Human-Seeded Attacks and Exploiting Hot-Spots in Graphical Passwords*

Julie Thorpe *P.C. van Oorschot*
School of Computer Science, Carleton University

Abstract

Although motivated by both usability and security concerns, the existing literature on click-based graphical password schemes that use a single background image like PassPoints (Wiedenbeck et al., 2005) has focused largely on usability. We examine the security of such schemes, including the impact of different background images, and strategies for cracking user passwords. We report on both short- and long-term user studies: one lab-controlled, involving 43 users and 17 diverse images, and the other a field test of 223 users. We provide empirical evidence that popular points (hot-spots) do indeed exist for many images, and exploit them. We create and evaluate two different types of attack to exploit this hot-spotting effect: a "human-seeded" attack based on harvesting click-points from a small set of users, and an entirely automated attack based on image processing techniques. Our most effective attacks are generated by harvesting password data from a small set of users to attack others' passwords. These attacks can crack 36% of user passwords within 2^{31} guesses (or 11% within 2^{17} guesses) on one image, and 20% within 2^{33} guesses (or 6% within 2^{22} guesses) on a second image. We perform an image-processing attack by implementing and adapting a bottom-up model of visual attention, resulting in a purely automated tool that cracks up to 30% of user passwords in 2^{35} guesses for some images, but under 3% on others. Our results show that these graphical passwords, even using the best among our tested background images, are at least as susceptible to offline attack as the traditional text-based passwords they have been proposed to replace.

1 Introduction

The bane of password authentication using text-based passwords is that users choose passwords which are easy to remember, which generally translates into passwords that are easily guessed. Thus even when the size of a password space may be theoretically "large enough" (in terms of number of possible passwords), the *effective* password space from which many users actually choose passwords is far smaller. Predictable patterns, largely due to usability and memory issues, thus allow successful search by variations of exhaustive guessing attacks. Forcing users to use "random" or other non-meaningful passwords results in usability problems. As an alternative, graphical password schemes require that a user remembers an image (or parts thereof) in place of a word. They have been largely motivated by the well-documented human ability to remember pictures better than words [24], and implied promises that the password spaces of various image-based schemes are not only sufficiently large to resist guessing attacks, but that the effective password spaces are also sufficiently large. The latter, however, is not well established.

Among the graphical password schemes proposed to date, one that has received considerable attention in the research literature is PassPoints [41, 42, 43, 3]. It and other click-based graphical password schemes [17, 4, 30] require a user to log in by clicking a sequence of points on a single background image. Usability studies have been performed to determine the optimal amount of error

*Version: Feb. 20, 2007. Contact author: jthorpe@scs.carleton.ca.

tolerance [42], whether people can create passwords in a reasonable period of time, error rates, and general perception [41, 43]. An important remaining question for such schemes is: how *secure* are they? This issue remains largely unaddressed, despite speculation that the security of these schemes likely suffers from hot-spots – areas of an image that are more probable than others for users to click. Indeed, the impact of hot-spots has been downplayed by some (e.g., see [41, Section 7]). In this paper, we focus on a security analysis of an implementation with the same parameters as used in the most recent PassPoints publication [43].

We confirm the existence of hot-spots through empirical studies, and show that some images are more susceptible to hot-spotting than others. We also explore the security impact of hot-spots, including a number of strategies for exploiting them under an offline attack model similar to that used by Ballard et al. [1]. Our work involves two user studies. The first used 17 diverse images (four used in previous studies [42], and 13 of our own chosen to represent a range of detail). We collected graphical passwords for 32-40 users per image in a lab setting, finding hot-spots on all images even from this relatively small sample size; some images had significantly more hot-spots than others. In the second study involving 223 users over a minimum of seven weeks, we explore two of these images in greater depth.

We implement and evaluate two types of attack: human-seeded and purely automated. Our human-seeded attack is based on harvesting password data from a small number of users to attack passwords from a larger set of users. We seed various dictionaries with the passwords collected in our lab study, and apply them to guess the passwords from our long-term field study. Our results demonstrate that this style of attack is quite effective against this type of graphical password: it correctly guessed 36% of user passwords within 2^{31} guesses (or 11% within 2^{17} guesses) on one image, and 20% within 2^{33} guesses (or 6% within 2^{22} guesses) on a second image. We implement and adapt a combination of image processing methods in an attempt to predict user choice, and employ them as tools to expedite guessing attacks on the user study passwords. The attack works quite well on some images, cracking up to 30% of passwords, but less than 3% on others within 2^{35} guesses. These results give an early signal that image processing can be a relevant threat, particularly as better methods emerge.

