on the touch sensor, e.g., screen for a smartwatch. Thus, the rhythm of tapping serves as the secret only known by the legitimate user. We call this rhythm-based password as the beat-PIN[1]. Basically, a beat-PIN can be easily created by the user, for example, extracting some beats from his/her favorite songs or jingles. A beat-PIN is characterized by the timing of its beats, which can be recorded by the device system clock.

Beat-PIN can serve as an ideal authentication method for wearable devices. First, unlike regular passwords or digit-PINs, which have to be entered either on a physical or virtual keyboard, or fingerprint and facial recognition based authentication methods, which require superior sensors, *Beat-PIN* can work on any wearable device with a simple touch sensor. Second, unlike the pattern-based passwords, which require a large-size screen to draw on, beat-PINs can be performed on a much smaller spot. Besides, it is resilient to infrared attacks and smudge attacks, as a user does not leave such kind of information on the screen when entering a beat-PIN. Note that traditional passwords/digit-PINs and pattern based authentications are reported vulnerable to these two types of attacks [11, 14], respectively. Moreover, it is also worth mentioning that our mechanism is friendly to sight impaired users.

[1]In this paper, we utilize the Italian font *Beat-PIN* to represent the authentication scheme, while the regular font beat-PIN as the password itself.

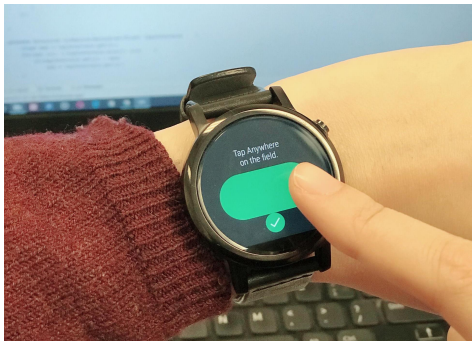

Figure 1: The prototype of *Beat-PIN* that we develop for the smartwatch. A beat-PIN is characterized by the timing of its beats.

Our mechanism is composed of two stages. In the enrollment stage, each user is asked to create his/her own beat-PIN and enter them multiple times for training purposes. During the login stage, the user simply enters the previously chosen beat-PIN to access the device. If it matches with the training samples, the user is authorized; otherwise, it is blocked. In order to achieve high authentication accuracy with low training overhead, we also propose a novel classification method, called vector comparison. To investigate the performance of *Beat-PIN*, we run two user studies. In phase-I, we recruit 124 volunteers and collect their beat-PINs via our data collection app on smartwatches. With the dataset, we derive their statistics, which are then used for the security analysis of our mechanism. Besides, its authentication accuracy is studied with respect to different parameter settings. In phase-II, we implement the prototype, setting the parameters as the ones that produce the best performance in the phase-I study. Another 49 volunteers are recruited. Multiple in-field experiments are conducted, evaluating performances in terms of time consumption, energy consumption, impact of user motions, and memorability. Notably, this work gives a formal security analysis over a proposed *Beat-PIN* authentication, which is missing from the existing works on wearable device authentication [16, 19, 26, 27, 34, 36, 38] and rhythm-based authentication [22, 28, 35].

2 RELATED WORK

2.1 User Authentication on Wearable Devices

There have been a few existing user authentication schemes for wearable devices. Recent development in glass-based devices has spurred work on iris recognition [26]. The basic idea is that users glance into a head mounted camera in order to authenticate. In order to address imitation attacks, Wang et al. [34] presented an algorithm that checks pupil size consistency in varying light conditions. Touch and movement based inputs have also been explored. Chauhan et al. [16], for instance, classified among a set of users based on gestures performed on the built-in touchpad on the side of Google Glass. Along this line, Li et al.'s Headbanger [27] authenticates users

by monitoring their head movement patterns in response to an external audio stimulus. Note that all the above schemes [16, 26, 27, 34] are designed specifically for glass-based devices. Besides, the authentication of [26, 34] can only be performed with the assistance of cameras. Thus, they cannot be easily adopted by other wearables. Yang et al.'s MotionAuth [36] collects movement data from a wrist-worn device during gesture performance and uses this to verify user identity. Their scheme, however, requires users to make awkward movements, such as drawing a circle in the air, which is impractical especially in public scenarios. Zeng et al. [38] proposed to use ambulatory activities (e.g. walking, running) as unique markers of the user to design an implicit authentication method. However, the accuracy performance under some activity modes is unsatisfactory. For instance, the accuracy rate is as low as 32% under user's jumping mode. Besides, as it is a continuous authentication scheme, it needs to run at the background, which can deplete a wearable device's battery quickly. Chun et al. [19] developed an electrocardiogram (ECG) biometric based user authentication. However, it is designed just for wearable ECG sensors that are capable of acquiring accurate ECG signals, and thus inapplicable to other general wearables.

2.2 Rhythm-Based Authentication

There are some existing rhythm-based authentication schemes. Wobbrock's TapSongs [35] enables user authentication on a single binary sensor by matching the rhythm of tap down/up events to a jingle timing model created by the user. However, its authentication accuracy is not perfect, with the false rejection rate as high as 16.8%. Lin et al. [28] developed a rhythm-based pairing scheme, called RhythmLink. It allows users to securely pair a peripheral with a host device via rhythmic taps. Notice that RhythmLink is not for user authentication. Marques et al. [13] transformed the timing information of taps into a sequence and leveraged Hamming distance to compare two tap patterns. However, performance accuracy is not evaluated. Chen et al. [12] developed a rhythm-based two-factor authentication, called RhyAuth, for multi-touch mobile devices. The two factors include a user-chosen rhythm and the behavioral metrics for inputting the rhythm. Recently, Das et al. [22] developed a group authentication scheme, called Thumprint, with a shared secret knock. All group members share one secret, but individual expressions of the secret are discernible. First of all, these schemes are not particularly designed for wearable devices. Thus, most of them are not readily applicable to our problem. For example, the input device for taps of TapSong [35] is the button on the earbuds' cord, which is unavailable in most wearables. Besides, a comprehensive study over the system utility to examine its practicality, in terms of, for example, enrollment/login time, energy consumption and impact of user motions, has been missing so far. More importantly, none of them provides a formal security analysis of rhythm-based authentication.

2.3 Keystroke Dynamics based Authentication

Keystroke authentication schemes leverage keystroke biometrics to characterize users. Extensive efforts have been devoted to this line of research. The preliminary work [20] conducts a feasible study of applying keystroke dynamics on mobile devices. It is extended in [21] where different neural networks, like the feed-forward multi-layered perceptron network, the radial basis function network and the generalized regression neural network, have been used. It achieves the EER at 13.3% and 12.8% for 4-digit PINs and 11-digit telephone numbers, respectively. New features are then explored to enhance the performance of keystroke authentication. For example, KenSens [23] passively authenticates users via examining the specific location touched on each key, the drift from finger down to finger up, the force of touch, the area of press. Zheng et al. [39] then proposed to rely on more sensors (e.g., accelerometers) other than purely touchscreen. Their approach reduces an average EER down to 3.65%. Frank et al. [24] later introduced the notion of continuous authentication by analyzing users' keystroke dynamics.

Note that keystroke authentication explores the biometric information inherent in people's typing behaviors, which relies on "something you are"; while in our case, a beat-PIN is a rhythm-based password, which can be viewed as "something you know". Besides, the adoption of keystroke authentication to wearable devices is largely hindered by their hardware constraints, as keystroke authentication must be performed on regular keyboards, either physical or soft ones, which are usually unavailable for wearables.

3 BEAT-PIN CHARACTERIZATION AND FEATURES

3.1 Definition of Beat-PINs

A beat-PIN is simply a set of time instances recored by the system clock when a user taps the device. Take a smartwatch as an example. A beat-PIN is generated when the user taps the screen. Figure 2 gives the example of two beat-PINs. Tapping onset (or action-down) and tapping offset (or action-up) mean that the screen is pressed and released, respectively. The time duration between two adjacent tapping onset and offset stands for one "beat". Similarly, the time duration between two adjacent tapping offset and onset stands for one "space". The length of a beat-PIN is considered as how many beats it contains. For example, the length of both beat-PINs shown in Figure 2 is 8. Essentially, beat-PINs could be interpreted as "rhythm passcode". A well-known existing rhythm passcode is the Morse code. It encodes letters and numbers as standardized sequences of short and long signals called "dots" and "dashes". The dash and dot are represented by some fixed time durations. However, Morse code falls short in user authentication due to its memorability issue, as there is a complicated mapping rule between a letter/number and its corresponding Morse code.

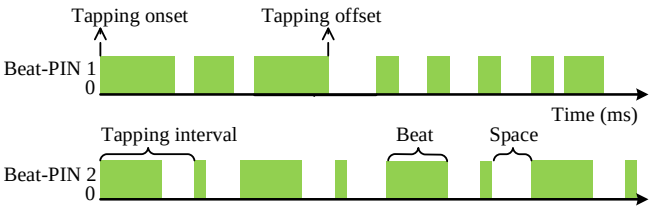

Figure 2: Two beat-PIN examples.

3.2 Features in Beat-PINs

Apparently, if two beat-PINs have different lengths, they are not the same. To distinguish between beat-PINs of the same length, we further explore their following features, *tapping time instance*, *tapping interval*, and *relative interval*.

Tapping time instance. Each beat-PIN can be uniquely identified by a set of tapping onsets and tapping offsets, indexed by their time instances, i.e., $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \cdots, \alpha_n\}$ and $\boldsymbol{\beta} = \{\beta_1, \beta_2, \cdots, \beta_n\}$, where n is the beta-PIN length. Besides, we index the time instance when the screen is first tapped as 0, indicating its start point. Thus, α_1 is always 0. Since a beat-PIN is uniquely identified by its set of tapping time instances, they naturally serve as one of the features.

