RESEARCH ARTICLE OPEN ACCESS

A New Security Captcha As Graphical Passwords On Hard AI Problems

M.Sneha Divya [1], Mr.P.Joshua Raju [2]

Dept. of CSE, Baba Institute of Technology and Sciences, Andhra Pradesh, India

ABSTRACT

Many security primitives are based on hard mathematical problems. Using hard AI problems for security is going onward as an exciting new paradigm, but has been under- researched. In this report, we present a new security primitive based on hard AI problems, namely, a novel family of graph- excel 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 system. Carp addresses a number of security problems altogether, such as online guessing attacks, replay 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 word is in the search set. CaRP also offers a novel plan of attack to address the well-known image hotspot problem in popular graphical password schemes, such as passports, that often leads to weak password choices. Cap is not a panacea, but it offers reasonable security and usability and appears to fit easily with some practical applications for improving online security.

Keywords:- Graphical password, password, hotspots, CaRP, Captcha, dictionary attack, password guessing attack, security primitive.

I. INTRODUCTION

A central task in security is to create crypto- graphic primitives based on hard mathematical problems that are computationally intractable. For instance, the problem of integer factorization is fundamental to the RSA public-key cryptosystem and the Rabin encryption. The discrete log problem is underlying to the ElGamal encryption, the Diffie- Hellman key exchange, the Digital Signature Algorithm, the elliptic curve cryptography and then on.

Using hard AI (Artificial Intelligence) problems for security, initially advised 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.

Is it possible to produce any new security primitive based on hard AI problems? This is a challenging and an interesting open problem. In this report, 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 applied to make a word.

The notion of CaRP is simple but generic. The carp can have multiple instantiations In theory, any Captcha scheme relying on multiple-object classification can be changed 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 word is a chronological sequence of occasions like a text password, but inserted by clicking the correct character sequence on CaRP images.

Carp offers protection against online dictionary attacks on passwords, which have been for a long time a major security threat for several online services. This threat is widespread and considered as a top cyber security risk [13]. Defense against online dictionary attacks is a more insidious problem than it might seem. 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 the final minutes of eBay auctions [12]) and incurs expensive help desk costs for account reactivation.
2) It is vulnerable to global password attacks [14] where by adversaries intend to break into any account rather than a specific one, and thus tries each password candidate on multiple accounts and ensure that the number of trials on each account is below the door to avoid triggering account lockout.

Carp also offers protection against relay attacks, an increase- in threat to bypass Captchas protection, wherein Captcha challenges are linked up to humankind to turn away. Carp is robust to shoulder-surfing attacks if combined with dual-view technologies.

Carp requires solving a Captcha challenge in every login. This impact on usability can be mitigated by adapting the CaRP image's difficulty level based on the login history of the explanation and the machine used to log in.

Typical application scenarios for CaRP include:

1) Carp can be applied on touchscreen devices wherein typing passwords are cumbersome. For secure Internet applications such as e-banks. Many e-banking systems have applied Captchas in user logins. For example, ICBC the largest bank in the world, requires solving a Captcha challenge for every online login attempt.
2) Cap increases spammer's operating cost and thus helps reduce spam emails. For an email service provider that deploys CaRP, a spam bot cannot log into an electronic mail account even if it recognizes the word. Instead, human involvement is compulsory to access an account.

The remaining paper is organized as follows:

Background and related work are presented in Section II. We outline CaRP in Section III, and exhibit a variety of CaRP schemes in Sections IV and V. Security analysis is provided in Section VI. A usability study on two CaRP schemes that we have ample- minted is reported in Section VII. Balance of security and usability is discussed in Section VIII. We resolve the paper with Section IX.

II. BACKGROUND AND RELATED WORK

A. Graphical Passwords

A large bit of graphical password schemes have been suggested. They can be split up into three categories according into the task involved in memorizing and entering passwords: recognition, recollection, and cued recall. Each role will be briefly described here. More can be found in a recent review of graphical passwords.

A recognition-based strategy requires identifying among decoys the visual objects belonging to a password portfolio. A typical scheme is Passfaces [2] where in a user supports a portfolio of faces from a database in creating a password. During certification, a panel of candidate faces is shown for the user to select the face belonging to her portfolio. This procedure is repeated several rounds, each round with a different jury. A successful login requires correct selection in each cycle. The set of images in a panel remains the same between logins, but their locations are permuted. Storey [20] is similar to Passfaces but the pictures in the portfolio are ordered, and a user must identify her portfolio images in the right order. Déjà Vu [21] is also similar but uses a thick set of information processing system- generated "random-art" images. Cognitive Authentication [22] requires a user to generate a path through a panel of icons as follows: starting from the top-left image, moving down if the image is in her portfolio, or right otherwise.

This process is duplicated, each time with a different jury. A successful login requires that the cumulative probability that correct responses were not read by chance exceeds a threshold within a committed number of cycles.

A recall-based strategy requires a user to reconstruct the same interaction result without cueing. Twirl-A-Secret (DAS) [3] was the first recall-based scheme proposed. A user finds out her parole on a 2D grid. The system encodes the sequence of grid cells along the drawing course as a user drawn password. Go-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 make more complex words.

In a cued-recall scheme, an external cue is offered to help memorize and enter a password. Passports [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 authenti- cation. Cued Click Points (CCP) [18] is similar to PassPoints but uses one image per click, with the next image selected by a deterministic design. Persuasive Cued Click Points (PCCP) [19] extends CCP by requiring a user to select a spot within a randomly positioned viewport when creating a password, resulting in more randomly distributed click-points in a word.

Among the three instances, recognition is considered the easiest for human memory, whereas pure recall is the hardest [1]. Identification is typically the weakest in resisting guessing attacks. Many proposed recognition-based schemes practically have a password space in the neighborhood of 213 to 216 passwords [1]. A study [6] reported that a significant percentage of passwords of DAS and Pass-Go [4] were successfully passed away with guessing attacks using dictionaries of 231 to 241 entries, as compared to the full password space of 258 entrances. The images contain hotspots [7], [8], i.e., Spots likely selected in creating passwords. Hotspots were exploited to mount successful guessing attacks on Pass Points [8] [11]: a significant portion of passwords was broken with dictionaries of 226 to 235 entries, as compared to the full space of 243 words.

