9-2017

EvoPass: Evolvable graphical password against shoulder-surfing attacks

Xingjie YU
Singapore Management University, xjyu@smu.edu.sg

Zhan WANG
RealTime Invent Inc

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Liang LI
Beijing Normal University

Wen Tao ZHU
Chinese Academy of Sciences

See next page for additional authors

DOI: https://doi.org/10.1016/j.cose.2017.05.006

Citation

YU, Xingjie; WANG, Zhan; LI, Yingjiu; LI, Liang; ZHU, Wen Tao; and SONG, Li. EvoPass: Evolvable graphical password against shoulder-surfing attacks. (2017). *Computers and Security*. 70, 179-198. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/3715

Author

Xingjie YU, Zhan WANG, Yingjiu LI, Liang LI, Wen Tao ZHU, and Li SONG

Published in Computers and Security, 2017 May, Volume 70, Pages 179-198
http://doi.org/10.1016/j.cose.2017.05.006

EvoPass: Evolvable graphical password against shoulder-surfing attacks ☆

Xingjie Yu [a,*], Zhan Wang [b], Yingjiu Li [a], Liang Li [c,d], Wen Tao Zhu [e,f], Li Song [e,f]

[a] Secure Mobile Centre, School of Information Systems, Singapore Management University, Singapore
[b] RealTime Invent, Inc., China
[c] College of Information Science and Technology, Beijing Normal University, China
[d] School of Computer and Control Engineering, University of Chinese Academy of Sciences, China
[e] Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, China
[f] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, China

Keywords:
Authentication security
Graphical password
Shoulder-surfing
Evolvable
Time-evolving

ABSTRACT

The passwords for authenticating users are susceptible to shoulder-surfing attacks in which attackers learn users' passwords through direct observations without any technical support. A straightforward solution to defend against such attacks is to change passwords periodically or even constantly, making the previously observed passwords useless. However, this may lead to a situation in which users run out of strong passwords they can remember, or they are forced to choose passwords that are weak, correlated, or difficult to memorize. To achieve both security and usability in user authentication, we propose *EvoPass*, the first evolvable graphical password authentication system. EvoPass transforms a set of user-selected *pass images* to *pass sketches* as user credentials. Users are required to identify their pass sketches from a set of challenge images for user authentication. Particularly, EvoPass improves password strength gradually over time through continually degrading pass sketches without annoying users to reselect pass images. The evolving feature makes it difficult for observational adversaries to identify the pass sketches, even though part of pass sketches may have been exposed to adversaries previously. We introduce two metrics, *Information Retention Rate* (IRR) and *Password Diversity Score* (PDS) to guide the process of generating pass sketches and a set of challenge images. Our experimental analysis reveals that applying reasonable IRR and PDS in EvoPass can remarkably improve the resistance to shoulder-surfing attacks without negatively affecting user experience. We also implement a prototype of EvoPass on Android platform with reasonable IRR and PDS applied. Our experimental results on the prototype further demonstrate that EvoPass could work efficiently and achieve a desired usability.

1. Introduction

Password-based user authentication is the most common authentication method used on mobile devices. In password-based user authentication, a user provides an alphanumerical or graphical password on a user interface as the user's credential. However, the password entry process is vulnerable to shoulder-surfing attacks, in which a password entered on a user interface is observed by a nearby adversary without any technical record devices, such as a hidden-camera. Although the abilities of a shoulder-surfing attacker are restricted to those of a human, since people are prone to carry and use mobile devices everyday and everywhere, even in crowded places, it is important to address such attacks so as to protect mobile users's passwords.

Some authentication systems require or encourage users to choose strong passwords, such as non-dictionary passwords, to increase the difficulty for shoulder-surfing attacker to remember such passwords. However, it also inevitably increases the burden on users for remembering such passwords. Even worse, when users enter such passwords, it is about 40 percent slower than dictionary words (Thomas et al., 2005), making the password entry process more vulnerable to shoulder-surfing attacks. Another widespread measure to mitigate the risk of shoulder-surfing attacks is to change the password periodically or continually, thus making revealed passwords useless for shoulder-surfing attackers. Unfortunately, such practice may still result in poor user experience and even make users fail in authentication, since eventually it forces users to memorize passwords that they can hardly remember. Therefore, it remains a challenge to mitigate shoulder-surfing attacks and achieve good usability at the same time for password-based user authentication on mobile devices.

In this work, we propose EvoPass, an evolvable graphical password-based authentication mechanism on mobile devices. EvoPass improves the resistance to shoulder-surfing attacks gradually over time without requiring users to replace their passwords. Users of EvoPass are required to identify pass sketches from a challenge set of images that contains pass sketches and decoy sketches. The pass sketches are transformed from pass images selected by users from their private images. Each pass sketch is generated by processing edge extraction on the original pass image, reserving only a subset of edges and basic outlines. Using pass sketches instead of pass images as user credentials makes it difficult for a shoulder-surfing attacker to memorize and identify pass sketches without any prior knowledge of the original pass images, while users who are familiar with their pass images can still easily identify the pass sketches.

When used, EvoPass evolves pass sketches to more shoulder-surfing resilient versions through periodically or continually reducing the recognizable information contained in each pass sketch. In this way, legitimate users could still identify the evolved pass sketches based on the visual memory of previous pass sketches in successive authentication practices. Meanwhile, the evolved pass sketches increase the difficulty for a shoulder-surfing attacker to identify them, especially in the presence of other confusing decoy sketches. To our best knowledge, this is the first work that introduces the time-evolving feature in graphical passwords for gradually improving the resistance to shoulder-surfing attacks without requiring users to change passwords.

To balance between security and usability of EvoPass, it is important to ensure that the recognizable information in each pass sketch should be as little as possible against the shoulder-surfing attacks and enough for users to easily identify the pass sketches. For such purpose, we introduce Information Retention Rate (*IRR*) as a metric to evaluate the reservation rate of information entropy between a pass sketch and its original pass image and thus determine the appropriate edge extraction level for each pass image. Meanwhile, we further add decoy sketches in the challenge set to confuse the adversaries in shoulder-surfing attacks. To guide the selection of decoy sketches, we introduce Password Diversity Score (*PDS*), a metric for evaluating the statistical characteristics of the challenge set.

Our experimental results show that, with the help of *IRR* and *PDS*, EvoPass can achieve better resistance to shoulder-surfing attackers than other graphical password-based authentication systems. To further explore the usability and efficiency of EvoPass, we implement a prototype on Android platform and demonstrate that users can quickly become skilled at using EvoPass and pass user authentication within an acceptable period time.

The rest of this paper is organized as follows. Section 2 describes the related work. Section 3 introduces the preliminaries. Section 4 presents the design of EvoPass. Section 5 analyzes the security of EvoPass. Section 6 presents a prototype of EvoPass on Android platform and evaluates its usability and efficiency. Section 7 concludes this paper.

2. Related work

2.1. *Human visual ability*

The recognition of visual objects has been studied rigorously for decades (Denning et al., 2003a, 2003b, 2003c, 2003e). EvoPass relies on human ability of recognizing images. Humans have extraordinary ability to recognize degraded images especially when they gain the knowledge of the original images. Biederman (1987) proposed a theory of human image understanding, which is named as Recognition-by-Components. This theory claims that if an arrangement of two or three generalized-cone components can be recovered from the input, objects can be quickly recognized even when they are occluded, novel, rotated in depth, or extensively degraded. Denning et al. (2003d) explored implicit memory for painless password recovery, and their results suggest that implicit memory can be potentially used for low-cognitive-overhead, high-stability, and knowledge-based authentication.

2.2. *Graphical passwords and attacks*

Graphical password systems (e.g. Angeli et al., 2005; Hong et al., 2004; Suo et al., 2005; Wiedenbeck et al., 2005) gain more attention recently as a promising alternative of text-based password systems in both academia and industry. Most graphical password mechanisms are based on recognizing system-provided images, such as faces (Passfaces corporation, 2001), identifying user uploaded images (Hayashi et al., 2008; Pering et al., 2003; Takada et al., 2006), recognizing the category of pass images (Khot et al., 2011), or recalling a sequence of actions, such as clicking (Chiasson, 2008; Chiasson et al., 2009a, 2009b; Dirik et al., 2007) and drawing (Dunphy and Yan, 2007). Various graphical password authentication systems have also been supported in industry. For example, the graphical identification and authentication (GINA) system is a component of Windows 7, Windows Server 2008, Windows Server 2008 R2, and Windows

Vista, which provides secure authentication and interactive login services (Microsoft, 2013). Gao et al. (2013) studied user choices in Windows 8 graphical password scheme and analyzed the hot-spots caused by user choices.

Among the graphical password systems that are used in mobile environment (e.g. Hayashi et al., 2008; Khan et al., 2011), that of Hayashi et al. (2008) is the most related work to ours. Hayashi et al. (2008) proposed a graphical password system named Use Your Illusion. This system, just like EvoPass, relies on human ability to recognize a degraded version of a previously seen image. Use Your Illusion utilizes an image process filter to eliminate most details in an image, while preserving some features such as color and rough shapes. However, Use Your Illusion does not solve the following problems. First, it allows users to upload their own images as pass images without checking the quality of these images, which may make these pass images vulnerable to some sophisticated adversaries. For example, if all the pass images are very similar, the pass images may attract the adversaries' attention and thus be selected for password guess attacks. Therefore, it is better to provide a guideline or validation process for choosing the pass images. Second, it applies a fixed distortion parameter for processing all pass images; however, since the entropy of a pass image is usually different from other pass images, applying a fixed distortion parameter for all pass images may cause some pass images containing little recognizable information, which makes such pass images hard to identify. Third, for each pass image, the distortion algorithm and parameters remain the same for all authentication practices, thus the challenge set of images remains the same. Its resistance to shoulder-surfing attacks can only be improved by requiring users to re-select pass images. Finally, the average login time (which is over 10 seconds) is not acceptable for most people to authenticate themselves on mobile devices.

Table 1 compares the features of different popular recognition-based password systems with EvoPass. We have the following observations from the comparison. First, most of the previous graphical password systems choose decoy images from system images, and in some cases, even pass images are chosen by the system from system images, rather than from users' owned images. In this way, it is difficult for users to remember pass images; it also enables an attacker who has captured a large number of decoy images to guess users' pass images (more details are given in Section 5.3). Some previous graphical systems are resilient to shoulder-surfing attacks. Like EvoPass, they do not require users to select original pass images during authentication processes. For example, DynaHand (Renaud and Olsen, 2007) requires a user to recognize an image in which a random sequence of numbers is displayed in the user's handwritten numerals rather than an image that contains the user's PIN. Tetrad (Renaud and Maguire, 2009) requires a user to align his/her pass images either horizontally, vertically or diagonally. Although these systems provide resistance to shoulder-surfing attacks, none of them improve password strength gradually over time without requiring users to update their passwords.

3. Preliminaries

3.1. Terminology

We will use the following security terminology in this paper:

Pass image – A pass image is an image chosen by a user from a set of private images and registered to EvoPass.

Decoy image – A decoy image is an image chosen by EvoPass from a set of images in system database according to the statistical characteristics of pass images.

Sketch – A sketch is a binary image generated by processing an image with a specific edge detection algorithm.

Pass sketch – A pass sketch is a binary image generated by processing a pass image with a specific edge detection algorithm.

Decoy sketch – A decoy sketch is a binary image generated by processing a decoy image with a specific edge detection algorithm.

Challenge set – A challenge set is a set of binary images, which is composed of all pass sketches and some decoy sketches.

Password Diversity Score (*PDS*) – *PDS* is a metric that is calculated based on the distance between each pair of two images in a set of images and used to guide the selection of decoy images.

Information Retention Rate (*IRR*) – *IRR* is a metric that is calculated between a sketch and its original image and used

Table 1 – A comparison between EvoPass and previous recognition-based graphical password schemes on various key features.