Tapping interval (Euclidean interval). We further extract inter-onset intervals of a beat-PIN. It is defined as the time duration between two adjacent tapping onsets (shown in Figure 2), $\boldsymbol{\gamma} = \{\gamma_1, \gamma_2, \cdots, \gamma_{n-1}\}$, where $\gamma_i = \alpha_{i+1} - \alpha_i$. Tapping intervals can effectively capture a beat-PIN's rhythm or tempo.

Relative interval. It is possible that the taping speed may vary under different scenarios, even when a user enters the same beat-PIN. For example, when a user is running for class, he/she is more likely to enter the beat-PIN faster. In order to tolerate this variance, we introduce another feature called relative interval. It is calculated by $\boldsymbol{\eta} = \{\eta_1, \eta_2, \cdots, \eta_{n-2}\}$, where $\eta_i = \frac{\gamma_{i+1}}{\gamma_i}$ $(i \in [1, n-2])$. Specifically, η_i measures the relative difference between two adjacent tapping intervals. Therefore, even a beat-PIN is entered faster or slower, as long as its pattern is the same with the original one, its relative interval set does not change.

The entire feature set is then written as $\boldsymbol{f} = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\eta}\}$. In all, to character a beat-PIN, we jointly consider its length and feature set $\boldsymbol{f}$.

4 DATA COLLECTION AND STATISTICS

4.1 Data Collection

To investigate the performance of beat-PINs in wearable device authentication, we perform two user studies. For phase-I, the objective is to collect different beat-PINs so as to derive their statistics. Besides, we also aim to identify suitable parameters to construct the classifier through testing over the dataset. For phase-II, a prototype of *Beat-PIN* is built. We then conduct another round of data collection through a set of in-filed experiments, based on which the security and utility of our system is evaluated. In this section, we focus on phase-I user study.

To facilitate the data collection, a specialized app was developed and implemented on Motorola Moto 360 smartwatches. Each runs Android Wear OS, and is equipped with a Cortex A7 processor, 4GB storage, 512MB memory, 1.37-inch circular backlit IPS display, Wi-Fi, Bluetooth, etc. A total of 124 volunteers were recruited to participate the phase-I data collection. They are all bachelor students aged from 18 to 33 from two introductory classes offered by the department. Among them, there are 29 females and 95 males. Before the data collection, they were explained how *Beat-PIN* works. They were also informed that their grades and course credits have no relation with their data. Our phase-I data collection consists of two steps. In the first step, each user was asked to choose his/her own beat-PIN independently and perform it for at least 25 times, all of which were recorded by the data collection app. Besides, a supervisor accompanied the user during the experiment in case there are any questions. One week later, in the second step, each user was asked to recall his/her previously chosen beat-PIN and re-enter it into the app for 3 times. In total, we collected more than 2904 data samples from 119 subjects after deleting erroneous samples.

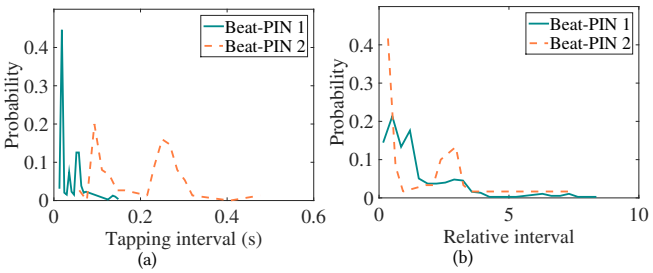

Figure 3: Feature distribution comparison between two beat-PINs. (a) Tapping interval. (b) Relative interval.

4.2 Statistics of Beat-PINs

To test if tapping interval and relative interval can serve as promising features to distinguish different beat-PINs, we first analyze their statistics. Without loss of generality, we randomly select two beat-PINs from the dataset. Figure 3(a) plots the tapping interval distributions of these two beat-PINs. The two distributions clearly distinguish from each other. Specifically, the distribution of beat-PIN 1 concentrates at the lower end, while that of the other one is more dispersed. It indicates that user 1 enters his/her beat-PIN faster than user 2. We have a similar observation over their relative interval distributions, as shown in Figure 3(b). Clearly, different beat-PINs demonstrate unique patterns in terms of tapping intervals and relative intervals. Thus, together with tapping time instances, we are able to explore them for user authentication.

5 SECURITY ANALYSIS OF *BEAT-PIN*

In this section, we first analyze the security of *Beat-PIN* in terms of its raw space size, which is then compared with that of the digit-PIN and traditional password. Besides, the comparison of their occurance frequencies in practice is provided as well.

5.1 Raw Size of Beat-PIN Space

The raw size of beat-PIN space is its information content assuming users equally pick different beat-PINs. It is an upper bound on the information content of the distribution that users choose in practice. Basically, the larger size the space has, the more robust *beat-PIN* is against the brute-force attack. Take the digit-PIN as an example. When it takes 6 digits as its size, as the case for iOS 10, its raw space size is calculated by 10^6, with the PIN chosen from "000000" to "999999". Similar to the digit-PIN, the size of beat-PIN pace is closely related to the number of beats in a beat-PIN. And we set this value as l. Besides, we assume that all beat-PINs of total length greater than some fixed value $L_{\max}$ have probability zero. That is, all users' chosen beat-PINs have a confined size no larger than $L_{\max}$. We further assume that the time duration of all beat-PINs is limited

by $T_{\max}$. It is recorded once the screen is pressed for the first beat and ends once the screen is released for the last beat. $L_{\max}$ and $T_{\max}$ are assumptions for practice, as it is challenging for users to remember the timing of a beat-PIN when its length, in terms of either beat numbers or time duration, becomes too large.

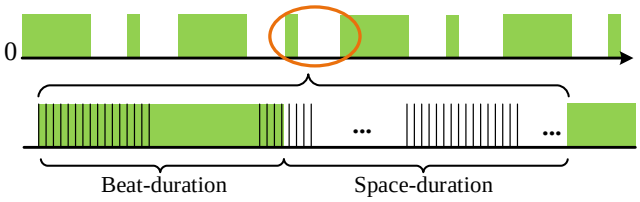

Figure 4: Example of a slotted beat-PIN.

Besides, due to the precision level of the system clock in a wearable device, the time domain can be evenly divided into a set of time slots, each with the size of the system clock unit. As a result, the beat-duration and space-duration of a beat-PIN can be represented by a set of time slots, as shown in Figure 4.

In addition, the finger tapping speed also influences the creation of beat-PINs. Specifically, let τ_b and τ_s be the minimum value of a beat-duration and that of a space-duration, respectively. The faster users can tap, the smaller values τ_b and τ_s are. We are now ready to analyze the raw size of beat-PIN space.

THEOREM 5.1. *The raw size of beat-PIN space is*

$$|\Pi| = \sum_{l=1}^{L_{\max}} \binom{\frac{T_{\max}}{\sigma} - (\frac{\tau_b}{\sigma} - 1) \times l - (\frac{\tau_s}{\sigma} - 1) \times (l-1)}{2l-1},$$

where $L_{\max}$, $T_{\max}$, σ, τ_b and τ_s stand for the maximum length, maximum time duration, the system clock unit, minimum value of a beat-duration duration and minimum value of a space-duration, respectively.

The formal proof is given in Appendix A.

To estimate the raw size of beat-PIN space following the result of Theorem 5.1, values of $T_{\max}$, σ, τ_b and τ_s should be provided. For an illustration purpose, we let $\sigma = 0.83\mu s$, which is the time unit for Moto 360's system clock. Appendix B provides detailed statistic analysis of the rest parameters. According to the discussion therein, we set $T_{\max}$ =5s, τ_s =0.12s and τ_b =0.08s. As shown in Table 1, when the beat-PIN length is upper bounded by 6, the corresponding space size is about 10^{42}. When the length is upper bounded by 10, the space size becomes 10^{70} approximately. Table 1 also shows the raw space size[2] of digit-PINs and traditional passwords. Apparently, beat-PIN's space size is significantly larger than the other two. In particular, when $L_{\max} = 6$, $|\Pi| = 10^{42}$, while that of digit-PINs and traditional passwords is 10^6 and 10^{13}, respectively. This is because, unlike digit-PINs (composed of pure numbers) or traditional passwords (composed of 128 ASCII characters), beat-PINs leverage more diverse "timing features", such as tapping-intervals, beat-durations, space-durations, etc. It indicates that users have a more ample choice over beat-PINs than the other two. Hence, our mechanism is more robust against brute-force attacks.

[2]This table shows the approximate raw space size. For example, when $L_{\max} = 4$, digit-PIN's raw space size is in fact calculated by $10^4 + 10^3 + 10^2 + 10^1 = 11,110$, which we use 10^4 to approximate. This is the same case for the password.

Table 1: Raw space size comparison.

$L_{\max}$	4	5	6	7	8	9	10
Beat-PIN	10^{28}	10^{35}	10^{42}	10^{49}	10^{56}	10^{63}	10^{70}
Digit-PIN	10^4	10^5	10^6	10^7	10^8	10^9	10^{10}
Password	128^4	128^5	128^6	128^7	128^8	128^9	128^{10}
	$\approx 10^8$	10^{11}	10^{13}	10^{15}	10^{17}	10^{19}	10^{21}

5.2 Beat-PIN Frequency in Practice

While the result shown in Table 1 is encouraging, in practice, not all beat-PINs are equally likely chosen by users, rendering a uniform distribution overly optimistic. Therefore, we further evaluate the distribution of beat-PINs based on our dataset. The comparison with digit-PINs and traditional passwords is conducted as well.