Our contributions include the first in-depth study of hot-spots in click-based graphical passwords schemes and their impact on security through two separate user studies. In the first (lab) study, we examine security for a diverse range of 17 background images; in a second (field) study, we further examine two of these images over 7 weeks or longer (varying by user). We propose the modification and use of image processing methods to expedite guessing attacks, and evaluate our implementation against the images used in our studies. Our implementation is based on Itti et al.'s [16] model of bottom-up visual attention and corner detection, which we found provided successful guessing attacks on some images, even with relatively naive dictionary strategies. Our most interesting contribution is applying a human-seeded attack strategy to graphical passwords, by harvesting password data in a lab setting from small sets of users, to attack other field-study passwords.

The remainder of this paper is organized as follows. Section 2 provides background and terminology. Section 3 presents our lab-controlled user study, and an analysis of observed hot-spots and the distribution of user click-points. Section 4 presents results on the larger (field) user study, and of our password harvesting attacks. Section 5 explores our use of image processing methods to expedite guessing attacks on the 17 images from the first user study and the two from the second user study. Related work is briefly discussed in Section 6. Section 7 provides further discussion and concluding remarks.

2 Background and Terminology

Click-based graphical passwords require users to log in by clicking a sequence of points on a single background image. Many variations are possible (see Section 6), depending on what points a user is allowed to select. We study click-based graphical passwords by allowing clicks anywhere on the image (i.e., PassPoints-style). We believe that most findings related to hot-spots in this style will apply to other variations using the same images, as the "interesting" clickable areas are still present.

We use the following terminology. Assume a user chooses a given click-point c as part of their password. The *tolerable error* or *tolerance* t is the error allowed for a click-point entered on a subsequent login to be accepted as c. This defines a *tolerance region (T-region)* centered on c, which for our implementation using $t = 9$ pixels, is a 19×19 pixel square. A *cluster* is a set of one or more click-points that lie within a T-region. The number of click-points belonging to a cluster is its *size*. A *hot-spot* is indicated by a cluster that is large, relative to the number of users in a given sample. To aid visualization and indicate relative sizes for clusters of size at least two, on figures we sometimes represent the underlying cluster by a shaded circle or *halo* with halo diameter proportional to its size. An *alphabet* is a set of distinct T-regions; our implementation, using 451×331 pixel images, results in an alphabet of $m = 414$ T-regions. Using passwords composed of 5-clicks, on an alphabet of size 414 provides the system with only a 43-bit full theoretical password space; we discuss the implications of this in the last paragraph of Section 7.

3 Lab Study (User Study #1) and Clustering Analysis

Here we report on the results of a university-approved 43-user study of click-based graphical passwords in a controlled lab environment. Each user session was conducted individually and lasted about one hour. Participants were all university students who were not studying (or experts in) computer security. Each user was asked to create a click-based graphical password on 17 different images.[1] Four of the images are from a previous click-based graphical password study by Wiedenbeck et al. [42]; the other 13 were selected to provide a range of values based on two image processing measures that we expected to reflect the amount of detail: the number of segments found from image segmentation [10] and the number of corners found from corner detection [15]. Seven of the 13 images were specifically chosen to be those we "intuitively" believed would encourage fewer hot-spots; this is in addition to the four chosen in earlier research [42] using intuition (no further details were provided on their image selection methodology).

Experimental Details. Each user was provided a brief explanation of what click-based graphical passwords are, and given two images to practice creating and confirming such passwords. To keep the parameters as consistent as possible with previous usability experiments of such passwords [43], we used $d = 5$ click-points for each password, an image size of 451×331 pixels, and a 19×19 pixel square of error tolerance.[2] Users were instructed to choose a password by clicking on 5 points, with no two the same.[3] We did not assume a specific encoding scheme (e.g., robust discretization [3] or other grid-based methods); the concept of hot-spots and user choice of click-points is general enough to apply across all encoding schemes. To allow for detailed analysis, we store and compare the actual click-points.

