ClickPattern: A Pattern Lock System Resilient to Smudge and Side-channel Attacks

Meriem Guerar[1]*, Alessio Merlo[2], Mauro Migliardi[1]
[1]*DEI - University of Padua, Padua, Italy*
guerar@dei.unipd.it, mauro.migliardi@unipd.it
[2]*DIBRIS - University of Genoa, Genoa, Italy*
alessio@dibris.unige.it

Abstract

Pattern lock is a very popular mechanism to secure authenticated access to mobile terminals; this is mainly due to its ease of use and the fact that muscle memory endows it with an extreme memorability. Nonetheless, pattern lock is also very vulnerable to smudge and side channels attacks, thus its actual level of security has been often considered insufficient. In this paper we describe a mechanism that enhances pattern lock security with resilience to smudge and side channel attacks, maintains a comparable level of memorability and provides ease of use that is still comparable with Pattern Lock while outperforming other schemes proposed in the literature. To prove our claim, we have performed a usability test with 51 volunteers and we have compared our results with the other schemes.

Keywords: Mobile Security, Authentication, Side Channel Attack, Smudge Attack.

1 Introduction

The possibility to tap into the wealth of information and services provided by the Internet has significantly changed our way of living and has become a deeply ingrained habit in the population of all the developed countries. This daily dependency on the accessibility of the Internet comes at a cost, in fact it makes us more and more susceptible to the malicious actions of hackers and requires an increased sensitivity to the problem of cybersecurity.

Recently, as a study from Statcounter shows (see Figure 1), the fruition of Internet services (be it pure information or more sophisticated services) has seen a steady evolutionary trend toward the use of mobile devices such as smartphones and tablet computers [1]. For this reason, when dealing with cyber-threats it is necessary to take into full account the peculiarities of each platform in general and, as it is quickly becoming the platform of choice, of mobile devices in particular. The first and fundamental step to g security is authentication: a secure identification of the user requesting an action, in fact, is a pre-requisite to decide if the requested action is to be allowed or forbidden (access-control); furthermore, the capability to allow access to resource only to the correct parties is a pre-requisite to guarantee resource integrity. Hence, authentication is the basis of all security. Any authentication mechanism, however, is not only required to be resilient to hacking, but it must also provide features such as ease of use, otherwise it turns into a nuisance that users actively struggle to deactivate. As the most commonly adopted authentication scheme is the one based on something you know, one of the more important features in terms of usability is memorability, i.e., the ease with which as user is capable of remembering what she needs to provide to the system to prove her identity. This fact has given to the pattern lock

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), 8:2 (June 2017), pp. 64-78
*Corresponding author: Dr. Meriem Guerar, Dipartimento di Ingegneria dell'Informazione (DEI), Room 419, Via Gradenigo 6g, 35131, Padova. Tel: +39-049 827 7955

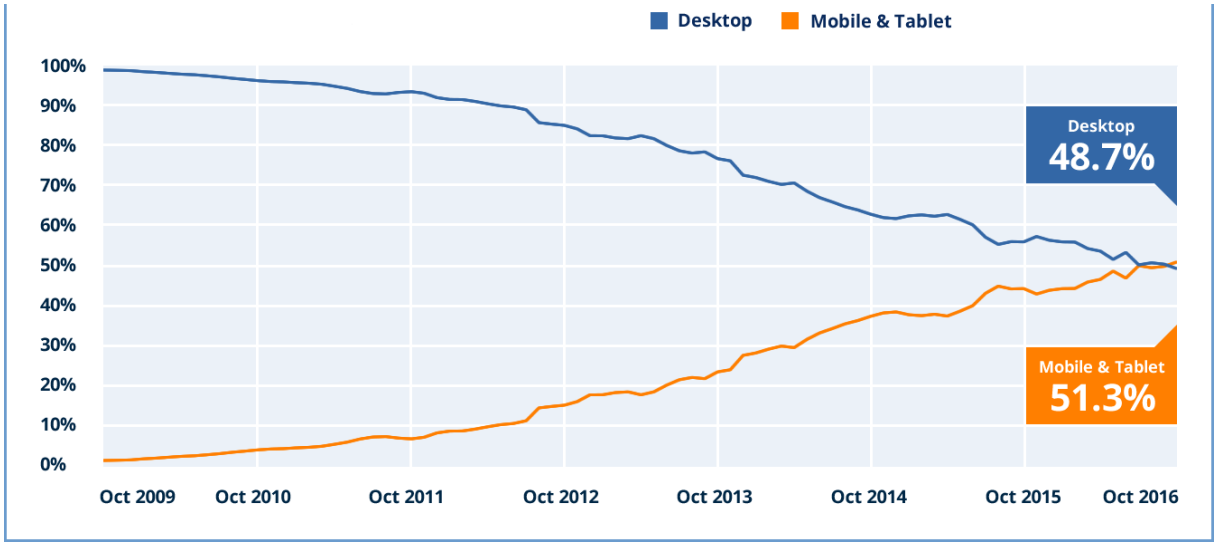

Figure 1: Evolutionary trend of the platform adopted for Internet fruition. Source: *Statcounter.*

mechanism a very large degree of adoption in mobile devices; in fact, it leverages muscle memory to limit the cognitive burden that it pushes users. Nonetheless, in its normal implementation the pattern lock mechanism has proved to be vulnerable to smudge attack [2] and side channel attack [3], a fact that greatly diminishes its appeal as the authentication mechanism of choice for many users.

In this paper we present ClickPattern, an improvement of traditional pattern lock mechanism that makes it resilient both to smudge and to side-channel attacks while keeping a high level of usability.

This paper is structured as follows: in Section 2 we provide a description of the state of the art and we analyze the related work; in Section 3 we present our mechanism to enhance pattern-lock authentication and make it resilient to smudge and side-channel attacks; in Section 4 we perform a security analysis of the described mechanism; in Section 5 we perform a usability test and we describe our experimental findings. Finally, in Section 6, we provide some concluding remarks.