As shown in Table 2, we list the top-16 most frequently used beat-PINs by analyzing 119 valid beat-PINs in the dataset. Four of them are the same, indexed as #1 beat-PIN, with their frequency calculated as 3.3%. Besides, there are also duplicates for #2–#5 beat-PINs, with their occurrence frequencies as 2.5%, 2.5%, 1.7% and 1.7%, respectively. We further show the frequency of digit-PINs, which is directly cited from a statistic study over 3.4 million 4-digit PINs [6]. Specifically, the most popular 4-digit PIN in use has its frequency at 10.7%. It implies that when someone picks up a phone that is locked by 4-digit PIN, if trying this #1 digit-PIN, he/she has more than 10% of chance to unlock it. We also show the frequency of traditional passwords, which is obtained from [9]. The most popular password has its frequency at 1.6%. Meanwhile, we notice that the frequency ratio between #1 and #16 beat-PINs is 4.1, which is much smaller than that of digit-PINs (35.7) and passwords (16.0), respectively. It implies that the beat-PIN is the most evenly distributed among these three. In another word, users are less likely to choose the same beat-PIN than the other two. Hence, our mechanism will be more robust against dictionary attacks.

We acknowledge that our dataset is limited in its size; only 119 beat-PINs are analyzed. Still, it provides a rough estimation of the beat-PIN frequency in practice. Users are less likely to choose the same beat-PIN than digit-PIN and password. This phenomenon is not surprising: compared with digit-PIN (only consisting of numbers) and password (consisting of numbers and characters), a beat-PIN can be characterized by a more rich set of features, including tapping time instances, tapping intervals, and relative intervals. As a result, it makes the choice over beat-PINs more diverse.

6 CLASSIFICATION METHODS

Once features of a beat-PIN have been extracted following Section 3.2, the remaining task is to apply classification methods for user authentication, i.e., to discriminate the legitimate user and imposters. Ideally, the authentication should be performed in a time-efficient manner. For this purpose, in this work we propose a classification method, called vector comparison algorithm. To evaluate its performance, we also apply the supervised machine learning (SVM) as a benchmark. The performance comparison between these two will

Table 2: Frequency comparison among beat-PINs, digit-PINs and traditional passwords.

Index	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16
Beat-PIN	3.3%	2.5%	2.5%	1.7%	1.7%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%
Digit-PIN	10.7%	6.0%	1.9%	1.2%	0.8%	0.6%	0.6%	0.5%	0.5%	0.5%	0.5%	0.4%	0.4%	0.4%	0.4%	0.3%
Password	1.6%	1.0%	0.6%	0.4%	0.2%	0.2%	0.2%	0.2%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%

be fully discussed in the next section. It is worth noting that for two beat-PINs with different lengths, we simply treat them as different. For example, if the legitimate beat-PIN has the length as 8, then any testing input with a different length will be rejected immediately. Hence, in the following we only focus on the classification over beat-PINs of the same length.

6.1 Vector Comparison based Classification

Denote by $\boldsymbol{f} = [f_1, f_2, \cdots, f_N]$ the feature vector of a beat-PIN, where N stands for the feature size. Assume that users are asked to input M training samples during acquisition. Then we derive the average and standard deviation vectors over these training samples as $\overline{\boldsymbol{f}} = [\overline{f}_1, \overline{f}_2, \cdots, \overline{f}_N]$, where $\overline{f}_i = \frac{\sum_{i=1}^{M} f_i}{M}$ and $\boldsymbol{\sigma} = [\sigma_{f_1}, \sigma_{f_2}, \cdots, \sigma_{f_N}]$, where $\sigma_{f_i} = |f_i - \overline{f}_i|$, $i \in [1, N]$. For a user with an input test vector $\boldsymbol{f'}$, it is accepted as legitimate if $\left\|\boldsymbol{f'} - \overline{\boldsymbol{f}}\right\|_2 \leq \alpha \cdot \|\boldsymbol{\sigma}\|_2$; otherwise, it is classified as an impostor. Here α represents the tolerance parameter chosen by the system and is tunable. Apparently, if we choose a large α, the detection rule is loose, which may lead to a high false acceptance rate; otherwise, we have a strict detection rule, which can result in a high false rejection rate. Hence, α should be carefully chosen to strive a balance between these two.

The idea of the vector comparison algorithm is simple. We first derive the "center" and "radius" of training samples as $\overline{\boldsymbol{f}}$ and $\boldsymbol{\sigma}$, respectively. Then, a "circle" is drawn based on this center and a modified radius $\alpha \cdot \sqrt{\boldsymbol{\sigma}}$. If the testing data falls into this "circle", it is classified as legitimate; otherwise, it is classified as an imposter.

6.2 One-class SVM Classification

In order to evaluate the performance of our proposed vector comparison algorithm, we also consider the SVM classification. Its performance will serve as a benchmark of our algorithm.

SVM generalizes the ideas of finding an optimal hyper-plane in a high-dimensional space to perform a classification. In the training phase, SVM builds models based on the training samples of the legitimate user. In the testing phase, the testing samples are projected onto the same high-dimensional space, and the distances between the samples and the hyper-plane are computed as the classification scores. If the classification score is over the threshold, we regard the sample as a legitimate one. Since we only have training data from legitimate users, we build a model based only on the legitimate user's data samples, and use that model to detect impostors. This model is known as one-class classification or anomaly detection.

7 PERFORMANCE EVALUATION

The purpose for this section is twofolds, to further validate the effectiveness of utilizing beat-PINs for user authentication and to compare the authentication performances of our proposed vector comparison algorithm with the one-class SVM. We leave the implementation of *Beat-PIN* on a smartwatch and the analysis of its system performances to the next section.

7.1 Evaluation Methodology and Metrics

To evaluate the authentication accuracy, we conduct a series of experiments based on the dataset. For each experiment, we designate one subject from the dataset as the legitimate user, and the rest as impostors. For the samples from the same subject, we randomly select a portion as training samples, while the rest serve as testing samples. Since we use a random sampling method to divide the data into training and testing sets, in order to account for the effect of this randomness, we repeat the procedure 50 times for each experiment, each time with independently selected samples from the entire dataset. As we have 119 subjects in total, there are 119 independent experiments accordingly. The performance discussed in this section is the result of these 119 experiments. The authentication accuracy is measured via the following metrics:

- *False rejection rate (FRR).* The probability that a legitimate user is treated as an imposter. It is calculated as the ratio of the number of a legitimate user's incorrect authentications to the total number of attempts.
- *False acceptance rate (FAR).* The probability that an imposter is treated as a legitimate user.
- *Equal Error Rate (EER).* It is the point at which FRR and FAR are equal.

Note that FRR reflects the user convenience in our system; a lower FRR implies that a legitimate user can successfully unlock the wearable device at a higher probability. FAR reflects the security aspect; a lower FAR implies that the imposter will be denied at a higher probability.

7.2 Performances of One-class SVM

With the Python SciKitLearn library [8], we utilize the OneClassSVM package to test a variety of parameters to determine the best possible performance with SVM. 15 training samples are used to train the classifier. It is observed from Table 3 that the performance are unsatisfactory even with 15 training samples. For example, the achievable FRR and FAR is 14.8% and 22.3%, respectively, using the linear kernel when ν=0.01. The parameter ν is an upper bound on the fraction of margin errors and a lower bound of the fraction of support vectors relative to the total number of training samples. For example, ν=0.01 means that at most 1% of the training samples are misclassified (at the cost of a small margin, though) and at least 1% of the training samples are support vectors. Although FAR can be reduced by applying RBF as the kernel, it produces a very poor FRR. When $\nu = 0.1$, its FRR is as high as 46.3%.

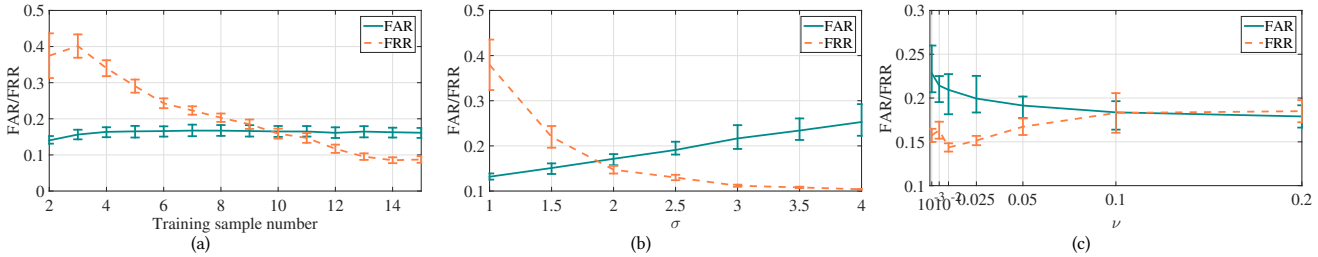

Figure 5: FAR and FRR achieved by one-class SVM under different parameter settings. (a) Training sample number. (b) σ. (c) ν.

Table 3: FRR and FAR with classic one-class SVM.

Kernel	ν	FRR	FAR
Linear	0.01	14.8%	22.3%
	0.025	15.2%	20.0%
	0.05	17.1%	18.8%
	0.1	18.1%	18.1%
RBF	0.01	14.2%	3.2%
	0.025	15.2%	3.1%
	0.05	29.1%	2.2%
	0.1	46.3%	1.7%

A batch run is then performed, analyzing the authentication performance of one-class SVM under more complex parameter settings.

