TMGuard: A Touch Movement-Based Security Mechanism for Screen Unlock Patterns on Smartphones

Weizhi Meng[1(✉)], Wenjuan Li[2], Duncan S. Wong[3], and Jianying Zhou[1]

[1] Infocomm Security Department, Institute for Infocomm Research, Singapore, Singapore
{mengw,jyzhou}@i2r.a-star.edu.sg
[2] Department of Computer Science, City University of Hong Kong, Hong Kong, China
wenjuan.li@my.cityu.edu.hk
[3] Applied Science and Technology Research Institute (ASTRI), Hong Kong, China
duncanwong@astri.org

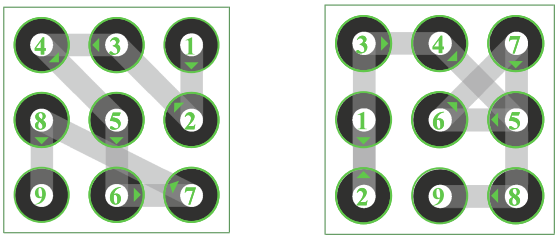
Abstract. Secure user authentication is a big challenge for smartphone security. To overcome the drawbacks of knowledge-based method, various graphical passwords have been proposed to enhance user authentication on smartphones. Android unlock patterns are one of the Android OS features aiming to authenticate users based on graphical patterns. However, recent studies have shown that attackers can easily compromise this unlock mechanism (i.e., by means of smudge attacks). We advocate that some additional mechanisms should be added to improve the security of unlock patterns. In this paper, we first show that users would perform a touch movement differently when interacting with the touchscreen and that users would perform somewhat stably for the same pattern after several trials. We then develop a touch movement-based security mechanism, called *TMGuard*, to enhance the authentication security of Android unlock patterns by verifying users' touch movement during pattern input. In the evaluation, our user study with 75 participants demonstrate that *TMGuard* can positively improve the security of Android unlock patterns without compromising its usability.

Keywords: Mobile security · User authentication · Android unlock patterns · Usability · Touch gestures · Behavioral biometric

1 Introduction

Smartphones like Android phones and iPhones have become extremely popular in our daily lives and routines, where the Android phones and iPhones captured nearly 82.8 % and 13.9 % global smartphone market share each in Q2 2015 [11]. With the increasing capability of current phones, users are likely to store their personal information such as passwords and credit card numbers on

M. Manulis et al. (Eds.): ACNS 2016, LNCS 9696, pp. 629–647, 2016.
DOI: 10.1007/978-3-319-39555-5_34



(a) (b)

Fig. 1. Cases of 9-dot Android unlock pattern generated by Berkeley Churchill.

their phones [12], and use the phones for sensitive tasks such as mobile banking [21]. However, according to a survey of mobile phone users in 2012 [24], among the most common issues, 67 % of respondents had dealt with lost or stolen mobile devices. In this case, user authentication on smartphones has become very crucial to protect the stored private and sensitive data.

At present, the most commonly used method for user authentication is based on text or PIN codes, in which users are required to input correct characters for authentication. However, several studies indicated that this kind of authentication had drawbacks regarding both usability and security [6]. For instance, users have difficulty in remembering complex and random passwords which is known as long-term memory (LTM) limitations [26]. Therefore, users are likely to choose a simple password to reduce the burden of memory. According to a report from SplashData, the worst password used in 2013 is "123456" [22].

To mitigate the drawbacks of the knowledge-based passwords, graphical passwords (*GPs*) have been developed as an alternative aiming to enhance the process of user authentication. Several psychological studies like [18] have indicated that the human brain was better at remembering and recognizing images than text. Current smartphones using the Android operating systems adopt a type of screen unlock mechanism that requires users to input correct patterns to unlock the phones within a 3×3 touch-enabled grid. Users can start touching on any one of the dots, swipe the fingers to touch more dots and construct a pattern. For example, Fig. 1 shows two patterns generated by an unlock pattern generator from Berkeley Churchill [3]. The number from 1 to 9 indicates the sequence of dots during the touch movement.

Motivations. Due to the popularity of the Android unlock patterns, many adversarial techniques have been explored in the literature aiming to compromise this mechanism. For instance, since users can only choose a minimum of 4 and a maximum of 9 dots to generate such a pattern, the total number of possible patterns is 389,112 [2], where it is still feasible for a brute-force attack. What is worse, by means of several other types of attacks, the password space of the unlock patterns can be greatly reduced. The details of potential attacks can be referred to Sect. 2.2. Therefore, it is very crucial for Android unlock patterns to improve its authentication security in practical usage.

Contributions. To enhance the authentication security of Android unlock patterns, it is reasonable that some additional mechanisms should be added to securing these patterns. Motivated by work [8,15,28,30], we believe that behavioral biometric is one of the potential solutions. Our main goal is to complement the existing solutions in enhancing authentication security on smartphones. The contributions of our work can be summarized as below:

- In this work, we begin by conducting a study with 50 participants to investigate how users would perform in creating unlock patterns. It is found that different users would input unlock patterns differently regarding touch movement, in which the average speed of touch movement may be varied. On the other hand, it is found that users are able to perform a more stable movement for inputting the same pattern after several trials.
- We then develop a security mechanism based on touch movement, called *TMGuard*, to authenticate users in terms of both their input patterns and extracted information from touch movement. Distinguished from other work, we develop two approaches of *dot-dot pattern computation* and *proportional matching* in order to better model and compare users' touch movements.
- In the evaluation, we conduct a user study with a total of 75 participants and it is found that *TMGuard* can enhance the authentication security of unlock patterns with good usability in practice.