2 Related Work

Graphical passwords are one of the most promising authentication methods that aim at overcoming the usability and memorability issues that affects text-based passwords. This mechanism is based on psychology studies which support the thesis that the human brain recognizes and remembers images much better than a string of characters [4]. Graphical passwords are commonly classified into three main categories; in the following sub-section we will describe this taxonomy.

2.1 Recall-based Graphical Passwords

These systems are based on using some form of recalling and then drawing a secret. One of the first recall-based graphical password system proposed as an alternative to traditional passwords was “Draw a Secret” (DAS) by Jermyn and al [5]. In this scheme, the user draws a secret symbol on an N x N grid. It is not required that users input the exact shape. Instead they have to pass through coordinates of the grid cells while drawing. This system boasts a large password space but suffers from some security weakness related to the kind of graphical passwords that users would draw and how this increases the guessability of a chosen shape [6]. In addition, a study by Thorpe et al. [7] suggests that users tend

to choose symmetric passwords because they are easy to recall, whereas this kind of password reduces the *dictionary* size which facilitates dictionary attacks. On a similar scheme, Tao and Adams [8] were inspired by an old Chinese game "Go" and proposed a new graphical password scheme named Pass-Go. Users input their password using grid intersection points, instead of grid cells. The authors suggest using a finer grid and add different pen colors as an additional parameter to further increase the theoretical password space. A special case of this scheme that has been widely deployed commercially by Google is the Android unlock pattern. However, when users swipe their fingers on a touch-enabled screen to form a pattern for unlocking the device, they leave behind oily residues or smudges on the touch screen surface itself, thus they make the system susceptible to the so-called smudge attack [9].

Nonetheless, the high level of both usability and memorability of Android unlock pattern has led researchers to suggest new approaches and modification aimed at improving its security level. Most of these studies focus on smudge attack.

Schneegass et al.[10] presented a system called SmudgeSafe. To authenticate, the user has to draw a shape between specific locations in the image known only to the user. As the system uses random geometric image transformations for each authentication session, smudge traces from a previous login will not match the current password image, which renders identification of password difficult or even impossible. In WhisperCore [2], authors suggested to add an additional task to the basic authentication. At the end of login process, the user is required to wipe parts of the screen to mask the smudge of his actual entered pattern. Kwon and Na [11] introduced a system called TinyLock which is similar to the previous scheme. In TinyLock, the user is required to draw the secret pattern on the tiny grid (i.e., 3x3 grid with small size). Then, the grid will be replaced by a virtual wheel with the same size which helps users to wipe their smudges on the screen. In [12], authors proposed three graphical password schemes. The most promising approach is the Marbles method, where the password is composed by a sequence of marbles (i.e., colors). In Marbles, a circle of 10 colored marbles are presented to the user. To authenticate, the user is required to drag the right sequence to the center. Upon each new attempt, the marbles are randomly arranged and thus, a different pattern will be drawn by the user, which prevents smudge attack. However, this method increases the login time by a factor of five over the Android grid unlocks. Furthermore, remembering colors is a difficult task for many users and fully impossible to the color blind ones. De Luca et al. [13] add an implicit authentication layer to pattern based authentication mechanism on Android phones. Instead of existing systems, access is not granted by just drawing the right pattern but also by the way the input is entered. This way, even if attackers are in possession of the device and know the right pattern, they will not necessarily have the access to the device. However, this approach has not been deployed into any actual system; furthermore, it has high false negative rate, which means there is a high chance that the legitimate owner of a mobile device would be mistakenly regarded as an attacker leading to delay in the ability of users to access the device in the best scenario and complete lock out in the worst one, where false negatives happen in a row.

Recently, Kim et al. [14] proposed an authentication scheme which is resilient against different kind of attacks. The approach they propose has its foundations in three elements: arrows in the same direction, the omission of authentication values and errors artificially introduced. The user memorizes the password's location in a 5 x 7 grid. During the login process, the user selects cells according to the arrows displayed in each password's cell but the starting cell position changes randomly each time. This method is secure against brute force attacks, smudge attacks, side-channel attacks and spyware-based recording attacks. However, while including errors is a key mechanism to increase the security level achieved, the users' study shows that it significantly decreases the level of usability of the proposed scheme.

Beside using graphical passwords, Guerar el al. [15] proposed a sensor based CAPPCHA to enhance the security of PIN code against a different kind of spyware attacks including side channel attack. To solve the challenge, the user needs to perform a tilting of the device to match a specific degree that is

displayed on the screen and hold it still in this position for one second. The successful completion of this operation grants access to the PIN pad. In [16], a usability study of CAPPCHA on smartphones shows very promising results.

2.2 Click-based Graphical Passwords

These systems are based on one of the most ancient mnemonic system, the Loci system [17]. The Loci system consists of memorizing a series of mental images of familiar locations in some natural order, then associate objects with these locations, in order to be easily remembered. The most known system in this category is PassPoints by Wiedenbeck et al. [18]. In PassPoints a user has to input password by clicking on predefined points of an image. Each click must be in the correct order and accurate within an acceptable tolerance of the original point. The main advantages of this scheme are the short login time and a large password space provided with only a small number of clicks. However, researchers highlighted how *hotspots*, i.e., parts of the image that attract the attention of the user, can be exploited to launch efficient dictionary attacks against this system [19, 20, 21]. In addition to security problems, Chiasson et al. [19] found that the need to select images significantly impacts the usability. In order to counter these problems, they proposed Cued Click-Points (CCP) [22] as an alternative to PassPoints. During login, the user has to recursively select one click-point on each of five distinct images displayed in sequence. The position of a click point on the current image determines the subsequent image. Thus, the legitimate users know that they selected a wrong click-point if an unexpected image appears. Similarly to what happens in most graphical passwords schemes, this technique is susceptible to shoulder surfing and malware attacks [22]. Recently, Ritter el al. [23] designed a multi-touch image-based authentication method for smartphones called MIBA. The login process is similar to CCP, with the difference that MIBA uses multiple fingers to enable simultaneous selection of up to four click points on an image per round instead of one click point, which improves resilience to shoulder surfing attacks and reduces authentication time.