Figure 5(a) shows the impact of training sample size to the authentication accuracy. When there are only 2 training samples, FRR is as high as 38.4%; when there are 15 training samples, it drops to 9.6%. FAR keeps relatively stable when the training sample number is increased from 4 to 15. Thus, in order to acquire an acceptable FRR, say 9.6%, 15 training samples are needed for one-class SVM. Still, its FAR performance is poor, with the value at 17.8%. Figure 5(b) shows the impact of σ to the authentication accuracy. Here, σ is the standard deviation of the kernel function. It influences the decision boundary qualitatively. As σ grows, FAR increases while FRR decreases, which means both legitimate users and impostors are more likely to get authenticated. In fact, for a larger σ, the decision criteria tends to be relaxed and avoids the hazard of overfitting. For a smaller σ, the decision boundary tends to be strict and sharp. In contrast to the former situation, it tends to overfit. Figure 5(c) shows the impact of ν to the authentication accuracy. Opposite to σ, a larger ν brings about a smaller FAR but a larger FRR.

To sum up, the one-class SVM produces unsatisfactory accuracy performances when serving as the classifier for *Beat-PIN*. Therefore, a different classification method should be used.

7.3 Performances of Vector Comparison based Classification

We now evaluate the authentication accuracy with our proposed vector comparison algorithm. We investigate the impact of training sample size and the tolerance parameter α to FAR and FRR in Figure 6(a) and Figure 6(b), respectively. It shows that a larger training sample size produces a smaller FRR but a slightly increased FAR. A similar observation is obtained with a larger α. This is because a larger α leads to a more loose detection rule, as discussed in Section

FAR	Training sample number														
σ	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.5	0	0	0	0.01	0.01	0.01	0.01	0.01	0.01	0	0	0	0	0	0
1	0	0.01	0.01	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.01	0.02	0.02	0.02
1.5	0	0.02	0.02	0.04	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
2	0	0.02	0.04	0.05	0.05	0.05	0.05	0.05	0.05	0.04	0.04	0.04	0.04	0.04	0.04
2.5	0	0.03	0.05	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.05	0.05
3	0	0.04	0.06	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
3.5	0.01	0.05	0.07	0.08	0.08	0.08	0.09	0.09	0.08	0.08	0.08	0.08	0.08	0.08	0.08
4	0.01	0.06	0.08	0.09	0.09	0.1	0.1	0.1	0.1	0.1	0.1	0.09	0.1	0.1	0.09
4.5	0.02	0.06	0.09	0.1	0.1	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
5	0.03	0.07	0.09	0.1	0.11	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.13	0.13	0.12
5.5	0.04	0.08	0.1	0.11	0.11	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.14	0.14	0.13
6	0.04	0.08	0.11	0.12	0.12	0.14	0.14	0.14	0.15	0.15	0.15	0.14	0.15	0.15	0.15
6.5	0.06	0.09	0.12	0.13	0.13	0.15	0.15	0.15	0.15	0.16	0.16	0.15	0.16	0.16	0.16
7	0.07	0.1	0.12	0.14	0.14	0.15	0.16	0.16	0.16	0.16	0.16	0.16	0.17	0.17	0.17
7.5	0.08	0.11	0.13	0.15	0.15	0.16	0.16	0.16	0.17	0.17	0.17	0.17	0.18	0.18	0.18
8	0.09	0.12	0.14	0.15	0.16	0.17	0.17	0.17	0.18	0.18	0.18	0.18	0.19	0.19	0.19
8.5	0.1	0.12	0.15	0.16	0.17	0.18	0.18	0.18	0.19	0.19	0.19	0.19	0.2	0.2	0.2
9	0.11	0.13	0.15	0.17	0.18	0.19	0.19	0.19	0.2	0.2	0.2	0.2	0.21	0.21	0.21
9.5	0.12	0.14	0.16	0.18	0.19	0.2	0.2	0.2	0.21	0.21	0.21	0.21	0.22	0.21	0.22
10	0.13	0.15	0.17	0.19	0.19	0.21	0.21	0.21	0.22	0.22	0.22	0.22	0.22	0.22	0.22

(a)

FRR	Training sample number														
σ	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.5	1	0.98	0.97	0.94	0.9	0.87	0.88	0.86	0.85	0.85	0.85	0.86	0.85	0.85	0.85
1	0.95	0.88	0.77	0.7	0.62	0.59	0.58	0.58	0.56	0.55	0.56	0.55	0.52	0.53	0.53
1.5	0.82	0.74	0.54	0.48	0.42	0.38	0.34	0.33	0.32	0.32	0.32	0.31	0.29	0.29	0.29
2	0.63	0.61	0.39	0.3	0.26	0.21	0.19	0.17	0.17	0.17	0.17	0.16	0.15	0.14	0.14
2.5	0.53	0.46	0.26	0.19	0.16	0.12	0.1	0.09	0.09	0.09	0.09	0.09	0.08	0.08	0.08
3	0.42	0.37	0.19	0.14	0.1	0.07	0.07	0.07	0.06	0.06	0.06	0.06	0.05	0.05	0.05
3.5	0.32	0.3	0.14	0.09	0.07	0.04	0.05	0.04	0.04	0.03	0.04	0.03	0.03	0.03	0.03
4	0.26	0.23	0.11	0.06	0.05	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.01	0.01
4.5	0.21	0.18	0.09	0.05	0.04	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
5	0.18	0.14	0.07	0.04	0.03	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0	0.01	0.01
5.5	0.16	0.11	0.05	0.03	0.03	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0	0	0
6	0.15	0.09	0.04	0.03	0.02	0	0	0.01	0.01	0.01	0.01	0.01	0	0	0
6.5	0.13	0.07	0.04	0.03	0.02	0	0	0	0	0.01	0.01	0.01	0	0	0
7	0.12	0.06	0.03	0.02	0.01	0	0	0	0	0	0	0	0	0	0
7.5	0.1	0.05	0.03	0.01	0.01	0	0	0	0	0	0	0	0	0	0
8	0.1	0.05	0.02	0.01	0.01	0	0	0	0	0	0	0	0	0	0
8.5	0.09	0.05	0.02	0.01	0.01	0	0	0	0	0	0	0	0	0	0
9	0.07	0.04	0.02	0.01	0.01	0	0	0	0	0	0	0	0	0	0
9.5	0.06	0.03	0.02	0.01	0.01	0	0	0	0	0	0	0	0	0	0
10	0.05	0.02	0.02	0.01	0.01	0	0	0	0	0	0	0	0	0	0

(b)

Figure 6: FAR and FRR with respect to training sample number and α. (a) FAR. (b) FRR.

Table 4: FRR and FAR of vector comparison based classification

Training sample size	1	2	3	4	5	6	7	8	9	10	11	12	13	14
α	8	6	4.5	3.5	3.5	3	3	3	3	3	3	3	3	3
FRR	10.2%	9.3%	9.2%	9.1%	7.4%	7.4%	**7.2%**	7.0%	6.4%	6.3%	6.2%	6.1%	5.3%	5.1%
FAR	9.4%	8.4%	8.9%	8.1%	7.8%	7.4%	**7.2%**	7.2%	7.4%	7.4%	7.3%	7.3%	7.2%	7.1%

6.1. Thus, suitable values of training sample size and α are needed to strive a balance between these two. For this purpose, based on Figure 6(a) and Figure 6(b), we list FAR and FRR under different combinations of training sample size and α in Table 4. We find that FRR=FAR, when α =3 and training sample size is either 6 or 7. More importantly, the lowest EER, i.e., 7.2%, is achieved when α =3 and the training sample size is 7.

To sum up, our proposed vector comparison classification achieves the lowest EER at 7.2% when α =3 and training sample size is 7. Compared with the one-class SVM, whose lowest EER is about 16.7% according to Figure 5, ours pertains a significantly higher authentication accuracy. More importantly, vector comparison classification acquires much less training samples than the one-class SVM. Hence, it can be expected that the former is more time-efficient than the latter, especially during the enrollment stage. Besides, our mechanism also outperforms TapSongs [35] and Thumprint [22], two existing rhythm-based authentication schemes, in terms of authentication accuracy. For TapSongs, its FRR and FAR is about 16.8% and 19.4%, respectively; for Thumprint, its FRR and FAR is around 9-15% and 13-19%, respectively. Both are significantly larger than ours.

8 *BEAT-PIN* IMPLEMENTATION AND EXPERIMENT EVALUATION

As a proof-of-concept implementation, we develop the prototype of *Beat-PIN* on the same Moto 360 smartwatch (as shown in Figure 1) that was used for phase-I user study. As observed in Section 7, our proposed vector comparison classification outperforms the classic one-class SVM. Hence, we implement the former as the classifier in the system. Since vector comparison classification demonstrates the best accuracy performance when taking training sample size as 7 and α as 3, we adopt these values as our setting in the implementation.

To evaluate performances of *Beat-PIN* in real scenarios, we design a set of in-field experiments and conduct a phase-II user study. Another 49 volunteers[3] are recruited. Screenshots for enrollment and login are shown in Figure 7. During the enrollment stage, users are asked to tap on the screen to enter their beat-PINs repeatedly until 7 valid samples have been collected. Users have the choice to either accept or drop any trial during this stage as shown in Figure 7(a). All collected samples are then used to train/generate the classifier, i.e., calculate $\overline{f}$ and σ. During the login phase, users are prompted to enter their pre-defined beat-PINs (see Figure 7(b)). If it is the correct one, the access will be granted (see Figure 7(c)); otherwise, the access will be denied with error message shown on the display (see Figure 7(d)).

[3]They are part of the 124 volunteers from the phase-I user study.

Figure 7: Screenshots of *Beat-PIN* prototype in enrollment (a) and login stage (b), (c) and (d).

8.1 Robustness Against Attacks

The adversary's goal is to impersonate a legitimate user and successfully authenticate to the device. In our case, it means that the adversary has to correctly enter the exact legitimate beat-PIN. Hence, we assume that the adversary has physical access to the device. In practice, such physical access can be gained in ways such as a thief stealing a device, finders finding a lost device, and a roommate temporarily holding a device when the owner is taking a shower. In experiments, we consider the following three common types of attacks, *zero-effort attacks*, *shoulder surfing attacks* and *statistical attacks*.