Sys.	Key src.	Decoy src.	User picks	Filtering	RSSA[1]	RSEA[2]	Time-evolving
VIP3 (Angeli et al., 2005)	Stock	Stock	No	Manual	No	Yes	No
Awase (Takada et al., 2006)	User	Stock	Yes	None	No	Yes	No
UYI (Hayashi et al., 2008)	User	Stock	Yes	None	Yes	No	No
Pering (Pering et al., 2003)	User	Peers	Yes	Manual	No	No	No
Déjà vu (Dhamija and Perrig, 2000)	Fractals	Fractals	Yes	Manual	No	Yes	No
Passfaces (Passfaces corporation, 2001)	Stock	Stock	No	Manual	No	Yes	No
GridMap (Balen and Wang, 2014)	Stock	Stock	Yes	None	Yes	No	No
DynaHand (Renaud and Olsen, 2007)	Stock	Stock	No	Automatic	Yes	No	No
Tetrad (Renaud and Maguire, 2009)	Stock	Stock	No	None	Yes	Yes	No
EvoPass	User	User	Yes	Automatic	Yes	Yes	Yes

[1] RSSA: Resilient to Shoulder-Surfing Attacks.

[2] RSEA: Resilient to Social Engineer Attacks.

to evaluate the recognizable information remained in the sketch.

Roll-back – Roll-back is a set of operations used to generate a new version of a challenge set. Each pass sketch in this new version contains more recognizable information than in the current version.

3.2. *Threat model*

In the authentication process, EvoPass presents a challenge set that contains both decoy sketches and pass sketches on a user interface, and a user is required to choose all pass sketches to pass the authentication. We use a threat model in which an adversary who is a shoulder-surfing attacker. A shoulder-surfing attacker intends to capture a legitimate user's pass sketches through observing the user's selection of pass sketches without any technical recording device. The capability of the shoulder-surfing attacker is restrained to a human, relying on only his/her memory and manual tools such as pencil and pen (Roth et al., 2004).

Out of the scope of this paper is the technology-based recording attacker who can enhance vision using binoculars or a low power telescope, record a login process using video cameras, video mobile phones, keystroke logging software, or malicious software, and capture user's actions using remote electro-magnetic sensors (Wiedenbeck et al., 2006). It remains a future work to evaluate EvoPass against this kind of attackers.

In addition to shoulder-surfing abilities, the adversary considered in our threat model may have a physical access to a user's device and analyze the oil residues left by a user on the user interface. The adversary can mount exhaustive password guessing attacks, dictionary attacks and social engineering attacks to obtain a user's pass images.

4. EvoPass

4.1. *Overview*

EvoPass is a graphical password authentication system on mobile devices. EvoPass authenticates users by requiring them to recognize all pass sketches in a challenge set. These pass sketches are transformed from the pass images uploaded by the user at registration. As time goes by, EvoPass evolves pass sketches to more shoulder-surfing resilient versions.

Fig. 1 illustrates an instance of EvoPass. A user uploads some of his/her private images as pass images (shown in Fig. 1a). In order to conceal such pass images in a challenge set, EvoPass chooses some decoy images and then transforms all the pass images and decoy images to sketches. After that, the challenge set is presented to the user for authentication, as shown in Fig. 1b, in which the pass sketches are highlighted in red frames. Over time, EvoPass further evolves the pass sketches by obscuring their outlines and reducing their edge information as shown in Fig. 1c.

EvoPass is a client–server system. However, a client only communicates with the server for registering the pass images. Other operations (e.g., user authentication, evolving pass sketches, roll-back) are just performed locally to reduce the burden on the network. *EvoPass* includes four functional subsystems: *Registration*, *Authentication*, *Time-evolving* and *Roll-back*.

4.1.1. *Registration*

The following steps are executed at registration:

- Private images uploading: first, a user chooses some images from his/her private images as pass images. Note that, in different implementations, the number of pass images should be decided based on certain security requirements. After the pass images are selected, the client performs a primary security evaluation of these pass images. If these pass images are identified insecure, i.e., any pair of these pass images are graphically similar (which can be identified by comparing a pair of images with perceptual hash algorithms (Zauner, 2010)), the user is required to rechoose pass images. Otherwise, the client sends pass images to the server.
- Challenge set generating: Upon receiving a registration request and pass images from a client, the server chooses a certain number of decoy images from the system database. The system database is constituted by registered pass images of all users to defend against harvest attacks (Section

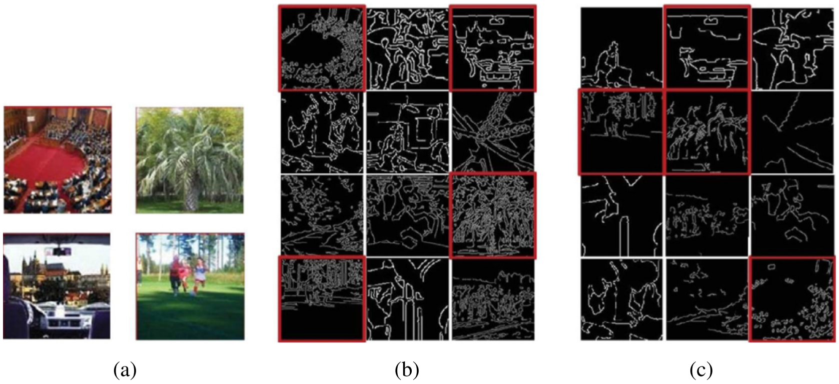


Fig. 1 – An instance of EvoPass. (a) shows the pass images uploaded by the user. (b) illustrates the sketches generated from the images in (a) through edge extraction. The sketches in (c) are evolved versions of those in (b), in which the edge information is further reduced. The red borders in (b) and (c) highlight the pass sketches. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

5.3). The server first transforms pass images and decoy images into gray-scale images. Then the server processes gray-scale pass images and decoy images with an edge detection algorithm generating pass sketches and decoy sketches. After that, the server sends the challenge set composed of pass sketches and decoy sketches to the client. Note that, to keep pass sketches recognizable to a user while increasing the difficulty for shoulder-surfing attackers to identify pass sketches, two metrics – Information Retention Rate (*IRR*) and Password Diversity Score (*PDS*) – are applied in generating a challenge set. More details about these two metrics are given in Section 4.2.
- User training: EvoPass provides a user training phase to make it easier for users to recognize pass sketches. In this phase, a client presents a received challenge set to a user. The user can practice several times in selecting his/her pass sketches in the challenge set until he/she finds no difficulty in identifying all pass sketches. Moreover, if a user is not satisfied with the challenge set, e.g., feeling difficult in recognizing pass sketches, the challenging set can be refreshed through restarting from the step of private images uploading.

4.1.2. *Authentication*

In authentication, EvoPass challenges a user with a challenge set presented on a user interface. A user is required to identify all pass sketches. To prevent shoulder-surfing attackers from recognizing pass sketches through remembering the positions of pass sketches in a challenge set, all sketches in a challenge set are displayed in a random sequence for each authentication attempt.

To mitigate the risk of password guessing attacks, EvoPass enforces a lock out policy to block a user who continuously fails in passing the authentication several times. The allowed number of failed authentications attempts should be specified based on certain security requirements of EvoPass implementations. Once a user is blocked, he/she should be authenticated by other authentication methods for unblocking. For example, EvoPass may authenticate a user in such situation by requiring him/her to provide a PIN code which is previously set by the user for security enhancement.

4.1.3. *Time-evolving*

EvoPass evolves pass sketches to more shoulder-surfing resilient versions through reducing the recognizable information contained in each pass sketch. The evolving operations can be performed periodically or constantly based on user configuration. Meanwhile, EvoPass allows users to activate evolving operations at any time as demand. There are two ways to generate an evolved version of a challenge set. One way is to process gray-scale pass images and decoy images with an edge detection algorithm in a higher edge detection level. Another way is to process current version of pass sketches and decoy sketches with an edge detection algorithm. More details about these two ways are given in Section 4.3.

4.1.4. *Roll-back*

In case a user feels difficult in recognizing evolved pass sketches, EvoPass allows a user to roll back a challenge set that contains more information that the user can recognize in each pass sketch and decoy sketch. The number of versions that a user can roll back from current version should be decided based on the trade-off between the usability and storage consumption. If the system supports rolling back a relatively large number of versions, it is more convenient for users. However, since roll-back operations are performed locally to reduce the burden on the network, supporting a relatively larger number of versions requires relatively more storage space for saving the parameters of previous versions. More details about the roll-back mechanism are given in Section 4.4.

4.2. *Metrics for generating a challenge set*

To achieve both desirable usability and security, EvoPass applies two metrics in generating a challenge set, *Information Retention Rate* (*IRR*) and *Password Diversity Score* (*PDS*). The pass images uploaded by a user may have different information entropy. If all pass images and decoy images are transformed to a fixed edge detection level (e.g., processing all pass images and decoy images with the same Canny parameter), some pass sketches may contain enough information for attackers to recognize them while other pass sketches are unrecognisable even by legitimate users. To solve this problem, *IRR* is proposed to evaluate the reservation rate of information entropy while transforming an image to a sketch and determining an appropriate edge detection level. On the other hand, if the decoy sketches in a challenge set can be clearly distinguished by attackers from the pass sketches, EvoPass can be easily broken in shoulder-surfing attacks. *PDS* is proposed to guide the selection of decoy sketches. With the help of *IRR* and *PDS*, EvoPass can achieve both practical usability and resistance to shoulder-surfing attacks.

4.2.1. *Information retention ratio (IRR)*

IRR is calculated between a sketch and its original image indicating that how many percentage of information is retained in the sketch after edge detection. *IRR* is defined as the information entropy of a sketch divided by the information entropy of its original image. In Shannon's information theory, entropy is a measurement of the uncertainty associated with a random variable (Shannon, 2001). Shannon's function is based on the concept that the information gain from an event is inversely related to its probability of occurrence (Pal and Pal, 1991). The entropy of an image can be defined as a statistical measure of its characteristics, representing the average amount of information contained in the image. For example, the one-dimensional (1*D*) entropy of an image *I* based on its one-dimensional gray level histogram can be calculated as follows (Kapur et al., 1985; Pun, 1981):

$$H_{1D}(I) = -\sum_{i=0}^{L} p_i \cdot \log(p_i) \tag{1}$$

where $L + 1$ is the number of gray levels, $p_i = f(i)/(M \times N)$ is the frequency of the occurrence $f(i)$ of gray level i, and $M \times N$ is the total number of pixels in image I.

1*D* entropy is used to characterize the accumulative distribution of gray levels in image *I*. However, it does not characterize the spatial correlation between pixels or the space distribution of gray levels in an image. Therefore, two images which have similar constituent gray levels but different space distributions would have similar 1*D* entropy. To make the entropy further reflect the space distribution of gray levels and the correlation between pixels, we also use two-dimensional entropy which is calculated with the surrounding pixels of a given pixel as another measurement of an image. Abutaleb (1989) defines two-dimensional entropy of an image *I* as follows:

$$H_{2D}(I) = -\sum_{i=0}^{L}\sum_{j=0}^{L} p_{ij} \cdot \log(p_{ij}) \tag{2}$$

where $p_{ij} = f(i, j)/(M \times N)$ is the frequency of the occurrence $f(i, j)$ of pixels' gray level *i* and their average gray level *j*; $L + 1$ is the total number of gray levels; and $M \times N$ is the total number of pixels in image *I*.

Similarly, for a color image (i.e., an RGB image), both 1*D* and 2*D* entropies can be calculated based on the color histograms in its red (R), green(G), and blue(B) color channels, respectively, using Eqs. (1) and (2) (Zachary, 2000).

Using 1*D* entropy and 2*D* entropy as measurements of an image, both accumulation and space properties of the image can be calculated. Hence, *IRR* based on both one-dimensional (1*D*) entropy and two-dimensional (2*D*) entropy are defined as follows:

$$\begin{aligned} IRR_{1D} &= \frac{H_{1D}(F(I))}{H_{1D}(I_R) + H_{1D}(I_G) + H_{1D}(I_B)}, \\ IRR_{2D} &= \frac{H_{2D}(F(I))}{H_{2D}(I_R) + H_{2D}(I_G) + H_{2D}(I_B)} \end{aligned} \tag{3}$$

where *F*(*I*) represents a sketch generated by processing image *I* with an edge detection algorithm, and I_R, I_G and I_B represent the R, G, B channel image of image *I*, respectively.