The remaining parts of this paper are organized as follows. In Sect. 2, we introduce the background of Android unlock patterns and present some potential attacks. Section 3 presents our first study to investigate how users would perform touch movement when inputting unlock patterns. Section 4 describes the proposed security mechanism of *TMGuard* in detail, and presents another user study to evaluate its performance. Finally, we conclude the paper in Sect. 5.

2 Background and Related Work

2.1 Android Unlock Patterns

Android unlock patterns are one of the graphical password schemes that requires users to swipe their finger to construct a pattern and unlock the device. Specifically, it is a modified version of Pass-Go [20] in order to adapt for the small touchscreens on typical smartphones. It allows users to create a pattern by means of 4 dots at least and 9 dots at most, within a 3×3 grid on the touchscreen, and to use it to unlock a mobile device. To create a valid unlock pattern, three major rules are applied as follows [19,23]: (1) One cannot use a dot more than once, since it is virtually removed after selection. In Fig. 1(b), it is shown that *dot 1* can be only selected once when touching back from *dot 2* to *dot 3*. (2) At least 4 dots must be chosen and only straight lines are allowed. (3) It is not possible to create a line using three dots, without selecting the middle one, unless the latter has been previously visited.

Based on these rules described above, it is not easy to compute the number of total patterns directly, but one can enumerate all possible patterns: there are 389,112 (2^{19}) possible patterns [2]. These possible patterns would be sufficient if users can select the patterns uniformly, however, the situation is much worse in practice (i.e., it offers less security than a three digit PIN [23]).

After users input one unlock pattern, this mechanism will convert the pattern to byte array, transform it to the SHA-1 hash function and save it in the phone (e.g., the stored file name is *gesture.key*). Due to the popularity of this mechanism, it has been available not only in Android OS, but also in iOS. For example, *Cydia Tweak* [25] currently allows users to add an Android-inspired pattern unlock system to a jailbroken iPhone handset.

2.2 Potential Attacks

Since an Android unlock pattern is composed of several dots, this mechanism suffers from the issue of 'hot-dot'. In [1], a pilot study has shown that users have some preferences on the *start points* and *end points* when drawing the pattern. For instance, they reported that about 52.08 % of the participants preferred to start their patterns from the top left node. In addition, Aviv et al. [2] indicated that unlock patterns can be retrieved by launching smudge attacks. The basic idea is that users may leave an oily residue or smudges when swiping their fingers on the device. In the experiments, they concluded that intentionally cleaning with cloth or putting the phone to pocket was not enough to prevent pattern retrieval. Therefore, an attacker can easily capture a photo of the touchscreen and perform necessary contrast and brightness adjustments to the captured photo to retrieve the pattern.

In addition, Android unlock patterns have an inherent limitation, in which only 9 touch dots can be used during the pattern creation. In such case, the total number of possible patterns is 389,112, which makes brute force attacks still feasible if a weak pattern is chosen by users. For instance, Pereira Botelho [19] conducted a preliminary study to explore the performance of 4-dot unlock patterns against brute force attacks. The experimental results indicated that the maximin time needed to crack a 4-dot pattern is less than 4 min.

As there are only 9 touch dots for creating an Android unlock pattern, we consider that additional mechanisms could be added to enhance the authentication security of unlock patterns. One of the possible solutions is to use *behavioral biometrics*, which use measurements from human actions [4]. As discussed in previous research such as [8,15,17,28,30], users may perform differently when using their phones, so that it is feasible to authenticate users based on their gestures. In this work, we thus aim to improve the authentication security of Android unlock patterns by combining it with users' touch behavior.

3 Study on Touch Movement for Unlock Patterns

As shown in Fig. 1, Android unlock patterns consist of 9 nodes in a 3×3 grid. In practice, to construct a valid pattern, users should use one touch movement

Fig. 2. (a) The interface of CyanogenMod Android OS; (b) The screen of Android unlock patterns; (c) An instance of raw data collection.

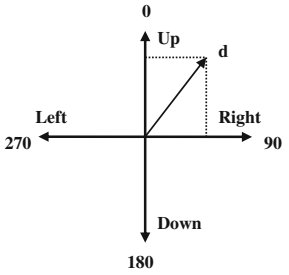


Fig. 3. Directions for a touch movement.

to draw a pattern by selecting dots in a certain sequence. A *basic* question here is how users would input patterns when performing touch movements on their phones. We have two intuitive hypotheses:

Hypothesis 1. Distinct users may perform the touch movement differently when inputting the patterns.

Hypothesis 2. Through some input trials, one user's touch behavior may become more stable.

To verify these hypothesis, we conduct a user study with 50 participants. In this section, we introduce how to collect raw data, select and define features for a touch movement, and analyze the collected results.

3.1 Data Collection

Although the unlock patterns will be hashed and stored in a pre-defined file like *gesture.key*, we do not use it directly in this work. Instead, to record and collect the input data, we used a modified Google/HTC Nexus One Android phone with a capacitive touchscreen (resolution 480×800 px). Specifically, we updated the phone with a modified Android OS version 2.2 based on *CyanogenMod*.[1] The modification consists of changes to the application framework layer to record raw data from the touchscreen, such as the timing of touch inputs, the coordinates of x and y, and the type of the input (e.g., single-touch or touch movement).

To facilitate the real observation, we installed a *log application* allowing us to more easily extract the recorded data from the phone. A Beta version of our customized-Android OS can be downloaded at Sourceforge website.[2] The major advantage of using our data collection is that we can collect all raw data during a user's input including users' behavioral data and input patterns, and then compute the related features, while using *gesture.key* can only extract those patterns. The interface of the *CyanogenMod* Android OS can be seen in Fig. 2(a), the interface of Android unlock patterns can be referred to Fig. 2(b), and an instance of raw data collection is given in Fig. 2(c).

