RESEARCH ARTICLE OPEN ACCESS

Captcha as Graphical Passwords (Security Primitive Based On Hard Ai Problems)

[1]. Valusani Shrenika, M.Tech student, [2]Mr.D.Uma Vishweshwar,
CMR Engineering College, Hyderabad
Asst.Professor, CMR Engineering College

Abstract
Many security primitives are based on hard mathematical problems. Using hard AI problems for security is emerging as an exciting new paradigm, but has been underexplored. In this paper, we present a new security primitive based on hard AI problems, namely, a novel family of graphical password systems built on top of Captcha technology, which we call Captcha as graphical passwords (CaRP). CaRP is both a Captcha and a graphical password scheme. CaRP addresses a number of security problems altogether, such as online guessing attacks, relay attacks, and, if combined with dual-view technologies, shoulder-surfing attacks. Notably, a CaRP password can be found only probabilistically by automatic online guessing attacks even if the password is in the search set. CaRP also offers a novel approach to address the well-known image hotspot problem in popular graphical password systems, such as PassPoints, that often leads to weak password choices. CaRP is not a panacea, but it offers reasonable security and usability and appears to fit well with some practical applications for improving online security.

INTRODUCTION

A FUNDAMENTAL task in security is to create cryptographic primitives based on hard mathematical problems that are computationally intractable. For example, the problem of integer factorization is fundamental to the RSA public-key cryptosystem and the Rabin encryption. The discrete logarithm problem is fundamental to the ElGamal encryption, the DiffieHellman key exchange, the Digital Signature Algorithm, the elliptic curve cryptography and so on. Using hard AI (Artificial Intelligence) problems for security, initially proposed in [17], is an exciting new paradigm. Under this paradigm, the most notable primitive invented is Captcha, which distinguishes human users from computers by presenting a challenge, i.e., a puzzle, beyond the capability of computers but easy for humans. Captcha is now a standard Internet security technique to protect online email and other services from being abused by bots.

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. Defense 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-serviceattacks (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. Koobface was a relay attack to bypass Facebook's Captcha in creating new accounts. CaRP is robust to shoulder-surfing attacks if combined with dual-view technologies.

BACKGROUND

A. Graphical Passwords

A large number of graphical password schemes have been proposed. They can be classified into three categories according to the task involved in memorizing and entering passwords:

recognition, recall, and cued recall. Each type will be briefly described here. More can be found in a recent review of graphical passwords [1].

A recognition-based scheme requires identifying among decoys the visual objects belonging to a password portfolio. A typical scheme is Passfaces [2] wherein a user selects a portfolio of faces from a database in creating a password.

During authentication, a panelof candidate faces is presented for the user to select the face belonging to her portfolio. This process is repeated several rounds, each round with a different panel. A successful login requires correct selection in each round. The set of images in a panel remains the same between logins, but their locations are permuted. Story [20] is similar to Passfaces but the images in the portfolio are ordered, and a user must identify her portfolio images in the correct order.

Cognitive Authentication [22] requires a user to generate a path through a panel of images as

follows: starting from the top-left image, moving down if the image is in her portfolio, or right otherwise. The user identifies among decoys the row or column label that the path ends. This process is repeated, eachtime with a different panel.

A successful login requires that the cumulative probability that correct answers were not entered by chance exceeds a threshold within a given number of rounds. A recall-based scheme requires a user to regenerate the same interaction result without cueing. Draw-A-Secret (DAS) [3] was the first recall-based scheme proposed. A user draws her password on a 2D grid. The system encodes the sequence of grid cells along the drawing path as a userdrawn password. Pass-Go [4] improves DAS's usability by encoding the grid intersection points rather than the grid cells.

BDAS [23] adds background images to DAS to encourage users to create more complex passwords.