After pass images are uploaded by a user, EvoPass performs edge detection on these images to generate their sketches and calculates IRR_{1D} and IRR_{2D} for each sketch. Only if both IRR_{1D} and IRR_{2D} of a sketch fall into an acceptable range, this sketch is accepted as a pass sketch. Otherwise, the pass image should be processed in a higher or lower edge detection level until both IRR_{1D} and IRR_{2D} fall into the acceptable range. Experimental analysis on the acceptable ranges of IRR_{1D} and IRR_{2D} are given in Section 5.1.1.

4.2.2. Password diversity score (PDS)

PDS is the variance of distances between each pair of images in a set of images. Hence, to calculate *PDS*, we first calculate the distance between each pair of images. For this purpose, a gray statistical histogram is introduced for a gray-scale image *p* as follows:

$$Histo_p = (g_{p1}, g_{p2} \ldots g_{pt}) \tag{4}$$

where g_{pi} is the total number of pixels in gray level *i* in the image *p*, and *t* is the total number of gray levels in the image. Since grayscale images for visual display are commonly stored with 8 bits per sampled pixel, *t* is set as 256 in our experiment. We calculate the Euclidean distance of two images *p* and *q*:

$$Dis_{pq} = \sum_{l=1}^{t} (g_{pl} - g_{ql})^2 \tag{5}$$

We can thus evaluate the similarity between each pair of images. The variance of the Euclidean distances indicates the fluctuation of these distances between images in a challenge set. A low variance means that the pass images are less conspicuous in a challenge set. We calculate *PDS* using the following formula:

$$PDS = Var(Dis_{12}, Dis_{13} \ldots Dis_{xy}) \tag{6}$$

where $1 \leq x, y \leq k + z$ and $x \neq y$; *k* and *z* are the number of pass images and decoy images, respectively.

Theoretically, *PDS* can be calculated based on either grayscale images or *RGB* images. The distance between a pair of images can be calculated based on color histograms in their R, G, and B color channels using Eq. (5). Consequently, *PDS* components should be calculated in their R, G, and B color channels using Eq. (6). Finally, *PDS* value of a set of *RGB* images is the mean value of *PDS* components calculated in R, G, and B color channels.

After receiving pass images, EvoPass server selects one decoy image based on *PDS* from an indeterminate challenge set *I*′ each time until all decoy images are selected. To select a decoy image, the server tests every image in the system database by tentatively adding the image into set *I*′ and calculating *PDS* of set *I*′. After all images in the system database have been tested, EvoPass chooses the image that results in the lowest *PDS* of set *I*′ as a decoy image and formally adds this decoy image into set *I*′. In this way, EvoPass continually finds the next decoy image which results in the lowest *PDS* of set *I*′ until all decoy images are identified.

For very large image databases, the computation of image distances results in poor performance (Zachary, 2000). According to our experimental results, *PDS* calculated based on grayscale images is effective enough for choosing appropriate decoy sketches. Hence, in our prototype, EvoPass server first transforms pass images received from a client into gray-scale images, and then calculates *PDS* only based on gray-scale images, which provides a substantial improvement in performance over color images. To further accelerate the computation of *PDS*, we can calculate *PDS* based on binary images (i.e., sketches), which is introduced in the Appendix.

4.3. Time-evolving feature

An evolved version of a challenge set can be generated by either processing gray-scale pass images and decoy images in a higher edge detection level or processing pass sketches and decoy sketches in the current version with an edge detection algorithm. Fig. 2a shows a sketch generated by processing an image with Canny algorithm while Canny parameter is set to 0.5. The other sketches in Fig. 2 are generated by processing the same image with Canny parameter setting to 0.6 (Fig. 2b) and 0.7 (Fig. 2c). Fig. 3a is the same sketch as Fig. 2a. Fig. 3b is generated by processing the sketch in Fig. 3a with Canny algorithm using Canny parameter 0.1, while Fig. 3c is generated by performing Canny algorithm on Fig. 3b with Canny parameter set to 0.1. We observe that performing edge detection on a sketch highlights the edges in the evolved sketch (Fig. 3b and Fig. 3c), while processing the original image in a higher edge detection level makes the evolved sketch (Fig. 2b and Fig. 2c) more

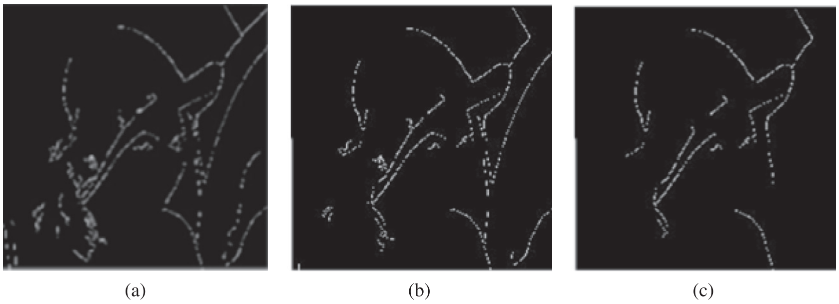

(a) (b) (c)

Fig. 2 – Sketches generated by processing an image with an edge detection algorithm in different edge detection levels.

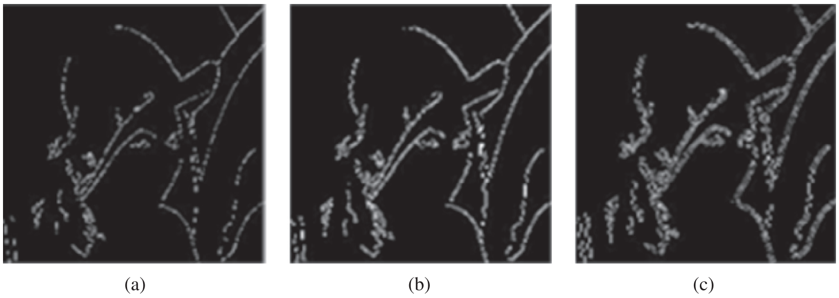

(a) (b) (c)

Fig. 3 – Sketches generated by processing the previous sketch with an edge detection algorithm.

difficult to recognize. Therefore, we suggest to generate evolved versions of a challenge set by processing gray-scale pass images and decoy images with an edge detection algorithm in a high edge detection level.

4.4. *Roll-back mechanism*

EvoPass supports two different roll-back mechanisms and both of these mechanisms can prevent roll-back operations activated by a shoulder-surfing attacker. Fig. 4 illustrates these two mechanisms, in which each node represents a version of the challenge set. For each hollow arrow in Fig. 4, the end node represents the next evolved version of the version represented by the start node.

As indicated by the dotted line in Fig. 4, EvoPass supports roll back to previously used versions of the challenge set. In this way, if a user activates roll-back operations while the *N*-th version of the challenge set is in use, the *N* – 1-th version of challenge set should be used for challenging the user as the result of roll-back operations. In this case, EvoPass validates the initiator of roll-back operations with other authentication mechanisms, such as PIN-based user authentication. Alternatively, users can also active the roll-back operations by answering some security questions which are widely used to authenticate users who have forgotten their passwords in the current authentication systems. In this case, users do not need to memorize additional PIN codes. Since the roll-back process is needed by chance, we suggest users to activate this roll-back operations in a security environment which could mitigate the risk of shoulder-surfing attacks in this process.

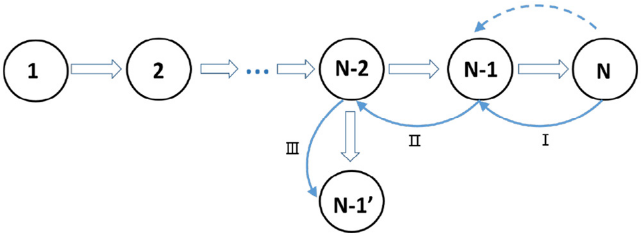


Fig. 4 – Roll-back mechanisms.

EvoPass also supports roll back to a new version of the challenge set which is composed of pass sketches and decoy sketches different from those previously used. This method is represented by the solid line in Fig. 4. In this way, once a user activates roll-back operations while the *N*-th version of the challenge set is in use, EvoPass first checks the parameters of the *N* – 1-th version including the edge detection algorithm used for generating the *N* – 1-th version and related parameters. Then EvoPass further checks the edge detection algorithm used for generating the *N* – 2-th version and related parameters. After that, EvoPass processes pass sketches and decoy sketches in the *N* – 2-th version with an edge detection algorithm that is different from the one used for generating the *N* – 1-th version, and thus generates the *N* – 1′-th version as the result of roll-back. In this case, the pass sketches in the *N* – 1-th version are different from the previously used pass sketches, which increases the difficulty for shoulder-surfing attackers to identify pass sketches even they have already obtained some information about the previous versions. Therefore, authenticating the roll-back initiator is not required for this solution. However, this solution consumes more storage space for keeping the parameters of previous versions, indicating the edge detection algorithms used for generating previous versions and corresponding function parameters.

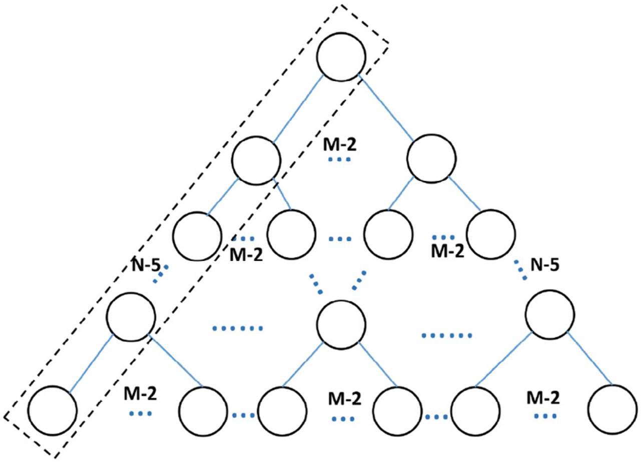


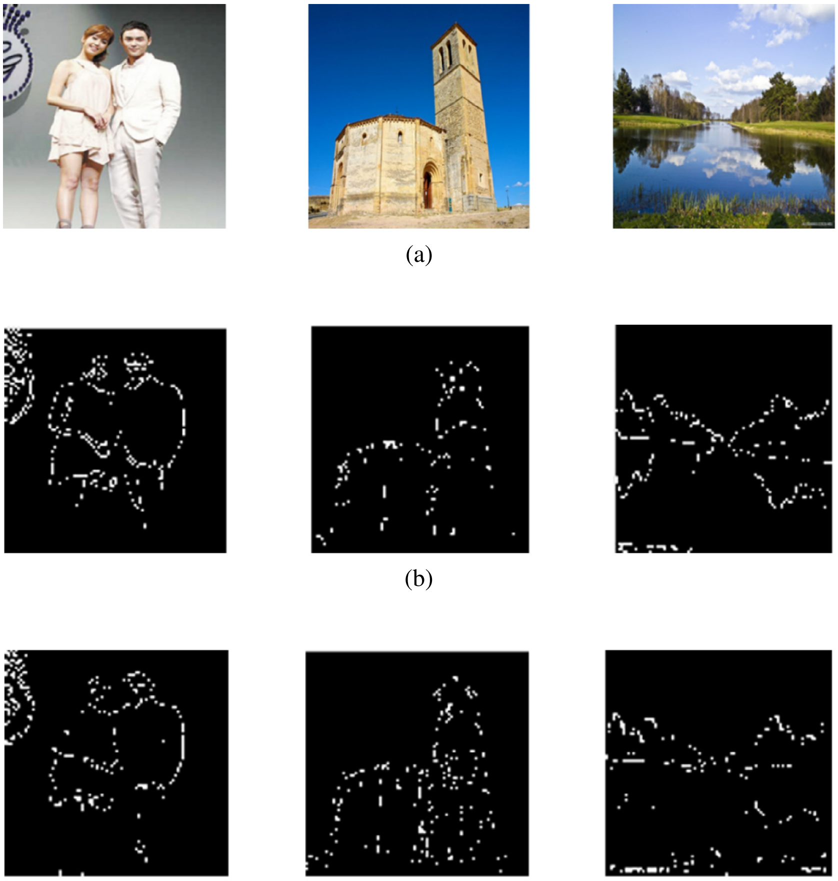
Fig. 5 – Space of evolved password.

