International Journal for Modern Trends in Science and Technology
Volume: 03, Issue No: 02, February 2017
ISSN: 2455-3778
http://www.ijmtst.com

Robust Discretization Security Scheme against Spyware Using Sequence Selection of Graphical Password

D Naina[1] | Y K Sundara Krishna[2]

[1]PG Scholar, Department of CSE, Krishna University, Machilipatnam, Krishna Dt, Andhra Pradesh, India.
[2]Professor, Department of CSE, Krishna University, Machilipatnam, Krishna Dt, Andhra Pradesh, India.

To Cite this Article
D Naina and Y K Sundara Krishna, "Robust Discretization Security Scheme against Spyware Using Sequence Selection of Graphical Password", *International Journal for Modern Trends in Science and Technology*, Vol. 03, Issue 02, 2017, pp. 25-34.

ABSTRACT

Now a day's many protection primitives are based on hard mathematical problems our proposed work will be based on Click-based graphical password schemes require a user to click on a set of points on one or more presented background images. With Pass Points, users create a password by clicking five ordered points anywhere on the given image. To log in, users must correctly repeat the sequence of clicks, with each click falling within an acceptable tolerance of the original point. To implement this aspect, along with a scheme converting the user entered graphical password into a cryptographic verification key, a "robust discretization" scheme. It consisted of three overlapping grids (invisible to the user) used to determine whether the click-points of a login attempt were close enough to the original points to be accepted.

KEYWORDS: *Graphical password, password, hotspots, CaRP, Captcha, dictionary attack, password guessing attack, security primitive.*

I. INTRODUCTION

The security and usability problems inherent in text-based password schemes have resulted in the development of graphical password schemes as a possible alternative. However, most of the current graphical password schemes are vulnerable to spyware which is a program that gathers information about a computer's use and relays that information back to a third party [1]. To date, there have been some schemes which have made contributions to the development of graphical password in term of spyware resistance [2, 3]. Using a challenge-response protocol, they have an advantage in that they are resistant to replay attacks. Namely, even the third party who observes a successful login session cannot perform a replay attack. Though they have a positive effect on protecting users' password, they are not yet sufficient to stop attackers from harvesting passwords. In this paper, CAPTCHA is used in a graphical password scheme to resist spyware. A

CAPTCHA (Completely Automated Public Turing tests to tell Computers and Humans Apart) is a program that generates and grades tests that are human solvable, but are beyond the capabilities of current computer programs [4]. CAPTCHA uses open algorithms based on hard AI problems, and has been discussed in text-based password schemes to resist dictionary attack [5]. Innovatively, we explore CAPTCHA in the context of graphical passwords to provide better protection against spyware. As long as the underlying open AI problems are not solved, CAPTCHA is a promising way to resist spyware attack in graphical password schemes. Based on this key idea, we have proposed a new graphical password scheme using CAPTCHA, designed to be

strongly resistant to spyware attack, either by purely automated software or via human participation. A preliminary user study indicates that our scheme needs to improve in terms of login time and memo ability. However, this new paradigm has achieved just a limited success as compared with the cryptographic primitives based on hard math problems and their wide applications. Is it possible to create any new security primitive based on hard AI problems?

This is a challenging and interesting open problem. In this paper, we introduce a new security primitive based on hard AI problems, namely, a novel family of graphical password systems integrating Captcha technology, which we call *CaRP (Captcha as graphical Passwords)*. CaRP is click-based graphical passwords, where a sequence of clicks on an image is used to derive a password. Unlike other click-based graphical passwords, images used in CaRP are Captcha challenges, and a new CaRP image is generated for every login attempt. The notion of CaRP is simple but generic. CaRP can have multiple instantiations. In theory, any Captcha scheme relying on multiple-object classification can be converted to a CaRP scheme. We present exemplary CaRPs built on both text Captcha and image-recognition Captcha. One of them is a text CaRP wherein a password is a sequence of characters like a text password, but entered by clicking the right character sequence on CaRP images. CaRP offers protection against online dictionary attacks on passwords, which have been for long time a major security threat for various online services. This threat is widespread and considered as a top cyber security risk [13]. Defence against online dictionary attacks is a more subtle problem than it might appear. Intuitive countermeasures such as throttling logon attempts do not work well for two reasons:

1) It causes denial-of-service attacks (which were exploited to lock highest bidders out in final minutes of eBay auctions [12]) and incurs expensive helpdesk costs for account reactivation.

2) It is vulnerable to global password attacks [14] whereby adversaries intend to break into any account rather than a specific one, and thus try each password candidate on multiple accounts and ensure that the number of trials on each account is below the threshold to avoid triggering account lockout. CaRP also offers protection against relay attacks, an increasing threat to bypass Captchas protection, wherein Captcha challenges are relayed to humans to solve.