In acued-recallscheme, an external cue is provided to help memorize and enter a password. PassPoints [5] is a widely studied click-based cued-recall scheme wherein a user clicks a sequence of points anywhere on an image in creating a password, and re-clicks the same sequence during authentication. Cued Click Points (CCP) [18] is similar to PassPoints but uses one image per click, with the next image selected by a deterministic function. Persuasive Cued Click Points (PCCP) [19] extends CCP by requiring a user to select a point inside a randomly positioned viewport when creating a password, resulting in more randomly distributed click-points in a password.

Among the three types, recognition is considered the easiest for human memory whereas pure recall is the hardest [1].

Recognition is typically the weakest in resisting guessing attacks. Many proposed recognition-based schemes practically have a password space in the range of 213 to 216 passwords [1].

A study [6] reported that a significant portion of passwords of DAS and Pass-Go [4] were successfully broken with guessing attacks using dictionaries of 231 to 241 entries, as compared to the full password space of 258 entries. Images contain hotspots [7], [8], i.e., spots likely selected in creating passwords.

Hotspots were exploited to mount successful guessing attacks on PassPoints [8]–[11]: a significant portion of passwords were broken with dictionaries of 226 to 235 entries, as compared to the full space of 243 passwords.

B. Captcha

Captcha relies on the gap of capabilities between humans and bots in solving certain hard AI problems. There are two types of visual Captcha: text Captcha and Image-Recognition Captcha (IRC). The former relies on character recognition while the latter relies on recognition of non-character objects. Security of text Captchas has been extensively studied [26]–[30]. The following principle has been established: text Captcha should rely on the difficulty of character segmentation, which is computationally expensive and combinatorial hard [30].

Machine recognition of non-character objects is far less capable than character recognition. IRCs rely on the difficulty of object identification or classification, possibly combined with the difficulty of object segmentation. Asirra [31] relies on binary object classification: a user is asked to identify all the cats from a panel of 12 images of cats and dogs. Security of IRCs has also been studied. Asirra was found to be susceptible to machine-learning attacks [24]. IRCs based on binary object classification or identification of one concrete type of objects are likely insecure [25]. Multi-label classification problems are considered much harder than binary classification problems.

Captcha can be circumvented through relay attacks whereby Captcha challenges are relayed to human solvers, whose answers are fed back to the targeted application.

C. Captcha in Authentication

It was introduced in [14] to use both Captcha and password in a user authentication protocol, which we callCaptcha-based Password Authentication (CbPA) protocol, to counter online dictionary attacks. The CbPA-protocol in [14] requires solving a Captcha challenge after inputting a valid pair of user ID and password unless a valid browser cookie is received. For an invalid pair of user ID and password, the user has a certain probability to solve a Captcha challenge before being denied access. An improved CbPA-protocol is proposed in [15] by storing cookies only on user-trusted machines and applying a Captcha challenge only when the number of failed login attempts for the account has exceeded a threshold. It is further improved in [16] by applying a small threshold for failed login attempts from unknown machines but a large threshold for failed attempts from known machines with a previous successful login within a given time frame. Captcha was also used with recognition-based graphical passwords to address spyware [40], [41], wherein a text Captcha is displayed below each image; a user locates her own pass-images from decoy images, and enters the characters at specific locations of the Captcha below each pass-image as her password during authentication. These specific locations were selected for each pass-image during password creation as a part of the password.

In the above schemes, Captcha is an independent entity, used together with a text or graphical password. On the contrary, a CaRP is both a Captcha and a graphical password scheme, which are intrinsically combined into a single entity.