2.2.1 Recognition-based Graphical Passwords

Scheme in this category require that users recognize their pre-selected images amongst a set of distractor images. The most prominent system is Passfaces [24]. Users authenticate by identifying previously selected faces from a grid of nine faces for each round. Comparative studies conducted by Brostoff and Sasse [25] showed that PassFaces is more memorable than text based passwords. However, users tend to choose faces of people that they find attractive; furthermore, they are prone to cultural biases and tend to select faces from same gender and race. These two factors greatly decrease the password space and thus the global security of this system [26]. In addition, a recent survey [27] pointed out that malware would need both user's touch coordinates and screen recording to acquire the password which means that while this system is secure against side-channel attacks, it is vulnerable to spyware-based recording attacks. Takada and Koike [28] proposed an image-based authentication for mobile devices that uses the user's favorite images. During authentication, the user has to select a pass-image for each round among decoy images or choose nothing if no pass-image is displayed. Even though using the favorite images increases memorability, it makes authentication predictable and thus less secure. Moreover, the system is vulnerable to replay attacks.

In order to highlight the main unresolved issues in existing graphical password schemes, we summarize the problems of some prominent schemes in Table 1. This table shows that none of the reviewed schemes provides adequate protection against smudge and side channel attacks while guaranteeing a reasonable authentication time and/or error rate.

Table 1: weakness of some existing graphical password schemes

Techniques	Schemes Approach	Weakness/Drawbacks
Recall-based Graphical Passwords	Android Pattern lock	It is vulnerable to side-channel, smudge and shoulder surfing attacks.
	DAS [5]	It is vulnerable to side-channel, smudge,dictionary and shoulder surfing attacks.
	Pass-go [8]	It is vulnerable to side channel and shoulder surfing attacks.
	Kim et al. [14]	High error rates (i.e 18 %).
	TinyLock [11]	It is vulnerable to side channel and shoulder surfing attacks.
Click-based Graphical Passwords	PassPoint [18]	It is vulnerable to side channel, brute force and shoulder surfing attacks
	CCP [22]	It is vulnerable to side channel and shoulder surfing attacks.
	MIBA [23]	It is vulnerable to side channel attacks.
Recognition-based Graphical Passwords	Passfaces [24]	It is vulnerable to brute force, dictionary, guess, shoulder surfing and spyware-based recording attacks.
	Takada and Koike [28]	It is vulnerable to brute force, guess and shoulder surfing attacks.

3 The *ClickPattern* concept

Comparative studies conducted by Von Zezschwitz et al. [29] (see Table 2) showed that PIN based authentication systems are better than pattern lock authentication systems in terms of input speed and error rates. However, pattern lock authentication systems were rated better in terms of ease-of-use, feedback, efficiency and memorability. For these reasons, in this paper, we introduce ClickPattern, an authentication method that enhances the security of pattern lock against both smudge attack and side channel attack.

To better explain how our proposed scheme works, we explain it with a stepwise description of how it differentiates from the pattern lock scheme. The first prototype of ClickPattern is very simple and similar to Android pattern lock, in fact, the only difference resides in how the user draws the pattern. To unlock the device, the user has to click directly on the sequence of dots that constitutes the nodes of the correct pattern instead of swiping his finger along the edges. This scheme is resilient to the smudge attack as the smudge left on the screen is very hard to distinguish from smudges caused by any other touchscreen

activity (use of any app) and, at worst, will just reveal the used dots but not the actual pattern. However, despite this prototype is simple and enhances the security of Pattern lock against smudge attacks, it still vulnerable to side channel attacks. In order to cope with this issue, we introduce an additional tweak to the user interface design of the pattern lock system. As illustrated in Figure 2, the final version of ClickPattern is composed of the three components on the touchscreen:

- **Grid 3 x 3**. Similar to Pattern lock systems, ClickPattern uses nine dots arranged in a 3 x3 grid, typically on a touchscreen. However, we label the dots in the 3 x 3 grid from 1 to 9. Thus, a pattern can be represented as an ordered sequence of connected dots. For example, the pattern shown in Figure 3 can be described as $1 \rightarrow 2 \rightarrow 6 \rightarrow 8$.
- **Numbered table**. Numbered table from 1 to 9 displayed in the bottom of the grid. The numbers are arranged randomly for each authentication attempts.
- **Ok button** is a button to validate the pattern.

Figure 2: The ClickPattern UI

Figure 3: Pattern of sequence |1268|

Authentication process. The user instead of clicking directly on the sequence of dots to construct the correct pattern (as we described previously, in the first prototype of ClickPattern), she clicks on their corresponding numbers in the table displayed at the bottom of the grid. When the user completes the selection, she clicks on the *Ok* button to validate the pattern. The undo button can be used to remove an

Table 2: Best rated security systems according to some usability categories

Evaluation Category	**Best Rated System**
Input Speed	PIN based
Error rate	PIN based
Ease of use	Pattern-lock based
Feedback	Pattern-lock based
Efficiency	Pattern-lock based
Memorability	Pattern-lock based

erroneous click and a consequently wrong edge. The authentication is successful if the pattern resulted from clicking numbers on the table is identical to the pattern originally defined by the user. The user must respect the following rules in order to create a valid pattern:

1. A pattern must contain at least 4 points.

2. No point can be used twice.

3. A pattern will always connect any unconnected dot along its path to reach the user indicated destination. As an example, consider the case of the z-like pattern shown in Figure 4; it must always contain points 2 and 5 even though the sequence obtained when a user does not click on either dot 2 or 5, more precisely, the sequence | 1378 |, generates the exact same pattern. In fact both un-clicked dots would be automatically selected (i.e., | 123578 |). Hence, patterns are actually patterns and not numeric sequences, in fact the numeric sequences | 12357 |, | 1357 |, | 137 |, | 1237 | all correspond to the same pattern and the secret is the pattern and not the numeric sequence. The numeric sequence is merely a tool to get to the pattern and a user has the choice to click on the intermediate point or not (i.e., the user may decide to click on 1→2→3 or directly on 1→3 and let dot number 2 be selected automatically).