CaRP is robust to shoulder-surfing attacks if combined with dual-view technologies. Commonly, a spyware is a software that, from a user's perspective, covertly gathers information about a computer's use and relays that information back to a third party [1]. Spyware has gradually become one of the most common security threats to computer systems. Password collection by spywares has rapidly increased [4, 5, 12, 13, 15]. The research community has expended much effort [4, 6, 7, 8,] on this topic. However, how to protect passwords effectively against spyware attack continues to be a problem. Observing that a practical spyware attack is done by an automated program, we propose a new approach where CAPTCHA is exploited. CAPTCHA (Completely Automated Public Turing tests to tell Computers and Humans Apart) is a program that generates and grades tests that are human solvable, but beyond the capabilities of current computer programs [7]. The robustness of CAPTCHA is found in its strength in resisting automatic adversarial attacks, automatic adversarial attacks, and it has many applications for practical security, including online polls, free email services, search engine bots, worms and spam, and preventing dictionary attacks [7]. Our proposal creates an innovative use of CAPTCHA in the context of graphical passwords to provide better password protection against spyware attacks. In this paper, we have proposed a new authentication scheme combining graphical passwords with text-based CAPTCHA. The scheme is easy for humans but makes it almost impossible for automated programs to harvest passwords. The novel scheme is friendly for legitimate users, while simultaneously raising the time and computer capacity cost to adversaries by several orders of magnitude. Experiments showed its effectiveness, but also indicated further research would improve its usability. The rest of the paper is organized as follows. Section 2 briefly reviews related work. Sections 3 and 4 present our scheme and analyses its security. Section 6 provides the results of experiments described in section 5. Section 7 discusses additional observations and possible extension to our scheme. Conclusions and future work are addressed in section 8.

II. RELATED WORKS

Most current graphical password schemes, such as [7, 11, 13, 14, 15], require users to enter the password directly, typically by clicking or drawing. Hence, passwords are easily exposed to a third party who has the opportunity to record a

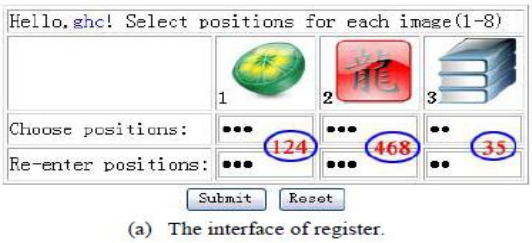
successful authentication session. There have been a few graphical password schemes devoted to secure passwords against spyware attacks. In the following, several representatives will be described. Man, et al [2] proposed that users remember a number of text strings as well as several images as pass-objects. To pass the authentication, users should enter the unique codes corresponding to the displayed pass-object variants and a code indicating the relative location of the pass-objects in reference to a pair of eyes. It is relatively hard to crack this kind of password, but the complex memory requirement is an obstacle to its popularity. In [6], users need to recognize pass objects and click inside the convex hull formed by all of the pass-objects. If properly designed, this method can provide good security. However, from time to time the convex hull is either too small to click or too large, creating a guessing problem. Moreover, to provide a large password space may result in a crowed screen and indistinguishable objects. The method in [2] to resist shoulder-surfing is a trivial trick, where a user must click a group composed of both the pass object and decoy-object rather than click the pass objects directly. The prototype presented in [2] does not provide sufficient security, having only two objects in each group. In 2006, Winchell proposed another challenge-response protocol that relied on a shared secret set of pictures [8]. To reduce the amount of information given out with each authentication session, the image set memberships are used to select a certain path on an image mosaic, with the user providing only a code that depends on the path's endpoint. This scheme was claimed to be so strong that an observer who fully records any feasible series of successful interactions could not compute the user's password. However, it was demonstrated by Golle and Wagner [9] that the attacker can learn a user's secret key with a SAT solver after observing as few as six successful user logins. In essence, the above methods adopt a challenge response protocol to confuse the spyware. They can prevent the passwords being cracked by the spyware and falling into the hand of an adversary, along with resisting replay attacks. Taking the previous mechanisms for reference, our scheme also uses a challenge-response protocol to enhance security. But, unlike these methods, our scheme innovatively applies CAPTCHA to graphical passwords to create a highly secure authentication method. Captcha is used to protect sensitive user inputs on an untrusted client [15]. This scheme protects the communication channel between user and Web server from keyloggers and spyware, while CaRP is a family of graphical password schemes for user authentication.

Figure1. The interface of the basic scheme (The pass-images are circled).

The paper [15] did not introduce the notion of CaRP or explore its rich properties and the design space of a variety of CaRP instantiations.