CAPTCHA AS GRAPHICAL PASSWORDS

A. A New Way to Thwart Guessing Attacks

In a guessing attack, a password guess tested in an unsuccessful trial is determined wrong and excluded from subsequent trials. The number of undetermined password guesses decreases with more trials, leading to a better chance of finding the password. Mathematically, let Sbe the set of password guesses before any trial,ρbe the password to find,T denote a trial whereas T_n denote the$n$-th trial, and $p(T = \rho)$be the probability that $\rho$is tested in trial T. Let E_n be the set of password guesses tested in trials up to (including) T_n. The password guess to be tested in$n$-th trial$T_n$is from set $S \backslash E_{n-1}$, i.e., the relative complement of E_{n-1} in S. If $\rho \in S$, then we have $p(T=\rho|T_1=\rho,...,T_{n-1}=\rho)>p(T=\rho)$, (1) and $E_n \rightarrow S$ $p(T=\rho|T_1=\rho,...,T_{n-1}=\rho) \rightarrow 1$ with$n \rightarrow |S|$, (2) where$|S|$ denotes the cardinality of S. From Eq. (2), the password is always found within $|S|$ trials if it is in S; otherwise S is exhausted after $|S|$ trials. Each trial determines if the tested password guess is the actual password or not, and the trial's result is deterministic.

To counter guessing attacks, traditional approaches in designing graphical passwords aim at increasing the effective password space to make passwords harder to guess and thus require more trials. No matter how secure a graphical password scheme is, the password can always be found by a brute force attack.

In this paper, we distinguish two types of guessing attacks: automatic guessing attacks apply an automatic trial and error process but$S$can be manually constructed whereas human guessing attacksapply a manual trial and error process. CaRP adopts a completely different approach to counter automatic guessing attacks. It aims at realizing the following equation:

$p(T=\rho|T_1,...,T_{n-1})=p(T=\rho), \forall n$ (3)

in an automatic guessing attack. Eq. (3) means that each trial is computationally independent of other trials. Specifically, no matter how many trials executed previously, the chance of finding the password in the current trial always remains the same. That is, a password in S can be found only probabilistically by automatic guessing (including brute-force) attacks, in contrast to existing graphical password schemes where a password can be found within a fixed number of trials. How to achieve the goal? If a new image is used for eachtrial, and images of different trials are independent of each other, then Eq. (3) holds. Independent images among different login attempts must contain invariant information so that the authentication server can verify claimants. By examining the ecosystem of user authentication, we noticed that human users enter passwords during authentication, whereas the trial and error process in guessing attacks is executed automatically. The capability gap between humans and machines can be exploited to generate images so that they arecomputationallyindependent yet retain invariants that only humans can identify, and thus use as passwords. The invariants among images must be intractable to machines to thwart automatic guessing attacks. This requirement is the same as that of an ideal Captcha [25], leading to creation of CaRP, a new family of graphical passwords robust to online guessing attacks.

B. CaRP: An Overview

In CaRP, a new image is generated for

every login attempt, even for the same user. CaRP uses an alphabet of visual objects (e.g., alphanumerical characters, similar animals) to generate a CaRP image, which is also a Captcha challenge. A major difference between CaRP images and Captcha images is that all the visual objects in the alphabet should appear in a CaRP image to allow a user to input any password but not necessarily in a Captcha image. Many Captcha schemes can be converted to CaRP schemes, as described in the next subsection.

CaRP schemes are clicked-based graphical passwords. According to the memory tasks in memorizing and entering a password, CaRP schemes can be classified into two categories: recognition and a new category, recognition-recall, which requires recognizing an image and using the recognized objects as cues to enter a password. Recognition-recall combines the tasks of both recognition and cued-recall, and retains both the recognition-based advantage of being easy for human memory and the cued-recall advantage of a large password space. Exemplary CaRP schemes of each type will
be presented later.

C. Converting Captcha to CaRP

In principle, any visual Captcha scheme relying on recognizing two or more predefined types of objects can be converted to a CaRP. All text Captcha schemes and most IRCs meet this requirement. Those IRCs that rely on recognizing a single predefined type of objects can also be converted to CaRPs in general by adding more types of objects. In practice, conversion of a specific Captcha scheme to a CaRP scheme typically requires a case by case study, in order to ensure both security and usability. Some IRCs rely on identifying objects whose types are not predefined. A typical example is Cortcha [25] which relies on context-based object recognition wherein the object to be recognized can be of any type. These IRCs cannot be converted into CaRP since a set of pre-defined object types is essential for constructing a password.