B. Captcha

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

Machine recognition of non-character objects is far less capable than character recognition. IRCs relies on the difficulty of object identification or categorization, perhaps combined with the difficulty of object

segmentation. Sierra [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. We guarantee the security of your transactions IRCs has also been studied. Sierra was found to be susceptible to machine-learning attacks [24]. IRCs based on binary object classification or designation of one concrete type of objects are likely insecure [25]. Multi-label classification problems are considered a great deal harder than binary classification problems.

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

C. Captcha in Authentication

At trial, whereas *TN* denote the *n*-the trial, and $p\ (T\ \rho)$ be the probability that ρ is tested in trial *T*. Let E_n be the set of password guesses tested in trials up to (including) *TN*. The password guesses to be tested in *n*-the trial *than* is from set $S\ E_{n-1}$, i.e., the relative complement of $E_{n\ 1}$ in *S*. If $\rho\ S$, then we have

$$p\ (T = \rho|T1\ /= \rho, \ldots, Tn{-}1\ /= \rho) > p(T = \rho), \quad (1)$$

It was presented in [14] to use both Captcha and password in a user authentication protocol, which we call Captcha-based Password Authentication (CRPA) protocol, to counter online.

$$\Sigma \text{ with } n \rightarrow |S|, (2)$$

It was shown in [14] to use both Captcha and password Dictionary attacks are good. The CRPA-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 selects a certain probability to go out a Captcha challenge before being denied access. An improved CRPA-protocol is proposed in [15] by storing cookies only on user-trusted machines and using a Captcha challenge only when the number of failed login attempts for the bill has exceeded a threshold. It is further improved in [16] by applying a low threshold for failed login attempts from unknown machines, but a big threshold for failed attempts from is known machines with a previously successful login within a collapsed time frame.

Captcha was also used with acknowledgement-based graphical passwords to address spyware [40], [41], wherein a text Captcha is displayed below each icon; a user locates her own pass-images from decoy images, and sets down the fibers in specific locations of the Captcha below each pass-image as her password during authentication. These specific locations were selected for each flip-image during password creation as a portion of the news.

In the above schemes, Captcha is an autonomous 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.

D. Other Related Work

Captcha is used to protect sensitive user inputs on an untrusted client [35]. This arrangement 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. The report [35] did not bind with the notion of CaRP or explore its rich properties and the design space of a variety of CaRP instantiations.

III. 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 set wrong and excluded from subs-Quent trials. The number of undetermined password guesses decreases with more tests, leading to a better chance of receiving the news. Mathematically, let *S* be the set of password guesses before any trial, ρ be the password to find, *T* denote Where *S* denotes the cardinality of *S*. From Eq. (2), the watchword 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 imagine and therefore need more tests. L's. No matter how secure a graphical password scheme is, the password can always be found by a brute force attempt. In this report, we identify two cases of guessing attacks: automatic guessing attacks apply an automatic trial and error process, but the S can be manually constructed whereas human guessing attacks apply a manual test and error procedure. Carp adopts a totally different attack to counter, automatic guessing attacks. It aims at making the following equation:

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

In an automatic guessing attack. Tantamount. (3) Means that each trial is computationally independent of other tests. Specifically, no topic how many trials executed previously, the hazard 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 word can be put up within a limited number of tests. How can you get to your goal? If a new image is used for each trial, and images of the different tests 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 studying the ecosystem of user documentation, we observed that human users enter passwords during authentication, whereas the trial and error process in guessing attacks is done automatically. The capability gap between world and automobiles can be exploited to generate images so that they are computed independently 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 the creation of CaRP, a novel family of graphical passwords robust to online guessing attacks.

B. *Carp: An Overview*

In CARP, a new picture is brought forth for every login attempt, even for the same user. Carp uses an alphabet of visual targets (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 optical 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 reported in the succeeding segment.

Carp schemes are clicked-based graphical passwords. Agreeing to the memory tasks in memorizing and enter in a password, CaRP schemes can be separated into two categories: recognition and a new category, recognition-recall, which calls for picking out an image and using the known objects as cues to enter a word. Recognition recall combines the jobs of both recognition and cued-recall, and retains both the credit-based advantage of being comfortable for human memory and the cued-recall advantage of a large password space. Exemplary CaRP schemes of each event will be presented later.

C. *Converting Captcha to CaRP*

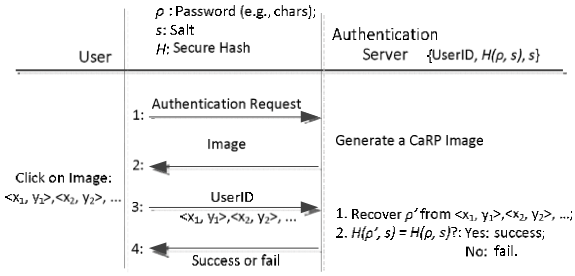

Fig. 1. Flowchart of basic CaRP authentication.

. This process is called the *basic CaRP authentication* and shown in Fig. 1.

Advanced authentication with CaRP, for lesson, challenge-response, will be established in Section V-B. We assume in the following that CaRP is used with the basic CaRP authentication unless explicitly stated otherwise.

To retrieve a password successfully, each user-clicked point in principle, any visual Captcha scheme relying on recognize- in two or more predefined types of targets can be shifted over to a CaRP. All text Captcha schemes and most IRCs meet this requirement. Those IRCs that rely on distinguishing a single predefined type of objects can also be converted to CaRPs in general by adding more types of objects. In the practice session, the reincarnation of a specific Captcha scheme to a CaRP scheme typically requires a case by case study, in lodge to ensure both security and usability. We will present in Sections IV and V several CaRPs built on top of text and image-recognition Captcha schemes.

Some IRCs relies on identifying objects whose types are not predefined. A typical example is Cortcha [25] which relies on context-based target recognition, wherein the object to be recognized can be of whatever type. These IRCs cannot be converted into CaRP since a band of pre-specified object types are entirely important for constructing a password.

D. *User Authentication With CaRP Schemes*

Like other graphical passwords, we assume that CaRP schemes are employed 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 authentica- tion server *AS* stores a salts and a hash value $H(\rho, s)$ for each user ID, where ρ is the password of the account and not stored. A CaRP password is a sequence of visual object IDs or clickable-points of optical objects that the user selects. Upon receiving a login request, AS generates a CaRP image, sets down the positions of the quarries in the image, and sends the image to the user to click her password. The coordinates of the clicked points are read and sent to AS along.