4. A pattern can go through a previously connected dot along its path in order to connect an unconnected dot. As an example for rule 4, consider the pattern shown in Figure 5; it is constructed by the sequence | 4537 |; in this case the intermediate point 5 is not automatically connected when the user selects points | 37 | because point 5 had been already connected.

5. Just as in Pattern Lock, patterns are directed. The order in which dots are selected is part of the secret and authentication with the same shape built in reverse order fails.

Discussion: Adopting this mechanism, contrarily to what happens in the case of the Android pattern lock, it is not possible to have the shape entered by the user directly from the smudge residue left on the screen. Furthermore, as the numbered table is randomized at each authentication attempt recording the user's touch coordinate does not allow pattern reconstruction either. An attacker can obtain only the position of numbers in the table that has been used to construct the pattern. However, he cannot identify which numbers belong to these positions due to the randomization of numbers position on the table at each login attempts. This way, ClickPattern is secure against smudge attack and side channel attacks. The same idea can be implemented using the color instead of number as shown in Figure 6. However, as we stated in the related work section, colors are less memorable than numbers for many users and authentication build on colors would discriminate color blind users.

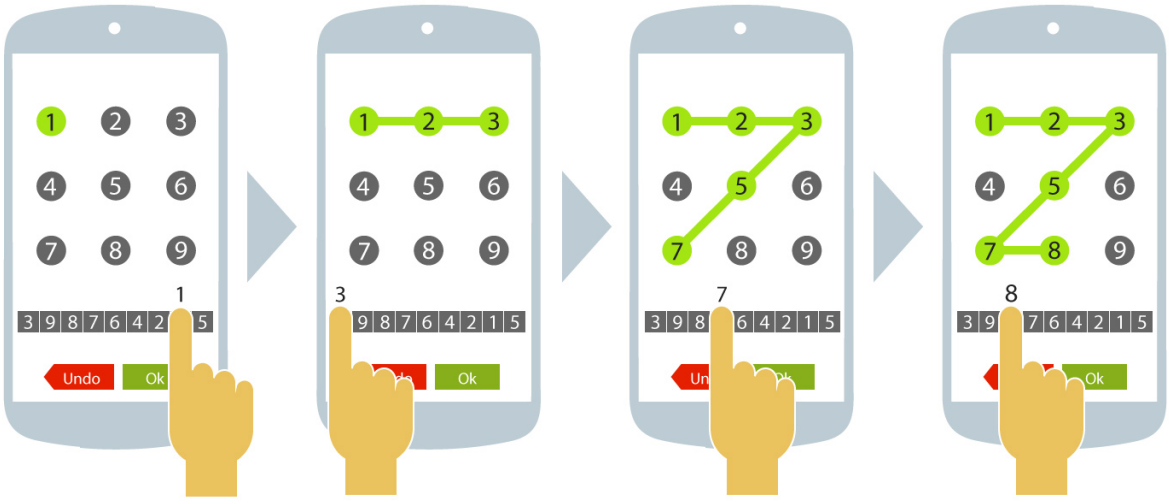

Figure 4: An example of authentication with ClickPattern using | 1378 | sequence

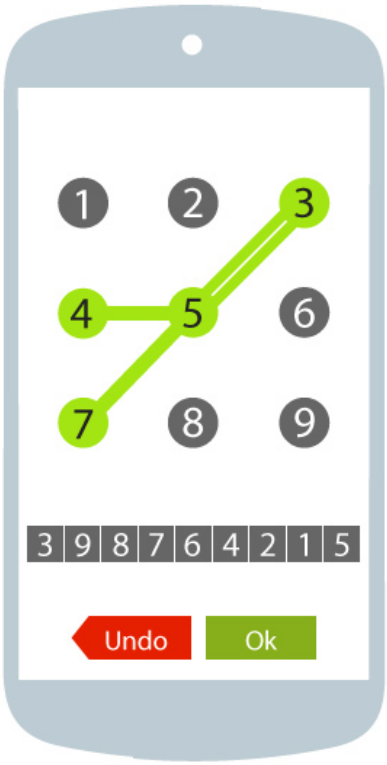

Figure 5: An example of authentication with ClickPattern using | 4537 | sequence

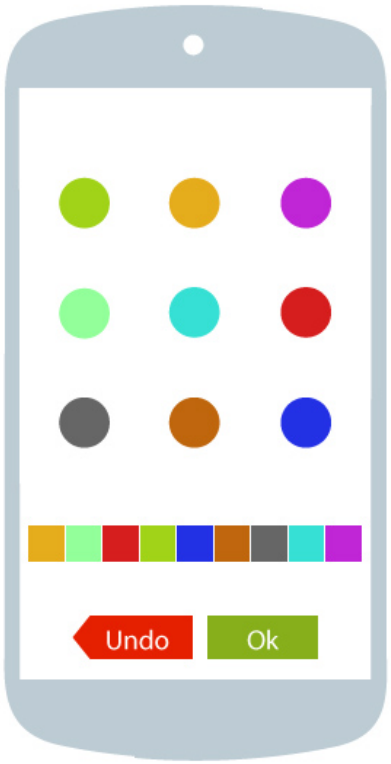

Figure 6: Color ClickPattern prototype

4 Security Analysis

In this section, the security of the ClickPattern is analyzed in relation to brute force attack, smudge attack, side-channel attacks and shoulder surfing attacks.