4.5. *Space of evolved passwords*

For one password (constructed by pass sketches) registered by a user, EvoPass gradually generates evolved versions of this password as time goes by. However, the information contained in a pass sketch will eventually become too little for a legitimate user to recognize after multiple evolving operations. Therefore, the number of evolved versions of a given password is limited. For a given password, if EvoPass processes pass sketches with only one specific edge detection algorithm and a user can no longer identify his/her pass sketches since the $N + 1$-th version of challenge set, the user can recognize the 1st version of pass sketches to the N-th version. Hence, in this case, the number of usable evolved passwords (i.e. the passwords of the 1st version to the N-th version) is N.

A promising way to expand the space of evolved passwords is to process pass sketches (as well as the decoy sketches) with more than one edge detection algorithm. In this way, the space of evolved passwords is expanded to a tree structure, as shown in Fig. 5 in which each node represents one evolved version of the given password, M represents the number of edge detection algorithms, and N represents the number of recognizable evolved passwords when pass sketches are processed by only one edge extraction algorithm. Through processing pass sketches with M different edge detection algorithms, the number of evolved passwords greatly increases to $(1 - M^N)/(1 - M)$ from N. Moreover, besides being processed by edge detection algorithms, pass sketches and decoy sketches can also be processed in other methods to reduce recognizable information contained in a sketch (e.g., adding random noise in a sketch).

Fig. 6 provides a comparison on processing images with two different edge detection algorithms – Canny and Roberts. The



(a)

(b)

(c)

Fig. 6 – Sketches generated by processing an image with different algorithms.

sketches shown in Fig. 6b are generated by processing the original images with Canny algorithm and the Canny parameter is set to 0.5. We also process the original images using Roberts algorithm with the Roberts parameter set to 0.15, and the generated sketches are shown in Fig. 6c. It can be observed that, for a given original image, the corresponding sketch in Fig. 6b contains different edge information from that in Fig. 6c. More particularly, some sketches generated by processing the original images with Canny algorithm contain more information than those generated by processing Roberts algorithm on the original images, while other sketches generated by processing the original images with Canny algorithm contain less information than those generated by processing the original images with Roberts algorithm. The metrics, including *IRR* and *PDS*, and the respective acceptable ranges (which are given in Section 5.1) can also be used to evaluate the sketches generated by processing the images with Roberts algorithm. To broaden the pass sketch space of EvoPass using different edge detection algorithms, both *IRR* and *PDS* of the generated sketches should fall into the known acceptable ranges. It can be achieved by applying appropriate parameters in respective edge detection algorithm.

5. Security analysis

In this section, we first analyze the security of our system based on the values of *IRR* and *PDS* and give some suggestions on choosing these values according to our experimental results. We then evaluate the resistance of EvoPass to shoulder-surfing attacks. Finally, we analyze other attacks to EvoPass, including smudge attacks, exhaustive attacks, dictionary attacks, image-harvest attacks and social engineering attacks.

5.1. *Trade-off between security and usability*

The values of *IRR* and *PDS* should be decided based on the trade-off between security and usability. For *IRR*, a relatively high value means that more information of an image remains in its sketch, which also means that it is relatively easy for both legitimate user and shoulder-surfing attacker to recognize pass sketches. For *PDS*, a high value would make pass sketches conspicuous in a challenge set, which also means that that is relatively easy for both legitimate users and shoulder-surfing attackers to recognize pass sketches. Since users are familiar with their pass images (which are chosen from their private images) and also their pass sketches (through user training and continuous authentication practices), they require much less recognizable information than shoulder-surfing attackers to identify pass sketches. The values of *IRR* and *PDS* should be selected so as to make it as difficult as possible for shoulder-surfing attackers to identify pass sketches, while ensuring that users can still recognize pass sketches. We carry out a series of experiments to explore how to choose *IRR* and *PDS* values appropriately.

5.1.1. *Information retention ratio*

We invited 103 participants to help determine the appropriate range of *IRR* in our experiments. We gave each participant one identical test image and asked him/her to identify its sketch from a challenge set. This set is composed of the test sketch transformed from the test image and the 8 decoy sketches transformed from 8 different decoy images. For each challenge set, the test image and decoy images are processed by Canny algorithm with the same Canny parameter. For one test image, we processed the test image and decoy images using Canny parameter from 0.9 to 0.1 with a decrement of 0.05 in each step for generating different challenge sets. Each participant was asked to identify the test sketch from these challenge sets. We challenged each participant with 20 different test images.

According to our experimental result, 98.6 percent of test sketches generated with Canny parameter 0.35 can be recognized by the participants, while 98.1 percent of test sketches generated with Canny parameter 0.4 can be recognized by participants. A pass sketch should still contain enough information for a legitimate user to recognize it. Considering that the percentage of sketches that can be recognized by participants decreases while the Canny parameter increases, we decide that the lower threshold of *IRR* should ensure that more than 98 percent of sketches generated with this threshold can be recognized by participants. However, to increase the difficulty for a shoulder-surfing attacker to identify users' pass sketches, the information contained in pass sketches should be as less as possible. Hence, we set the lower threshold of *IRR* as 0.4. 97.3 percent of test sketches generated with Canny parameter (0.4–0.55) can be easily identified by participants while only 42.6 percent of test sketches generated with Canny parameter 0.6 can be recognized. Hence, we use the range of average *IRR* values corresponding to Canny parameter (0.4–0.55) as the acceptable range of *IRR* while generating pass sketches.

Then, we calculated the average values of IRR_{1D} and IRR_{2D} corresponding to each Canny parameter (0.9–0.1) from ten thousand images in Matlab. 20 test images and corresponding decoy images used in our experiment were also chosen from these ten thousand images. The relationship between IRR_{1D} and Canny parameter is shown in Fig. 7, and that between IRR_{2D} and Canny parameter is shown in Fig. 8. The acceptable range of IRR_{1D} 0.773%–1.26% corresponds to Canny parameter (0.4–0.55); and

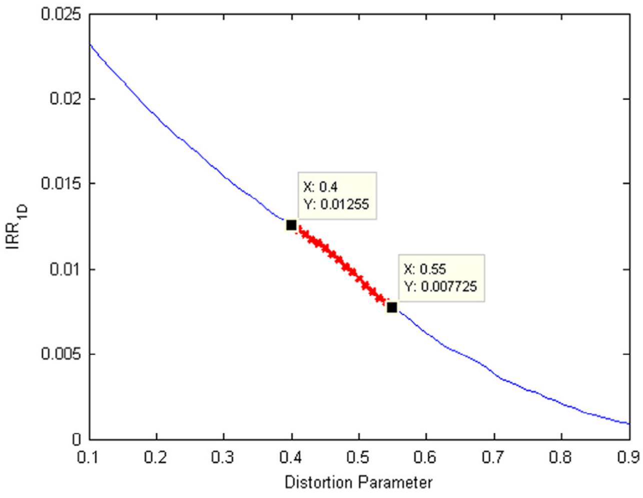


Fig. 7 – The relationship between IRR_{1D} value and Canny parameter.

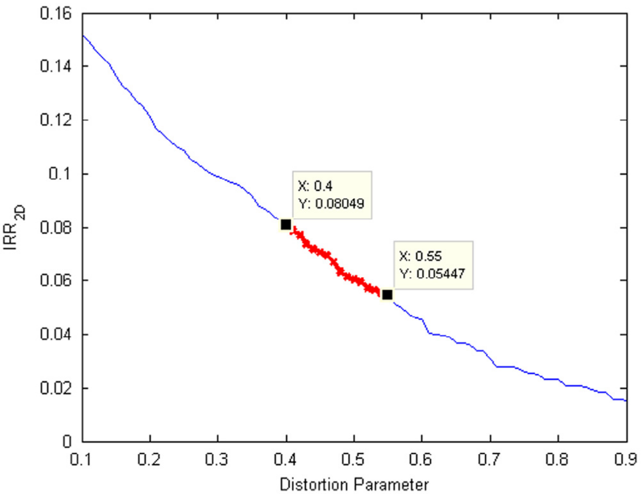


Fig. 8 – The relationship between IRR_{2D} value and Canny parameter.

the acceptable range of IRR_{2D} 5.45%–8.05% corresponds to Canny parameter (0.4–0.55).

After pass images are uploaded by a user, we suggest to generate pass sketches by processing pass images with default edge detection function parameters (at Canny parameter 0.55) and calculate IRR_{1D} and IRR_{2D} for each pass sketch. If IRR_{1D} of a pass sketch falls into 0.773%–1.26% and IRR_{2D} falls into 5.45%–8.05%, this pass sketch is accepted. Otherwise, the pass image is processed in a higher or lower edge detection level until both IRR_{1D} and IRR_{2D} fall into the acceptable range.

5.1.2. *Password diversity score*

Using *PDS* in challenge set generation, the difficulty of recognizing pass sketches can be quantified. To investigate the influence of *PDS* on the choice of decoy sketches, we compare a challenge set generated based on the highest *PDS* value with a challenge set generated based on the lowest *PDS* value. All decoy images are chosen from the same ten thousand images in our experiments.

Fig. 9 shows two challenge sets given the same pass images. Fig. 9a shows the challenge set generated using the highest *PDS* value, while Fig. 9b is the challenge set generated using the lowest *PDS* value. In Fig. 9a, some sketches contain clear outlines, while others contain large-area background and few outlines. In this situation, attackers may choose the sketches with clear outlines as pass sketches. In contrast, in Fig. 9b, there is no such difference among the sketches. Choosing images with relatively low *PDS* reduces the risk of exposing pass sketches in a challenge set.

A relatively low *PDS* value of a challenge set also means that pass sketches are concealed better in the challenge set, which raises the bar for a shoulder-surfing attacker to recognize pass sketches. However, it may also raise the difficulty for a user to identify pass sketches. To help a user recognize pass sketches in a challenge set generated based on the lowest value of *PDS*, EvoPass provides a user training phase to overcome the difficulty in identifying pass sketches. More details on generating the challenge set based on *PDS* are given in Section 6.

5.2. *Resistance to shoulder-surfing attacks*

A shoulder-surfing attack can be explained as password entry being observed by an attacker at the real-time, and replayed at a later time. To quantify the impact of shoulder-surfing attacks on EvoPass, we invited 20 participants to act as shoulder-surfing attackers observing password entry in EvoPass. Every participant stands behind a user; the accurate position is decided by the participant to ensure that he/she has a clear view to observe the user's authentication actions. We compare the resistance to shoulder-surfing attacks between EvoPass and two popular graphical password schemes – Awase-e (Takada et al., 2006) and Use Your Illusion (Hayashi et al., 2008). Awase-e is a graphic password system without image distortion features, while Use Your Illusion is a distortion-based graphic mechanism in which the recognizable information is reduced in a challenge set. We implement EvoPass and these two systems in Matlab.

The participants in our experiment are asked to observe the same user entering passwords in all three systems. Each system supports 3 different sizes of challenge set, which are 6-image,

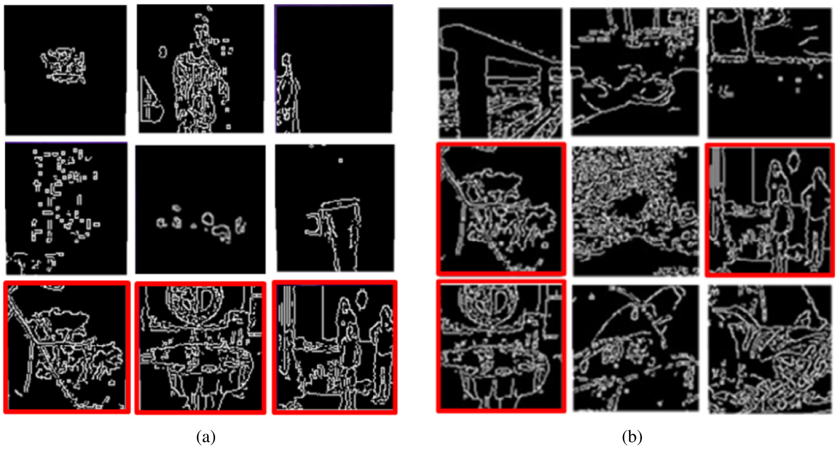


Fig. 9 – Challenge sets generated with different requirements on *PDS* value.