Must belong to a single object or a clickable-point of an object. Targets in a CaRP image may overlap slightly with neighboring objects to resist segmentation. Users should not descend into position in an overlapping region to avoid ambiguity in identifying the clicked object. This is not a usability concern in practice since overlapping areas generally require a tiny portion of an object.

IV. RECOGNITION-BASED CARP

For this type of CaRP, a word is a sequence of visual objects in the ABCs. Per view of traditional recognition-based graphical passwords, credit-based CaRP seems to receive admittance to an infinite number of different visual objects. We present two recognitions-based CaRP schemes and a variation next.

A. *Click Text*

Click Text 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 Click Text password is a chronological sequence of eccentrics in the first principle, e.g., ρ "AB#9CD87", which is standardized to a text password. A Click Text image is generated by the underlying Captcha engine as if a Captcha image was generated except that all the alphabet characters should appear in the image. During generation, each character's position is drawn over to produce ground truth for the emplacement of the character in the generated image. The authentication server relies on the ground truth to identify the components corresponding to user-clicked point. In Click Text, 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) and 6 6 grid (right) determined by red turkey's bounding rectangle.

A single stud in a 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 Click Text image with an ABC of 33 events. 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 password ρ = "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 puts them on a cluttered desktop. A user clicks all the horses in a challenge to broaden the image test. Fig. 3 presents a sample challenge wherein all the horses are circled red.

ClickAnimal is a credit-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 chronological sequence of animal names such as ρ "Turkey, Cat, Horse, Dog,…." For each animal, one or more 3D models are created. The Captcha generation process is applied to generate ClickAnimal images: 3D models are used to generate 2D animals by planting of different backgrounds, textures, colors, lightning effects, and optionally distortions. The resulting 2D animals are then put on a cluttered background such as grassland. Some animals may be blocked by other animals in the film, but their core ingredients are not blocked in order for mankind to identify each of them. Fig. 4 shows a ClickAnimal image with an ABC of 10 animals. Note that different positions used in mapping, 3D models to 2D animals, together with occlusion in the following step, produce many different configurations 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 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 usable fibers. ClickAnimal has a smaller alphabet, and therefore a smaller password space, then Click Text. The cop should hold a sufficiently-large effective password space to resist human guessing attacks. AnimalGrid's password space can be increased by mixing it with a grid-based graphical password, with the grid depending on the size of the chosen instrument.

DAS [3] is a chance, but requires drawing along the power system. To be consistent with ClickAnimal, we change from dragging to clicking: Click-A-Secret (CAS) wherein a user clicks the grid cells in her intelligence. *AnimalGrid* is a combination of ClickAnimal and CAS. The number of grid-cells in a grid should be a great deal bigger than the alphabet size. Unlike DAS, grids in our CAS are object-dependent, as we will find out next. It receives 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 probably make a wrong grid-cell on the authentication server side when the correct grid is employed.

To enter a password, a ClickAnimal image is displayed first. Later on an animal is chosen, an icon of a grid appears, with the grid-cell size equaling the bounding rectangle of the chosen instrument. Each grid-cell is labeled to help identify users. Fig. 4 indicates a 6 grid when the red turkey in the left image of Fig. Comment number 4 was chosen. A user can select zero to multiple grid-cells matching her password. Hence a word is a chronological succession of animals interleaving with grid-cells, e.g., ρ "Dog, Grid 2, Grid 1; Cat, Horse, Grid 3 ", where Grid 1 means the grid-cell indexed as 1, and grid-cells after an animal means that the power system is

define by the bounding rectangle of the beast. A word must get down with an animal.

When a ClickAnimal image appears, the user clicks the animal on the word picture that fits the first animal in her intelligence. The coordinates of the clicked point are recorded. The bounding rectangle of the clicked animal is then found, interactively as follows: a bounding rectangle is calculated and displayed, e.g., the white rectangle shown in Fig. 4. The user specifies the displayed rectangle and corrects inaccurate edges by dragging if needed. This procedure is reiterated until the user is satisfied with the accuracy of the bounding rectangle. In most instances, the calculated bounding rectangle is accurate enough without calling for manual correction.

Once the bounding rectangle of the chosen animal is identified, an icon of a grid with the identified bounding rectangle as its grid-cell size is selected forth and let out. If the grid image is excessively big or too minuscule 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 correspond the grid-cells following the first creatures in her password, and then comes back to the ClickAnimal image age. For the example password ρ given previously, she clicks a point inside grid-cell 2, and then a point inside grid-cell 1 to select the two grid-cells. The coordinates of user-clicked point on the grid image (the original one before scaling if the grid image is scaled) are presented. The above procedure is reiterated until the user has ended up coming in her news. The resulting sequence of coordinates of user-clicked points, e.g., "AP 150,50, GP 30,66, GP 89,160, AP 135,97,…" where "AP x, y " denotes the point with coordinates x, y on a ClickAnimal image, and "GP x, y " denotes the point with coordinates x, y on a grid image, is transported to the authentication server.

Using the ground truth, the server retrieves 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 computed and compared with the stored hash.

V. RECOGNITION-RECALL CaRP

In recognition-recall CaRP, a word is a chronological succession 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 distinguish the objects in a CaRP image, and then utilizes 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 satisfactory as the password point. Most people make a tricky variation of 3 pixels or less [18]. TextPoint, a recognition- recall CaRP scheme with an alphabet of characters, is presented next, succeeded by a variation of challenge- response authentication.

A. TextPoints

Characters contain invariant points. Fig. 5 shows some invariant points of the alphabetic character "A", which proffers a strong cue to memorize and locate its invariant points. A bit is said to be an inner point of an object if its distance to the closest limit of the object exceeds a threshold. A lot of internal invariant points of characters is selected to make a lot of clickable points for TextPoints. The internality ensures that a clickable point is unlikely occluded by a neighboring character and that its tolerance region unlikely overlaps with any tolerance region of a neighboring 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 case 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 brought into thoughtfulness. For instance, if the effect of a stroke segment in one role is required, we should avoid selecting the meat of a similar stroke segment in another case. Rather, we should take

Fig. 5. Some invariant points (red crosses) of "A".

Rather, we should look at a different level from the stroke segment, e.g., A sprinkle at one-third length of the stroke segment to a destination. This change 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 case that the detail lies in. Finer recognition is required in locating clickable points on a TextPoints image, although the clickable points are known for each exercise. This is a task beyond a bot's capability.