Brute Force Attack. A Brute Force Attack is a password cracking method that uses an automated process to try all possible character combinations until the password is found. The number of possible password combinations of the proposed scheme is influenced by the grid size and the password length. As the grid size of the implemented prototype is 3 x 3, we wrote a brute force program to generate all possible patterns that follow the rules described in Section 3. We found that there are 389,112 valid patterns in total, which is similar to the possible password combination of Android Pattern Lock [9]. The statistics of the sizes of all valid patterns are as shown in the Table 3.

Table 3: Valid patterns for ClickPattern

Pattern length (dots)	4	5	6	7	8	9
Number of valid patterns	1624	7152	26016	72912	140704	140704

The number of possible password combination of Click pattern is about 39 times more than the commonly used 4-digit PIN (i.e., 10000).

Smudge Attack. The Smudge Attack is aimed at finding the password by using smudges left by the user's finger on the touch screen surface. As in the commonly used Android pattern lock, the user swipe his finger on the touch screen to draw the correct pattern which leaves behind oily residues or smudges that can easily determine the user's input path [2, 9]. In ClickPattern, the information that could be obtained from the smudge attack is the position of numbers on the table that has been used to construct the pattern. However, as these positions are randomly arranged for each authentication attempts, it is not possible to know which numbers belong to the smudges. Thus, the password cannot be inferred by the smudges left on the touch screen and cannot be replayed in the next login attempts which make ClickPattern resilient against smudge attacks.

On the other hand, the chance that a brute force attack or guessing attack succeeds depends on the size of the password space. Thus, some attackers exploit smudge attack to reduce the size of the password space to perform brute force or guessing attacks. For example, guessing the right PIN in a list of 10000 possibilities is difficult. However, through smudge attack, an attacker could reveal the digits typed by the user and has only to guess the order of these four digits. So, she reduced the size of password space from 10000 to 24.

ClickPattern has two elements for protection against this kind of attack. The first is the randomization of numbers position on the table at each login attempt. The second is the possibility to automatically select the intermediate dots by the system which will not leave smudge on the screen. As a consequence, the length of the numbers sequence obtained from the smudges is not necessarily the length of the correct pattern. For example, the size of **Z** shape pattern shown in the Figure 3 is 7 (i.e., the number of dots the pattern connects). To draw this shape the user has the choice to click on one of the following sequences $|1235789|$, $|135789|$, $|13789|$ or $|1379|$.... If we assume that the user choose the simple sequence and click on $1 \rightarrow 3 \rightarrow 7 \rightarrow 9$, the intermediate dots 2, 5 and 8 will be selected automatically to have the right sequence (i.e., $|1235789|$). Thus, the smudges left by the user fool the attacker and could let him think that the size of the pattern equal to 4. Thus, an attacker will not succeed to have the right pattern between 1624 possible pattern with 4 dots. In this case, the attack should perform an extended scan using the whole password space (i.e., 389,112).

Side channel Attack. Most of current smartphones use ARM's TrustZone technology in order to provide stronger isolation for sensitive applications (e.g. payment, banking, corporate emails, etc.). This technology ensures that when users interact with the trusted OS, the malware in the Android OS cannot access the screen [30]. However, a new kind of spyware that emerged recently is capable of exploiting resources that are shared between the mobile OS and the trusted OS (e.g., the accelerometer [31], the camera and the microphone [30], the Gyroscope [32, 33]) to steal the user's keystrokes.

To provide adequate protection against this kind of attack, ClickPattern uses indirect input to draw the pattern. Instead of swiping the finger to connect between dots that construct the pattern on the grid, the user has to click on the number that identifies each dot on the table. In addition to that, ClickPattern randomizes the position of these numbers on the table for each authentication attempt, making the actual user's input different every time. Thus recording the user's touch coordinates is useless.

Shoulder Surfing Attack. Shoulder surfing attacks are aimed at stealing sensitive personal information by looking over someone's shoulder. Similar to Android pattern lock, ClickPattern has two modes of drawing operations, the normal mode and the stealth mode. In the normal mode, the feedback of swipes drawn by the user is visually shown on the screen. On the contrary, the stealth mode hides them in order to mitigate the shoulder surfing attack.

Obviously, when a pattern connects more dots, it gets longer and acquires more intersections and overlapping sections; this makes the pattern more complex from the visual point of view and thus enhances its security against shoulder surfing attack. The protection derives directly from the fact that the attacker's ability to identify and remember the correct password is limited and cannot cope with long and complex patterns.

ClickPattern introduces an additional *hidden complexity* to the pattern by providing to the user the possibility to hold some dots that construct the pattern by holding their identifier on the table (i.e., instead of simple click) without any visual feedback. In addition to this, it also provides the possibility to automatically select the intermediate unconnected dots. Finally, the authentication time of Click pattern is short and in general the attacker is capable to focus only on the final shape not on the actual user input, thus our mechanism mitigates the threat posed by shoulder surfing attacks consistently.

5 Experimental results

Designing a new authentication mechanism requires taking into account the following requirements: memorability, usability and security. Thus, in addition to the security factor, the secret should be easily memorized, take a short time to input to the device, and have a low error rate. To this end, in this section, we evaluate the usability of the proposed scheme and we compare it to the most commonly used authentication schemes, namely Android lock pattern and 4-digit PIN. The times and error rates of the Android lock pattern authentication scheme have been taken from the literature, thus there might be some procedural differences in the way they were actually calculated. However, even if the complete procedural equivalence is not guaranteed, they provide a significant baseline for our comparison. The development tools used for the implementations were Android Studio, android SDK 25.2.5 and Java 8.

5.1 Participants

We recruited 51 volunteers among students at the University of Genoa. The age of the participants to the study ranges from 23 to 26, among the participants 9 are females, while 42 are males. Some participants preferred to use their own smartphone, while many others used the smartphones that we provided for the test.