D. User Authentication With CaRP Schemes

Like other graphical passwords, we assume that CaRP schemes are used with additional protection such as secure channels between clients and the authentication server through Transport Layer Security (TLS). A typical way to apply CaRP schemes in user authentication is as follows. The authentication server ASstores a salt s and a hash value H(ρ,s) for each user ID, whereρis the password of the account and not stored. A CaRP password is a sequence of visual object IDs or clickable-points of visual objects that the user selects. Upon receiving a login request, as generates a CaRP image, records the locations of the objects in the image, and sends the image to the user to click her password. The coordinates of the clicked points are recorded and sent to ASalong Fig. 1.Flowchart of basic CaRP authentication with the user ID.ASmaps the received coordinates onto the CaRP image, and recovers a sequence of visual object IDs or clickable points of visual objects,ρ , that the user clicked on the image. Then AS retrieves salts of the account, calculates the hash value of ρ with the salt, and compares the result with the hash value stored for the account. Authentication succeeds only if the two hash values match. This process is called the basic CaRP authentication and shown in Fig. 1.Advanced authentication with CaRP, for example, challenge-response, will be presented in Section V-B. We assume in the following that CaRP is used with the basic CaRP authentication unless explicitly stated otherwise. To recover a password successfully, each user-clicked point must belong to a single object or a clickable-point of an object. Objects in a CaRP image may overlap slightly with neighboring objects to resist segmentation. Users should not click inside an overlapping region to avoid ambiguity in identifying the clicked object. This is not a usability concern in practice since overlapping areas generally take a tiny portion of an object.

RECOGNITION-BASED CaRP:

For this type of CaRP, a password is a sequence of visual objects in the alphabet. Per view of traditional recognitionbased graphical passwords, recognition-based CaRP seems to have access to an infinite number of different visual objects. We present two recognition-based CaRP schemes and a variation next.

A. ClickText

ClickText is a recognition-based CaRP scheme built on top of text Captcha. Its alphabet comprises characters without any visually-confusing characters. For example, Letter “O” and digit “0” may cause confusion in CaRP images, and thus one character should be excluded from the alphabet. A ClickText password is a sequence of characters in the alphabet, e.g., ρ =“AB#9CD87”, which is similar to a text password. A ClickText image is generated by the underlying Captcha engine as if a Captcha image were generated except that all the alphabet characters should appear in the image. During generation, each character’s location is tracked to produce ground truth for the location of the character in the generated image. The authentication server relies on the ground truth to identify the characters corresponding to user-clicked points.

In ClickText images, characters can be arranged randomly

Fig. 2. A ClickText image with 33 characters.

Fig. 3. Captcha Zoo with horses circled red.

Fig. 4. A ClickAnimal image (left)

on 2D space.

This is different from text Captcha challenges in which characters are typically ordered from left to right in order for users to type them sequentially. Fig. 2 shows a ClickText image with an alphabet of 33 characters. In entering a password, the user clicks on this image the characters in her password, in the same order, for example “A”, “B”, “#”, “9”, “C”, “D”, “8”, and then “7” for passwordρ=“AB#9CD87”

B. ClickAnimal

Captcha Zoo [32] is a Captcha scheme which uses 3D models of horse and dog to generate 2D animals with different textures, colors, lightings and poses, and arranges them on a cluttered background. A user clicks all the horses in a challenge image to pass the test. Fig. 3 shows a sample challenge wherein all the horses are circled red. Click Animal is a recognition-based CaRP scheme built on top of Captcha Zoo [32], with an alphabet of similar animals such as dog, horse, pig, etc. Its password is a sequence of animal names such as ρ =“Turkey, Cat, Horse, Dog,….” For each animal, one or more 3D models are built. The Captcha generation process is applied to generate ClickAnimal images: 3D models are used to generate 2D animals by applying different views, textures, colors, lightning effects, and optionally distortions. The resulting 2D animals are then arranged on a cluttered background such as grassland. Some animals may be occluded by other animals in the image, but their core parts are not occluded in order for humans to identify each of them. Fig. 4 shows a ClickAnimal image with an alphabet of 10 animals. Note that different views applied in mapping 3D models to 2D animals, together with occlusion in the following step, produce many different shapes for the same animal’s instantiations in the generated images. Combined with the additional anti-recognition mechanisms applied in the mapping step, these make it hard for computers to recognize animals in the generated image, yet humans can easily identify different instantiations of animals.