[1] Some of these are reproduced herein; others are available from the authors.

[2] Wiedenbeck et al. [43] used a tolerance of 20×20, allowing 10 pixels of tolerated error on one side and 9 on the other. To keep the error tolerance consistent on all sides, we approximate this error tolerance using 19×19.

[3] The software did not enforce this condition, but subsequent analysis showed that the effect on the resulting cluster sizes was negligible for all images except *pcb*; for more details, see caption of Table 1.

Once the user had a chance to practice a few passwords, the main part of the experiment began. For each image, the user was asked to create a click-based graphical password that they could remember but that others will not be able to guess, and to pretend that it is protecting their bank information. After initial creation, the user was asked to confirm their password to ensure they could repeat their click-points. On successful confirmation, the user was given 3D mental rotation tasks [32] as a distractor for at least 30 seconds. This distractor was presented to remove the password from their visual working memory, and thus simulate the effect of the passage of time. After this period of memory tasks, the user was provided the image again and asked to log in using their previously selected password. If the user could not confirm after two failed attempts or log in after one failed attempt, they were permitted to reset their password for that image and try again. If the user did not like the image and felt they could not create and remember a password on it, they were permitted to skip the image.[4]

To avoid any dependence on the order of images presented, each user was presented a random (but unique) shuffled ordering of the 17 images used. Since most users did not make it through all 17 images, the number of graphical passwords created per image ranged from 32 to 40, for the 43 users. Two users had a "jumpy" mouse, but we do not expect this to affect our present focus – the location of selected click-points. This short-term study was intended to collect data on initial user choice; although the mental rotation tasks work to remove the password from working memory, it does not account for any effect caused by password resets over time due to forgotten passwords. The long-term study (Section 4) does account for this effect, and we compare the results.

3.1 Results on Hot-Spots and Popular Clusters Observed

To explore the occurrence of hot-spotting in our lab user study, we assigned all the user click-points observed in the study to clusters as follows. (1) For each click-point c_i, let C_i be the cluster associated with the T-region centered on c_i. Count the total number of user click-points within c_i's T-region; these click-points are candidate members of the cluster. (2) Sort all clusters in decreasing order by their size c_i. (3) Greedily make permanent assignments of click-points to clusters as follows. First the largest cluster claims all its candidate members, which are then removed as candidates from all other clusters. Continue the process until each click-point is assigned to a unique cluster.

This process determines a set of (non-empty) clusters and their sizes. We then calculate the observed "probability" p_j (based on our user data set) of the cluster j being clicked, as cluster size divided by total clicks observed. When the probability p_j of a certain cluster is sufficiently high, we can place a confidence interval around it for future populations,[5] using (1) as discussed below.

Each probability p_j estimates the probability of a cluster being clicked for a *single* click. Since a password has 5 clicks, we approximate the probability that a user chooses cluster j in a password by $P_j = 5 \times p_j$. Note that the probability for a cluster j increases slightly as other clicks occur (due to the constraint of 5 distinct clusters in a password); we ignore this in our present estimate of P_j.

Our results in Table 1 indicate a significant number of hot-spots for our sample of the full population (between 32 and 40 users per image). Previous "conservative" assumptions [43] were that half of the available alphabet of T-regions would be used in practice – or 207 in our case. If this were the case, and all T-regions in the alphabet were equi-probable, we would expect to see some clusters of size 2, but none of size 3 after 40 participants; we observed significantly more on all 17 images. Table 1 shows that some images were clearly worse than others. There were many clusters of size at least 5, and some even as large as 16 (see *tea* image). If a cluster in our lab

[4]Only two images had a significant number of skips: *paperclips* and *bee*. We note that this implies some passwords for these images were not memorable, and thus our results for them might be an overestimate of image security.