5.2 Procedure

At the beginning, we showed to the participants a short instructional video, then the prototype was explained in detail to each participant. They were encouraged to train until they felt familiar with it. When the participants felt ready, they were asked to define a simple and complex password patterns. Afterwards, they were required to repeat the authentication process ten times for each predefined pattern. Thus, the results are based on 1020 authentication sessions performed by 51 participants. Finally, the same volunteers were invited to perform 10 authentication attempts with the four-PIN digits authentication scheme.

5.3 Performance Comparison

The ClickPattern authentication time was measured for each successful unlocking task from the first key press to releasing the OK button (to confirm the pattern). The PIN authentication time was measured from the first key press to the completion of PIN entry. The logged data stored on the smartphones were used to calculate the average authentication time and the error rate. The Figure 7 illustrates the graphical results of the entry time (for successful authentication) for ClickPattern, Android pattern lock and the PIN method. Table 4 summarizes the numerical results.

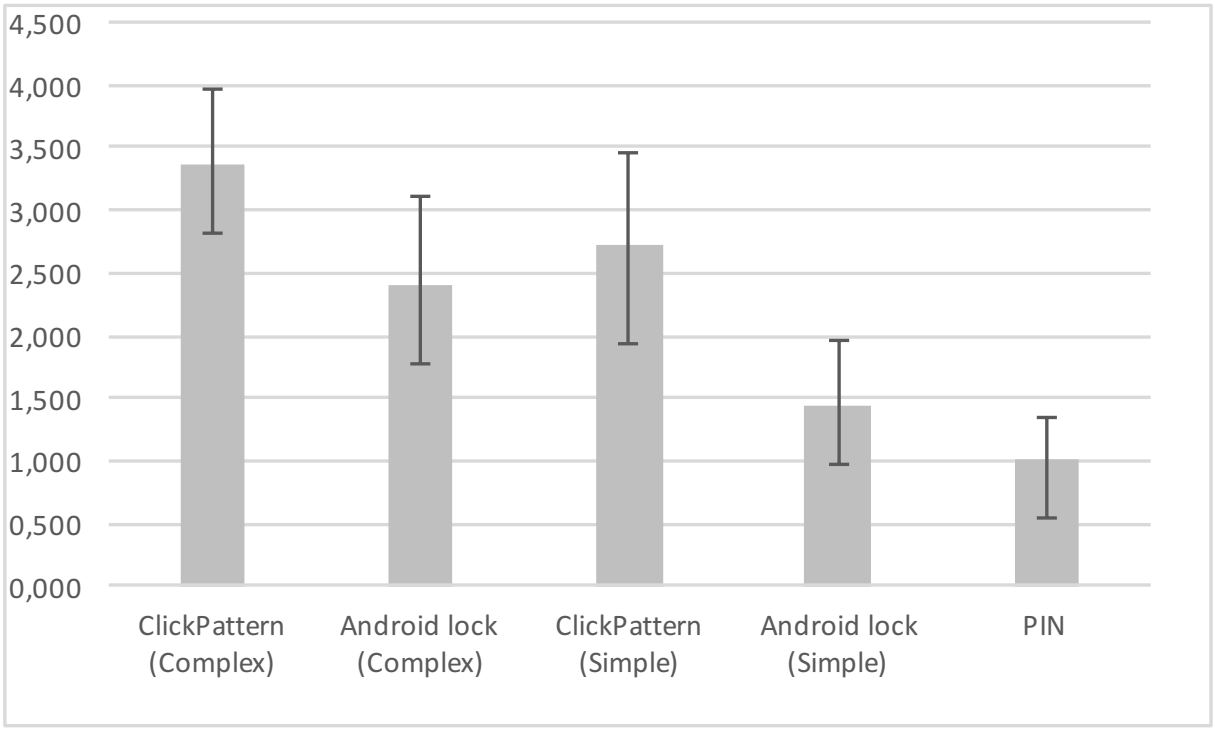

Figure 7: Entry time of successful authentication for ClickPattern, Android pattern lock and the four-PIN digits.

The experimental results show that the fastest method is the 4-digit PIN (median: 1.015 s; average: 1.011 s; sd: 0.183 s) followed by Android lock with simple pattern (median: 1.336 s; average: 1.431 s; sd: 0.286 s), Android lock with complex pattern (median: 2.313 s; average: 2.392 s; sd: 0.420s), ClickPattern with simple pattern (median: 2.785 s; average: 2.728 s; sd: 0.356 s), ClickPattern with Complex pattern (median: 3.352 s; average: 3.358 s; sd: 0.220 s).

Table 4: Numerical results of the entry time

Authentication methods	**Median [s]**	**Average [s]**	**Min [s]**	**Max [s]**	**StDev [s]**
4-digit PIN	1.015	1.011	0.534	1.332	0.183
ClickPattern with simple pattern	2.785	2.728	1.924	3.457	0.356
Android lock with simple pattern [11]	1.336	1.431	0.941	1.964	0.286
ClickPattern with complex pattern	3.352	3.358	2.820	3.953	0.220
Android lock with complex pattern [11]	2.313	2.392	1.781	3.112	0.420

Hence, we notice a trade-off between enhanced security and speed. In fact, ClickPattern provides enhanced security compared to both the 4-digit PIN and Android lock pattern schemes but introduces an additional cost in terms of input time. However, the goal of ClickPattern is to be more secure while avoiding the introduction of unacceptably long times. Furthermore, while remaining just slightly less convenient than the PIN and Android lock pattern, ClickPattern keeps the same level of memorability boasted by Android lock Pattern.

As we stated before, the numbers provided in these table and figure summarize the result of an experimental campaign comprising 1020 authentication sessions. We now compare the performance of our scheme with some other schemes proposed in the literature. Table 5 shows this comparison for the average authentication time and error rate. It is possible to notice that ClickPattern has the fastest

Table 5: User test results for ClickPattern and the existing schemes.