III. OUR SCHEME

Our approach is motivated by the observation that effective spyware attacks are launched from automated programs. We realized that to increase security passwords should be accompanied by a product of a "computation" that is difficult for machines. As an authentication method, the scheme should also be user friendly. Considering these requirements, we applied CAPTCHA to graphical password schemes. CAPTCHA is a program designed to test whether the user is a computer or a human, by creating a task easy for humans but difficult for machines [7]. It is based on hard AI problems which cannot be solved with any greater accuracy than what is currently known to the AI community [3]. CAPTCHA is now almost a standard security mechanism for addressing undesirable or malicious Internet bot programs [8] and major web sites such as Google, Yahoo and Microsoft all have their own CAPTCHAs. The state-of-the-art CAPTCHAs mainly include three types: text-based schemes, sound-based schemes and image-based schemes. The most widely deployed schemes are text-based CAPTCHAs and we also use this in our schemes



(a) The interface of register.

Figure2. Interface Register

After introducing a basic scheme with a hidden safety loophole, we will describe an improved

scheme that is designed to fill the hole. The performances of the both schemes depend extremely on the property of CAPTCHA.

A. The Basic Scheme

The basic scheme embeds a text-based CAPTCHA into a simple graphical password scheme. Each image has a CAPTCHA instance called adjunctive string and the strings are generated at random by the system. In the register phase, users are required to select and remember images as their password images (pass-images). To be authenticated, users need to distinguish his/her pass-images as well as solve a test by recognizing and typing the adjunctive string below each pass image. For example, in Figure 1, assume the three images with red circles are pass-images, users should input the adjunctive strings 'mewo', 'xnco' and 'nvso' correctly to pass the authentication. For simplicity, we assume that the CAPTCHA here is an ideal CAPTCHA that is hard enough for machines to recognize while easy for humans to solve. In the case that adversaries are automated programs without human intervention, the scheme has a strong resistance to replay attack. Namely, even if it observes a successful login, a spyware program cannot launch a replay attack. This can be illustrated from two aspects. Firstly, pass-images are entered by typing random adjunctive strings rather than clicking directly. In other words, the entered strings are the trap instead of the real password. Secondly, machines have no ability to recognize the characters embedded in each image. It follows that it is rather difficult for an automated program to find pass images according to the recorded strings. The loophole in this scheme occurs if the adversary is a person and the spyware is an assistant. The password will be in danger because CAPTCHA is easy for a person. In this case, the person can see what the spyware has gathered, a successful login scene along with the entered characters. Then, a person can crack the passwords without much effort. For 26 lower case letters in the scheme, the probability that different images have the same string is 1/456976, which can be ignored. One useful method for password cracking is to divide the gathered strings with four characters into groups and then compare each segment with that below each image. To close this loop hole, we constructed an improved version.

B. The Improved Scheme

The vulnerability of the basic scheme lies in two factors. One is the requirement that CAPTCHAs should be human user friendly. The other is the reversible relationship between passwords and what is entered. That is, pass-images determine what is entered and vice versa. What's more, we noted that the reversible relationship depends greatly on the fact that the probability of different images with the same adjunctive string is close to zero and that the trap of each pass-image has a uniform length. While the former is necessary for a popular authentication scheme, we are encouraged to disturb the latter. One possible method is increasing the probability by decreasing the types of letters or the length of adjunctive string. This method might work, but it will increase the probability of illegal login by random guessing. Thereby, it is ineffective as a security method. Our alternative is to replace the uniform length with a random one predefined by users. In other words, the number of characters entered is determined by users. In our improved scheme, users are required to select and remember letter positions, ie choose several specific letter positions within a string of letters; for example, letters in 1st, 4th and 5th position in the string will become the code. These letter positions are the called pass-positions for each pass-image. During the authentication, users should enter the characters shown in the pass-positions of each pass-image. An example is shown in Figure 2. In Figure 2(b), the three circled images are passimages, the strings with them are 'qarwrxex', 'heeqseio', and 'mvgqqebh' respectively, and the corresponding pass-positions are (1, 2, 4), (4, 6, 8), and (3, 5) shown in Figure 2 (a). A user can input any combination of the three sequences, 'qaw', 'qeo', and 'gq' to be authenticated successfully. This scheme is strongly resistant to attacks launched by humans with spyware, while simultaneously preserving the advantages of graphical password schemes. The related security analysis will be given in the following section and usability problems will be discussed in Section 5, 6 and 7 through experiments.