[1] http://www.cyanogenmod.com/.
[2] https://sourceforge.net/projects/touchdynamicsauthentication/files/Android_OS/.

3.2 Touch Movement Features

In this work, we mainly consider 4 standard directions for a touch movement: up, down, left and right. Figure 3 defines each direction and thus we can use a degree d to describe the direction of a touch movement.

We use two features to describe a specific touch movement: the speed of touch movement (STM) and the angle of touch movement (ATM). Suppose a touch movement selects two dots *D1* and *D2* with coordinates (x1, y1) and (x2, y2) respectively, while the event system time is *S1* and *S2*. As shown below, Eq. (1) describes how to calculate STM and Eq. (2) describes how to calculate ATM (e.g., with an angle d).

$$STM = \frac{\sqrt{(x2 - x1)^2 + (y2 - y1)^2}}{S2 - S1} \tag{1}$$

$$ATM\ (d) = \arctan \frac{y2 - y1}{x2 - x1}, \theta \in [0, 360^{\circ}] \tag{2}$$

3.3 Study Design and Result Analysis

In the study, we have recruited 50 participants who are volunteers and interested in this topic. Among them, 60 % are males and the remainder are females. All participants are regular mobile phone users and aged between 15 and 60. Among them, 76 % currently use Android OS while the others use iOS. But all of them have used or experienced Android unlock patterns before. As incentives, $20 gift vouchers were given to each participant. The detailed information of participants is shown in Table 1.

More specifically, we introduced our objectives to all participants before they joined the study, showed what kind of data would be collected and acknowledged that all data collected in the study was used in an anonymized way. Overall, there are two phases in the user study:

- *Phase1.* Each participant has to create a total of 3 different patterns, while for each pattern they should re-enter it three times (recorded) after two practice (not recorded) in one day. This makes us collect 150 patterns and 450 trials in total.
- *Phase2.* We provide each participant with an Android phone equipped with our modified Android OS. Each participant should choose one of their created patterns in *Phase1* as the phone's unlock pattern, and freely use the phone for another 2 days. After that, all participants were asked to return and input their patterns in our lab for three times.

The objective of *Phase1* is to explore whether users can perform touch movement differently when inputting the patterns, while the objective of *Phase2* is to investigate whether users can input the pattern stably after a number of trials.

We show the average speed of touch movement ($ASTM$) for different users in Fig. 4. In particular, Fig. 4(a) shows the $ASTM$ for user ID from 1 to 25 while

Table 1. Participants' information in the first user study.

Age range	Male	Female
< 25	7	5
25–35	13	9
35–45	6	3
> 45	4	3

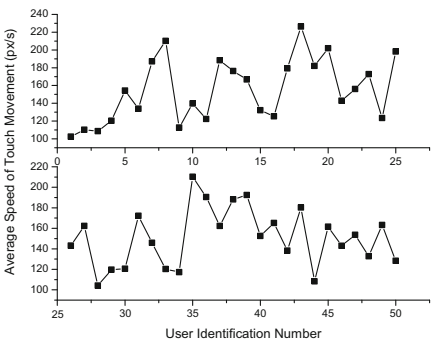


Fig. 4. Average speed of touch movement (users from 1 to 50).

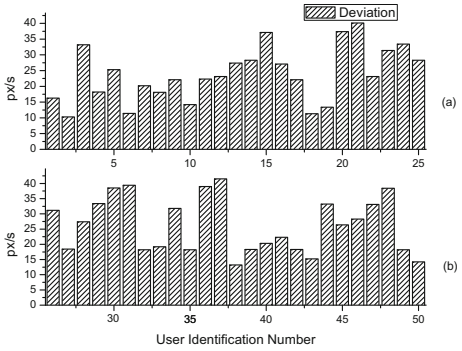


Fig. 5. Deviation for average speed of touch movement (users from 1 to 50).

Fig. 4(b) shows the *ASTM* for user ID from 26 to 50. The calculation of average movement speed is based on the collected 9 trials for each user. The average speed is ranged from nearly 100 px/s to 230 px/s. The figure shows that users would perform differently when swiping their fingers on the touchscreen. For example, it is seen that User 8, 18, 25 and 35 could perform a high movement speed over 200 px/s, while User 1, 3, 28 and 44 might perform a very slow speed less than 100 px/s. Others may perform a speed between these two.

In addition, as shown in Fig. 5, we compute the deviations for each user based on their 9 trials. It is noticeable that several users like User 2, 6, 10, 18, 19, 38 and 50 could perform more stably than other users (i.e., the deviation is less than 15 px/s), but some users like User 15, 20, 21, 30, 31, 36, 37 and 48 would perform not stably (i.e., the deviation is more than 35 px/s). The results reveal that users would not perform consistently when inputting different patterns, which is in line with our common sense. However, our interests are focus on whether users would perform consistently to draw a same pattern, or whether the deviations are below an appropriate threshold. To explore these questions, we further compute the deviations for all users when drawing the same pattern (3 trials for the same pattern) in Fig. 6(a). We have two key observations based on the comparison between Fig. 6(a) and Fig. 5:

– The deviation for the same pattern is much lower than that for inputting all patterns (by comparing Fig. 6(a) with Fig. 5). This is reasonable as users

may perform different movement speeds according to distinct patterns. For example, for a complex pattern, users may slow down the speed while for some 'easy' patterns, users may perform a touch movement fast.
– Nearly 75 % deviations are below 25 px/s while only 3.3 % deviations are over 30 px/s. This observation shows that users could perform more consistently to some degree, when inputting the same pattern as compared to inputting different patterns. It also shows that the speed of touch movement can be used to distinguish different users when inputting unlock patterns.