Authentication methods	**Authentication time [s]**	**Error rate [%]**
DAS [14]	9.87	12
Passfaces [14]	14.55	14
Kim et al. [14]	11.47	18
TinyLock with simple pattern [11]	2.10	0
TinyLock with complex pattern [11]	3.15	0
ClickPattern with simple pattern	2.73	0
ClickPattern with complex pattern	3.36	0

authentication time and the lower error rates among all the evaluated schemes with the single exception of TinyLock. However, while the difference in performance is barely significant, ClickPattern is more secure as TinyLock does not provide security against side channel attacks. The error rate of ClickPattern equals to zero, this might be due to two facts. First, the dots identifiers in the table, displayed in the bottom of the grid, are arranged randomly for each authentication attempts, forcing the user to pay more attention when he clicks on these identifiers to draw the correct pattern. Second, ClickPattern allows the user to backtrack one step in case of mistakes through the undo button.

6 Conclusion and Future Work

Recently, our dependency on data and services available through the Internet has steadily grown. At the same time, the number of users tapping into the Internet by means of mobile devices has overcome the number of desktop based Internet users. The combination of these makes security for mobile devices paramount. Authentication is the cornerstone of any security system and in mobile devices the Pattern Lock mechanism is very popular for its ease of use and its high level of memorability.

However, the Pattern Lock mechanism is vulnerable to both smudge attacks and to side-channel attacks, thus it represents an insufficient security measure for any user that performs critical operations

(e.g., mobile payments) on his mobile device.

To overcome these vulnerabilities we have proposed an enhanced version of the Pattern Lock mechanism, namely ClickPattern, that adopts a separated taps approach to recreate the authentication pattern.

Our security analysis shows that ClickPattern is resilient to both smudge attacks and side-channel attacks as well as to traditional brute force attacks; besides, our solution is also more difficult to crack with shoulder surfing attacks than normal Pattern Lock.

To prove the efficacy of our proposal, we performed a usability test with 51 volunteers. Our results show that, even if there is a trade-off between additional security and speed of execution when compared to traditional schemes such as 4-digit PIN and Android Pattern Lock, our scheme maintains both a high level of memorability and a very good ease of use. Besides, when compared to other schemes proposed in the literature, our scheme guarantees the best security level, one of the shortest execution time and the lowest error rate. In future work we will evaluate the possibility to port our scheme to completely different devices such as ATM machines.

References

[1] A. Lella and A. Lipsman, "The u.s. mobile app report," 2014, http://www.comscore.com/Insights/Presentations-and-Whitepapers/2014/The-US-Mobile-App-Report, [Online; Accessed on June 1, 2017].

[2] K. Airowaily and M. Alrubaian, "Oily residuals security threat on smart phones," in *Proc. of the 1st International Conference on Robot, Vision and Signal Processing (RVSP'11), Kaohsiung, Taiwan*. IEEE, November 2011, pp. 300–302.

[3] A. Nahapetian, "Side-channel attacks on mobile and wearable systems," in *Proc. of the 13th IEEE Annual Consumer Communications Networking Conference (CCNC'16), Las Vegas, Nevada, USA*. IEEE, January 2016, pp. 243–247.

[4] F. F. Patrick Elftmann, C. Bischof, and M. Mink, "Diploma thesis secure alternatives to password-based authentication mechanisms," 2006.

[5] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin, "The design and analysis of graphical passwords," in *Proc. of the 8th Conference on USENIX Security Symposium (SSYM'99), Washington, D.C., USA*. USENIX Association, August 1999.

[6] D. Nali and J. Thorpe, "Analyzing user choice in graphical passwords," May 2004.

[7] J. Thorpe and P. C. van Oorschot, "Graphical dictionaries and the memorable space of graphical passwords," in *Proc. of the 13th Conference on USENIX Security Symposium (SSYM'04), San Diego, California, USA*, vol. 13. USENIX Association, August 2004, pp. 10–10.

[8] H. Tao and C. Adams, "Pass-go: A proposal to improve the usability of graphical passwords," 2006.

[9] A. J. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. M. Smith, "Smudge attacks on smartphone touch screens," in *Proc. of the 4th USENIX Conference on Offensive Technologies (WOOT'10). Washington, D.C., USA*. USENIX Association, August 2010.

[10] S. Schneegass, F. Steimle, A. Bulling, F. Alt, and A. Schmidt, "Smudgesafe: Geometric image transformations for smudge-resistant user authentication," in *Proc. of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'14), Seattle, Washington, USA*. ACM, September 2014, pp. 775–786.

[11] T. Kwon and S. Na, "Tinylock: Affordable defense against smudge attacks on smartphone pattern lock systems," *Computers & Security*, vol. 42, no. 1, pp. 137–150, May 2014.

[12] E. von Zezschwitz, A. Koslow, A. De Luca, and H. Hussmann, "Making graphic-based authentication secure against smudge attacks," in *Proc. of the 2013 International Conference on Intelligent User Interfaces (IUI'13), Santa Monica, California, USA*. ACM, March 2013, pp. 277–286.

[13] A. De Luca, A. Hang, F. Brudy, C. Lindner, and H. Hussmann, "Touch me once and i know it's you!: Implicit authentication based on touch screen patterns," in *Proc. of the SIGCHI Conference on Human Factors in Computing Systems (CHI'12), Austin, Texas, USA*. ACM, May 2012, pp. 987–996.

[14] T. Kim, J. H. Yi, and C. Seo, "Spyware resistant smartphone user authentication scheme," *International Jounal of Distributed Sensor Networks*, vol. 2014, no. 1, March 2014.

[15] M. Guerar, M. Migliardi, A. Merlo, M. Benmohammed, and B. Messabih, "A completely automatic public physical test to tell computers and humans apart: A way to enhance authentication schemes in mobile devices," in *Proc. of the 2015 International Conference on High Performance Computing Simulation (HPCS'15), Amsterdam, Netherlands*. IEEE, July 2015, pp. 203–210.