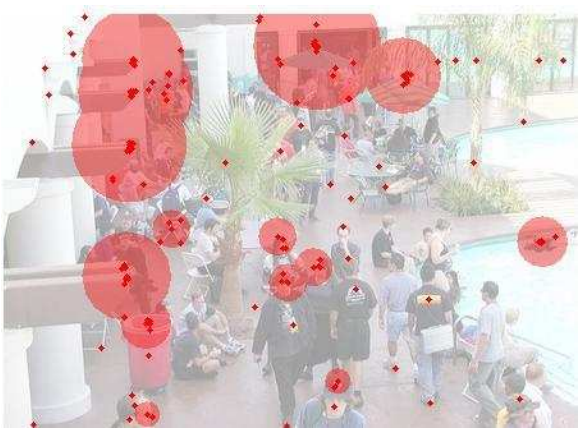
[5]More precisely, for future populations of users who are similar in background to those in our study.

study received 5 or more clicks – in which case we call it a *popular* or *high-probability* cluster – then statistically, this allows determination of a confidence interval, using:[6]

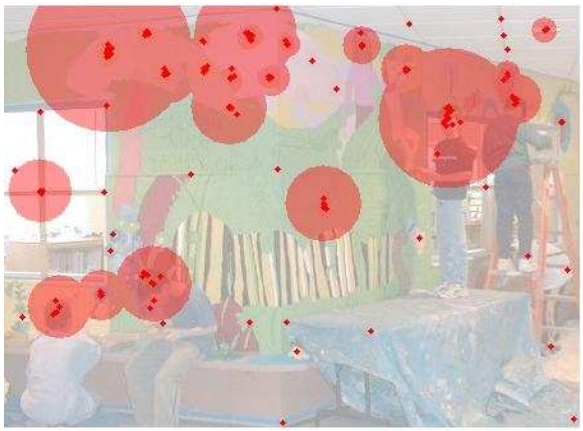
$$p \pm z_{\alpha/2}\sqrt{\frac{pq}{n}} \tag{1}$$

Here n is the total number of clicks (i.e., five times the number of users), p takes the role of p_j, $q = 1 - p$, and $z_{\alpha/2}$ is from a z-table. A confidence interval can be placed around p_j (and thus P_j) using (1) when $np \geq 5$ and $nq \geq 5$. For clusters of size $k \geq 5$, $p = \frac{k}{n}$, then $np = k$ and $nq = n - k$. In our case, $n \geq 32 \cdot 5$ and $n - k \geq 5$, as statistically required to use (1).

Table 2 shows these confidence intervals for four images, predicting that in future similar populations many of these points would be clicked by between 10-50% of users, and some points would be clicked by 20-60% of users with 95% confidence ($\alpha = .05$). For example, in Table 2(a), the first row shows the highest frequency cluster (of size 13); as our sample for this image was only 35 users, we observed 37.1% of our participants choosing this cluster. Using (1), between 17.7% and 56.6% of users from future populations are expected to choose this same cluster (with 95% confidence).

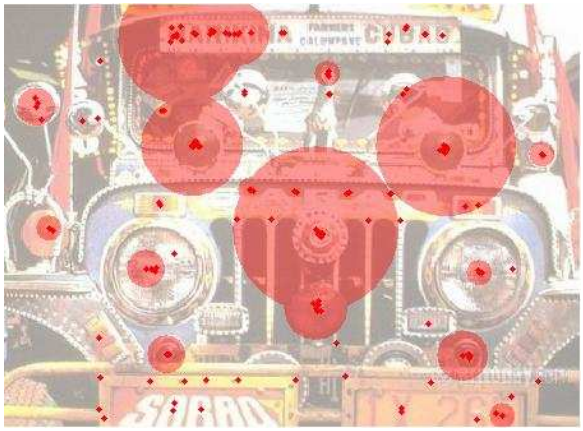


(a) *pool* (originally from [42, 43]).



(b) *mural* (originally from [42]).

(c) *philadelphia* (originally from [42]).



(d) *truck* (originally from [11]).

Figure 1: Observed click-points. Halo diameters are 10 times the size of the underlying cluster, illustrating its popularity.

[6]Equation (1) provides the $100(1 - \alpha)\%$ confidence interval for a population proportion [8, page 288].