C. AnimalGrid

The number of similar animals is much less than the number of available characters. ClickAnimal has a smaller alphabet, and thus a smaller password space, than ClickText. CaRP should have a sufficiently-large effective password space to resist human guessing attacks. Animal Grid’s password space can be increased by combining it with a grid-based graphical password, with the grid depending on the size of the selected animal. DAS [3] is a candidate but requires drawing on the grid. To be consistent with ClickAnimal, we change from drawing to clicking: Click-A-Secret (CAS)wherein a user clicks the grid cells in her password. Animal Grid is a combination of ClickAnimal and CAS. The number of grid-cells in a grid should be much larger than the alphabet size. Unlike DAS, grids in our CAS are object-dependent, as we will see next. It has the advantage that a correct animal should be clicked in order for the clicked grid-cell(s) on the follow-up grid to be correct. If a wrong animal is clicked, the follow-up grid is wrong. A click on the correctly labeled grid-cell of the wrong grid would likely produce a wrong grid-cell at the authentication server side when the correct grid is used. To enter a password, a ClickAnimal image is displayed first. After an animal is selected, an image ofn×ngrid appears, with the grid-cell size equaling the bounding rectangle of the selected animal. Each grid-cell is labeled to help users identify. A user can select zero to multiple grid-cells matching her password. Therefore a password is a sequence of animals interleaving with grid-cells. A password must begin with an animal. When a ClickAnimal image appears, the user clicks the animal on the image that matches the first animal in her password. The coordinates of the clicked point are recorded. The user checks the displayed rectangle and corrects inaccurate edges by dragging if needed. This process is repeated until the user is satisfied with the accuracy of the bounding rectangle. In most cases, the calculated bounding rectangle is accurate enough without needing manual correction. Once the bounding rectangle of the selected animal is identified, an image of n×ngrid with the identified bounding rectangle as its grid-cell size is generated and displayed. If the grid image is too large or too small for a user to view, the grid image is scaled to a fitting size. The user then clicks a sequence of zero to multiple grid-cells that match the grid- cells following the first animals in her password, and then gets back to the ClickAnimal image. For the example password ρ given previously, she clicks a point inside to select the two grid-cells.

The coordinates of user-clicked points on the grid image (the original one before scaling if the grid image is scaled) are recorded. The above process is repeated until the user has finished entering her password. The resulting sequence of coordinates of user-clicked points, Using the ground

truth, the server recovers the first animal from the received sequence, regenerates the grid image from the animal's bounding rectangle, and recovers the clicked grid-cells. This process is repeated to recover the password the user clicked. Its hash is then calculated and compared with
the stored hash.

BALANCE OF SECURITY AND USABILITY

Some configurations of CaRP offer acceptable usability across common device types, e.g. our usability studies used 400×400 images, which fit displays of smart phones, iPads, and PCs. While CaRP may take a similar time to enter a password as other graphical password schemes, it takes a longer time to enter a password than widely used text passwords. We discuss two approaches for balancing CaRP's security and usability.

A. Alphabet Size

Increasing alphabet size produces a larger password space, and thus is more secure, but also leads to more complex CaRP images. When the complexity of CaRP images gets beyond a certain point, humans may need a significant amount of time to recognize the characters in a CaRP image and may get frustrated. The optimal alphabet size for a CaRP scheme such as ClickText remains an open question.