8.1.1 Zero-effort Attacks. Zero-effort attacks may be the most common type of attacks against an authentication system where the attacker guesses the secret or tries the authentication procedure without much knowledge of the legitimate password. In our case, each volunteer (attacker) is asked to randomly pick beat-PINs without any prior knowledge of the legitimate one and tries to pass the authentication by chance. Up to three authentication attempts can be made. An attack is considered to succeed if any one of them passes the authentication.

Table 5 shows the success rate of zero-effort attacks, which is directly the FAR of our mechanism. According to the statistic results shown in Figure 15, most of the beat-PINs, more than 95%, take their lengths from 6 to 10. Hence, we conduct tests over beat-PINs

Table 5: FAR (Success rate) of zero-effort attacks under different beat-PIN lengths.

Beat-PIN length	6	7	8	9	10
FAR (Success rate)	8.2%	6.1%	2.0%	0.0%	0.0%

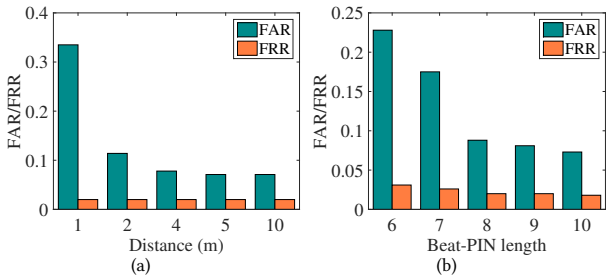

Figure 8: FAR/FRR under type-I shoulder surfing attacks. (a) Different distances between the legitimate user and the attacker. (b) Beat-PIN length.

with their lengths falling within this range. Apparently, the beat-PIN length plays a critical role in the attacker's success rate. The longer a beat-PIN, the stronger it is against zero-effort attacks. Specifically, when the length is 6, the success rate is 8.2%. When the length becomes 8, only 1 attacker succeeds. Moreover, with the length even longer, the success rate becomes 0. Therefore, in the practical implementation of *Beat-PIN*, the system can impose a hard constraint over a valid beat-PIN's minimum length, say 8, to defeat zero-effort attacks.

8.1.2 Shoulder Surfing Attacks. Shoulder surfing [15, 25, 30, 33] is a targeted attack leveraging the visual aspect of a certain specific user authentication method. There are two types of shoulder surfing attack: direct observation attack, in which authentication information is obtained by an attacker who is directly monitoring the authentication sequence, and recording attack, in which the authentication information is obtained by recording the authentication process for later analysis. In the experiment, we evaluate the robustness of *Beat-PIN* against the above two types of shoulder surfing attacks. We first consider the mild one, in which attackers learn a beat-PIN via direct visual observation. We grouped 20 volunteers into 10 pairs. Each of them was told to replay his/her partner's beat-PIN. Firstly, one user of the pair acts as an attacker, the other one as a legitimate user, and then the roles are exchanged. During the experiment, the legitimate user repeats the same beat-PIN for three times and there is a pause in between. Then, the attacker watches the entire process, acquires its rhythm information, and tries to reproduce it. Every attacker makes three access attempts. The attacker is considered to success in a shoulder surfing if any one of the three trials passes the authentication. In addition, we also conduct a controlled experiment, to illustrate the errors and bias in the experiment. Specifically, this is captured by the FRR, i.e., the percentage that legitimate users' beat-PINs get denied.

Figure 8(a) plots the attacker's performance with respect to its distance to the legitimate user. Specifically, FAR represents the success rate of shoulder surfing. As shown in the figure, the success rate decreases as the distance gets longer. This is intuitive, as a shorter distance enables the attacker to have a closer observation over the legitimate user's login. Thus, it has a better chance to correctly replay the genuine beat-PIN. Luckily, when the distance is longer than 2 meters, this advantage diminishes. The result implies that the rhythm is difficult to mimic through vague visual observation. As a control measurement, we further show the FRR of legitimate users in shoulder surfing attack experiments. We observe that FRR keeps as low as 2.1% under all distance values, as the distance does not effect the classification of legitimate beat-PINs. More importantly, the result indicates that the experiment imposes rather limited errors and bias toward the measurement over the attacker's performance. In particular, these errors and bias mainly come from two factors, i.e., the error caused by our proposed vector comparison based classification and the confusion caused by the inconsistency when the legitimate user enters his/her beat-PIN in different trials.

Figure 8(b) illustrates the impact of beat-PIN length to shoulder surfing attack. We have a similar observation as in zero-effort attacks; it is more difficult to compromise a beat-PIN with a larger length. Besides, together with Table 5, it tells that the adversary can indeed leverage visual observation to assist the attack. However, the advantage is limited. The FRR is also given for the purpose of control measurement. The value is low, ranging from 3.2% with the beat-PIN length as 6 to 1.8% with the length as 10. First of all, it complies with the observation from Figure 8(a) that errors and bias in our experiments are well confined. Besides, FRR experiences a slight decrease as the beat-PIN length grows. This is because a longer beat-PIN bears richer features to distinguish from others. As a result, the chance that it is wrongly classified decreases.

The resistance performance of *Beat-PIN* to type-II shoulder surfing attack is discussed in Appendix C.

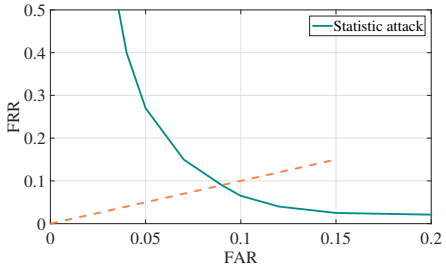

Figure 9: FAR-FRR under statistic attacks.

8.1.3 Statistic Attacks. This type of attackers employ knowledge obtained from the statistics of a group of users as hints to generate authentication attempts. The basic approach is to estimate the distribution of features from a group of users and then use the most probable feature values to generate the forgery. In our case, we use all samples from the 119 subjects collected from our phase-I user study as the input. Hence, we estimate the worst situation where the attacker has full knowledge about statistics of the whole

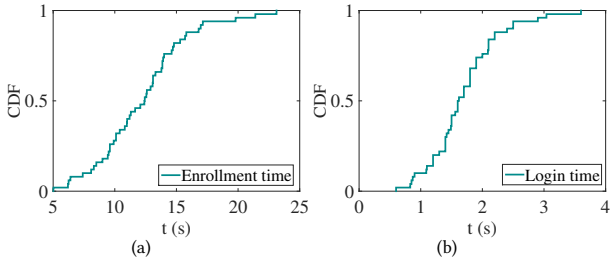

Figure 10: Distribution of time involved in *Beat-PIN*. (a) Enrollment time. (b) Login time.

population. A forgery beat-PIN under statistic attacks is produced in the following way. First, the beat-PIN length is randomly chosen following the distribution obtained in Figure 15. Second, for each beat-duration and space-duration, the value is randomly chosen following the distribution given by Figure 16(c) and Figure 16(b), respectively. We generate 10000 forgery beat-PINs to attack all the 119 legitimate ones. Performances on against statistic attacks are given in Figure 9 by tuning parameters of the vector comparison classifier. The red dash line stands for the possible points where FAR=FRR. The crossover of the red dash line and FAR-FRR curve is exactly the location of the EER, which is 9.6%; when the attacker's success rate is 9.6%, the chance that a legitimate user is blocked is 9.6% as well. Although our system is slightly more vulnerable to statistic attacks than zero-effort attacks, the attacker's success rate is still within an acceptable range. We'd like to point out that even the recently proposed touch dynamics plus hand geometry based authentication scheme [31] has its EER about 13%, which is higher than ours.

From the above experiment results, we observe that *Beat-PIN* is the most robust to zero-effort attacks. Besides, an adversary does benefit from mimicking the legitimate user's login patterns via visual observation or fabricating a synthetic one via statistic analysis. Nonetheless, the benefit is marginal.

8.2 Usability

Besides security, usability is another critical criteria in evaluating the performance of a user authentication mechanism. We measure the usability of *Beat-PIN* from aspects of time consumption, energy consumption, impact of user motions, and memorability (Appendix D). Moreover, we present a comprehensive survey result in Appendix E based on volunteers' feedbacks.

8.2.1 Enrollment Time and Login Time. We examine the enrollment time and login time needed for *Beat-PIN*. Specifically, the former is the total duration for the user to provide training samples and for the system to derive the classifier, while the latter is the total duration for a user to enter a test beat-PIN and for the system to make an authentication decision. These two parameters directly determine *Beat-PIN*'s usage convenience. We depict in Figure 10(a) and Figure 10(b) the distribution of enrollment time and login time, separately, according to our dataset. We observe that the enrollment time spans from 5.0s to 23.1s, with its average value as 12.3s. Besides, 90% of collected beat-PINs have their enrollment time shorter than 17.4s. The login time spans from 0.6s to 3.6s, with its average value as 1.7s. 90% of collected beat-PINs have their login time shorter than 2.6s. Hence, the most time-consuming part is the enrollment stage. Still, it can be performed within a relatively short time.

Table 6: Enrollment time and login time comparison among different authentication schemes.

Method	Enroll. time (s)	Login time (s)
Beat-PIN	**12.3**	**1.7**
Pattern [18]	22.5	4.5
Gesture [37]	69.4	16.5
Graphical password [17]	42.3	15.1
Touch dynamics[32]	120.0	0.3
Touch dynamics +hand geometry [31]	63.0	1.0

We further compare the time consumption of *Beat-PIN* with some recently proposed authentication schemes, which leverage "pattern" [18], "gesture" [37], "graphical password" [17], "touch dynamics" [32], and "touch dynamics+hand geometry" [31], in Table 6. Among them, *Beat-PIN* has the lowest average enrollment time, because it only takes 7 training samples. Besides, it ranks as the third among all these schemes in terms of login time and is much faster than "pattern", "gesture", and "graphical password" based authentications. This is because these schemes apply some computationally complex algorithms as their classifiers, such as neural networks and random forest, which are time-consuming in both training and testing.