[16] M. Guerar, A. Merlo, and M. Migliardi, "Completely automated public physical test to tell computers and humans apart: A usability study on mobile devices," *Future Generation Computer Systems*, March 2017.

[17] K. Higbee, *Your Memory: How it Works and how to Improve it*. Marlowe & Company, March 2001.

[18] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, "Passpoints: Design and longitudinal evaluation of a graphical password system," *International Journal of Human-Computer Studies*, vol. 63, no. 1-2, pp. 102–127, July 2005.

[19] S. Chiasson, R. Biddle, and P. C. van Oorschot, "A second look at the usability of click-based graphical passwords," in *Proc. of the 3rd Symposium on Usable Privacy and Security (SOUPS'07), Pittsburgh, Pennsylvania, USA*. ACM, July 2007, pp. 1–12.

[20] J. Thorpe and P. C. van Oorschot, "Human-seeded attacks and exploiting hot-spots in graphical passwords," in *Proc. of 16th USENIX Security Symposium on USENIX Security Symposium (SS'07), Boston, Massachusetts, USA*. USENIX Association, August 2007, pp. 8:1–8:16.

[21] P. C. V. Oorschot and J. Thorpe, "On predicting and exploiting hotspots in click-based graphical passwords," School of Computer Science, Carleton University, Tech. Rep., 2010.

[22] S. Chiasson, P. C. van Oorschot, and R. Biddle, *Computer Security - ESORICS 2007*. Springer Berlin Heidelberg, September 2007, pp. 359–374.

[23] D. Ritter, F. Schaub, M. Walch, and M. Weber, "Miba: Multitouch image-based authentication on smartphones," in *Proc. of the CHI'13 Extended Abstracts on Human Factors in Computing Systems (CHI EA'13), Paris, France*. ACM, April 2013, pp. 787–792.

[24] P. Corporation, "Real user coorp: Pass faces," 1998, http://www.realuser.com, [Online; Accessed on June 1, 2017].

[25] S. Brostoff and M. A. Sasse, *People and Computers XIV — Usability or Else!*, ser. Proc. of the Human Computer Interaction 2000 (HCI'00). Springer-Verlag London, 2000, pp. 405–424.

[26] D. Davis, F. Monrose, and M. K. Reiter, "On user choice in graphical password schemes," in *Proc. of the 13th Conference on USENIX Security Symposium (SSYM'04), San Diego, California, USA*. USENIX Association, August 2004, pp. 11–11.

[27] R. Biddle, S. Chiasson, and P. Van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Computing Survey*, vol. 44, no. 4, pp. 19:1–19:41, September 2012.

[28] T. Takada and H. Koike, "Human-computer interaction with mobile devices and services," in *Proc. of the 5th International Symposium on Mobile Human-Computer Interaction with Mobile Devices and Services (MobileHCI'03), Udine, Italy*, ser. Lecture Notes in Computer Science, vol. 2795. Springer-Verlag Berlin Heidelberg, September 2003, pp. 347–351.

[29] E. von Zezschwitz, P. Dunphy, and A. De Luca, "Patterns in the wild: A field study of the usability of pattern and pin-based authentication on mobile devices," in *Proc. of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI'13), Munich, Germany*. ACM, August 2013, pp. 261–270.

[30] L. Simon and R. Anderson, "Pin skimmer: Inferring pins through the camera and microphone," in *Proc. of the 3rd ACM Workshop on Security and Privacy in Smartphones & Mobile Devices (SPSM'13), Berlin, Germany*. ACM, November 2013, pp. 67–78.

[31] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang, "Accessory: Password inference using accelerometers on smartphones," in *Proc. of the 12th Workshop on Mobile Computing Systems & Applications (HotMobile'12), San Diego, California, USA*. ACM, February 2012, pp. 9:1–9:6.

[32] L. Cai and H. Chen, "Touchlogger: Inferring keystrokes on touch screen from smartphone motion," in *Proc. of the 6th USENIX Conference on Hot Topics in Security (HotSec'11), San Francisco, California, USA*. USENIX Association, August 2011, pp. 9–9.

[33] Z. Xu, K. Bai, and S. Zhu, "Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors," in *Proc. of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WISEC'12), Tucson, Arizona, USA*. ACM, April 2012, pp. 113–124.

Author Biography

Meriem Guerar received the Master degree in Information Systems and Networks from the University of Sciences and the Technology of Oran (USTO), Algeria, in 2011, and her Ph.D. in 2017. Her main research interests include the areas of authentication and identity management, security and usability, smartphone security and payment systems security.

Alessio Merlo. got a M.Sc. in Computer Science in 2005 at University of Genova. He received his Ph.D. in Computer Science from University of Genova (Italy) in 2010 where he worked on performance and access control issues related to Grid Computing. He is currently serving as an Assistant Professor at the Universita' degli Studi di Genova, Italy, where he collaborates with the CSec Lab at DIBRIS. His currently research interests are focused on performance and security issues related to Web, distributed systems (Grid, Cloud) and mobile (Android platform). He is a member of the IEEE Computer Society and ACM. He is committees of international conferences (IFIP-SEC, AINA, participates to program ARES, HPCS,. . .) and he is member of the Editorial Board of an International Journal (Journal of High Speed Networks).

Mauro Migliardi got his Ph.D. in Computer Engineering in 1995. He was a Research Associate and Assistant Professor at the University of Genoa and Research Associate at Emory University; currently he is Associate Professor at the University of Padua, Adjunct Professor at the University of Genoa, member of the Steering Committee of the Center for Computing Platforms Engineering and of the Scientific Board of Circle Garage s.r.l. startup. His main research interest is distributed systems engineering, with a focus on security, pervasive systems, human memory support services and energy awareness. He has won the 2013 Canada–Italy Innovation Award, tutored more than 100 among Bachelor, Master and Ph.D. students at the Universities of Genoa, Padua and Emory, and (co-)authored more than 120 scientific papers.