A word is a sequence of clickable points. A type can typically contribute multiple clickable points. Therefore TextPoints has a much larger password space than Click Text. **Image Generation.** Text Points images look identical to Click Text, images and are generated in the same manner except that the positions of all the clickable points are selected to insure that none of them is occluded or its tolerance region overlaps another clickable point's. We just make another image if the verification fails. As such failures occur rarely due to the fact that clickable points are all internal points, the restriction due to the confirmation has a negligible impact on the security of generated icons.

Authentication. When creating a word, all clickable points are labelled with the corresponding qualities in a

CaRP image for a user to read. During authentication, the user first identifies her is chosen characters, and clicks the password points on the right sections. The authentication server maps each user-clicked point on the image to see the closest clickable point. If their distance exceeds a tolerable range, login fails. Other than 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 TextPoints and traditional click-based graphical passwords such as passports [5]. In PassPoints, salient points should be avoided since they are readily picked up by adversaries to mount dictionary attacks, but avoiding salient points would increase the essence to remember a word. This difference does not exist in TextPoints. Clickable points in TextPoints are the salient details of their lawsuits and therefore 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., Cases) change from one picture to another.** The clickable points in one image are computationally Inde- pendent of the clickable points in another picture, as we will note in Section VI-B.
- **Context:** Whether a similarly structured point is a clickable point or not depends on its context. It is only if within the right context, ie, at the correct location of a serious quality.

These two features require recognizing the correct settings, i.e., Characters, first. By the identical nature of Captcha, recognize- in theatrical roles in a Captcha image is a task beyond computer's capability. Thus, these salient details of cases cannot be exploited to mount dictionary attacks on Text Points.

B. TextPoints4CR

For the CaRP schemes presented up to now, the coordinates of the user-clicked points are transmitted immediately to the authentication server during authentication. For more complex protocols, say a challenge-response authentication protocol, a response is mailed to the authentication server instead. Text Points can be modified to fit challenge-response authentication. This fluctuation 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 explanation. Another deviation is that each piece comes along only one time with 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 discretized grid-cells independently. That takes a singular path to fetch forth the sequence from the shared secret, i.e., Word. Repeated characters would go to several posh- simple sequences for the same word. 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 x-axis 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 TextPoints image is used. And then the next operation is used to make every click- able point at least τ distance from the boundaries of the grid-cell it lies in. All the clickable points, denoted as set F, are settled on the image. For every point in F, we compute its distance along x-axis or y-axis to the middle of the grid-cell it lies in. A chip is supposed 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 F, a nearby internal point in the same grid-cell is chosen. The selected period is called a target point of the boundary point. After treating all the points in F, we obtain a new set Fj comprising internal points; these are either internal clickable points or target points of boundary clickable points.

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. Hence the sequence of grid-cells generated from user-clicked points are 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 pass- words instead of their hash values. Stored passwords must be protected from insider attacks; for instance, 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. SECURITY ANALYSIS

A. Security of Underlying Captcha

Computational intractability in recognizing objects in CaRP images is fundamental to CaRP. Existing analyses of Captcha security were mostly case by case or used an approximate process. No theoretic security model has been produced thus far. Object segmentation is viewed as a computationally- expensive, combinatorically-hard problem [30], which modern text Captcha schemes rely on. According to [30], the complexity of object

segmentation, C, is exponentially dependent of the number M of objects marked in a challenge, and polynomial dependent of the size N of the Captcha alphabet: C αM P (N), where α > 1 is a parameter, and P () is a polynomial function. A Captcha challenge typically contains 6 to 10 characters, whereas a CaRP image typically takes

Use 30 or more events.. The complexity to break a Click- Text, image is about α^{30} *P (N) /(*α^{10} *P (N))* α^{20} times the complexity to break a Captcha challenge generated by its underlying Captcha scheme. Therefore Click Text is much harder to get round than its underlying Captcha scheme. Further more characters in a CaRP scheme are arranged two- dimensionally, further increasing segmentation difficulty due to one more dimension to segment. As a consequence, we can reduce distortions in Click Text images for improved usability yet maintain the same protection level as the underlying text Captcha. ClickAnimal relies on both object segmentation and multiple-label classification. Its security remains an undecided question.

As a framework of graphical passwords, CaRP does not rely on any specific Captcha scheme. If one Captcha scheme gets broken, a raw and more robust Captcha scheme may appear and be used to construct a new CaRP scheme. In the remaining security analysis, we take for granted that it is intractable for computers to distinguish any objects in any challenge image generated by the underlying Captcha of CaRP. More accurately, the Captcha is assumed to be chosen-pixel attack (CPA) - secure defined by the following experiment: an adversary at first reads from an arbitrary bit of challenge images by querying a ground- truth oracle O as follows: A selects an arbitrary act of intimate objects-points and sends to O, which reacts with the aim that each point lies in. Then A receives a new challenge image, and chooses an internal object-point to query O again.

This time *O* chooses a random bit *b* 0, {1 to determine what to return: It returns the true object if *b* 1; otherwise a false object selected with a certain strategy. *A is needed to indicate whether the given object is the true object that the internal object-point lies in or not.* A Captcha scheme is taken to be CPA-secure if A cannot succeed with a probability non-negligibly higher than ½, the probability of a random guess.

B. Automatic Online Guessing Attacks

In automatic online guessing attacks, the trial and error process is posted out automatically, whereas dictionaries can be made manually. If we ignore negligible probabilities, CaRP with underlying CPA-secure Captcha has the accompanying attributes:

1. Internal object-points on one CaRP image are *computed-independent* of internal object-points on another CaRP image. Particularly, clickable points on one image are computed-independent of clickable points on some other flick.
2. Equivalent. (3) Holds, i.e., Trials in guessing attacks are mutually independent.

The first property can be proved by contradiction. Presume that the property does not prevail, i.e., There exists an inter- null object-point α on one image A that is non-negligibly dependent on an inner object-point β on another image B. An opponent can exploit this dependency to launch the following chosen-pixel approach. In the learning stage, image A is used to ascertain the objective that contains point α. In the testing stage, point β on image B is used to question the oracle. Since point α is non-negligibly dependent of point β, this CPA-experiment would result in a success probability non-negligibly higher than a random guess, which contradicts the CPA-secure assumption. We reason 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 computed-independently of user-clicked internal object-points in another trial due to the first attribute. 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 performed. 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 make out the objects in the CaRP image to input the password Correctly. This is a big contrast to automatic online guessing attacks on existing graphical passwords which are determined is tic, 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 adjust by a circumscribed number of tests. Particularly, brute-force attacks or dictionary attacks with the targeted password in the dic- tionary would always succeed in attacking existing graphical passwords.