B. Advanced Mechanisms

The CbPA-protocols described in Section II-C require a user to solve a Captcha challenge in addition to inputting a password under certain conditions. For example, the scheme described in [16] applies a Captcha challenge when the number of failed login attempts has reached a threshold for an account. A small threshold is applied for failed login attempts from unknown machines but a large threshold is applied for failed attempts from known machines on which a successful login occurred within a given time frame. This technique can be integrated into CaRP to enhance usability:

1. A regular CaRP image is applied when an account has reached a threshold of failed login attempts. As in [16], different thresholds are applied for logins from known and unknown machines.
2. Otherwise an "easy" CaRP image is applied. An "easy" CaRP image may take several forms depending on the application requirements. It can be an image generated by the underlying Captcha generator with less distortion or overlapping, a permuted "keypad" wherein undistorted visual objects (e.g. characters) are permuted, or even a regular "keypad" wherein each visual object (e.g., character) is always located at a fixed position. These different forms of "easy" CaRP images allow a system to adjust the level of difficulty to fit its needs. With such a modified CaRP, a user would always enter a password on an image for both cases listed above. No extra task is required. The only difference between the two cases is that a hard image is used in the first case whereas an easy image is used in the second case.

CONCLUSION AND FUTURE SCOPE

We have proposed CaRP, a new security primitive relying on unsolved hard AI problems. CaRP is both a Captcha and a graphical password scheme. The notion of CaRP introduces a new family of graphical passwords, which adopts a new approach to counter online guessing attacks: a new
CaRP image, which is also a Captcha challenge, is used for every login attempt to make trials of an online guessing attack computationally independent of each other. A password of CaRP can be found only probabilistically by automatic online guessing attacks including brute-force attacks, a desired security property that other graphical password schemes lack. Hotspots in CaRP images can no longer be exploited to mount automatic online guessing attacks, an inherent vulnerability in many graphical password systems. CaRP forces adversaries to resort to significantly less efficient and much more costly human-based attacks. In addition to offering protection from online guessing attacks, CaRP is also resistant to Captcha relay attacks, and, if combined with dual-view technologies, shoulder-surfing attacks. CaRP can also help reduce spam emails sent from a Web email service. Our usability study of two CaRP schemes we have implemented is encouraging. For example, more participants considered AnimalGrid and ClickText easier to use than PassPoints and a combination of text password and Captcha.

Both AnimalGrid and ClickText had better password memorability than the conventional text passwords. On the other hand, the usability of CaRP can be further improved by using images of different levels of difficulty based on the login history of the user and the machine used to log in. The optimal tradeoff between security and usability remains an open question for CaRP, and further studies are needed to refine CaRP for actual deployments. Like Captcha, CaRP utilizes unsolved AI problems. However, a password is much more valuable to attackers than a free email account that Captcha is typically used to protect. Therefore there are more incentives for attackers to hack CaRP than Captcha. That is, more efforts will be attracted to the following win-win game by CaRP than ordinary Captcha: If attackers succeed, they contribute to improving AI by providing solutions to open problems such as segmenting 2D texts. Otherwise, our system stays secure, contributing to practical security. As a framework, CaRP does not rely on any specific Captcha scheme. When one Captcha scheme is broken, a new and more secure one may appear and be converted to a CaRP scheme. Overall, our work is one step forward in the paradigm of using hard AI problems for security. Of reasonable security and usability and practical applications, CaRP has good potential for refinements, which call for useful future work. More importantly, we expect CaRP to inspire new inventions of such AI based security primitives.

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] (2012, Feb.). The Science Behind Passfaces [Online]. Available: http://www.realuser.com/published/ScienceBehindPassfaces.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 userdrawn 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 passpoints graphical password scheme," inProc. Symp. Usable Privacy Security, 2007, pp. 20–28.

[9] J. Thorpe and P. C. van Oorschot, "Human-seeded 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 passpoints-style 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 clickbase 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/