In conclusion, *Beat-PIN* is time-efficient during both enrollment and login stage.

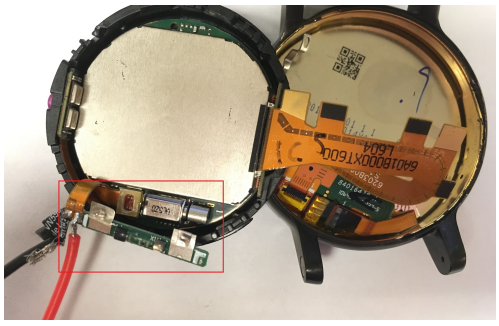

Figure 11: Power measurement of an Moto 360 smartwatch. A compatible battery interface circuit (as shown in the red box) was carved out from the same smartwatch and used as an adapter between the watch and the power monitor.

8.2.2 Energy Consumption. Typically, wearable devices have a much shorter battery life compared with regular mobile devices. As pointed out by [3], the battery life for Apple Watch Series 3 is about 18 hours after an overnight charge under normal use, including 90 time checks, 90 notifications, 45 minutes of app use, and a 30-minute workout with music playback from Apple Watch via Bluetooth. Therefore, it is desirable to design energy-efficient

authentication for wearable devices, especially with the fact that users may access their wearables dozens of times every day. In the in-field experiments, we extensively test energy consumption performances of *Beat-PIN* to validate its usability. Note that such evaluation has been merely discussed in existing works on authentication design for regular mobile devices, since their batteries can easily last for a couple of days, rendering energy consumption a less critical issue.

Due to the lack of software-based approaches for wearables (although those for smartphones are available), such kind of study has rarely been conducted so far [16, 19, 26, 27, 34, 36, 38]. To bridge this gap, in this work we measure the precise power consumption of our system using the dedicated hardware, the Monsoon power monitor [7]. To facilitate the measurement, as shown in the red box of Figure 11, we carved out a compatible battery interface circuit from the same smartwatch, and then used the interface circuit as an adapter between the watch and the power monitor. During the measurement, we kept other components offline (e.g., Wi-Fi and Bluetooth).

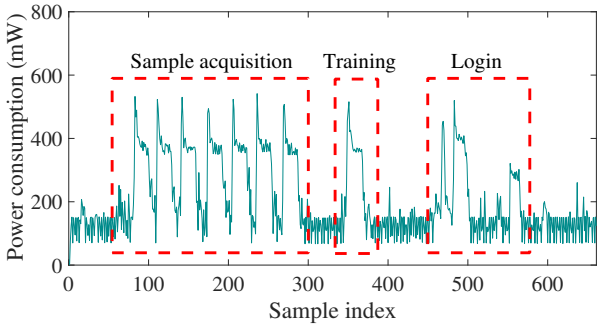

Figure 12: Instant power reading via the power monitor.

Figure 12 shows the instant power reading via the power monitor when executing one beat-PIN. We clearly specify the part dedicated to sample acquisition, training and login in the figure. We can tell that sample acquisition consumes the largest amount of power among these three, which is about 341.5mW on average, while the average power for training and login take 337.2mW and 181.4mW, separately. Figure 13 further depicts the statistic result of power consumption according to our dataset. We observe that the power consumption during enrollment spans from 272.4mW to 374.1mW, with 90% of samples lower than 367.0mW. The power consumption during login spans from 165.8mW to 197.6mW, with 90% of the samples lower than 191.4mW. Hence, in general, the enrollment stage costs about twice the power than the login stage.

We also compare the average power consumption of *Beat-PIN*'s login process with some common smartwatch tasks in Table 7. We notice that the power consumption of *Beat-PIN* login, 181.4mW, is only slightly higher than that for screen on operation, 161.5mW. It is much more energy-efficient than most of the operations, such as voice assistant, map service, sending a text message, etc. For instance, the average power for sending a voice message is 524.7mW, which is about three times of our power consumption. Together with the fact that *Beat-PIN* can be performed within a short period,

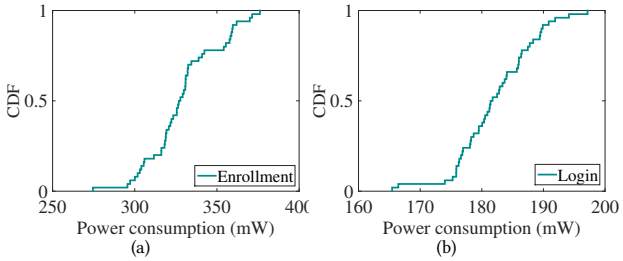

Figure 13: Distribution of power consumption in executing beat-PINs. (a) Enrollment. (b) Login.

Table 7: Power consumption by different operations at the smartwatch.

Operation	Power consumption (mW)
***Beat-PIN* login**	**181.4**
Screen on	161.5
Voice assistant	585.0
Map service	508.7
Sending a text message	410.5
Sending a voice message	524.7
Web browsing	574.8
Measuring heartbeat	428.0

we conclude that our authentication mechanism only imposes mild energy consumption overhead to the smartwatch. This is a desirable property for the battery-constrained wearable devices.

8.2.3 Impact of User Motions. Ideally, *Beat-PIN* should be insensitive to user motions because wearable devices are mainly used in mobile environments and the user motion always introduces noises. In this experiment, we test the impact of user motions to the authentication accuracy. Three types of user motions are considered: sitting, slow walk and fast walk. Results are shown in Figure 14. We observe that the best authentication accuracy is realized when a user is in the sitting status, with the corresponding FRR as 4.5% and FAR as 8.0%. The lowest accuracy takes place when a user is in the fast walk status, with the corresponding FRR as 9.0% and FAR as 8.3%. It means that user motion does impact the authentication performance. This meets our expectation because the faster motion will increase vibration in one's smartwatch and thus causing more noise. On the other hand, we notice that FAR keeps relatively stable under all three motion patterns, for example, FAR=8.2% under slow walk while FAR=8.1% under fast walk. It implies that user motion has more impact on usability, i.e., a legitimate user has a higher chance to get denied when walking; while the impact on security is limited, an imposter gets rejected at a stable rate regardless his/her motion status.

9 CONCLUSIONS

As wearable devices are increasingly weaved into our everyday life, providing security to the data acquired by or accessed through these devices becomes critically important. In this study, we have developed a user authentication mechanism, *Beat-PIN*, which relies

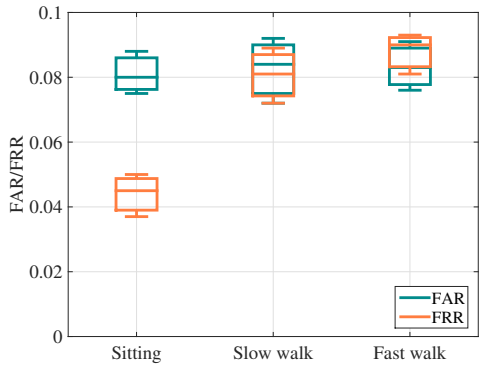

Figure 14: Impact of user motions on authentication accuracy.

on timing of beat sequences for direct authentication to a wearable device. Compared to existing authentication solutions, our solution delivers accurate authentication, incurs low processing overhead, and offers great convenience to users.

Through an extensive evaluation that involves 124 participants, we observe that the average EER of our approach is 7.2% with only 7 training samples. We have also implemented a prototype on Moto 360 smartwatches, and measured its security, in terms of robustness against various types of attackers, and its utility, from aspects of time consumption, power consumption, impact of user motions and memorability. As a conclusion, we believe *Beat-PIN* is a realistic authentication mechanism applicable on resource-constrained wearables.

ACKNOWLEDGMENTS

The work of Ming Li was supported by the U.S. National Science Foundation under grants CNS-1566634 and ECCS-1711991. The work of Lei Yang was supported in part by the U.S. National Science Foundation under Grants CNS-1559696 and IIA-1301726.