Image Name	Size of most popular clusters					# clusters of size ≥ 5
	# 1	# 2	# 3	# 4	# 5	
1. *paperclips*	7	5	4	4	4	2
2. *cdcovers*	8	8	8	8	7	10
3. *philadelphia*	10	10	9	9	7	13
4. *toys*	11	8	8	7	6	8
5. *bee*	11	9	8	7	7	17
6. *faces*	11	10	7	6	5	5
7. *citymap-nl*	11	10	10	8	8	13
8. *icons*	12	7	7	6	6	13
9. *smarties*	12	9	7	7	6	12
10. *cars*	13	7	6	5	4	4
11. *pcb*†	13	8	7	7	6	6
12. *citymap-gr*	13	9	8	7	7	9
13. *pool*	13	12	12	11	11	9
14. *mural*	14	13	10	8	7	15
15. *corinthian*	14	13	12	8	6	7
16. *truck*	15	14	13	13	13	10
17. *tea*	16	6	6	5	5	8

Table 1: The five most popular clusters (in terms of size, i.e., # of times selected), and # of popular clusters ($size \geq 5$). Results are from 32-40 users, depending on the image, for the final passwords created on each image. †For *pcb*, which shows only 6 clusters of size ≥ 5, the size of clusters 2-5 become 5, 5, 4, and 3 when counting at most one click from each user (cf. footnote in Section 3).

(a) *pool* image			(b) *mural* image		
Cluster size	P_j	95% CI (P_j)	Cluster size	P_j	95% CI (P_j)
13	0.371	(0.177; 0.566)	14	0.400	(0.199; 0.601)
12	0.343	(0.156; 0.530)	13	0.371	(0.177; 0.566)
12	0.343	(0.156; 0.530)	10	0.286	(0.114; 0.458)
11	0.314	(0.134; 0.494)	8	0.229	(0.074; 0.383)
11	0.314	(0.134; 0.494)	7	0.200	(0.055; 0.345)

(c) *philadelphia* image			(d) *truck* image		
Cluster size	P_j	95% CI (P_j)	Cluster size	P_j	95% CI (P_j)
10	0.286	(0.114; 0.458)	15	0.429	(0.221; 0.636)
10	0.286	(0.114; 0.458)	14	0.400	(0.199; 0.601)
9	0.257	(0.094; 0.421)	13	0.371	(0.177; 0.566)
9	0.257	(0.094; 0.421)	13	0.371	(0.177; 0.566)
7	0.200	(0.055; 0.345)	13	0.371	(0.177; 0.566)

Table 2: 95% confidence intervals for the top 5 clusters found in each of four images. The confidence intervals are for the percentage of users expected to choose this cluster in future populations.

Tables 1 and 2 show the popularity of the hottest clusters; Table 1's last column also shows the number of popular clusters. The clustering effect evident in Tables 1 and 2 and Fig.1 clearly establishes that hot-spots are very prominent on a wide range of images. These hot-spots might significantly impact the practical security of full 5-click passwords, provided they can be translated into predictable full (or even partial) click-patterns. We pursue this further in Section 4.2.

As a partial summary, our results suggest that many images have significantly more hot-spots than would be expected if all T-regions were equi-probable. The *paperclips*, *cars*, *faces*, and *tea* images are not as susceptible to hot-spotting as others (e.g., *mural*, *truck*, and *philadelphia*). For example, the *cars* image had only 4 clusters of size at least 5, and only one with frequency at least 10. The *mural* image had 15 clusters of size at least 5, and 3 of the top 5 frequency clusters had frequency at least 10. Given our sample size for the *mural* image was only 36 users, these clusters are quite popular. This demonstrates the range of effect the background image can have (for the images studied).

Although previous work [42] suggests using intuition for choosing more secure background images (no further detail was provided), our results apparently show that intuition is not a good indicator. Of the four images used in other click-based graphical passwords studies, three showed a large degree of clustering (*pool*, *mural*, and *philadelphia*). Furthermore, two other images that we "intuitively" believed would be more secure background images were among the worst (*truck* and *citymap-nl*). The *truck* image had 10 clusters of size at least 5, and the top 5 clusters had frequency at least 13. Finding reliable automated predictors of more secure background images remains an open problem.[7] Thus, we use our lab study data to help choose two images for further analysis in our field study. We next explore the impact of hot-spotting across images.