In *Phase2*, all users are required to input their selected patterns to unlock the phone for three times after a 2-day usage. The results of deviation are shown in Fig. 6(b). Similarly, we have two key observations as follows:

– All deviations are below 17 px/s. As compared to Fig. 6(a), Fig. 6(b) shows that the deviation can be greatly decreased after more practices. We also interview users after they input the selected patterns, and it is found that users would input the patterns to unlock the phone at least 6 times and at most 25 times each day, depending on different usage of the phones. Thus, before they input the patterns in our lab, they have already input the pattern at least 12 times.
– Only 6 % deviations are over 12 px/s and up to 84 % deviations are very close to, or even below 10 px/s. As compared to Fig. 6(a), this observation positively indicates that users would perform a touch movement much more stably after a period of time. Based on this observation, we believe that it is feasible and promising to enhance the authentication security of Android unlock patterns by combining it with behavioral biometrics.

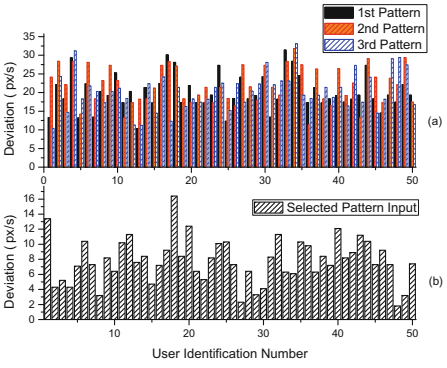


Fig. 6. Deviation for average speed of touch movement (users from 1 to 50): (a) Deviation in *Phase1* and (b) Deviation in *Phase2*.

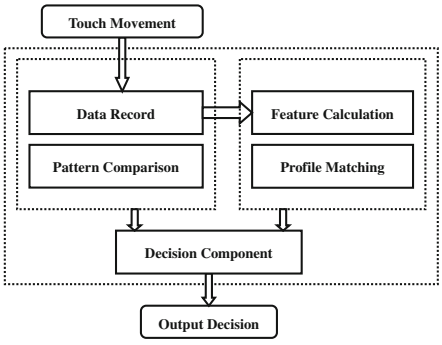


Fig. 7. The high-level architecture of TM-Guard.

3.4 Discussions

The results illustrated above demonstrate the feasibility of applying behavioral biometrics to improving the security of Android unlock patterns. From the study, we verify our two hypothesis: users would perform a touch movement differently when inputting the patterns and they would perform more stably after inputting a pattern several times. These users' behavioral habits benefit the application of touch gestures in user authentication on smartphones.

However, we should still pay attention to an issue. It is noted that the average touch speed of some users may be similar. For example, based on Fig. 4, we find that User 1 can perform an *ASTM* of 102.5 px/s while User 28 can perform an *ASTM* of 104.2 px/s. Therefore, it is still too vague for a mechanism to use *ASTM* only to distinguish different users without considering their deviations. Otherwise, this problem can cause many usability problems (e.g., a high false rejection rate). To mitigate this issue, motivated by [9,15,16], we believe that some parameters/features like the *angle of touch movement* can be combined to better distinguish users.

What is more, we further develop and introduce two methods called *dot-dot pattern computation* and *proportional matching* in our proposed security mechanism (see next section) to maintain a balance between security and usability. The *dot-dot pattern computation* aims to describe a user's touch movement more accurately by separating a pattern into several segments while the *proportional matching* attempts to provide better usability through allowing reasonable touch deviations.

4 TMGuard: A Security Mechanism for Android Unlock Patterns

As illustrated above, it is identified that distinct users would perform the touch movement differently when drawing a pattern, while they would perform more stably for the same pattern after several trials. Based on the observations, it is feasible to apply behavioral biometrics to enhancing the security of Android unlock patterns. In this section, we therefore develop a security mechanism based on touch movement, called *TMGuard*, attempting to improve the authentication security of drawing unlock patterns. This mechanism can be utilized to complement the existing security solutions.

4.1 Mechanism Design

We present the high-level architecture of *TMGuard* in Fig. 7, which consists of five major components: *Data Record*, *Feature Calculation*, *Pattern Comparison*, *Profile Matching* and *Decision Component*.

– *Data Record.* This component is mainly used to record users' input when they interact with the touch screen and to collect relevant data for speed and angle calculation (e.g., timing and coordinates).

- *Feature Calculation.* This component is responsible for calculating the speed and angle of a touch movement based on the collected data.
- *Pattern Comparison.* This component is used to compare the unlock pattern input with the stored pattern and to report the result like acceptance or decline to the *Decision Component*.
- *Profile Matching.* This component is responsible for establishing the normal profile of users' input (e.g., touch movement) and matching the current input behavior with the normal profile. The result will be forwarded to the *Decision Component*.
- *Decision Component.* This component is responsible for collecting the results and making the final decision whether the current user is legitimate. Users can only be authenticated by both inputting the correct pattern and passing the examination of *Profile Matching*.

4.2 Profile Matching

As discussed earlier, it is not good enough to use only one $ASTM$ to distinguish different users due to false rates. To address this issue, we add the *angle of touch movement* (ATM) in the profile construction. Moreover, in order to establish a more reliable normal profile, we develop another method called *dot-dot pattern computation*. This method aims to construct an accurate normal profile by separating a pattern into several segments. That is, it records pairs of (STM, ATM) for any two sequential touched dots in a pattern.

Dot-Dot Pattern Computation. Taking the pattern in Fig. 1 as an example, our mechanism records the speed and angle when the finger swipes from *dot 1* to *dot 2*. When the finger swipes from *dot 2* to *dot 3*, *TMGuard* then calculates the speed and angle for this movement in-between. Similarly, all pairs of (STM, ATM) will be recorded during the construction of a pattern. In this case, when the pattern is finished, *TMGuard* would log a collection of pairs regarding average touch speed and touch angle between any two sequential touched dots in a pattern. For a 9-dot pattern, there will be 8 pairs (or segments) to construct a normal profile.