Human Guessing Attacks

In human guessing attacks, humans are used to enter passwords in the trial and error process. Mankind are a good deal slower than computers in mounting guessing attacks. For 8-character passwords, the theoretical password space is

338240 for Click Text with an ABC of 33 events, 108226 for ClickAnimal with an ABC of 10 animals, and 10467242 for AnimalGrid with the setting as ClickAnimal plus 66 grids. 6 grids. If we assume that 1000 people are employed to work 8 hours per day without any stopover in a human guessing attack, and that each person takes 30 minutes to finish one trial. It would lead them on average 0.5 338 30/ (3600 8 1000 365)2007 years to break a Click Text password, 0.5 108 30/(3600 8 1000)52 days to break a ClickAnimal password, or 0.5 10 46730/(36008 1000 365)6219 years to break an AnimalGrid password. Human guessing attacks on TextPoints require a much longer time than those in Click Text since TextPoints has a much larger password space.

Exactly like any password scheme, a longitudinal evaluation is needed to establish the effective password space for each CaRP instantiation. This calls for a

separate study similar to what Bonneau [42] did for text passwords.

A recent survey on text passwords [42] shows that users tend to choose passwords of 6–8 characters and have a firm dislike of using non-alphanumeric characters, and that an acceptable benchmark of effective password space is the anticipated number of optimal guesses per account needed to snap off 50% of accounts, which is equivalent to 21.6 bits for Yahoo! Users. If we assume that Click Text has roughly the same effective password space as text passwords, it requires on average 1000 people to work 1.65 days or one person to work 4.54 years to find a Click Text password.

C. Relay Attacks

Relay attacks may be borne out in various 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 stay surfing the Website, or related to sweatshops where humans are employed to solve Captcha challenges for small payments. Is CaRP vulnerable to replay attacks? We reach the same assumption as Van Oorschot and Stubblebine [15] in discussing CbPA-protocol's robustness to relay attacks: a person will not deliberately participate in relay attacks unless paid for the job. The task to perform and the image used in CARP are very dissimilar from those employed to solve a Captcha challenge. This noticeable difference makes it difficult for a person to mistakenly help test a password guess by trying to figure out a Captcha challenge. And then it would be unlikely to make a great number of unknowing people to mount human guessing attacks on CaRP. In increase, human input obtained by performing a Captcha task on a CaRP image is useless for testing a password guess.

If sweatshops are hired to put on a human guessing attack, we can draw a crude idea of the price. We presume that the cost to click one password, on a CaRP image is the same as working out a Captcha challenge. Using the lowest retail price, $1,

Reported [34] to solve 1000 Captcha challenges, the average cost to break a 26-bit password is $0.5\ 2^{26}\ 1/1000$, or about
33.6 thousand US dollars.

D. Shoulder-Surfing Attacks

Shoulder-surfing attacks are a threat when graphical passwords are inserted in a public place such as bank ATM machines. Carp is not robust to shoulder-surfing attacks by itself. Still, combined with the following dual-view technology, CaRP can thwart shoulder-surfing attacks.

By solving the technological 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 icons on an LCD screen concurrently, one public image viewable at most view-angles, and the other private image viewable only at a specific aspect-angle [38]. When a CaRP image is exhibited as the "private" image of the dual-view system, a shoulder-surfing attacker can get user- clicked point on the sieve, but cannot capture the "private" CaRP image that only the user can view.

Withal, 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 interpret the correct word on the new image anymore.

To the contrary, common implementations of graphical password schemes such as passports use a static input image in the same location of the screen for each login attempt. Although this icon can be shrouded as the private image by the dual-view technology from being overwhelmed by a shoulder- surfer, the user-clicked points captured in a successful login are still the valid password for the next login attempt. That is, getting the points alone is sufficient for an effective attack in this lawsuit.

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.

E. Others

CaRP is not bulletproof to all possible attacks. Carp is vulnerable if a customer is compromised, such that both the image and user-clicked points can be earned. Too many other graphical passwords such as CCP and PCCP, CaRP schemes using the basic CaRP authentication are vulnerable to phishing since user-clicked points are sent to the authentication server. However, CaRP schemes such as TextPoints4CR used with challenge-response authentication are robust to phishing to a certain level: a phishing adversary has to mount offline guessing attacks to differentiate out the password using the verifiable data obtained through a successful phishing attack.

VII. EMPIRICAL EVALUATIONS

A. Executions

Click Text and AnimalGrid were implemented using ASP.NET. Click Text was implemented by calling a config- urable text Captcha engine commercially used by Microsoft. This Captcha engine accepts only capital letters. As a consequence, we chose the following 33 characters in our usability studies: capital letters except I, J, O, and Z, digits except 0 and 1, and three special characters "#", "@", and "&". The last three special cases were selected to balance security and users' strong dislike of using non-alphanumeric characters in text passwords [42]. The examples were arranged in 5 categories. Each case was randomly rotated from 30◦ to 30◦ and scaled from 60% to 120%.. Neighboring characters could overlap up to 3 pixels. Warping effect was set to the luminosity level. Each icon was set to 400 by 400 pixels.

Fig. 2 in Section IV-A shows an image brought forth with the above setting.

In our implementation of AnimalGrid, we used an ABC of 10 animals: bird, cow, buck, dog, giraffe, pig, rabbit, camel, element, and dinosaur.. Each instrument had three 3D models. The number of beasts in a ClickAnimal image ranged randomly from 10 to 12, with the extra animals randomly selected from the first principle. In producing an animal, object, one of the three 3D animal models was randomly selected, and set at a random view in generating a 2D object. Each animal was set apart a color randomly picked out from a set of 12 colors. Generated 2D objects were situated at random on a grass ground, with the main character of each animal not occluded by other creatures. Each ClickAnimal image was also set to 400 by 400 pixels. A6 grid was used for CAS. Cells were labeled clockwise starting from cell 0. Fig. 4 in Section IV shows an example of generating ClickAnimal images and an example of grid images. Thither was a cross icon on top of a grid image that a user could lose it to shut down the grid icon.