10-image and 14-image. In each system, a participant observes the user entering 10 different passwords in challenge sets of each size. For each password, after observing the user choosing pass images (i.e. pass sketches in EvoPass) in a login phase, the participant has one chance to authenticate himself/herself. If the participant can identify all pass images (i.e. pass sketches in EvoPass) that he/she has previously observed, the attack is successful, otherwise unsuccessful. The processes of observing password entry and performing the attack are repeated until the participant can successfully identify all pass sketches in a single attack.

In our experiments, the pass images that are supplied by a user are processed with the Canny algorithm in Matlab so as to generate pass sketches in EvoPass. Each sketch is generated with the default Canny parameter 0.55; a decrement of 0.05 is applied in the next try until IRR_{1D} falls into 0.773%–1.26% and IRR_{2D} falls into 5.45%–8.05%, which are chosen according to the experimental results in Section 5.1.1. After that, decoy images are chosen from 1000 system images based on the lowest *PDS*, and then decoy sketches are generated with Canny algorithm in the same way as pass sketches. Finally, all pass sketches and decoy sketches are presented in the size of 100 (pixels) × 100 (pixels); the pass images and decoy images of Awase-e and Use Your Illusion are presented in the same format, which is reasonable for mobile devices. To evaluate the improvement on the resistance to shoulder-surfing attacks, we further process each pass image and decoy image with the Canny parameter set to 0.05 higher than the one used for generating the first version of sketch. In Figs. 10–15, this case is labeled as *EvoPass – evolved*.

Table 2 – Number of participants for each level of self-estimated memory accuracy.

Self-estimated memory accuracy	100%	90%	80%	70%	60%	50%
Number of participants	1	2	3	5	6	3

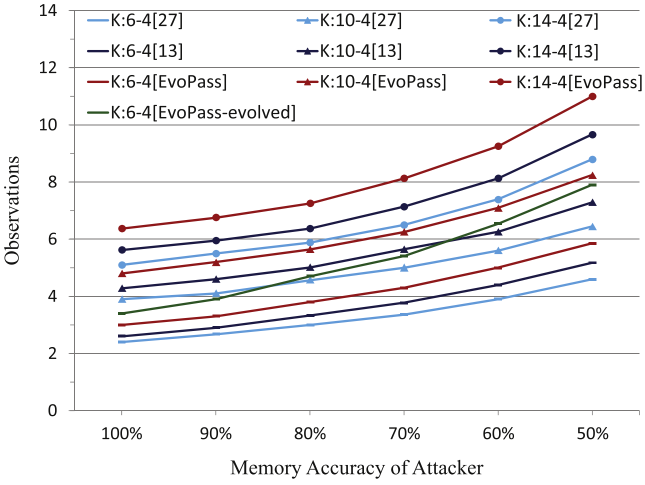


Fig. 10 – Number of observations required by an attacker to learn one pass sketch.

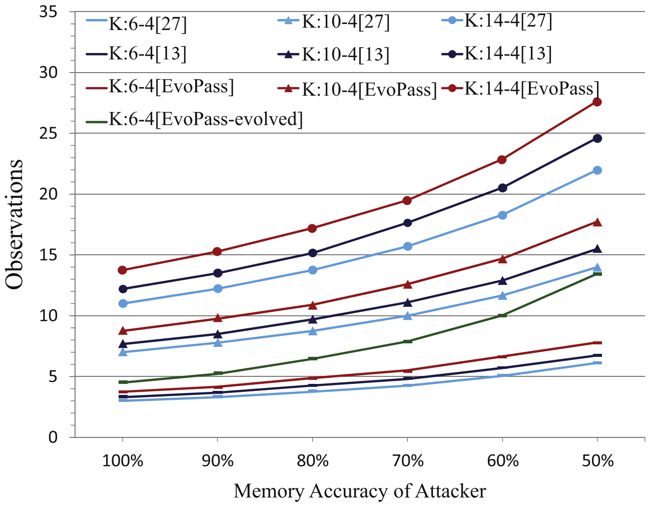


Fig. 11 – Number of observations required by an attacker to learn all pass sketches.

The participants are asked to estimate their memory accuracy ranging from 100% to 50%. The number of participants for each level of self-estimated memory accuracy are given in Table 2. Figs. 10 and 11 report the numbers of observations required by an attacker to learn only one pass sketch and all pass sketches, respectively. Figs. 12–14 provide more accurate statistical results on the number of observations required by an attacker to learn all pass sketches in each test system with error bars displayed. For the label "*K:a – b*" in these figures, "*a*" indicates the total number of images (i.e. sketches in EvoPass) in the challenge set; "*b*" indicates the number of pass images (i.e. pass sketches in EvoPass) required for one successful login, which is set to 4 in our experiments.

From Figs. 10–14, we have the following observations: (1) Given any size of the challenge set in each system, the attackers who are more confident with their memory accuracy need less observations to learn either one pass image/sketch or all pass images/sketches. (2) Given any memory accuracy of attacker's perception in each system, when the number of

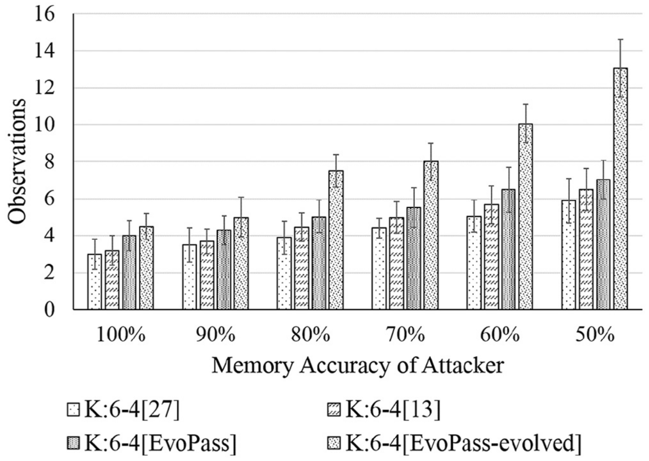


Fig. 12 – Number of observations required by an attacker to learn all pass sketches in a 6-image challenge set.

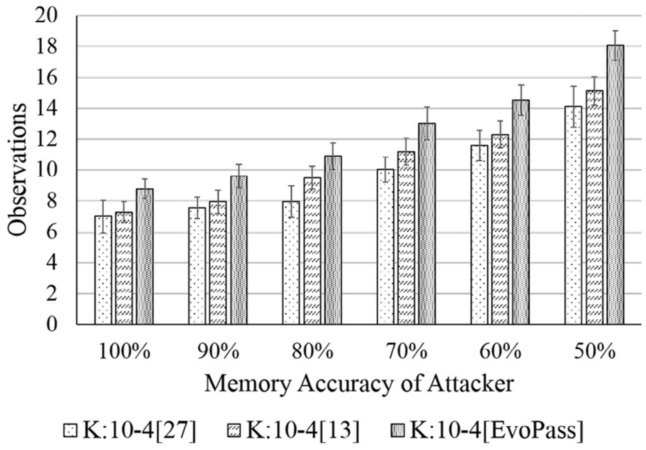


Fig. 13 – Number of observations required by an attacker to learn all pass sketches in a 10-image challenge set.

decoy images increases, the attackers need more observations to learn either one pass image/sketch or all pass images/sketches.

(3) Given any memory accuracy of attacker's perception, the distortion-based graphical password systems, including Hayashi et al. (2008) and EvoPass, require more observations than the distortion-free graphical password system (Takada et al., 2006) for an attacker to learn either one pass image/sketch or all pass images/sketches. (4) After evolving pass sketches and decoy sketches, the attackers, who estimate their memory accuracy to be less than 60%, need more observations to learn 1 pass sketch in a 6-image challenge set in EvoPass than in a 10-image challenge set in Takada et al. (2006) and Hayashi et al. (2008). (5) For the attackers who estimate their memory accuracy to be less than 60%, the observations to learn all 4 pass sketches in an evolved 6-image challenge set in EvoPass is close to those in a 10-image challenge set in Takada et al. (2006).

The success attack rates after different numbers of observations are recorded in Figs. 15–17. Once an attacker identifies all 4 pass images (i.e., pass sketches in EvoPass), which happens after observing a user's password entry *k* times, we call this attack a *successful attack after k observations*. In each system, for a given size of challenge set, the *success attack rate after k observations* is the total number of *successful attacks after k observations* divided by the total number of *successful attacks*. The total number of *successful attacks* is 200, which is also the total number of passwords that has been attacked. This is because, in our experiments, an attacker is allowed to repeatedly observe and attack until he/she can successfully identify all pass images (i.e. pass sketches in EvoPass).

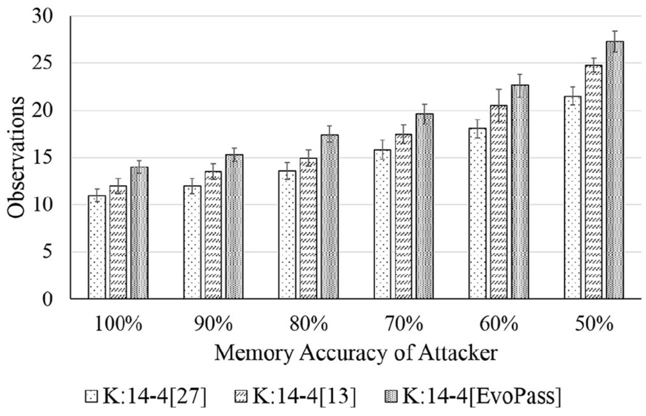


Fig. 14 – Number of observations required by an attacker to learn all pass sketches in a 14-image challenge set.

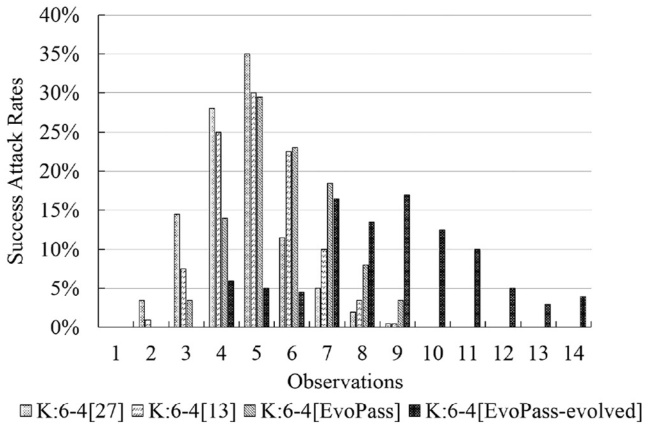


Fig. 15 – Success attack rates after different numbers of observations against a 6-image challenge set.

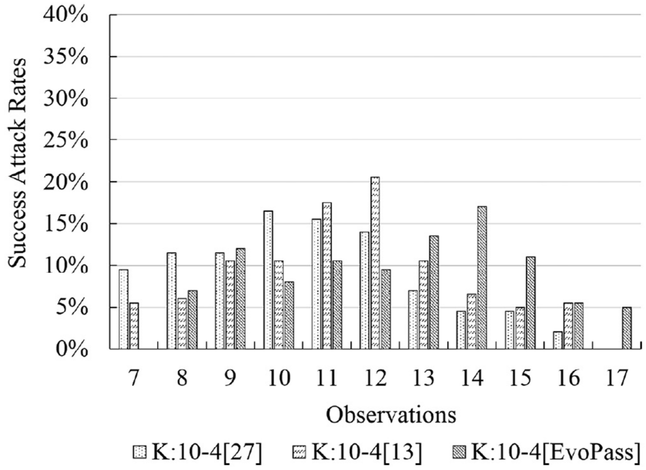


Fig. 16 – Success attack rates after different numbers of observations against a 10-image challenge set.