3.2 Measurement and Comparison of Hot-Spotting for Different Images

To compare the relative impact of hot-spotting on each image studied, we calculated two formal measures of password security for each image: entropy $H(X)$, in equation (2), and in equation (3), the expected number of guesses $E(f(X))$ to correctly guess a password assuming the attacker knows the probabilities $w_i > 0$ for each password i.[8] Of course in general, the w_i are unknown, and our study gives only very coarse estimates; nonetheless, we find it helpful to use this to develop an estimate of which images will have the least impact from hot-spotting. For (2) and (3), n is the number of passwords (of probability > 0), random variable X ranges over the passwords, and w_i is calculated as described below.

$$H(X) = -\sum_{i=1}^{n} w_i \cdot log(w_i) \qquad \text{where } w_i = Prob(X = x_i) \tag{2}$$

$$E(f(X)) = \sum_{i=1}^{n} i \cdot w_i \quad \text{where } w_i \geq w_{i+1}, \text{ and } f(X) \text{ is number of guesses before success} \tag{3}$$

We calculate these measures based on our observed user data. For this purpose, we assume that users will choose from a set of click-points (following the associated probabilities), and combine 5 of them randomly. This assumption almost certainly over-estimates both $E(f(X))$ and $H(X)$ relative to actual practice, as it does not consider click-order patterns or dependencies. Thus, popular clusters likely reduce security by more than we estimate here.

We define C^V to be the set of all 5-permutations derivable from the clusters observed in our user study (as computed in Section 3.1). Using the probabilities p_j of each cluster, the probabilities w_i of

[7] Our preliminary work with simple measures (image segmentation, corner detection, and image contrast measurement) does not appear to offer reliable indicators.

[8] The relationship between $H(X)$ and $E(f(X))$ for password guessing is discussed by Massey [25].

each password in C^V are computed as follows. Pick a combination of 5 observed clusters $j_1, \ldots, j_5$ with respective probabilities $p_{j1}, \ldots, p_{j5}$. For each permutation of these clusters, calculate the probability of that permutation occurring as a password. Due to our instructions that no two click-points in a password can fall in the same T-region, these probabilities change as each point is clicked. Thus, for password $i = (j_1, j_2, j_3, j_4, j_5)$, $w_i = p_{j1} \cdot [p_{j2}/(1 - p_{j1})] \cdot [p_{j3}/((1 - p_{j1}) \cdot (1 - p_{j2}))] \cdot \ldots)$.

The resulting set C^V is a set of click-based graphical passwords (with associated probabilities) that coarsely approximates the effective password space if the clusters observed in our user study are representative of those in larger similar populations. We can order the elements of C^V using the probabilities w_i based on our user study.[9]

For comparison to previous "conservative" estimates that simply half of the available click-points (our T-regions) would be used in practice [43], we calculate C^U. C^U is the set of all permutations of clusters, assuming a uniformly random alphabet of size 207. We compare to C^U as it is a baseline that approximates what we would expect to see after running 32 users (the lowest number of users we have for any image), if previous estimates were accurate, and T-regions were equi-probable.

Fig. 2 depicts the entropy and expected number of guesses for C^V. Notice the range between images, and the drop in $E(f(X))$ from C^U to values of C^V. Comparison to the marked C^U values for (1) $H(X)$ and (2) $E(f(X))$ indicates that previous rough estimates are a security overestimate for practical security in all images, some much more so than others. This is at least partially due to click-points not being equi-probable in practice (as illustrated by hot-spots), and apparently also due to the previously suggested effective alphabet size (half of the full alphabet) being an overestimate. Indeed, a large alphabet is precisely the theoretical security advantage that these graphical passwords have over text passwords. If the effective alphabet size is not as large as previously expected, or is not well-distributed, then we should reduce our expectations of the security.

These results appear to provide fair approximation of the entropy and expected number of guesses for the larger set of users in the field study; we performed these same calculations again using the field study data. For both of the two images, the entropy measures were within one bit of values measured here (less than a bit higher for *pool*, and about one bit lower for *cars*). The number of expected guesses increased for both images (by 1.3 bits for *cars*, and 2.5 bits for *pool*).