IV. SECURITY ANALYSIS OF THE IMPROVED SCHEME

A.Capability to Withstand Spyware

There are many different kinds of spyware [1, 2], such as browser hijackers, key loggers and spy bots. We have focused on the spyware cluster that runs in the background collecting passwords. The security of our scheme relies on the robustness of CAPTCHA in resisting automatic adversarial attacks. However, it is not clear whether there is a true CAPTCHA at all and some reports show that some text-based CAPTCHAs can be partly or

almost broken by automatic programs [3, 4, and 5]. With the assumption that spyware is capable of detecting and recording screen snapshots, entered strings and the system feedback, we will analyze the security of the improved scheme from two extreme aspects. Firstly, it is impossible for machines to solve the CAPTCHAs in our scheme, the ideal case. Secondly, CAPTCHAs can be completely solved by machines, the worst case. Under ideal conditions, spywares have no chance of gaining the passwords without human invention, similar to the discussion in sections 3.1. If people are involved, spyware assistance can help users to break the scheme. What the spyware needs to do s to catch the password string entered by the legal user. To crack passwords, adversaries should solve the CAPTCHA himself or by employing human workers. It is costly to obtain a password because the pass-positions of each pass-image are unknown and thereby it is hard to manually find the correspondence between pass-images and what is entered. Even for the lowest level security, adversaries must recognize 400 CAPTCHAs. In this case, there are three pass-images, each with a pass-position and then the attacker can easily divide the entered string into three segments each with a specific character. The probability of a letter displayed below one image is 0.27 26 100 images on screen in our scheme with about 27 images which have a common specific character. That is, there are 27 candidates including a pass-image and 26 decoys. Through interaction, the attacker can gradually get rid of all the decoys. For the second observation, third observation, there will only be about three CAPTCHAs which contain the specific character. The attacker can find the users passwords correctly in four sessions. So the attacker must solve approximately 400 CAPTCHAs and conduct many observations and comparisons, which is time consuming and costly. More complex work is required if the correspondence between passim ages and entered strings are unknown. Therefore, our scheme has a strong resistance against spywares under the ideal environment. Projecting the worst condition, that CAPTCHAs can be completely solved by machines, it is possible that spywares could crack passwords because each successful login reveals some information about the password. One method is to divide the entered strings into different segments and find the passwords from images which contain the same segments from analyzing different login sessions. Another method is to find the common images by excluding images without any character of the entered string. For instance, when the passwords lie in the lowest security level, it is possible to crack the passwords in four sessions, as discussed above. This worst case scenario is not probable, unless spywares can gather sufficient information in the background and can break CAPTCHAs quickly. Currently, no programs can break a CAPTCHA automatically in a short time. Furthermore, even if the currently applied CAPTCHAs are effectively broken, there will always be versions with higher security in production. In addition, as long as the hard AI problems underlying CAPTCHA are unsolved, successful attacks will advance the development of more robust CAPTCHAs. Therefore, it is demonstrated that our scheme is secure against spyware as long as CAPTCHAs cannot be broken by automated programs. Any defeated CAPTCHAs will be substituted by more robust ones. If humans are involved, the cost of cracking a password is significantly increased.

B.Automatic Online Guessing Attacks

In automatic online guessing attacks, the trial and error process is executed automatically whereas dictionaries can be constructed manually. If we ignore negligible probabilities, CaRP with underlying CPA-secure Captcha has the following properties: 1. internal object-points on one CaRP image are computationally-independent of internal object points on another CaRP image. Particularly, clickable points on one image are computationally independent of clickable points on another image. 2. Eq. (3) holds, i.e., trials in guessing attacks are mutually independent. The first property can be proved by contradiction. Assume that the property does not hold, i.e., there exists an internal objectpoint a on one image A that is non-negligibly dependent of an internal object-point β on another image B. An adversary can exploit this dependency to launch the following chosen-pixel attack. In the learning phase, image A is used to learn the object that contains point a. In the testing phase, point β on image B is used to query the oracle. Since point a is non-negligibly dependent of point β, this CPAexperiment would result in a success probability non negligibly higher than a random guess, which contradicts the CPA-secure assumption. We conclude that the first property holds. The second property is a consequence of the first property since user-clicked internal object-points in one trial are computationally-independent of user-clicked internal object-points in another trial due to the first property. We have ignored background and

boundary object-points since clicking any of them would lead to authentication failure. Eq. (3) indicates that automatic online guessing attacks can find a password only probabilistically no matter how many trials are executed. Even if the password guess to be tested in a trial is the actual password, the trial has a slim chance to succeed since a machine cannot recognize the objects in the CaRP image to input the password correctly. This is a great contrast to automatic online guessing attacks on existing graphical passwords which are deterministic, i.e., that each trial in a guessing attack can always determine if the tested password guess is the actual password or not, and all the password guesses can be determined by a limited number of trials. Particularly, brute-force attacks or dictionary attacks with the targeted password in the dictionary would always succeed in attacking existing graphical passwords.