From Figs. 15–17, we have the following two observations which further demonstrate that the time-evolving feature significantly improves the resistance to shoulder-surfing attacks: (1) In EvoPass, given any 6-image challenge set, all successful attacks require no more than 9 observations when the challenge sets have not been evolved; however, after evolving the challenge sets, 53.5% successful attacks require no less than 9 observations. Hence, the time-evolving feature improves the resistance to shoulder-surfing attacks without increasing the number of decoy sketches. (2) Given any evolved 6-image challenge set in EvoPass or any 10-image challenge set in Takada et al. (2006), over 50% successful attacks require no less than 9 observations (53.5% in EvoPass 77% in Takada et al. (2006)). Compared to Takada et al. (2006), it is not difficult for

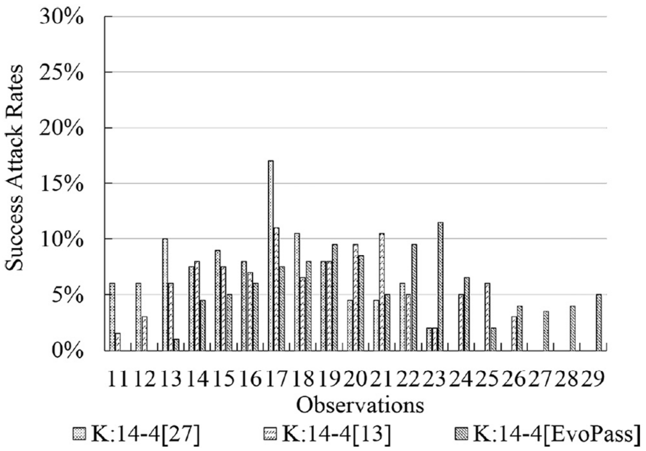


Fig. 17 – Success attack rates after different numbers of observations against a 14-image challenge set.

EvoPass to achieve the same or even stronger resistance to shoulder-surfing attacks with less decoy images by further evolving the evolved version of challenge set.

5.3. *Resistance to other attacks*

Besides the resistance to shoulder-surfing attacks, EvoPass also take countermeasures to the following known attacks and a new attack, which is named as image-harvest attack by us.

5.3.1. *Exhaustive attack*

This attack happens when an attacker selects images at random until all pass sketches are obtained. Assuming that an EvoPass application contains *P* pass sketches and Q decoy sketches, a random attacker attempting choose a user's pass sketches in the challenge set would get this right with the probability of $\frac{P}{P+Q}\times\frac{P-1}{P+Q-1}\times\ldots\times\frac{P-(P-1)}{P+Q-(P-1)}$. For a random 4-digit PIN which is widely used in current mobile authentication systems, the probability that an attacker can successfully guess a user's PIN is 1/1000. Hence, if an EvoPass implementation requires a user to select 3 pass sketches from an 18-image challenge set, the probability that the attacker can successfully choose a user's all 3 pass sketches is 1/816 which is close to the probability in 4-digit PIN-based authentication systems. In this case, an 18-image challenge set can be displayed on 2 pages; and each page contains 9 images which is a reasonable size for most mobile devices. Moreover, to mitigate the risk of exhaustive attacks, EvoPass supports a lock out policy which blocks a user after several continuous failed authentication attempts.

5.3.2. *Dictionary attack*

In a textual password system, users tend to select simple passwords following some regular patterns, which are vulnerable to dictionary attacks. However, in EvoPass, users select pass images differently and unpredictably. Moreover, unlike previous graphical password systems in which all pass sketches are selected from system images, a user of EvoPass selects pass images from his/her private images, which further increases the difficulty of mounting dictionary attacks.

5.3.3. *Social engineering attack*

Some attackers who know a user very well may recognize or guess the user's private images among several other images. However, EvoPass uses sketches to challenge users rather than original images. The edge information in a sketch only reveals partial content of an original image, which increases the difficulty for such attacker to identify the user's pass sketch.

5.3.4. *Smudge attack*

EvoPass presents pass sketches and decoy sketches on the user interface in a random sequence. Therefore, the oily residues on the screen does not carry any information or pattern about pass sketches.

5.3.5. *Images-harvest attack*

Images-harvest attack is a new attack to graphic password systems, which has not been considered before. If decoy images are generated from a dedicated system image database as in most previous graphical password systems, an attacker may register many accounts and/or frequently change pass images so as to get a large number of challenge sets and collect plenty of decoy images. Assuming that a system image database contains *M* images, and a challenge set contains *d* decoy images and *s* pass images. Thus, an images-harvest attacker can collect *d* decoy images from a challenge set. Through uploading pass images *L*/*d* times, the attacker can get *L*/*d* challenge sets and collect *L* decoy images if decoy images contained in a new challenge set are all different from the previous decoy images. Normally, the attacker needs more than *L*/*d* challenge sets to collect all *L* decoy images since some decoy images might be repeatedly displayed in several challenge sets. When the attacker tries to login to a target account after collecting *L* decoy images, the probability that he/she can recognize all decoy images is

$$P=\frac{L}{M}\times\frac{L-1}{M-1}\times\ldots\times\frac{L-(d-1)}{M-(d-1)} \tag{7}$$

Hence, an attacker who has collected all *M* images in the system database can even recognize all decoy images in a target account's challenge set, and then identify all pass images with 100 percent accuracy.

To keep EvoPass resilient to images-harvest attacks, the database used for choosing the decoy sketches in EvoPass is constructed by all users' pass images rather than system images. In this way, even if an attacker has collected all decoy sketches from a challenge set, he/she still cannot identify a target account's pass sketches by excluding all decoy sketches since both the target account's pass sketches and decoy sketches are contained in his/her collection of decoy sketches. Thus, only if a decoy sketch in the target account's challenge set is generated from a pass image uploaded by the attacker, the attacker can identify it as a decoy sketch. Still assuming that a system image database contains *M* images, and a challenge set contains *d* decoy sketches and *s* pass sketches. After uploading pass sketches *L*/*d* times, an attacker can only collect $L' = (L/d)\times s$ sketches which would be used as other users' decoy sketches, provided that the decoy images contained in a new challenge set are different from the previous decoy images. Then the system image database would contain $M' = M + (L/d) \times s$ images. The probability that such attacker can recognize all decoy images in a target account's challenge set is

$$P' = \frac{L'}{M'} \times \frac{L'-1}{M'-1} \times \ldots \times \frac{L'-(d-1)}{M'-(d-1)} \tag{8}$$

Once a challenge set contains more decoy sketches than pass sketches, $s < d$. Then, $L' = (L/d) \times s < L$. Because $L' < L$, and also because $M' = M + (L/d) > M$, we have $P' < P$. Hence, after uploading pass images L/d times, the probability that an attacker can recognize all decoy images in a target account's challenge set in EvoPass is usually much smaller than in other graphical systems that use system images as decoy images. For example, when a system image database contains 1000 images, and a challenge set contains 6 decoy images/sketches and 3 pass images/sketches, after uploading pass images 100 times, $P = 4.61\%$ and $P' = 0.0145\%$. More importantly, unlike in these other systems, the attacker can never ensure himself to recognize all decoy sketches in EvoPass by frequently uploading pass sketches, since these M sketches which are already contained in the database still might be used as the target account's decoy sketches.

5.3.6. *Technology-based recording attack*

EvoPass improves the resistance to technology-based recording attacks. There are two types of technology-based recording attackers. The technology-based recording attacker without the capability of image process is a weak adversary. After recording legitimate users' selection of pass sketches with automatic recording devices, such as hidden cameras or video cameras, such attackers can remember the pass sketches through repeatedly observing the records. However, such attackers may face an evolved version of challenge set when they try to pass authentication with recorded pass sketches. Since such attackers have no knowledge of original pass images, it is relatively difficult for them to recognize the evolved pass sketches. On the other hand, the technology-based recording attacker is a strong adversary who can process the recorded pass sketches with image process algorithms, including edge detection algorithms. Since the edge detection functions used in EvoPass are analogous to one-way functions used in cryptography, the attacker cannot recover the pass images using recoded pass sketches. Meanwhile, as we elaborated in Section 4.3, processing the current pass sketches with an edge detection algorithm may not always yield an evolved version of pass sketches.

6. Implementation and evaluation

In this section, we implement EvoPass as a screen unlock application on an Android 4.2.2 smartphone. Furthermore, we evaluate the usability and efficiency of our prototype.

6.1. *Implementation on Android platform*

We implement a prototype of EvoPass on an Android 4.2.2 smartphone. For the purpose of running the whole system correctly, we also build a web server using Apache TOMCAT and an FTP server using Serv-u. Considering the limited size of the screen, in our prototype, a challenge set is displayed as a 3 × 3 matrix, including three pass sketches and six decoy sketches. Each sketch in a challenge set is displayed in 100 (pixels) × 100 (pixels).

6.1.1. *Registration*

In our prototype, once the server receives a registration request and pass images, it chooses decoy sketches to generate a challenge set. In our prototype, we choose decoy skecthes from 1000 randomly chosen images in the database. According to our experimental results, it takes an average of 20.3 s to choose decoy images among 1000 randomly chosen images in the database based on *PDS*.

After all decoy images are identified, we use OPENCV Canny function to process pass images and decoy images with the default edge detection threshold set to [180,450]. We perform OPENCV Canny function on 1000 images with different thresholds to determine an appropriate default edge detection threshold. Finally, we set the default threshold as [180,450] which results in the average value of IRR_{1D} of those 1000 sketches being 0.811% and the average value of IRR_{2D} being 6.26%. According to our evaluations in Section 5.1.1, both values are acceptable. To keep pass sketches recognizable to a legitimate user, we calculate IRR_{1D} and IRR_{2D} of each pass sketch. Only if both IRR_{1D} and IRR_{2D} of a pass sketch fall into the selected range as given in Section 5.1.1, this pass sketch is accepted. Otherwise, we reprocess the pass images by increasing 10% of both high threshold and low threshold, until both IRR_{1D} and IRR_{2D} fall into the range.

6.1.2. *Evolving password*

In our prototype, we evolve the sketches in the challenge set by processing pass images and decoy images in a higher edge detection level to reduce the information contained in the sketches. Our prototype supports automatically evolving and the default evolving period is set to two weeks. The Canny edge detection threshold increases 2 percent each time to generate an evolved version of the challenge set. However, the default increment of threshold can be configured to a different value in other implementation based on different security requirements.

We set this default increment of threshold based on a heuristic view on the influences of different threshold increments. 100 participants are invited to identify evolved pass sketches generated with different edge detection thresholds. We divide 100 participants into 4 groups evenly. Participants in Group I, Group II, Group III and Group IV deal with evolved versions of challenge set generated by setting the default increment of threshold to 1%, 2%, 5% and 10%, respectively. Every participant chooses 3 pass sketches from his/her private images and tries twenty evolved versions of password. We record the success rate for each authentication period in Table 3. We observe that most participants are still satisfied with the 15th evolved version when the default variation of threshold is set to 2%.

6.1.3. *Roll-back*

Our prototype supports roll back to a new version of the challenge set, rather than to previous versions of the challenge

Table 3 – Success rate for each authentication period (the number of participants whose first request to increase the edge detection threshold/decrease the edge detection threshold appears in relative interval of attempts).

	Group I	Group II	Group III	With time-evolving II
[1st, 5th]	12/0	0/0	0/0	0/11
[6th, 10th]	5/0	8/0	2/13	2/5
[11th, 15th]	0/4	3/2	1/4	0/2
[16th, 20th]	0/9	0/9	0/3	0/2

Table 4 – Success rate for each authentication period (the number of successful authentications through 1 attempt/2 attempts/3 attempts/roll-back once/total number of authentications).

	Without time-evolving	With time-evolving I	With time-evolving II
1st day	317/18/5/–/340	320/19/1/0/340	313/25/2/0/340
2nd day	324/14/2/–/340	322/18/0/0/340	327/12/0/0/340
3rd day	324/15/1/–/340	316/23/1/0/340	326/20/3/1/340
1 week later	285/23/2/–/310	263/59/8/1/330	291/40/7/2/340
2 weeks later	256/29/5/–/290	290/39/9/2/340	247/45/6/1/300

set (which is introduced in Section 4.4). First, EvoPass checks the parameters of the current version of the challenge set and two latest versions as illustrated in Section 4.4. Then, the client processes the first version of pass sketches and decoy sketches with OPENCV Canny function using the same threshold used for generating the third from the end version of the challenge set. After that, it further processes the sketches with OPENCV Roberts function, an edge detection function, generating a new version of the challenge set. Note that, although we used Roberts algorithm in our prototype, other edge detection algorithms may be used in other EvoPass implementations.