The variation across all images shows how much of an impact the background image can be, even when using images that are "intuitively" good. For example, the image that showed the most impact from hot-spotting was the *mural* image, chosen for an earlier PassPoints usability study [42]. We note that the *paperclips* image scores best in the charted security measures (its $H(X)$ measure is within a standard deviation of C^U); however, 8 of 36 users who created a password on this image could not perform the subsequent login (and skipped it – as noted earlier), so the data for this image might be a security overestimate.

Overall, one can conclude that image choice can have a significant impact on the resulting security, and that developing reliable methods to filter out images that are the most susceptible to hot-spotting would be an interesting avenue for future research.

We used these formal measures to make an informed decision on which images to use for our field study. Our goal was to give the PassPoints scheme the best chance (in terms of anticipated security) we could, by using one image (*cars*) that showed the least amount of clustering (with the best user success in creating a password), and also using another that ranked in the middle (*pool*).

[9] An ordered C^V could be used as the basis of an attack dictionary; this ordering could be much improved, for example, by exploiting expected patterns in click-order. See Section 4.2 for more details.

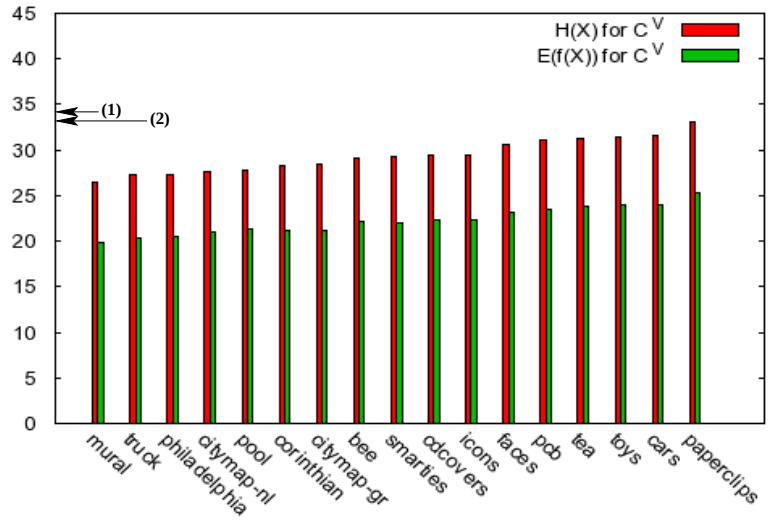


Figure 2: Security measures for each image (in bits). C^V is based on observed data from lab user study of 32–40 passwords (depending on image). For comparison to a uniform distribution, (1) marks $H(X)$ for C^U, and (2) marks $E(f(X))$ for C^U.

4 Field Study (User Study #2) and Harvesting Attacks

Here we describe a 7-week or longer (depending on the user), university-approved field study of 223 users on two different background images. We collected click-based graphical password data to evaluate the security of this style of graphical passwords against various attacks. As discussed, we use the entropy and expected guesses measures from our lab study to choose two images that would apparently offer different levels of security (although both are highly detailed): *pool* and *cars*. The *pool* image had a medium amount of clustering (cf. Fig. 2), while the *cars* image had nearly the least amount of clustering. Both images had a low number of skips in the lab study, indicating that they did not cause problems for users with password creation.

Experimental Details. We implemented a web-based version of PassPoints, used by three first-year undergraduate classes: two were first year courses for computer science students, while the third was a first year course for non-computer science students enrolled in a science degree. The students used the system for at least 7 weeks to gain access to their course notes, tutorials, and assignment solutions. For comparison with previous usability studies on the subject, and our lab study, we used an image size of 451×331 pixels. After the user entered their username and course, the screen displayed their background image and a small black square above the image to indicate their tolerance square size. For about half of users (for each image), a 19×19 T-region was used, and for the other half, a 13×13 T-region.[10] The system enforced that each password had to be 5 clicks and that no click-point could be within $t = 9$ pixels of another. To complete initial password creation, a user had to successfully confirm their password once. After initial creation, users were permitted to reset their password at any time using a previously set secret question and answer.