C.Relay Attacks

Relay attacks may be executed in several ways. Captcha challenges can be relayed to a high volume Website hacked or controlled by adversaries to have human surfers solve the challenges in order to continue surfing the Website, or relayed to sweatshops where humans are hired to solve Captcha challenges for small payments. Is CaRP vulnerable to relay attacks? We make the same assumption as Van Oorschot and Stubblebine [5] in discussing CbPA-protocol's robustness to relay attacks: a person will not deliberately participate in relay attacks unless paid for the task. The task to perform and the image used in CaRP are very different from those used to solve a Captcha challenge. This noticeable difference makes it hard for a person to mistakenly help test a password guess by attempting to solve a Captcha challenge. Therefore it would be unlikely to get a large number of unwitting people to mount human guessing attacks on CaRP. In addition, human input obtained by performing a Captcha task on a CaRP image is useless for testing a password guess. If sweatshops are hired to mount human guessing attack, we can make a rough estimation of the cost. We assume that the cost to click one password on a CaRP image is the same as solving a Captcha challenge. Using the lowest retail price, $1, reported [3,4] to solve 1000 Captcha challenges, the average cost to break a 26-bit password is 0.5 · 226 · 1/1000, or about 33.6 thousand US dollars.

D.Shoulder-Surfing Attacks

Shoulder-surfing attacks are a threat when graphical passwords are entered in a public place such as bank ATM machines. CaRP is not robust to shoulder-surfing attacks by itself. However, combined with the following dual-view technology, CaRP can thwart shoulder-surfing attacks. By exploiting the technical limitation that commonly used LCDs show varying brightness and color depending on the viewing angle, the dual-view technology can use software alone to display two images on a LCD screen concurrently, one public image viewable at most view-angles, and the other private image viewable only at a specific viewangle [8]. When a CaRP image is displayed as the "private" image by the dual-view system, a shoulder-surfing attacker can capture user clicked points on the screen, but cannot capture the "private" CaRP image that only the user can see. However, the obtained user-clicked points are useless for another login attempt, where a new, computationally-independent image will be used and thus the captured points will not represent the correct password on the new image anymore. To the contrary, common implementations of graphical password schemes such as Pass Points use a static input image in the same location of the screen for each login attempt. Although this image can be hidden as the private image by the dual-view technology from being captured by a shoulder surfer, the user-clicked points captured in a successful login are still the valid password for next
login attempt. That is, capturing the points alone is sufficient for an effective attack in this case. In general, the higher the correlation of user clicked points between different login attempts is, the less effective protection the dual-view technology would provide to thwart shoulder surfing attacks.

V. RECOGNITION-RECALL CaRP

In recognition-recall CaRP, a password is a sequence of some invariant points of objects. An *invariant point* of an object (e.g. letter "A") is a point that has a fixed relative position in different incarnations (e.g., fonts) of the object, and thus can be uniquely identified by humans no matter how the object appears in CaRP images. To enter a password, a user must identify the objects in a CaRP image, and then use the identified objects as cues to locate and click the invariant points matching her password. Each password point has a tolerance range that a click within the tolerance range is acceptable as the password point. Most people have a click variation of 3 pixels or less [8]. Text Point, a recognition recall CaRP scheme with an alphabet of characters, is presented next,

followed by a variation for challenge response authentication.

A. Text Points

Characters contain invariant points. Fig. 5 shows some invariant points of letter "A", which offers a strong cue to memorize and locate its invariant points. A point is said to be an *internal point* of an object if its distance to the closest boundary of the object exceeds a threshold. A set of internal invariant points of characters is selected to form a set of *clickable points* for Text Points. The internality ensures that a clickable point is unlikely occluded by a neigh boring character and that its tolerance region unlikely overlaps with any tolerance region of a neigh boring character's clickable points on the image generated by the underlying Captcha engine. In determining clickable points, the distance between any pair of clickable points in a character must exceed a threshold so that they are perceptually distinguishable and their tolerance regions do not overlap on CaRP images. In addition, variation should also be taken into consideration.

Fig.3. some invariant points (red crosses) of "A".

For example, if the center of a stroke segment in one character is selected, we should avoid selecting the center of a similar stroke segment in another character. Instead, we should select Fig3. Some invariant points (red crosses) of "A". a different point from the stroke segment, e.g., a point at onethird length of the stroke segment to an end. This variation in selecting clickable points ensures that a clickable point is context-dependent: a similarly structured point may or may not be a clickable point, depending on the character that the point lies in. Character recognition is required in locating clickable points on a Text Points image although the clickable points are known for each character. This is a task beyond a bot's capability. A password is a sequence of clickable points. A character can typically contribute multiple clickable points. Therefore Text Points has a much larger password space than Click Text.