In real usage, *TMGuard* will record three trials from users in inputting their patterns, and use the average value to establish the normal profile aiming to improve the reliability. In this case, the construction of a normal profile can be represented by means of Eq. (3).

$$Profile = \{\bigcup^{j}(ASTM,\ AATM)_i^{i+1}\} \quad (4 \leq j \leq 9; i = 1, ..., j) \tag{3}$$

In the equation, j means the number of selected dots in an Android unlock pattern, i means dot number (or dot sequence number). $ASTM$ means the average speed of touch movement between *dot* i and *dot* $i+1$, while $AATM$ means the average angle of touch movement between *dot* i and *dot* $i+1$. Thus for a j-dot pattern, the number of collected pairs is $j-1$. There are two major objectives of using the *dot-dot pattern computation* in *TMGuard*:

- We identify that it is not reliable to authenticate users by means of only one *ASTM* for the whole pattern. In this case, the use of *dot-dot pattern computation* can provide more segments of *STM* during the authentication, so that users' touch behavior can be examined more precisely. In other words, *dot-dot pattern computation* attempts to describe a touch movement more accurately by recording the data between any two dots. This can improve the authentication security of Android unlock patterns.
- The same in our previous user study, it is found that the overall *ASTM* can be significantly affected by an abnormal (or unexpected) touch movement between two dots. Therefore, separating these dots and computing their *ASTM* respectively may eliminate these negative effects to some extent and improve the usability of *TMGuard*.

To authenticate a user, the component of *profile matching* will record his/her current inputs, calculate the pairs of $(ASTM,\ AATM)$ between any two touched dots, and compare these pairs with the stored normal profile.

Tradeoffs Between Security and Usability. Traditionally, users should perform a similar touch movement to unlock the pattern with the same pairs of $(ASTM,\ ATM)$ in a right sequence. However, we notice that users are often hard to exactly perform the same behavior. For example, the speed and angle of a touch movement between two dots may be a bit different. This is actually a big challenge for behavioral biometric authentication. It is also a big difference between *pattern comparison* and *profile matching*. If we do not improve the traditional profile matching, it can definitely increase false rejection rate and decrease usability. Thus, tradeoffs should be made between security and usability. Below we develop a novel scheme for profile matching.

Proportional Matching Scheme. For many existing behavioral biometric schemes like [9,15,28], machine learning techniques have been widely used in profile matching. But a major limitation is that it is hard to train an appropriate classifier in real scenario [13]. To avoid this issue, in this work, we develop a statistic-based scheme in *TMGuard*, called *proportional matching*, aiming to improve its usability, and make a balance between security and usability.

This method specifically utilizes a *confidence threshold* during the authentication. That is, users are only allowed to perform a touch movement within a defined deviation. For instance, if we set the *confidence threshold* to 0.98, thus, it is allowed a deviation less than 0.02 $(= 1 - 0.98)$ as compared to the stored normal profile. For a numerical example, if we have a pair of $(110.5, 23^o)$, with a *confidence threshold* of 0.98, users then can be authenticated if the touch movement speed and the angle fall into an interval of $[108.3, 112.7]$ and $[22.54^o, 23.46^o]$ respectively. The effectiveness of this scheme is based on our observation that users would perform more stably when they have several input trials.

We have two major objectives of developing such a scheme in *TMGuard*:

- Users' inconsistent behaviors are a big challenge (open problem) for any behavioral biometric authentication scheme, which can significantly reduce the effectiveness of behavioral authentication. *TMGuard* attempts to provide another

protection for Android unlock patterns, so that we do not expect to compromise the usability; otherwise, users may lose interests in using the application. The *proportional matching scheme* is thus used with the purpose of improving the usability of *TMGuard*.
– During the previous user study, it is found that users may perform more stably after inputting the selected patterns several times. This makes us believe that loosing the profile matching appropriately would not compromise the authentication security. On the other hand, according to specific scenarios, it is very easy to adjust the *confidence threshold* of the *proportional matching scheme*, making *TMGuard* more flexible in practical applications.

4.3 User Study for TMGuard

To investigate the performance of *TMGuard*, we conduct another user study with a total of 75 participants. All participants are regular mobile phone users and 40 % of them were joined our previous study in drawing unlock patterns. There are 45 males and 30 females and aged in the range from 18 to 60. Among them, 66.67 % are students while the others are company employees, senior citizens and businessmen. As incentives, $20 gift vouchers were given to each participant. The detailed information is shown in Table 2.

Table 2. Participants' information in the second user study.

Information	Male	Female	Occupation	Male	Female
Age < 25	10	7	Students	26	24
Age 25–35	20	15	Company employees	3	2
Age 35–45	9	5	Business people	8	4
Age > 45	6	3	Senior citizens	5	3

During the lab study, all participants were provided with our modified Android phones to avoid any implementation differences. There are two major phases in the study.

– *Phase1: in-lab study.* Users require to create a 4-dot and 9-dot pattern respectively and re-draw the pattern for three times. *TMGuard* will collect these trials, calculate the data and build the corresponding normal profile. The *confidence threshold* is set to 0.9. Then after 5 practice trials, users input the same pattern for another three real trials for authentication.
– *Phase2: out-of-lab study.* Users can freely create a pattern as their phone lock (note that they should also re-draw the pattern for three times to build normal profiles) in the lab and freely use the phone for another 2 days out of lab. When users input patterns, records will be stored. Finally, they should input the same pattern for three times in our lab.

The objective of *phase1* is to explore the initial performance of *TMGuard* and investigate how to decide an appropriate *confidence threshold*, while the latter aims to study the performance of *TMGuard* in a real scenario.