REFERENCES

[1] Fitbit. http://en.wikipedia.org/wiki/Fitbit.
[2] Gartner Says Worldwide Wearable Devices Sales to Grow 18.4 Percent in 2016. https://www.gartner.com/newsroom/id/3198018.
[3] Apple Watch Series 3 Battery Information. https://www.apple.com/watch/battery.html.
[4] Google glass. http://en.wikipedia.org/wiki/Google Glass.
[5] IDC Forecasts Worldwide Shipments of Wearables to Surpass 200 Million in 2019, Driven by Strong Smartwatch Growth and the Emergence of Smarter Watches. https://www.businesswire.com/news/home/20160317005136/en/IDC-Forecasts-Worldwide-Shipments-Wearables-Surpass-200.
[6] PIN Analysis. http://www.datagenetics.com/blog/september32012/.
[7] Power Monitor Software. http://msoon.github.io/powermonitor/.
[8] scikit-learn-Machine Learning in Python. http://scikit-learn.org/stable/.
[9] Unmasked: What 10 million passwords reveal about the people who choose them. https://wpengine.com/unmasked/.
[10] Wearable Technology Market - Global Opportunity Analysis and Industry Forecast, 2014-2022. https://www.prnewswire.com/news-releases/wearable-technology-market—global-opportunity-analysis-and-industry-forecast-2014-2022-300460342.html.
[11] Here's How iPhone Thermal Cameras Can Be Used to Steal Your Pin Codes. https://petapixel.com/2014/08/29/heres-iphone-thermal-cameras-can-used-steal-pin-codes/. (2014).
[12] Yimin Chen, Jingchao Sun, Rui Zhang, and Yanchao Zhang. 2015. Your song your way: Rhythm-based two-factor authentication for multi-touch mobile devices. In *Proceedings on IEEE Conference on Computer Communications (INFOCOM).*
[13] Diogo Marques, Tiago Guerreiro, Luís Duarte, and Luís Carriço. 2013. Under the table: tap authentication for smartphones. In *Proceedings on International BCS Human Computer Interaction Conference (BCS-HCI).*
[14] Adam J Aviv, Katherine L Gibson, Evan Mossop, Matt Blaze, and Jonathan M Smith. 2010. Smudge attacks on smartphone touch screens. In *Proceedings on USENIX conference on Offensive technologies (WOOT).*
[15] Michael Backes, Markus Dürmuth, and Dominique Unruh. 2008. Compromising reflections-or-how to read LCD monitors around the corner. In *IEEE Symposium on Security and Privacy.*
[16] Jagmohan Chauhan, Hassan Jameel Asghar, Anirban Mahanti, and Mohamed Ali Kaafar. 2016. Gesture-Based Continuous Authentication for Wearable Devices: The Smart Glasses Use Case. In *International Conference on Applied Cryptography and Network Security.* Springer, 648–665.
[17] Hsin-Yi Chiang and Sonia Chiasson. 2013. Improving user authentication on mobile devices: A touchscreen graphical password. In *Proceedings of ACM conference on Human-computer interaction with mobile devices and services (MobileHCI).*
[18] Geumhwan Cho, Jun Ho Huh, Junsung Cho, Seongyeol Oh, Youngbae Song, and Hyoungshick Kim. 2017. SysPal: System-guided Pattern Locks for Android. In *Proceedings of IEEE Symposium on Security and Privacy.*
[19] Se Young Chun, Jae-Hwan Kang, Hanvit Kim, Chungho Lee, Ian Oakley, and Sung-Phil Kim. 2016. ECG based user authentication for wearable devices using short time Fourier transform. In *Proceedings of IEEE International Conference on Telecommunications and Signal Processing (TSP).*
[20] Nathan L Clarke, SM Furnell, BM Lines, and Paul L Reynolds. 2003. Keystroke dynamics on a mobile handset: a feasibility study. *Information Management & Computer Security* 11, 4 (2003), 161–166.
[21] Nathan L Clarke and Steven M Furnell. 2007. Authenticating mobile phone users using keystroke analysis. *International Journal of Information Security* 6, 1 (2007), 1–14.
[22] Sauvik Das, Gierad Laput, Chris Harrison, and Jason I Hong. 2017. Thumprint: Socially-Inclusive Local Group Authentication Through Shared Secret Knocks. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI).*
[23] Benjamin Draffin, Jiang Zhu, and Joy Ying Zhang. 2013. KeySens: Passive User Authentication through Micro-behavior Modeling of Soft Keyboard Interaction. In *Proceedings of International Conference on Mobile Computing, Applications, and Services.*
[24] Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. 2013. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Transactions on Information Forensics and Security* 8, 1 (2013), 136–148.
[25] Benjamin Laxton, Kai Wang, and Stefan Savage. 2008. Reconsidering physical key secrecy: Teleduplication via optical decoding. In *Proceedings of the ACM conference on Computer and Communications Security.*
[26] Jeong Jun Lee, Seungin Noh, Kang Ryoung Park, and Jaihie Kim. 2004. Iris recognition in wearable computer. In *Biometric Authentication.* Springer, 475–483.
[27] Sugang Li, Ashwin Ashok, Yanyong Zhang, Chenren Xu, Janne Lindqvist, and Macro Gruteser. 2016. Whose move is it anyway? Authenticating smart wearable devices using unique head movement patterns. In *Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom).*
[28] Felix Xiaozhu Lin, Daniel Ashbrook, and Sean White. 2011. RhythmLink: securely pairing I/O-constrained devices by tapping. In *Proceedings of ACM symposium on User interface software and technology.*
[29] Jim Pitman. 1999. Probability. *Springer Science & Business Media* (1999).
[30] Volker Roth, Kai Richter, and Rene Freidinger. 2004. A PIN-entry method resilient against shoulder surfing. In *Proceedings of the ACM conference on Computer and Communications Security.*
[31] Yunpeng Song, Zhongmin Cai, and Zhi-Li Zhang. 2017. Multi-touch Authentication Using Hand Geometry and Behavioral Information. In *Proceedings of IEEE Symposium on Security and Privacy.*
[32] Jingchao Sun, Rui Zhang, Jinxue Zhang, and Yanchao Zhang. 2014. Touchin: Sightless two-factor authentication on multi-touch mobile devices. In *Proceedings on IEEE Conference on Communications and Network Security (CNS).*
[33] Furkan Tari, Ant Ozok, and Stephen H Holden. 2006. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In *Proceedings of The Symposium on Usable privacy and security.*
[34] Tianzi Wang, Zheng Song, Jian Ma, Yongping Xiong, and Yun Jie. 2013. An anti-fake iris authentication mechanism for smart glasses. In *Proceedings of IEEE International Conference on Consumer Electronics, Communications and Networks (CECNet).*
[35] Jacob Otto Wobbrock. 2009. Tapsongs: tapping rhythm-based passwords on a single binary sensor. In *Proceedings of ACM symposium on User interface software and technology.*
[36] Junshuang Yang, Yanyan Li, and Mengjun Xie. 2015. MotionAuth: Motion-based authentication for wrist worn smart devices. In *Proceedings of IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops).*

[37] Yulong Yang, Gradeigh D Clark, Janne Lindqvist, and Antti Oulasvirta. 2016. Free-form gesture authentication in the wild. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI)*.
[38] Yunze Zeng, Amit Pande, Jindan Zhu, and Prasant Mohapatra. 2017. WearIA: Wearable Device Implicit Authentication based on Activity Information. In *Proceedings of IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*.
[39] Nan Zheng, Kun Bai, Hai Huang, and Haining Wang. 2014. You are how you touch: User verification on smartphones via tapping behaviors. In *Proceedings of IEEE International Conference on Network Protocols*.

A PROOF OF THEOREM 5.1

We first write down Theorem 5.1 here again.

THEOREM A.1. *The raw size of beat-PIN space is*

$$|\Pi| = \sum_{l=1}^{L_{\max}} \binom{\frac{T_{\max}}{\sigma} - (\frac{\tau_b}{\sigma} - 1) \times l - (\frac{\tau_s}{\sigma} - 1) \times (l-1)}{2l-1},$$

where $L_{\max}$, $T_{\max}$, σ, τ_b and τ_s stand for the maximum length, maximum time duration, minimum unit of the system clock, minimum value of a beat-duration duration and minimum value of a space-duration, respectively.

PROOF. Our objective is to find out how many different beat-PINs there are, when the beat-PIN's length and time duration are limited by $L_{\max}$, $T_{\max}$, respectively. Meanwhile, due to the constraint of human tapping speed, we assume that the beat-duration and space-duration are lower bounded by τ_b and τ_s, respectively.

First of all, with the minimum system clock unit σ, the corresponding maximum number of time slots in a beat-PIN is calculated by $N = \frac{T_{\max}}{\sigma}$. We now analyze the number of all possible beat-PINs when their lengths, i.e., number of beats, are l ($l \in [1, L_{\max}]$).

By treating each beat-duration as a "white" bin, each space-duration as a "black" bin, and the entire N time slots as N stars, our problem is equivalent to find out the number of all possible configurations to fit N stars into l "white" bins and $l-1$ "black" bins. Besides, there should be no less than $\frac{\tau_b}{\sigma}$ stars in each "white" bin and no less than $\frac{\tau_s}{\sigma}$ stars in each "black" bin.

Instead of coping with the above problem directly, we first check a simpler question. There are N stars and $l + l - 1 = 2l - 1$ bins, including both the black ones and white ones. How many configurations are there to place at least one star into each bin? Note that bins are distinguishable, while stars are not. In fact, this is a standard "Stars and Bins" (or called "Stars and Bars") problem [29]. Its solution is calculated by $\binom{N}{2l-1}$.

We are now ready to solve our problem. The idea is to first place $\frac{\tau_b}{\sigma} - 1$ stars in each "white" bin and $\frac{\tau_s}{\sigma} - 1$ stars in each "black" bin. Then there are $N - (\frac{\tau_b}{\sigma} - 1) \times l - (\frac{\tau_s}{\sigma} - 1) \times (l-1)$ stars left. The remaining issue is to find out the number of configurations to place $N - (\frac{\tau_b}{\sigma} - 1) \times l - (\frac{\tau_s}{\sigma} - 1) \times (l-1)$ stars into $2l-1$ bins, such that each of them contains at least one star. Following the solution to the standard "Stars and Bins" problem discussed above, it gives the result of our problem as $\binom{N-(\frac{\tau_b}{\sigma}-1)\times l-(\frac{\tau_s}{\sigma}-1)\times(l-1)}{2l-1}$. Note that this is the number of all possible beat-PINs of length l. When their lengths range from 1 to $L_{\max}$, we have

$$\begin{aligned}|\Pi| &= \sum_{l=1}^{L_{\max}} \binom{N - (\frac{\tau_b}{\sigma} - 1) \times l - (\frac{\tau_s}{\sigma} - 1) \times (l-1)}{2l-1} \\ &= \sum_{l=1}^{L_{\max}} \binom{\frac{T_{\max}}{\sigma} - (\frac{\tau_b}{\sigma} - 1) \times l - (\frac{\tau_s}{\sigma} - 1) \times (l-1)}{2l-1},\end{aligned}$$

which ends the proof. □

B STATISTIC ANALYSIS OVER $L_{\max}$, $T_{\max}$, τ_b AND τ_s

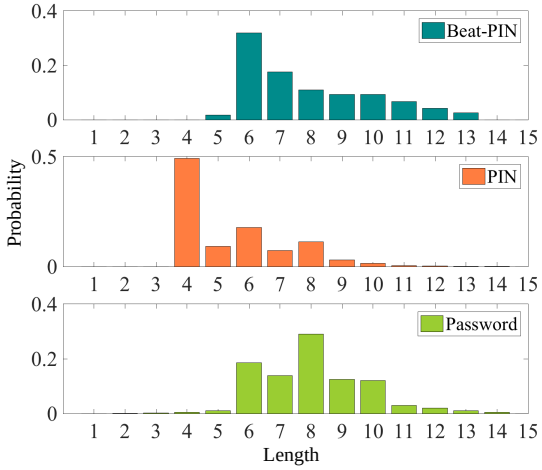

Figure 15: Statistics on lengths of beat-PINs, digit-PINs and traditional passwords.