B. Usability Study

1) Experimental Settings

We took an in-lab usability study to compare Click-Text, AnimalGrid, PassPoints, text, password (Text), and text password combined with text Captcha (P C). *C)*. PC was used to simulate a CbPA-protocol when a Captcha challenge was used during login. In PC, a user was required to enter a password and solve a Captcha challenge generated by the same Captcha engine used in Click Text. Each Captcha challenge contained 6 to 8 random characters. Keyboard input was used to produce and enter passwords for Text and P C as well as to enter user IDs for all the systems. As explained later, Text and P C were conducted as if they were a single outline for participants.

We recruited 40 (30 males and 10 females) voluntary senior and grad students majoring in engineering and sciences, with ages running from 20 to 28 years (the average age 23.4 and the standard deviation 1.74).). For pragmatic rea- sons, they were recruited from interns working at Microsoft Research Asia. None of them had studied security or was involved in any security usability study before. They were regarded in this work solely as participants in our usability survey. All participants were trained to get cozy with each authentication scheme and their experimental tasks before our data collecting. During the experiment, one of the authors got each participant when it was time for the participant to gain a test, which insures that we could gather the needed information from every participant.

Each system was tried in the following setting: a partic-paint used a web browser to interact with an authentication server, creating passwords or logging into the host. One time a participant submitted his/her credentials to the waiter, the browser would show the login result.

The systems were divided into two classes according to their cases of passwords: AnimalGrid and PassPoints in the inaugural year, and the remaining schemes in the second degree. A word for the schemes in the second class was a chain of roles.

Each participant was required to make a new password never used previously for each scheme, 4 in total, and a user ID for all the systems. Each created password consisted of 8 characters or click-points. We also made it explicit that participants were not allowed to write down their passwords. Each password must conform to the following minimum come- plexity requirements. A word must carry at least one letter, one digit, and individual non-alphanumerical character for both Text and Click Text, and at least three different animals for AnimalGrid. No repeating patterns such as "A#A#…" or "Dog, Dog,…" were allowed. For PassPoints, click-points in a word must be distinct (i.e., No click-point was inside another click-point's tolerance range). Each password was verified immediately after creation.

The subject area was zoned into two phases. II systems were examined at each level. On the commencement level, two schemes, one from each category, were randomly selected for a participant to examine. One scheme had a chain of text references as a password while the other held a string of animals and grid cells or click- points as a rallying call.

During the survey, each participant was required to log in with the following intervals between two consecutive login tests: one hour after creation, one day, one week, and three weeks. In each trial, a participant was permitted three attempts to log in. If he/she failed three attempts, his/her password was considered forgotten, and no more test would be taken by the participant for that specific system.

In the second phase, the remaining two schemes were examined in the same manner as above.

In the terminal, each participant was asked to fill a question- nor to compare Click Text and AnimalGrid with PassPoints and Text, and to compare Click Text with P C, in terms of ease of habit as a password scheme, taking both memorizing and entering a password into consideration..

A participant's login time in each trial was brought out by the host. We limit the login time as the duration from the time when the server had a login request to the time when the server gave its response to the login request, which includes the time to introduce a user ID and password, to generate a CaRP image, and to communicate between the host and a participant's browser.

For Text and P C, a participant was taken to insert a word. If successful, the server recorded the fourth dimension at the login time for Text, and then generated a Captcha challenge and sent to the user to discover out. If the participant failed with the challenge, another challenge was brought off and practiced.

This procedure was iterated until the server received a right response to a challenge. Then the server recorded the fourth dimension at the login time for P C, which included the time that the participant failed to resolve a challenge.

2) Experimental Results

Usability. Among all the recorded login attempts, 24.4% died. Trials after a larger interval tended to engage in more failed efforts. Some participants contributed significantly more failed attempts than others. At the end of tests, 40 (100%) participants remembered their passports passwords, 39 (97.5%) remembered their passwords of both Click Text and AnimalGrid, and 34

T (s) 27.22 29.20 21.62 28.24 10.34

vs. PassPoints vs. Text vs. P+C

2.5 7.5 7.5 15.0 25.0

(85%) remembered their Text passwords. One participant forgot the AnimalGrid password at the one-hour test, and another one forgot the Click Text password at the one-week trial. For Text, two participants forgot their passwords at the single-week test, and four forgot at the three- week trial. Passports scored the best in memorability whereas Text scored the most spoiled. This may be partly due to the fact that hotspots were allowed in PassPoints passwords, and that Text passwords had a much larger alphabet than both Click Text and AnimalGrid.

Scheme Click Text PassPoints P+C Text

17.38 19.24 12.29 12.55 6.08

Max.(s) 63.62 88.51 [illegible] [illegible] [illegible]

Min.(s) 10.41 13.46 8.36 13.7 3.58

Table I presents the login time averaged over the 40 participants' successful login attempts and the sample standard deviation as well as the maximum and minimum login times for each scheme.. Click Text, AnimalGrid and P C had similar average login time, whereas PassPoints had a little shorter average login time. The textbook had a much shorter average login time than the other systems. Each system had a large sample standard deviation relative to the average login time, indicating large variations of login time for each schema, which is supported by the large difference between the lower limit and maximum login times in each column shown in Table I. This is chiefly done by large individual differences. We did not find obvious patterns indicating that a test with a longer interval had a larger login time than an interrogation with a shorter interval. We did note that some participants experienced a much larger login time when the preceding trial failed, but many other participants didn't take after this notice.

Easier (%) 40.0 47.5 25.0 40.0 47.5

The language in our trials were applied much less free-quently than typical usage of a password in practice since we would like to test password memorability for each system. We expect improved results when a parole is used more often.

Table II shows the comparison results of different scheme for ease of utilization as a password scheme. We put a value ranging from 1 to 5 to each category, indicating the spectrum from "much more difficult" to "much more leisurely". Click Text has a mean value of 3.2 and a median value of 3 as compared to PassPoints, and a level of 2.85 and a median of 2 as compared to Text. AnimalGrid has a mean of 3.325 and a median of 4 as compared to PassPoints, and a mean of 3.5 and a median of 4 as compared to Text. Click Text has a mean of 3.875 and a median of 4 as compared to P C.

Security. S We also analyzed the security of the passwords we collected for Text and Click Text, with a popular password-checking tool, John the Ripper version 1.7.9 [43]. Given our sample size of 40 users, the distribution of passwords will not be as complete as a larger study with significantly more users,

TABLE I

LOGIN TIME FOR DIFFERENT SCHEMES: AVERAGE (T), SAMPLE STANDARD DEVIATIO)(σ), MAX. AND MIN.