B.Image Generation:

Text Points images look identical to Click Text images and are generated in the same way except that the locations of all the clickable points are checked to ensure that none of them is occluded or its tolerance region overlaps another clickable point's. We simply generate another image if the check fails. As such failures occur rarely due to the fact that clickable points are all internal points; the restriction due to the check has a negligible impact on the security of generated images.

C.Authentication:

When creating a password, all clickable points are marked on corresponding characters in a CaRP image for a user to select. During authentication, the user first identifies her chosen characters, and clicks the password points on the right characters. The authentication server maps each user-clicked point on the image to find the closest clickable point. If their distance exceeds a tolerable range, login fails. Otherwise a sequence of clickable points is recovered, and its hash value is computed to compare with the stored value. It is worth comparing potential password points between Text Points and traditional click-based graphical passwords such as Pass Points [5]. In Pass Points, salient points should be avoided since they are readily picked up by adversaries to mount dictionary attacks, but avoiding salient points would increase the burden to remember a password. This conflict does not exist in Text Points. Clickable points in Text Points are salient points of their characters and thus help remember a password, but cannot be exploited by bots since they are both *dynamic* (as compared to static points in traditional graphical password schemes) and *contextual*:

Dynamic:

Locations of clickable points and their contexts (i.e., characters) vary from one image to another. The clickable points in one image are computationally independent of the clickable points in another image, as we will see in Section VI-B.

Contextual:

Whether a similarly structured point is a clickable point or not depends on its context. It is only if within the right context, i.e., at the right location of a right character these two features require recognizing the correct contexts, i.e., characters, first. By the very nature of Captcha, recognizing characters in a Captcha image is a task beyond computer's capability. Therefore, thesesalient points of characters cannot be exploited to mount dictionary attacks on Text Points.

C.TextPoints4CR

For the CaRP schemes presented up to now, the coordinates of user-clicked points are sent directly to the authentication server during authentication.

For more complex protocols, say a challenge response authentication protocol, a response is sent to the authentication server instead. Text Points can be modified to fit challenge-responseauthentication. This variation is called *Text Points for Challenge-Response* or *TextPoints4CR*. Unlike Text Points wherein the authentication server stores a salt and a password hash value for each account, the server in TextPoints4CR stores the password for each account. Another difference is that each character appears only once in a TextPoints4CR image but may appear multiple times in a Text Points image. This is because both server and client in TextPoints4CR should generate the same sequence of discredited grid-cells independently. That requires a unique way to generate the sequence from the shared secret, i.e., password. Repeated characters would lead to several possible sequences for the same password. This unique sequence is used as if the shared secret in a conventional challenge response authentication protocol. In TextPoints4CR, an image is partitioned into a fixed grid with the discretization grid-cell of size μ along both directions. The minimal distance between any pair of clickable points should be larger than μ by a margin exceeding a threshold to prevent two clickable points from falling into a single grid-cell in an image. Suppose that a guaranteed tolerance of click errors along both xaxis and y-axis is τ, we require that $\mu \geq 4\tau$.

Image Generation:

To generate a TextPoints4CR image, the same procedure to generate a Text Points image is applied. Then the following procedure is applied to make every clickable point at least τ distance from the edges of the grid-cell it lies in. All the clickable points, denoted as set, are located on the image. For every point in, we calculate its distance along x-axis or y-axis to the center of the grid-cell it lies in. A point is said to be an *internal point* if the distance is less than $0.5\mu-\tau$ along both directions; otherwise a *boundary point*. For each boundary point in, a nearby internal point in the same grid-cell is selected. The selected point is called a *target point* of the boundary point. After processing all the points in, we obtain a new set comprising internal points; these are either internal clickable points or target points of boundary clickable points. Mesh warping [6], widely used in generating text Captcha challenges, is then used to warp the image so that maps to. The result is a TextPoint4CR image wherein every clickable point would tolerate at least τ of click errors. Selection of target points should try to reduce warping distortion caused by mapping to.

Authentication

In entering a password, a user-clicked point is replaced by the grid-cell it lies in. If click errors are within τ, each user-clicked point falls into the same grid-cell as the original password point. Therefore the sequence of grid-cells generated from user clicked points is identical to the one that the authentication server generates from the stored password of the account. This sequence is used as if the shared secret between the two parties in a challenge-response authentication protocol. Unlike other CaRP schemes presented in this paper, Text Points4CR requires the authentication server to store passwords instead of their hash values. Stored passwords must be protected from insider attacks; for example, they are encrypted with a master key that only the authentication server knows. A password is decrypted only when its associated account attempts to log in.