Result Analysis for Phase1. In this phase, each user can perform the authentication three times for both 4-dot and 9-dot pattern respectively, so that we can obtain 225 trials for each pattern. We show the results of authentication attempts differentiated by gender in Table 3. The table shows that male participants can achieve a successful login with a rate of 98.5 % for a 4-dot pattern, while they can reach a successful rate of 97.8 % for a 9-dot pattern. The slight decrease is due to that more pairs should be authenticated for a 9-dot pattern (e.g., 8 pairs of dot-dot patterns) as compared to a 4-dot pattern (e.g., 3 pairs of dot-dot patterns). The results are reasonable as more pairs of dot-dot patterns will increase the uncertainty during a touch movement (i.e., increasing the deviation of inputting patterns). Regarding female participants, it is noticed that they perform very similarly for 9-dot pattern, but achieve better performance for 4-dot pattern than males.

After the user study, we interviewed all users and found that 78.7 % of the participants are satisfied with the login experience, and encouragingly 80 % of them consider that *TMGuard* can improve the security of Android unlock patterns. In addition, 73.3 % of them acknowledge that they would like to try this mechanism in regular use. As this is a scientific and security related study, we notice that users' answers may be affected by the environment. Even so, the feedback can still positively support the performance of *TMGuard*.

In contrast, Table 4 shows the authentication results if we do not use *dot-dot pattern computation*. It is noticeable that the successful authentication rate decreases significantly for both male and female participants. Taking 9-dot patterns as an example, the successful rate is decreased from 97.8 % to 91.1 % for males and from 97.8 % to 88.9 % for females respectively. To study the effect of *proportional matching scheme*, we further present the authentication results with different *confidence thresholds* for the 9-dot patterns in Fig. 8(a). The figure

Table 3. Authentication results of users' trials with *TMGuard* including *confidence threshold* and *dot-dot pattern computation*.

Successful rate	4-dot pattern	9-dot pattern
Males	133/135 (98.5 %)	132/135 (97.8 %)
Females	90/90 (100 %)	88/90 (97.8 %)

Table 4. Authentication results of users' trials without *dot-dot pattern computation*.

Successful rate	4-dot pattern	9-dot pattern
Males	127/135 (94.1 %)	123/135 (91.1 %)
Females	83/90 (92.2 %)	80/90 (88.9 %)

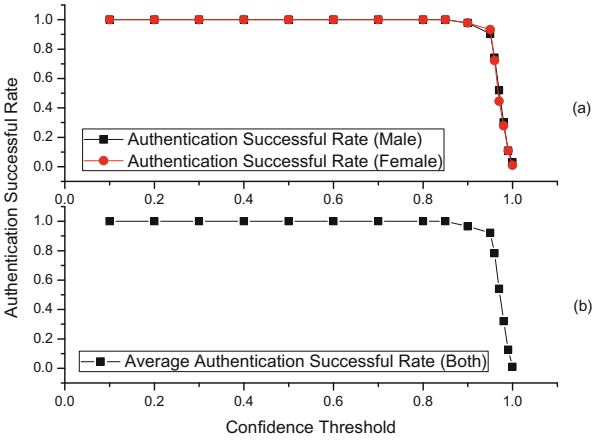


Fig. 8. Authentication results of users' trials with different *confidence thresholds*.

shows that the *confidence threshold* can make a crucial impact on user authentication. We have three major observations:

- On the whole, the authentication rate will be decreased through increasing the *confidence threshold*. When the *confidence threshold* reaches 1, which means conducting the user authentication without *proportional matching scheme* where users should exactly input their patterns, it is found that the authentication rate will be significantly reduced below 1 %. This observation demonstrates the importance of *proportional matching scheme* on improving the usability of *TMGuard*.
- In Fig. 8(b), we compute the average authentication successful rate for both 4-dot and 9-dot patterns. It is found that 0.9 is a turning point, where before this point, the authentication rate can be quickly increased to 1, while after this point, the authentication rate would have a quick drop. At this point, Table 3 presents that the successful authentication rate is about 98 %. Thus, we consider that it is an appropriate threshold in *TMGuard*.
- In addition, we find that there is no significant statistical difference between male and female participants. The collected data shows that gender information would not greatly affect the performance of *TMGuard*.

Result Analysis for Phase2. In this phase, we expect to simulate a real scenario on how users may use their phones. We have two collected datasets. (1) After an informal interview, we find that all users have input their selected patterns to unlock their phones 10 times at least and 33 times at most during the 2 days, and a total of 1856 trials were collected after analyzing the record. (2) In addition, since all users should input their patterns three times in our lab, we can further record 225 real trials in the lab. The *confidence threshold* is also set to 0.9. It means that there allows a 20 px/s deviation for a high speed at 200 px/s and a 10 px/s deviation for a low speed at 100 px/s.

For the first dataset, we present the successful authentication rate in Fig. 9. The figure shows that the successful authentication rate keeps increasing and becomes much stable after 4 trials. In addition, we show the DET curve regarding the false rejection rate (FRR) and false acceptance rate (FAR) with different *confidence thresholds* in Fig. 10, based on the recorded 1856 trials. The FAR and FAR are computed by authenticating all users trials against their templates under different thresholds. It is seen that when the *confidence threshold* is 0.9, a better FAR of 2.12 % and FRR of 2.23 % could be achieved.

Table 5. Authentication results of users' trials with *TMGuard* in *Phase2*.