		Animal Grid			
σ(s)					

TABLE II

COMPARING DIFFERENT SCHEMES FOR EASE OF USE

	Click Text	Animal Grid	Click Text	Animal Grid	Click Text
Much easier (%)					
More difficult (%)					
Much more difficult (%)					

But such an analysis can supply at least an indication of their shelter.

John the Ripper has three performance modes: "single crack", "wordlist", and "incremental". In the "single crack" mode, login names and other invoice information as well as a large set of mangling rules are given to generate password guesses. In the "wordlist" mode, a list of words and word mangling rules are given to generate password guesses. In the "incremental" mode, a brute force attack is used, with guesses being tested in the descending order of their likelihood to be a word.

In our field, the default parameters and contexts were used for John the Ripper except that length of a password was set to 8 characters, and that the recommended wordlist "all. Lst" from [44] was used in the "wordlist" mode. When operated- in in both "single crack" and "wordlist" modes, John the Ripper didn't find any word for either Text or Click Text. When playing in the "incremental" mode for 24 hours on an HP Compaq Elite

8100 PC with Intel Core i7-870 CPU, 8GB RAM and 64-bit Windows 7 OS, John the Ripper found two of 40 (i.e., 5.0%) passwords for Text but didn't meet any password in Click Text. The small difference in the effects of the "incremental" mode is likely because the unusual alphabet set in Click Text increased a user's chance to read more random passwords in order to satisfy the same password complexity requirement applied to both Text and Click Text. A separate longitudinal study is required to fully interpret the distribution and security of Click Text passwords that users would pick out.

We conclude from the cracking results that the pass- words the participants selected for Text and Click Text were reasonably firm, which meets our expectation of the Password complexity requirement described in the previous subsection.

3) Treatments

Wiedenbeck et al. [5] Studied memorability of text pass- words and PassPoints with 20 participants for each scheme: their recall rate after one week was 65% and 70%, respectively. Our memorability results are higher than theirs for both text and PassPoints passwords.

The sample sizes in both [5] and our subject fields were too pressed down to explain conclusively the difference in the outcomes. Still, this difference could probably be excused from the remainder of the issues in both fields. First, although randomly chosen, our participants were from the pool of interns working at Microsoft Research Asia, who were typically removed from leading universities. Second, our participants were used to strong text passwords as their Microsoft accounts were strictly applied with strong password policies (e.g. Each account must apply a complex password, and a password has to be changed regularly, with the new password having to be significantly different from previously used ones). Third, our participants were much younger, with an average age of 23.4 years as compared to the average of 32.9 years in Wiedenbeck et al.'s experiment.

However, we do not consider this sample bias introduced an undue impact on the main goal of our experimentation, namely, comparing the performance of CaRP with other authentication organizations. However, user studies with a diverse sample pool are needed for CaRP, which are our future work.

C. Computation Load on Server

Compared with many graphical password schemes, genre- action of CaRP images is an extra burden on the host side. We tested our implementations of Click Text and AnimalGrid on the same HP Compaq Elite 8100 PC we used to run John the Ripper, as identified in Section VII-B. For images of 400x400 pixels, the average speed was 10.68 images per second in generating a Click Text image with 33 characters and 0.86 images/s in generating an AnimalGrid image with 10 to 12 animals. Our implementations were single-threaded without code optimization, and did not take any advantage of the multi - core capability of the test car. Much faster image generation should be feasible by exploiting multi-core architecture of today's hosts and by optimizing the code.

VIII. 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 insert a password as other graphical password schemes, it necessitates a longer time to insert a password than widely used text passwords. We have the cat out of the bag about two approaches for balancing CaRP's security and serviceability.

A. Alphabet Size

Increasing alphabet size produces a larger password space, and therefore is more dependable, but also leads to more complex CaRP Pictures.

When the complexity of CaRP images gets beyond a sure level, humans may need a substantial total of time to know the qualities in a CaRP image and may start out frustrated. The optimal alphabet size for a CaRP scheme such as Click Text remains an unresolved inquiry.

It is likely to apply a modified subset of the alphabet to generate CaRP images for a user if the server receives her user ID before posting an image. In this example, the authentication server allows a user to create her password from the full alphabet. One time the password is created, the server creates a desirable subset of a reasonable size, which holds all the symbols in the parole. The host stores the subset or its index for the history, and regains it later when the bill attempts to log in to generate a CaRP image. This system is suited when the ABC must be heavy while some people would log in on small-screen devices for which an icon using the entire alphabet would be too complex to quickly place the objects in the picture.

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 weather. For instance, the scheme described in [16] uses a Captcha challenge when the number of failed login attempts has reached a threshold for an explanation. A smaller 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 collapsed time frame. This proficiency can be integrated into CaRP to enhance usability:

1. A regular CaRP image is used when an account has reached a threshold of failed login attempts. As in [16], different thresholds are used for logins from known and strange machines.
2. Otherwise an "easy" CaRP image is used.

An "easy" CaRP image may contain several kinds depending on the application demands. 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. References) are permuted, or even a regular "keypad" wherein each visual object (e.g., Character) is invariably situated at a specified location. These different varieties of "easy" CaRP images allow a scheme to adjust the degree of difficulty to suit its needs.

With such a modified CaRP, a user would always insert a password on an image for both cases listed in a more eminent place. No special labor is needed. The sole difference between the two fonts is that a strong image is used in the first plate, whereas an easy image is used in the second example.

IX. CONCLUSION

We have proposed CaRP, a new security primitive relying on unsolved hard AI problems. Cap is both a Captcha and a graphical password system. The notion of CaRP intro- duces a new family of graphical passwords, which espouses a new plan of attack to counter online guessing attacks: a new CaRP image, which is also a Captcha challenge, is employed for every login attempt to make trial 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-strength attempts, a desired security property that other graphical password schemes lack. Hotspots on CaRP images can no longer be exploited to mount automatic online guessing attacks, an inherent vulnerability in many graphical password schemes. Carp forces adversaries to resort to significantly less efficient and a good lot 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. Corps can also help cut back spam emails sent from a Web email service.

Our usability study of two CaRP schemes we have lived through is encouraging. For instance, more participants considered AnimalGrid and Click Text easier to use than PassPoints and a combination of text password and Captcha. Both AnimalGrid and Click Text had better password memora- ability than the conventional text passwords. On the other hand, the usability of CaRP can be farther ameliorated by utilizing images of different points of difficulty based on the login history of the user and the machine used to log in. The optimal tradeoff between protection and usability remains an unresolved question for CaRP, and further surveys are required to refine CaRP for actual deployments.