An instance of roll-back operations in our prototype is shown in Fig. 18. Fig. 18a illustrates a new version of challenge set resulting from a roll-back request, and Fig. 18b is the current version of the challenge set when the user asks for roll-back. Fig. 18c shows the latest version of the challenge set before Fig. 18b. It is observed that the pass sketches and decoy sketches in Fig. 18a and Fig. 18c are graphically different, while either Fig. 18a or Fig. 18c contains more recognizable information than Fig. 18b.

6.2. *Usability and efficiency*

In order to evaluate the usability and efficiency of EvoPass with the default threshold increment set to 2%, we design a set of experiments and invite 103 participants in our experiments. Participants are required to install EvoPass client on their mobile phones, register their pass images at the first day, and pass the authentications in five periods during two weeks. In the fourth (one week later) and fifth period (two weeks later), after E-mail invitation, 98 and 93 participants respectively returned to our experiments.

We divide all the participants into three groups evenly. The participants in the first group are authenticated without time-evolving feature. Other two groups experience time-evolving feature with different evolving intervals. Group “with time-evolving I” reduces the recognizable information every two days. Group “with time-evolving II” reduces the recognizable information every day. We ask each participant to try on authentication every 6 minutes within one hour in each period. We allow participants when challenged with evolved versions of the challenge set to activate the roll-back operations. According to the experimental results shown in Table 4, participants in all three groups pass authentication in a high rate. Moreover, the success rate with time-evolving feature is close to that without time-evolving feature.

We record the login time for all successful authentications within three attempts. The login time is measured cumulatively, which means that the clock is not reset after a failed login attempt but runs until the user successfully logs in, or fails in three consecutive times. The average login time

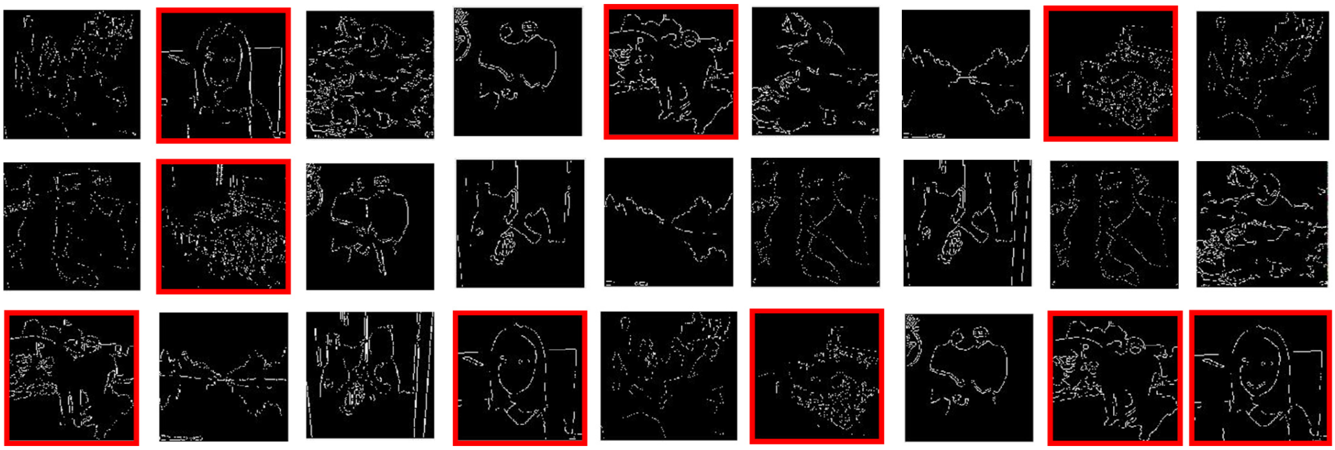

(a) (b) (c)

Fig. 18 – An instance of roll-back.

is 4.1 seconds, which is significantly faster than the average login time in Use Your Illusion (Hayashi et al., 2008) (which is over 10 seconds). This is because a user of Use Your Illusion is required to select three pass images out of a set of 27 images. These images are displayed on 3 pages, so that a user needs to switch back and forth between these pages several times to identify his/her pass images. In comparison, a 9-image challenge set in our prototype is displayed on a single page. To enhance the security of EvoPass against exhaustive attacks, users may be required to choose 3 pass sketches in a set of 18 or more images. These images would also be display on 2 or more pages, which will increase the login time accordingly. Since the average login time for choosing 3 pass sketches in a 9-image challenge set displayed on one page is only 4.1 seconds, the login time for choosing 3 pass sketches in an 18-image challenge set displayed on two pages should also be acceptable.

7. Conclusion

In this paper, we propose an evolvable graphical password system named EvoPass. EvoPass is resilient to shoulder-surfing attacks without requiring users to change their pass images. To achieve both desirable usability and resistance to shoulder-surfing attacks, we apply two metrics – *IRR* and *PDS* in generating a challenge set. Our experimental results show that, using edge detection as the image distortion feature in EvoPass improves its resistance to shoulder-surfing attacks compared to other graphical password systems without the image distortion feature. Furthermore, with the help of *IRR* and *PDS*, more observations of password entry are needed for a shoulder-surfing attacker to break a target account in EvoPass than in another graphical system featuring image distortion (Hayashi et al., 2008). Particularly, with the time-evolving feature, EvoPass can achieve the same resistance to shoulder-surfing attacks with less decoy images than other graphical systems. We further explore the usability and efficiency of EvoPass through implementing a prototype of EvoPass on Android 4.2.2 platform. The experimental results on our prototype demonstrate that users are skilled to use EvoPass with acceptable login time and a desired success login rate even after evolving their challenge sets.

Acknowledgment

This work was supported by the Singapore National Research Foundation under NCR Award Number NRF2014NCR-NCR001-012. We thank the associate editor and the two reviewers whose insightful comments and suggestions helped improve and clarify this manuscript.

Appendix A. Calculation of *PDS* based on binary images

To decrease the computational overhead caused by *PDS* calculation, one may calculate *PDS* based on binary images (i.e., sketches). PDS_s is the variance of distances between each pair of sketches in a set of sketches. While calculating PDS_s, the distance between two sketches is calculated based on the proportion of the pixel value 255 in a sketch. A sketch is evenly divided into N × N parts and the proportion of the pixel value 255 is calculated in each part. After that we build an N × N-dimensional coordinate system, in which the proportion of the pixel value 255 in each part represents the coordinate in one dimensional space correspondingly. In this way, we map a sketch into this N × N-dimensional coordinate system. Fig. A19 shows the way to map a sketch 1 into a 9-dimensional coordinate system, where m_i is the number of the pixel value 255 in part *i*, and n_i is the size of part *i*.

After mapping sketches into the N × N-dimensional coordinate system, the distance of image p and q is calculated by

$$Dis_{pq} = \sum_{i=1}^{N\times N} (x_{pi} - x_{qi})^2 \tag{A.1}$$

Then PDS_s is calculated as

$$PDS_s = Var(Dis_{12}, Dis_{13} \ldots Dis_{xy}) \tag{A.2}$$

where $1 \le x, y \le k + z$ and $x \ne y$; *k* and z are the number of pass sketches and decoy sketches, respectively.

Fig. A20 shows two challenge sets generated based on different PDS_s values in Matlab. The decoy sketches of these two challenge sets are chosen from the same ten thousand images used for generating the challenge sets in Fig. 9. Fig. A20a shows a challenge set generated based on the highest PDS_s value, while

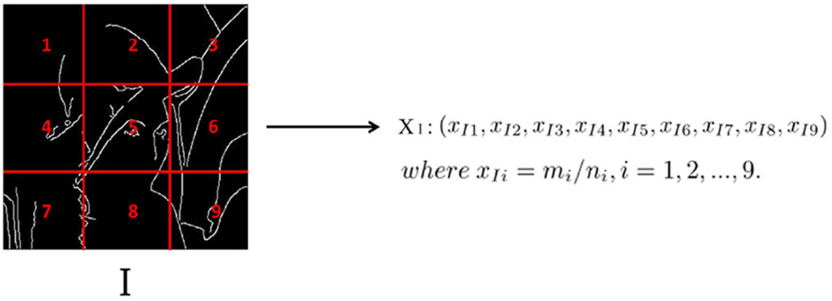
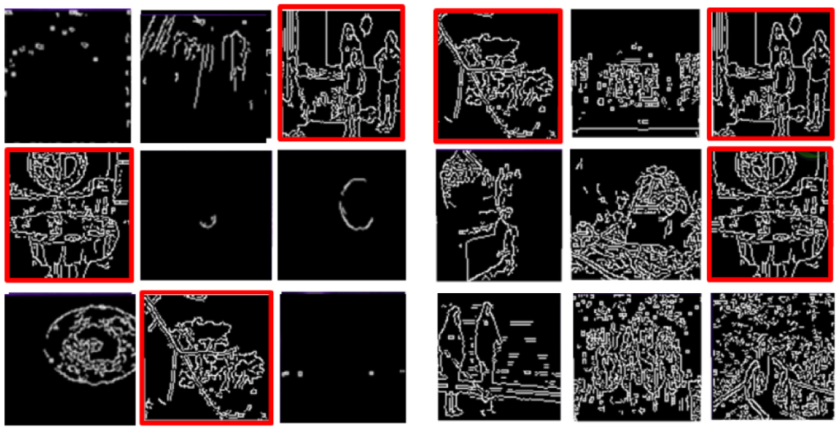

1
2
3
4
5
6
7
8
9
I
$X_I: (x_{I1}, x_{I2}, x_{I3}, x_{I4}, x_{I5}, x_{I6}, x_{I7}, x_{I8}, x_{I9})$
$where\ x_{Ii} = m_i/n_i, i = 1, 2, ..., 9.$

Fig. A19 – Mapping a sketch to a 9-dimensional system.



(a) (b)

Fig. A20 – Challenge sets generated with different requirements on PDS_s value.

Fig. A20b shows a challenge set generated based on the lowest PDS_s value. In Fig. A20b, there is no obvious difference among the sketches.

Hence, EvoPass can also choose decoy sketches based on PDS_s. In this way, when the server receives pass images from a client, it first processes each pass image with edge detection algorithm and calculates IRR_{1D} and IRR_{2D} of each pass sketch to ensure that enough recognizable information remains in the pass sketch. After that, the server processes all images in the database (or a set of images if appropriate) to sketches and chooses sketches that result in smallest value of PDS_s as decoy sketches. We also implement the selection of decoy sketches using PDS_s in our prototype. After transferring 3 pass images to pass sketches, the average time to choose 6 decoy images among 1000 randomly chosen images in the database based on PDS_s is 235 ms, while the average time is 20.3 s using PDS calculated based on gray-scale images.

The computational complexity of PDS_s is $O((N \times N)^2)$, while the computational complexity of PDS_g (which is calculated using Eqs. (5) and (6) based on gray-scale images) is $O(t^2)$, where t is the number of all pixels in an image. Therefore, choosing decoy images based on PDS_s results in a higher efficiency compared to PDS_g. While calculating PDS_s, the smaller the value of N is assigned, the higher the efficiency is achieved. However, less characteristics are used in calculating the distance between a pair of images with a smaller N. As a result, PDS_s is calculated based on less characteristics which may reduce the quality of a challenge set.

Appendix B. An example on evolved versions of a challenge set

Fig. B21 shows every 4th over 20 generations of a challenge set with the default variation threshold set to 2%. Fig. B21a is the first version of the challenge set, and the subsequent subfigures are the evolved challenge sets after every 4 rounds of password evolving.