Gender	Trials and successful rate
Males	135/135 (100 %)
Females	90/90 (100 %)

Similarly, for the second dataset collected in our lab, we compute the results of authentication attempts in Table 5, which shows a perfect authentication rate that all users can successfully input the patterns and unlock their phones. After interviewing with the participants, we found that many participants would pay attention to their touch behavior when inputting the patterns. They indicated that this may bring a little burden for them, but it is not a hard job for them to keep their behavior within the threshold. That is, users can adapt to a new mechanism when they pay attention to it and practice with several trials. This is the major reason for the perfect authentication results. It is worth noting that increasing user awareness is one of the important factors to improve the authentication security [5].

Based on the results in our study, we believe that setting the *confidence threshold* to 0.9 is appropriate without compromising the usability of inputting unlock patterns. These results also showed that the use of *dot-dot pattern computation* and *proportional matching* can encouragingly improve the usability of *TMGuard* in real applications.

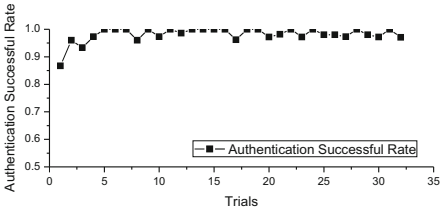


Fig. 9. Authentication results of users' performance with 1856 trials in *Phase2*.

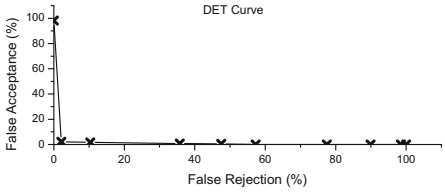


Fig. 10. DET curve shows how FRR and FAR vary when different confidence thresholds are used.

Discussions. In the literature, De Luca et al. [8] proposed an implicit approach to improve unlock patterns by extracting touchscreen data including pressure, size, X-coordinate, Y-coordinate and timings. They then conducted two studies and gave a conclusion: it is possible to distinguish users and improve the security of password patterns and screen unlocks by integrating behavioral biometrics. In their studies, the top user could reach an accuracy of 96 %, while the overall accuracy is 77 % for all users. Their work is the most referred and similar one to our work in the following aspects:

- Both research studies advocate that the security of unlock patterns should be improved by integrating an additional layer.
- Both research studies attempt to combine behavioral biometrics with Android unlock patterns.
- Both studies employ a non-machine-learning approach in the process of profile matching, where our work uses statistic-based method while De Luca et al. [8] use dynamic time warping (DTW).[3]

Although the main idea of these research studies are similar, it is not applicable to directly compare the results of these two articles. For example, the authentication accuracy in our work is above 97 % in average, but the results in [8] are much lower (i.e., 77 %). However, we should notice that the evaluation processes and research focuses are different. Those differences can be summarized as below:

- *Goals.* The main goal of [8] is to investigate the feasibility of applying behavioral biometrics to unlock patterns, while thanks to their conclusion, the goal of our work is to design a better mechanism to enhance the security of Android unlock patterns.
- *Schemes.* According to different goals, in [8], they did not propose a specific scheme to process the collected data while only apply dynamic time warping to the data. In contrast, our work first conducts a study to learn user behaviors during inputting Android unlock patterns and then designs a concrete mechanism based on touch movement.
- *Evaluation.* Obviously, the evaluation steps are different in these two studies. Moreover, behind the evaluation, the two articles have different views on user awareness. In [8], they would like to reduce users awareness in which users can perform not the same for a pattern input. In contrast, our work aims to remind users of their unlock inputs. Actually, user should increase their awareness during the authentication, since it is a basic requirement for behaviorial biometric authentication.
- *Algorithms.* It is impossible to say whose algorithm is better, since these two studies have different goals and focuses. It is understandable that both algorithms are performed well in their own scenarios. In addition, our work dose not aim to replace the existing algorithms, but provide alternatives for enhancing the security of unlock patterns.

[3] Dynamic time warping (DTW) is an algorithm for measuring similarity between two temporal sequences which may vary in time or speed.

Overall, [8] is a feasibility study that provides useful insights for combining behavioral biometrics with Android unlock patterns, and its results are positive and encouraging. Thanks to this, our work designs a more specific scheme in data processing and uses a statistic-based approach in profile matching. In practice, these two studies are complementary to each other. For example, our work does not include pressure and size, which can be considered in our future studies.

5 Conclusion and Future Work

In this paper, we develop a security mechanism, called *TMGuard*, attempting to enhance the authentication security of Android unlock patterns by combining it with behavioral biometrics. We totally conduct two studies in this work. In the first study, we find that users would perform touch movement differently when interacting with the touchscreen and that users would perform touch movement more stably for the same pattern after several trials. In the second user study, the experimental results and users' feedback demonstrate that *TMGuard* can promisingly improve the authentication security of Android unlock patterns without compromising its usability. Future work includes adding more features to our mechanism (i.e., from accelerometer and sensors [7,10]) and simulating advanced attacks. Our efforts aim to complement the existing solutions and to stimulate more research in this area.

Acknowledgments. We would like to thank all participants for their hard work and collaboration in the user studies such as data collection, and thank all anonymous reviewers for their helpful comments.