Figure 15 shows the distribution of beat-PIN length according to our collected dataset. Besides, we also depict those of digit-PINs (composed of pure numbers) and traditional passwords (composed of ASC-II characters), according the result from [6, 9]. We can see that the majority of beat-PINs have length ranging from 6 to 10, which takes about 90% of all the samples. Interestingly, this result matches well with the distribution of traditional password length, whose major components also lie between 6 and 10. Regarding to the length of digit-PINs. We notice that they mainly range between 4 and 8. Besides, the most existing ones are 4 and 6. This may due to the fact that most smartphones require a PIN at length of 4 or 6. Just like digit-PINs and traditional passwords, the length of beat-PINs plays a significant role in its security. This is because as the beat-PIN gets longer, the number of ways its constituent parts can be shuffled into a new combination gets exponentially larger and therefore, much harder to take wild guesses at.

Figure 16(a) shows time duration distribution of beat-PINs. We can see that most of beat-PINs, about 95.2% of them, last between 1 to 5 seconds. It indicates that beat-PINs can be entered very fast. Figure 16(b) and Figure 16(c) illustrate the distribution of beat-duration τ_b and space-duration τ_s, respectively. For τ_b, its value spreads from 0.08s to 2.91s; for τ_s, its value spreads from 0.12s to 3.23s. These

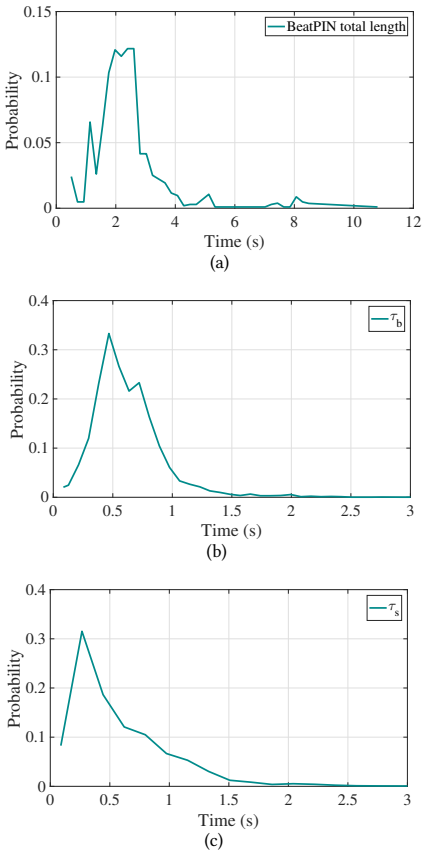

Figure 16: Distribution of three characters of beat-PINs. (a) Beat-PIN length. (b) τ_s. (c) τ_b.

statistic results show that users demonstrate sufficient diversity in terms of overall time duration, beat-duration and space-duration when generating beat-PINs, which can substantially enlarge the beat-PIN space.

C SHOULDER SURFING ATTACK ASSISTED WITH CAMERA

In addition to the type-I shoulder surfing attack discussed in Section 8.1.2, we further consider the stronger shoulder surfing attacks that are powered by camera. We prepared a video that filmed legitimate users entering 10 different beat-PINs. The video is recorded by an iphone 7. We seated the camera about 0.5 meters away from the legitimate user and had user's finger movement fully captured. The film is then shown in the same iphone 7 to 20 volunteers who play as attackers. Attackers are allowed to replay the film arbitrary times to recognize and remember the beat-PIN rhythms. As before, each of them can make up to three access attempts. It is considered to success if any one of the three trials passes the authentication.

Table 8 shows that the success rate of type-II shoulder surfing attack drops quickly when legitimate users choose longer beat-PINs. Specifically, when the length is 10, the attacker's FAR is as low as 11.3%. This is because rhythms of a beat-PIN is more difficult to

Table 8: FAR comparison under type-II shoulder surfing attacks

Length	**6**	**7**	**8**	**9**	**10**
Beat-PIN	39.3%	36.8%	17.6%	15.0%	11.3%
Digit-PIN	85.5%	82.1%	75.4%	72.7%	70.4%
Password	82.3%	79.6%	71.5%	68.2%	65.7%
Pattern lock	78.1%	71.2%	66.7%	54.5%	34.8%

Table 9: The impact of beat-PIN length to its memorability.

Beat-PIN Length	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Total number	2	38	21	13	11	11	8	5	3
Fail number	0	2	0	1	1	0	0	0	0

mimic and recall as it becomes longer. However, we have to admit that the resistance performance for shorter beat-PINs is less attractive. Therefore, users should be advised to choose longer beat-PINs in order to resist type-II shoulder surfing attacks. Besides, we further compare the attacker's success rate of *beat-PIN* with other commonly used "something you know" style authentication, such as digit-PIN, password and patter lock. The shoulder surfing attack over these authentication is conducted under a similar setting as with *Beat-PIN*. Note that the length of a pattern lock passcode is considered as the number of "dots" the line passes by. We observe that *beat-PIN* outperforms the other three. Specifically, when the length is 10, the attacker's success rate over *beat-PIN*, digit-PIN, password and patter lock is 11.3%, 70.4%, 65.7% and 34.8%, respectively. Note that we confine the comparison within "something you know" style authentication. This is because the secret information involved therein are observable by attackers, while that from "something you are" or "something you have" style authentication is not, e.g., fingerprint, iris and hand geometry. Therefore, generally speaking, "something you are" or "something you have" style authentication will be more robust against shoulder surfing attack than "something you know" style authentication, including *beat-PIN*.

D MEMORABILITY

In order to test the memorability performance of beat-PINs, we design a recall test. It was conducted one week after the main data collection session in phase-I user study. Specifically, all the volunteers were asked to re-enter their previously chosen beat-PINs three times. If any of the three trials passes the authentication, then we consider the user can recall his/her beat-PIN correctly. The test result is really promising, as only 4 out of all 112 fail, leading to the overall recall fail ratio as low as 3.6%. We would also like to mention that no user was exposed to the system during the week. Thus, we can expect a higher success recall rate if users practice beat-PINs multiple times every day. Moreover, we further illustrate the impact of beat-PIN length to its memorability. There are 2 fails among total 38 samples when the length is 6, and one fail for each of the cases when the length is 8 and 9, leading to the fail ratio as 5.2%, 7.7%, 9.1%, respectively. Therefore, there experiences a light increase in terms of recall fail ratio as the beat-PIN length grows.

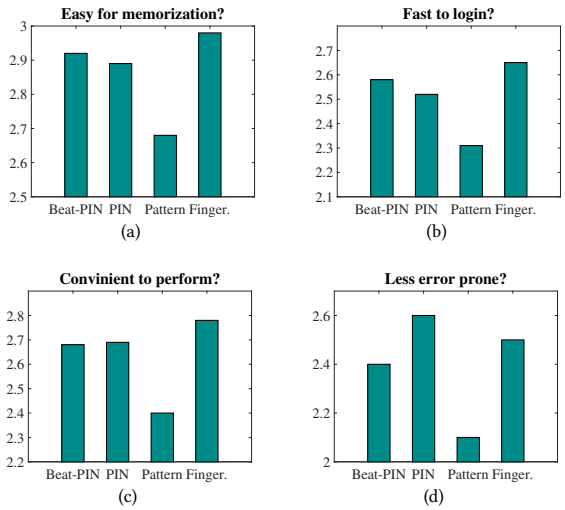

Figure 17: Average ratings for the four usability questions.

E SURVEY RESULTS

In addition to the experiments, we further analyze the usability of *Beat-PIN* via the survey results. Specifically, at the end of phase-II experiments, questionaries were distributed among all volunteers. They were asked to rate *Beat-PIN* from the following 4 perspectives and compare it to the commonly used methods on mobile devices, including digit-PIN, pattern lock and fingerprint. 1) Is it easy to memorize? 2) Is it fast to login in? 3) Is it convenient to perform? 4) Is it less error prone? For each question, we use 3 (1-3) levels representing responses of disagree, neutral and agree. The average ratings for different authentication methods calculated from the resulted 49 questionnaires are shown in Figure 17.

Specifically, Figure 17(a) compares the memorization of *Beat-PIN*, digit-PIN, pattern lock and fingerprint-based authentication. The result shows that the average score regarding "easy for memorization" of *Beat-PIN* is 2.92, which is slightly higher than that for digit-PINs, i.e., 2.89, and significantly higher than that for pattern lock, i.e., 2.68. Note that all these three belong to the "something you know" style authentication, where users have to remember some secret information and correctly "prove" it to get authenticated. *Beat-PIN* has the best performance among these three. Meanwhile, fingerprint authentication belongs to the "something you have" style authentication and thus requires the least effort for memorization. Figure 17(b) tells that beat-PINs are regarded as the second fastest to log in, even faster than digit-PINs. According to Figure 17(c) and Figure 17(d), our scheme is generally regarded as convenient and less error prone. In particular, it is rated better than pattern lock in these two aspects.