VI. EXPERIMENTAL ENTHODOLOGY

During the testing phase, fifty images of 60×60cpixels and Corresponding CAPTCHAs were displayed on the screen in the prototype of the improved scheme. All the images were downloaded from http://www.chinaz.com freeware website and processed for study only. The length of CAPTCHA strings was 8, and the characters contained 26 lowercase letters. The CAPTCHA algorithm was designed to generate crowded, distorted and rugged strings similar to the CAPTCHA being used in Google email service for its acknowledged robustness. A total of 36 participants were invited to complete the experiments and answer some questions. The participants, of whom there were 15 women and 21 men, were staff and students from a university community and unfamiliar with our scheme. The average age of the participants was 27 years (Std Dev=4.5), and ranged from 21 to 39 years. All the participants were required to complete the following operations individually. Firstly, they need answer a demographic questionnaire, which collected information including age, sex, highest degree earned and computer experience. At this session the scheme and procedures for the experiments were explained to them in detail. Secondly, the user was required to select three or more pass-images. After selecting the pass-images, the user set the pass-positions for each image. During the testing phase, if the participants forgot the pass-images or the pass positions, the password which they have just set

was shown to them. In the testing phase, the data were collected longitudinally: first, at end of the training session (P1), then one week later (P2), and finally one month later (P3). For P1, each participant was asked to set a password, and authenticate ten times. For P2 and P3, if a participant entered an incorrect password, he or she was allowed to re-enter the password. Three login attempts were permitted for each participant.

VII. RESULTS

A. The Mean Success Login Percentage

In P1 testing session, 9 of 36 participants completed with no mistakes in ten times of login, while the others, to a greater or less extent, made some incorrect submissions. The mean success login percentage is 87.8% (StdDev=9.29). The reasons offered by the participants for the incorrect submissions included difficulty in identifying the text-based CAPTCHAs generated by our algorithms and sometimes in locating the exact pass-positions.

B. The Mean Login Time

P1 testing session, the mean login time of all participants is 22.04 seconds (StdDev=10.9) which is acceptable for most participants. The results show that there is a significant difference in terms of time to respond to a challenge ($F(35,280) = 15.48$, $p<0.01$). The main reason may be that the CAPTCHAs are randomly generated so that sometimes they are easy to recognize but sometimes more difficult. As the images are randomly located, the time for recognition also differs. Results show that the majority of participants chose three to five pass-images, with only three participants choosing more than five pass-images. Mean times and standard deviations of logins with different pass-images are.

C. Password Memo ability

In P2 testing session, 80.6 percent of participants successfully logged into his/her account in three attempts, and in P3 session, 72.2 percent participants were successful. Interviews with participants provided the following reasons for memory lapses: a) the difficulty of remembering the pass positions and b) the difficulty of remembering the relationships between pass-positions and pass-images.

VIII. DISCUSSION

In comparison to other graphical password schemes, such as [7,8,12], there are some advantages and disadvantages in our improved scheme. One disadvantage is that it is more complex and increases users' memory load. Users have to remember both the pass-images and pass positions. To be authenticated, users need to recognize the pass-images and input the characters of the text-based CAPTCHAs on the pass positions correctly. These factors have increased the complexity of the login process. However, although it is complex and cumbersome, the improved scheme is strongly resistant to spywares, which is our primary focus. A comparison of login time for our scheme shows that, our scheme, as other graphical schemes, is longer than that of text based schemes. However, when compared to other graphical password schemes our login time is shorter. For instance, the mean login time of CHC is 72 seconds and Déjà vu is 27 to 32 seconds because there are multiple rounds of challenges in these schemes [9]. In [11], a typical entry takes over 3 minutes using a high-complexity protocol and over 1.5 minutes with a low-complexity protocol. Moreover, schemes against spywares [1, 2] also challenge user's memory capacity to a great extent. In [18], the high-complexity protocol asks the user to remember 30 pictures. And in [2], the user needs to remember 16 random strings for corresponding 16 pass images. The mean login time of our improved scheme is 22.04 seconds. We believe that our login times will decrease with familiarity with the scheme. All experiments were undertaken in lab and all the participants were new to our scheme. The users' login speed should be faster with the extended use. If the scheme is moved to real usage, the settings of the parameters can be adjusted to adapt to different security demands and application situations. There are M images randomly generated including N pass images, and there are S rounds of challenges for one login. For each round, m images are displayed with n pass-images. With the increasing of the numbers of total images, pass-images and length of text-based CATPCHAs, the security of the scheme will be enhanced. For example, when $M = 250$ and $N = 10$, the spyware will detect 10 pass-images and the corresponding pass-positions for 250 images. This requires recording of hundreds of logins and recognition of a huge number of CAPTHCAs. Gathering so much information may take a long time and recognizing the CAPTCHAs also needs an extensive manpower. Certainly, increasing the setting for high security is at the expense of usability. There are also some user behaviors which create risks for our scheme. First, the passwords selected by user often accord with a