References

1. Andriotis, P., Tryfonas, T., Oikonomou, G., Yildiz, C.: A pilot study on the security of pattern screen-lock methods, soft side channel attacks. In: Proceedings of WiSec, pp. 1–6. ACM (2013)
2. Aviv, A.J., Gibson, K., Mossop, E., Blaze, M., Smith, J.M.: Smudge attacks on smartphone touch screens. In: Proceedings of the 4th USENIX Conference on Offensive Technologies, pp. 1–7. USENIX Association (2010)
3. Churchill , B.:Unlock Pattern Generator (2013). https://www.berkeleychurchill.com/software/android-pwgen/pwgen.php
4. Bergadano, F., Gunetti, D., Picardi, C.: User authentication through keystroke dynamics. ACM Trans. Inf. Syst. Secur. **5**(4), 367–397 (2002)
5. Bisson, D.: The state of security-Authentication and awareness: the anti-cybercrime duo, 30 October 2014. http://www.tripwire.com/state-of-security/security-awareness/authentication-and-awareness-the-anti-cybercrime-duo/
6. Brown, A.S., Bracken, E., Zoccoli, S., Douglas, K.: Generating and remembering passwords. Appl. Cogn. Psychol. **18**, 641–651 (2004)
7. Conti, M., Zachia-Zlatea, I., Crispo, B.: Mind how you answer me! (transparently authenticating the user of a smartphone when answering or placing a call). In: Proceedings of the 6th ASIACCS, pp. 249–259 (2011)

8. De Luca, A., Hang, A., Brudy, F., Lindner, C., Hussmann, H.: Touch me once and i know it's you!: implicit authentication based on touch screen patterns. In: Proceedings of CHI, pp. 987–996. ACM (2012)
9. Frank, M., Biedert, R., Ma, E., Martinovic, I., Song, D.: Touchalytics: on the applicability of touchscreen input as a behavioral biometric for continuous authentication. IEEE Trans. Inf. Forensics Secur. **8**(1), 136–148 (2013)
10. Giuffrida, C., Majdanik, K., Conti, M., Bos, H.: I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics. In: Dietrich, S. (ed.) DIMVA 2014. LNCS, vol. 8550, pp. 92–111. Springer, Heidelberg (2014)
11. IDC. Smartphone OS Market Share, Q2 2015, December 2015. http://www.idc.com/prodserv/smartphone-os-market-share.jsp
12. Karlson, A.K., Brush, A.B., Schechter, S. Can i borrow your phone?: understanding concerns when sharing mobile phones. In: Proceedings of the 27th CHI, pp. 1647–1650. ACM (2009)
13. Kotthoff, L., Gent, I.P., Miguel, I.: An evaluation of machine learning in algorithm selection for search problems. AI Commun. **25**(3), 257–270 (2012)
14. Li, L., Zhao, X., Xue, G.: Unobservable re-authentication for smartphones. In: Proceedings of the 20th Annual Network and Distributed System Security Symposium (NDSS), pp. 1–16 (2013)
15. Meng, Y., Wong, D.S., Schlegel, R., Kwok, L.: Touch gestures based biometric authentication scheme for touchscreen mobile phones. In: Kutyłowski, M., Yung, M. (eds.) INSCRYPT 2012. LNCS, vol. 7763, pp. 331–350. Springer, Heidelberg (2013)
16. Meng, W., Wong, D.S., Kwok, L.F.: The effect of adaptive mechanism on behavioural biometric based mobile phone authentication. Inf. Manag. Comput. Secur. **22**(2), 155–166 (2014)
17. Meng, W., Wong, D.S., Furnell, S., Zhou, J.: Surveying the development of biometric user authentication on mobile phones. IEEE Commun. Surv. Tutorials **17**(3), 1268–1293 (2015)
18. Nelson, D.L., Reed, V.S., Walling, J.R.: Pictorial superiority effect. J. Exp. Psychol.: Hum. Learn. Mem. **2**(5), 523–528 (1976)
19. Pereira Botelho, B.A., Nakamura, E.T., Uto, N.: Security analysis of touch inputted passwords. In: Lopez, J., Huang, X., Sandhu, R. (eds.) NSS 2013. LNCS, vol. 7873, pp. 714–720. Springer, Heidelberg (2013)
20. Tao, H., Adams, C.: Pass-go: a proposal to improve the usability of graphical passwords. Int. J. Netw. Secur. **7**(2), 273–292 (2008)
21. Van Thanh, D.: Security issues in mobile eCommerce. In: Proceedings of the 11th International Workshop on Database and Expert Systems Applications (DEXA), pp. 412–425 (2000)
22. SplashData Inc, Password unseated by "123456" on SplashData's annual Worst Passwords list (2013). http://splashdata.com/press/worstpasswords2013.htm
23. Uellenbeck, S., Dürmuth, M., Wolf, C., Holz, T.: Quantifying the security ofgraphical passwords: the case of Android unlock patterns. In: Proceedings of the 2013 ACM Conference on Computer and Communications Security (CCS), pp. 161–172 (2013)
24. Webroot. SURVEY: Mobile Threats are Real and Costly (2012). http://www.webroot.com/shared/pdf/byod-mobile-security-study.pdf
25. J. White. Cydia Tweak: How To Add An Android-Inspired Pattern Unlock Screen To The iPhone, 26 June 2013. http://appadvice.com/appnn/2013/06/cydia-tweak-how-to-add-an-android-inspired-pattern-unlock-screen-to-the-iphone

26. Yan, J., Blackwell, A., Anderson, R., Grant, A.: Password memorability and security: empirical results. IEEE Secur. Priv. **2**(5), 25–31 (2004)
27. Yan, Q., Han, J., Li, Y., Zhou, J., Deng, R.: Designing leakage-resilient password-dentry on touchscreen mobile devices. In: Proceedings of the 8th Asia CCS, pp. 37–48 (2013)
28. Zahid, S., Shahzad, M., Khayam, S.A., Farooq, M.: Identification, keystroke-based user on smart phones. In: Proceedings of RAID, pp. 224–243 (2009)
29. Zhang, Y., Xia, P., Luo, J., Ling, Z., Liu, B., Fu, X.: Fingerprint attack against touch-enabled devices. In: Proceedings of the 2nd ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, pp. 57–68 (2012)
30. Zhao, X., Feng, T., Shi, W., Kakadiaris, I.A.: Mobile user authentication using statistical touch dynamics images. IEEE Trans. Inf. Forensics Secur. **9**(11), 1780–1789 (2014)