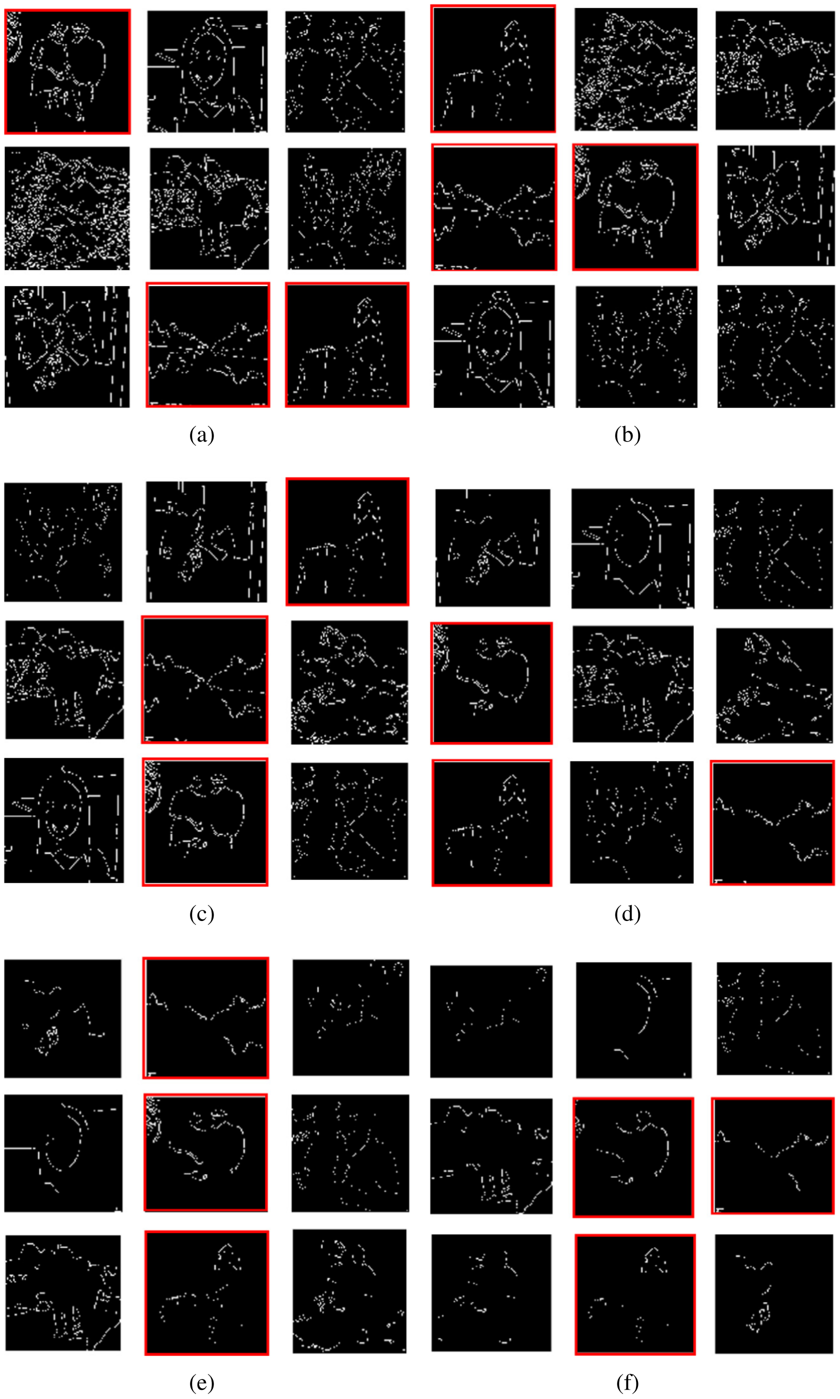

(a) (b) (c) (d) (e) (f)

Fig. B21 – Every 4th of 20 generations of evolved sketches.

Appendix. Supplementary material

Supplementary data to this article can be found online at doi:10.1016/j.cose.2017.05.006.

REFERENCES

Abutaleb AS. Automatic thresholding of gray-level pictures using two-dimensional entropy. Comput Vis Graph Image Proc 1989;47(1):22–32.

Angeli AD, Coventry L, Johnson G, Renaud K. Is a picture really worth a thousand words? Exploring the feasibility of graphical authentication systems. Int J Hum Comput Stud 2005;63(1–2):128–52.

Balen NV, Wang H. Effects of visual experience on the representation of objects in the prefrontal cortex. In: 10th International conference on security and privacy in communication networks; 2014.

Biederman I. Recognition-by-components: a theory of human image understanding. Psychol Rev 1987;94(2):115.

Chiasson S. Usable authentication and click-based graphical passwords [Ph.D. thesis]. School of Computer Science, Carleton University, Ottawa, Canada; 2008.

Chiasson S, Forget A, Biddle R, van Oorschot P. User interface design affects security: patterns in click-based graphical passwords. Int J Inf Secur 2009a;8(6):387–98.

Chiasson S, Forget A, Stobert E, van Oorschot P, Biddle R. Multiple password interference in text passwords and click-based graphical passwords. In: ACM conference on computer and communications security, pp. 500–511; 2009b.

Denning T, Bowers K, van Dijk M, Juels A. Brain regions associated with retrieval of structurally coherent visual information. Int J Comput Vis 2003a;53(3):225–43.

Denning T, Bowers K, van Dijk M, Juels A. Effects of visual experience on the representation of objects in the prefrontal cortex. Int J Comput Vis 2003b;53(3):225–43.

Denning T, Bowers K, van Dijk M, Juels A. The effects of visual object priming on brain activation before and after recognition. Int J Comput Vis 2003c;53(3):225–43.

Denning T, Bowers K, van Dijk M, Juels A. Exploring implicit memory for painless password recovery. Int J Comput Vis 2003d;53(3):225–43.

Denning T, Bowers K, van Dijk M, Juels A. Neuroimaging evidence for dissociable forms of repetition priming. Int J Comput Vis 2003e;53(3):225–43.

Dhamija R, Perrig A. Déjà vu: a user study using images for authentication. In: Proc. of the 9th conference on USENIX security symposium; 2000.

Dirik AE, Memon N, Birget J-C. Modeling user choice in the passpoints graphical password scheme. In: Proceedings of the 3rd symposium on usable privacy and security (SOUPS '07). New York, NY, USA: ACM; 2007. pp. 20–28.

Dunphy P, Yan J. Do background images improve "draw a secret" graphical passwords? In: Proceedings of the 14th ACM conference on computer and communications security (CCS '07). New York, NY, USA: ACM; 2007. pp. 36–47.

Gao H, Jia W, Liu N, Li K. The hot-spots problem in Windows 8 graphical password scheme. In: Cyberspace Safety and Security. Springer; 2013. p. 349–62.

Hayashi E, Dhamija R, Christin N, Perrig A. Use your illusion: secure authentication usable anywhere. In: Symposium on usable privacy and security. 2008. p. 35–45.

Hong D, Man S, Hawes B, Mathews M. A password scheme strongly resistant to spyware. In: Proc. international conference on security and management; 2004.

Kapur JN, Sahoo PK, Wong AK. A new method for gray-level picture thresholding using the entropy of the histogram. Comput Vis Graph Image Proc 1985;29(3):273–85.

Khan WZ, Aalsalem MY, Xiang Y. A graphical password based system for small mobile devices. CoRR abs/1110.3844; 2011.

Khot RA, Srinathan K, Kumaraguru P. Marasim: a novel jigsaw based authentication scheme using tagging. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI '11). New York, NY, USA: ACM; 2011. pp. 2605–2614.

Microsoft. Credentials management in windows authentication. 2013.

Pal NR, Pal SK. Entropy: A new definition and its applications. IEEE Trans Syst Man Cybernet 1991;21(5):1260–70.

Passfaces corporation. The science behind passfaces. Company white paper. 2001.

Pering T, Sundar M, Light J, Want R. Photographic authentication through untrusted terminals. IEEE Pervasive Comput 2003;2(1):30–6.

Pun T. Entropic thresholding, a new approach. Comput Graph Image Process 1981;16(3):210–39.

Renaud K, Maguire J. Armchair authentication. In: Proceedings of the 23rd British HCI group annual conference on people and computers: celebrating people and technology. British Computer Society; 2009. pp. 388–397.

Renaud K, Olsen ES. Dynahand: observation-resistant recognition-based web authentication. IEEE Technol Soc Magazine 2007;26(2):22–31.

Roth V, Richter K, Freidinger R. A pin-entry method resilient against shoulder surfing. In: Proceedings of the 11th ACM conference on computer and communications security. ACM; 2004. pp. 236–245.

Shannon C. A mathematical theory of communication. ACM SIGMOBILE Mobile Comput Commun Rev 2001;5(1):3–55.

Suo X, Zhu Y, Owen G. Graphical passwords: a survey. In: Proc. annual computer security applications conference, pp. 463–472; 2005.

Takada T, Onuki T, Koike H. Awase-e: recognition-based image authentication scheme using users' personal photographs. In: Innovations in information technology, pp. 1–5; 2006.

Thomas RC, Karahasanovic A, Kennedy GE. An investigation into keystroke latency metrics as an indicator of programming performance. In: Proceedings of the 7th Australasian conference on Computing education, vol. 42. Australian Computer Society, Inc.; 2005. pp. 127–134.

Wang Z, Jing J, Li L. Time evolving graphical password for securing mobile devices. In: Proceedings of the 8th ACM SIGSAC symposium on information, computer and communications security (ASIA CCS '13). ACM; 2013. pp. 347–352.

Wiedenbeck S, Waters J, Birget J, Brodskiy A, Memon N. Authentication using graphical passwords: effects of tolerance and image choice. In: Proc. symposium on usable privacy and security, pp. 1–12; 2005.

Wiedenbeck S, Waters J, Sobrado L, Birget J-C. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In: Proceedings of the working conference on advanced visual interfaces. ACM; 2006. pp. 177–184.

Zachary JM Jr An information theoretic approach to content based image retrieval [Ph.D. thesis]. Citeseer; 2000.

Zauner C. Implementation and benchmarking of perceptual image hash functions. na; 2010.

Xingjie Yu received her B.E. degree in Electronic Engineering from Nanjing Normal University, China, in 2010 and Ph.D. degree in Information Security from University of Chinese Academy of Sciences, China, in 2015. She is currently a research fellow in the Secure Mobile Centre, School of Information System, Singapore Management University, Singapore. Her research interests include mobile security and cloud security.

Zhan Wang received her Ph.D. degree from University of Chinese Academy of Sciences in 2013. She visited George Manson University from 2011 to 2013. Since July 2013, she has been a faculty member with State Key Laboratory of Information Security, which is now part of Institute of Information Engineering, Chinese Academy of Sciences. Her research interests include cloud and mobile security, authentication and deniable encryption. She joined Amazon Web Services since September 2015. She also started a security service start-up RealTime Invent, Inc. since December 2015.

Yingjiu Li is currently an Associate Professor in the School of Information Systems at Singapore Management University (SMU). His research interests include RFID Security and Privacy, Mobile and System Security, Applied Cryptography and Cloud Security, and Data Application Security and Privacy. He has published over 130 technical papers in international conferences and journals, and served in the program committees for over 80 international conferences and workshops. Yingjiu Li is a senior member of the ACM and a member of the IEEE Computer Society. The URL for his web page is http://www.mysmu.edu/faculty/yjli/.

Liang Li received his Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2013. He received his B.S. degree in Computer Science from Xian Jiaotong University, Shaanxi, China, in 2008. He is currently an assistant professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His research interests include image processing, large-scale image retrieval, image semantic understanding, multimedia content analysis, computer vision, and pattern recognition.

Wen Tao Zhu received his B.E. and Ph.D. degrees from University of Science and Technology of China, in 1999 and 2004, respectively. Since July 2004, he has been a faculty member with State Key Laboratory of Information Security, which is now part of Institute of Information Engineering, Chinese Academy of Sciences, where he has been a full professor since Oct. 2011. His research interests include network and information security as well as applied cryptography. He is a senior member of the IEEE Computer Society. Since Aug. 2011, he has been on the editorial board of Journal of Network and Computer Applications published by Elsevier.

Li Song received her B.E. degree in Communication Engineering from Jilin University, Jilin, China, in 2011 and M.S. degree in Computer Application from University of Chinese Academy of Sciences, Beijing, China, in 2014. She is currently working as a research assistant in the Institute of Information Engineering, Chinese Academy of Sciences. Her research interests include mobile security and certificate system.