Like Captcha, CaRP utilizes unsolved AI problems. Withal, a password is much more valuable to attackers than a spare email account that Captcha is typically employed to protect. Then at that point 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 the attackers succeed, they lead to improving the AI by providing solutions to unresolved problems such as segmenting 2D texts. Otherwise, our system stays secure, going to practical security. As a framework, CaRP does not rely on any specific Captcha scheme. When one Captcha scheme is discontinued, a raw and more secure one may appear and be converted to a CaRP scheme.

Overall, our study is one step onward in the epitome 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 significantly, we expect CaRP to inspire new innovations of such AI based security primitives.

REFERENCES

[1] R. Biddle, S.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 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 passpoints graphical password scheme," in *Proc. 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 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.

[17] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford, "CAPTCHA: Using hard AI problems for security," in *Proc. Eurocrypt*, 2003, pp. 294–311.

[18] S. Chiasson, P. C. van Oorschot, and R. Biddle, "Graphical password authentication using cued click points," in *Proc. ESORICS*, 2007, pp. 359–374.

[19] S. Chiasson, A. Forget, R. Biddle, and P. C. van Oorschot, "Influencing users towards better passwords: Persuasive cued click-points," in *Proc. Brit. HCI Group Annu. Conf. People Comput., Culture, Creativity, Interaction*, vol. 1. 2008, pp. 121–130.

[20] D. Davis, F. Monrose, and M. Reiter, "On user choice in graphical password schemes," in *Proc. USENIX Security*, 2004, pp. 1–11.

[21] R. Dhamija and A. Perrig, "Déjà Vu: A user study using images for authentication," in *Proc. 9th USENIX Security*, 2000, pp. 1–4.

[22] D. Weinshall, "Cognitive authentication schemes safe against spyware," in *Proc. IEEE Symp. Security Privacy*, May 2006, pp. 300–306.

[23] P. Dunphy and J. Yan, "Do background images improve 'Draw a Secret' graphical passwords," in *Proc. ACM CCS*, 2007, pp. 1–12.

[24] P. Golle, "Machine learning attacks against the Asirra CAPTCHA," in *Proc. ACM CCS*, 2008, pp. 535–542.

[25] B. B. Zhu *et al.*, "Attacks and design of image recognition CAPTCHAs," in *Proc. ACM CCS*, 2010, pp. 187–200.

[26] J. Yan and A. S. El Ahmad, "A low-cost attack on a microsoft CAPTCHA," in *Proc. ACM CCS*, 2008, pp. 543–554.

[27] G. Mori and J. Malik, "Recognizing objects in adversarial clutter," in *Proc. IEEE Comput. Society Conf. Comput. Vis. Pattern Recognit.*, Jun. 2003, pp. 134–141.

[28] G. Moy, N. Jones, C. Harkless, and R. Potter, "Distortion estimation techniques in solving visual CAPTCHAs," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jul. 2004, pp. 23–28.

[29] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, "Computers beat humans at single character recognition in reading-based human interaction proofs," in *Proc. 2nd Conf. Email Anti-Spam*, 2005, pp. 1–3.

[30] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, "Building segmentation based human-friendly human interaction proofs," in *Proc. 2nd Int. Workshop Human Interaction Proofs*, 2005, pp. 1–10.

[31] J. Elson, J. R. Douceur, J. Howell, and J. Saul, "Asirra: A CAPTCHA that exploits interest-aligned manual image categorization," in *Proc. ACM CCS*, 2007, pp. 366–374.

[32] R. Lin, S.-Y. Huang, G. B. Bell, and Y.-K. Lee, "A new CAPTCHA interface design for mobile devices," in *Proc. 12th Austral. User Inter. Conf.*, 2011, pp. 3–8.

[33] N. Joshi. (2009, Nov. 29). *Koobface Worm Asks for CAPTCHA* [Online]. Available: http://blogs.mcafee.com/mcafee-labs/koobface-worm-asks- for-CAPTCHA

[34] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, "Re: CAPTCHAs—Understanding CAPTCHA-Solving Services in an Economic Context," in *Proc. USENIX Security*, 2010, pp. 435–452.

[35] M. Szydlowski, C. Kruegel, and E. Kirda, "Secure input for web applications," in *Proc. ACSAC*, 2007, pp. 375–384.

[36] G. Wolberg, "2-pass mesh warping," in *Digital Image Warping*. Hoboken, NJ, USA: Wiley, 1990.

[37] HP TippingPoint DVLabs, New York, NY, USA. (2011). *The Mid-Year Top Cyber Security Risks Report* [Online]. Available: http://h20195.www2.hp.com/v2/GetPDF.aspx/4AA3-7045ENW.pdf

[38] S. Kim, X. Cao, H. Zhang, and D. Tan, "Enabling concurrent dual views on common LCD screens," in *Proc. ACM Annu. Conf. Human Factors Comput. Syst.*, 2012, pp. 2175–2184.

[39] S. Li, S. A. H. Shah, M. A. U. Khan, S. A. Khayam, A.-R. Sadeghi, and R. Schmitz, "Breaking e-banking CAPTCHAs," in *Proc. ACSAC*, 2010, pp. 1–10.

[40] H. Gao, X. Liu, S. Wang, and R. Dai, "A new graphical password scheme against spyware by using CAPTCHA," in *Proc. Symp. Usable Privacy Security*, 2009, pp. 760–767.

[41] L. Wang, X. Chang, Z. Ren, H. Gao, X. Liu, and U. Aickelin, "Against spyware using CAPTCHA in graphical password scheme," in *Proc. IEEE Int. Conf. Adv. Inf. Netw. Appl.*, Jun. 2010, pp. 1–9.

[42] J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in *Proc. IEEE Symp. Security Privacy*, Jun. 2012, pp. 20–25.

[43] *John the Ripper Password Cracker* [Online]. Available: http://www. openwall.com/john/

[44] *Openwall Wordlists Collection* [Online]. Available: http://www. openwall.com/wordlists/

M.Sneha Divya is presently pursuing M.Tech (CST) Department of Computer Science Engineering from Baba Institute of Technology and Sciences,Visakhapatnam.

Mr. P. Joshua Raju, pursuing his Ph.D. at Gitam University is currently working at BITS Vizag as an Assistant Professor of Computer Science and Engineering in Baba Institute of Technology and Sciences, Visakhapatnam.