particular trend. For example, in order to make the password easily remembered, most users select the same position for different pass-images, first or anterior positions, consecutive positions or one position for each pass image. And certain images were selected by a number of users as pass images. All the factors mentioned above can reduce the practical password space and increase the possibility of "guessing" attacks. Second, we find that there is always a significant time gap when entering characters belonging to two different pass images. The reason is that users are used to enter corresponding characters after he finds a pass-image. Such a situation will be recorded and utilized by spywares. This problem can be solved by entering characters by turns which belong to different CATPCHAs in a certain order. In summary, our improved scheme is resistant to spyware attack, and the rules for setting passwords have increased the cost and time of the human intervention attack.

IX. CONCLUSION AND FUTURE WORKS

In this paper, we have presented a new approach to protect user's password against spyware attack. Our main contribution is that we introduce CAPTCHA into the realm of graphical passwords to resist spyware programs. From a security viewpoint, this exploration is expected to advance the development of graphical passwords. While the design of CAPTCHA is an interdisciplinary topic and the current collective understanding of this topic is still in its infancy, we do not claim that our scheme is immediately feasible. However, we believe that our method will enhance current security and as CAPTCHA increases in effectiveness our method will also increase computer security. The results of our experiments show that the future research should concentrate on improving the login time and memo ability. Furthermore, when a user inputs the corresponding substrings which belong to different CAPTCHAs, the time gap is longer than the time between two characters in one substring. So a method for narrowing the time gap in the entering process and reduction of the impact of user's choice trend on security, provide other areas for future research.

REFERENCES

[1] R. Biddle, S. Chiasson, and P. C. van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Comput. Surveys*, vol. 44, no. 4, 2012.

[2] *The Science Behind Passfaces* [Online].Available:http://www.realuser.com/published/Scien ceBehindPassfaces.pdf

[3] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin, "The design and analysis of graphical passwords," in *Proc. 8th USENIX Security Symp.*, 1999, pp. 1–15.

[4] H. Tao and C. Adams, "Pass-Go: A proposal to improve the usability of graphical passwords," *Int. J. Netw. Security*, vol. 7, no. 2, pp. 273–292, 2008.

[5] S. Wiedenbeck, J. Waters, J. C. Birget, A. Brodskiy, and N. Memon, "PassPoints: Design and longitudinal evaluation of a graphical password system," *Int. J. HCI*, vol. 63, pp. 102–127, Jul.2005.

[6] P. C. van Oorschot and J. Thorpe, "On predictive models and user drawn graphical passwords," *ACM Trans. Inf. Syst. Security*, vol. 10, no. 4, pp. 1–33, 2008.

[7] K. Golofit, "Click passwords under investigation," in *Proc. ESORICS*, 2007, pp. 343–358.

[8] A. E. Dirik, N. Memon, and J.-C. Birget, "Modeling user choice in the pass points graphical password scheme," in *Proc. Symp. Usable Privacy Security*, 2007, pp. 20–28.

[9] J. Thorpe and P. C. van Oorschot, "Humanseeded attacks and exploiting hot spots in graphical passwords," in *Proc. USENIX Security*, 2007, pp. 103–118.

[10] P. C. van Oorschot, A. Salehi-Abari, and J. Thorpe, "Purely automated attacks on pass pointsstyle graphical passwords," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 3, pp. 393–405, Sep.2010.

[11] P. C. van Oorschot and J. Thorpe, "Exploiting predictability in click based graphical passwords," *J. Comput. Security*, vol. 19, no. 4, pp. 669–702, 2011.

[12] T. Wolverton. (2002, Mar. 26). *Hackers Attack eBay Accounts* [Online]. Available: http://www.zdnet.co.uk/news/networking/2002/03/ 26/hackers-attack-ebay-accounts-2107350/

[13] HP TippingPoint DVLabs, Vienna, Austria. (2010). *Top Cyber Security Risks Report, SANS Institute and Qualys Research Labs* [Online].Available: http://dvlabs.tippingpoint.com/toprisks2010

[14] B. Pinkas and T. Sander, "Securing passwords against dictionary attacks," in *Proc. ACM CCS*,2002, pp. 161–170.

[15] P. C. van Oorschot and S. Stubblebine, "On countering online dictionary attacks with login histories and humans-in-the-loop," *ACM Trans. Inf. Syst. Security*, vol. 9, no. 3, pp. 235–258, 2006.

[16] M. Alsaleh, M. Mannan, and P. C. van Oorschot, "Revisiting defenses against large-scale online password guessing attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 9, no. 1, pp. 128–141, Jan./Feb. 2012.