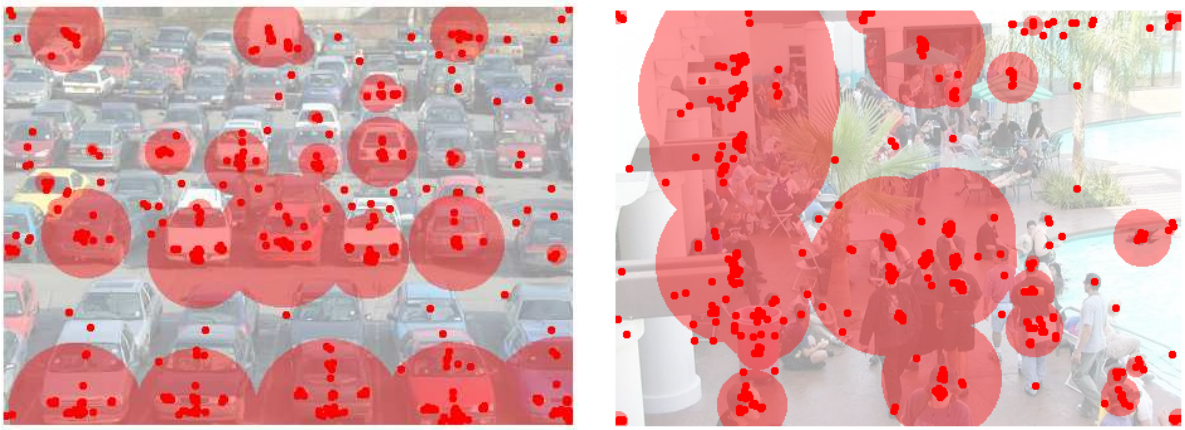
Users were permitted to login from any machine (home, school, or other), and were provided an online FAQ and help. The users were asked that they keep in mind that their click-points are a password, and that while they will need to pick points they can remember, not to pick points that someone else will be able to guess. Each class was also provided a brief overview of the system, explaining that their click-points in subsequent logins must be within the tolerance shown by a

[10] Analysis showed little difference between the points chosen for these different tolerance groups.

small square above the background image, and that the input order matters. We only use the final passwords created by each user that were demonstrated as successfully recalled at least once subsequently (i.e., at least once after the initial create and confirm). We also only use data from 223 out of 378 users that used the system, as this was the number that provided the required consent. Of the 223 users, 114 used *pool* and 109 used *cars* as a background image.

4.1 Field Study Hot Spots and Relation to Lab Study Results

Here we present the clustering results from the field study, and compare results to those on the same two images from the lab study (Section 3). Fig. 3b shows that the areas that were emerging as hot-spots from the lab study (recall Fig. 1a) were also popular in the field study, but other clusters also began to emerge. Fig. 3b shows that even our "best" image from the lab study (in terms of apparent resistance to clustering) also exhibits a clustering effect after gathering 109 passwords. Table 3 provides a closer examination of the clustering effect observed.



(a) *cars* (originally from [5])

(b) *pool* (originally from [42, 43])

Figure 3: Observed clustering (field study). Halo diameter is 5× the number of underlying clicks.

Image Name	Size of most popular clusters					# clusters of size ≥ 5
	# 1	# 2	# 3	# 4	# 5	
cars	26	25	24	22	22	32
pool	35	30	30	27	27	28

Table 3: Most popular clusters (field study).

These values show that on *pool*, there were 5 points that 24-31% of users chose as part of their password. On *cars*, there were 5 points that 20-24% of users chose as part of their password. The clustering on the *cars* image indicates that even highly detailed images with many possible choices have hot spots. Indeed, we were surprised to see a set of points that were this popular, given the small amount of observed clustering on this image from our smaller lab study.

The prediction intervals calculated from our lab study (recall Section 3) provide reasonable predictions of what we observed in the field study. For *cars*, the prediction intervals for 3 out of the 4 popular clusters were correct. For *pool*, the prediction intervals for 8 out of the 9 popular clusters were correct. The anomalous cluster on *cars* was still quite popular (chosen by 12% of users), but the lower end of the lab study's prediction interval for this cluster was 20%. The anomalous cluster

on *pool* was also still quite popular (chosen by 18% of users), but the lower end of the lab study's prediction interval for this cluster was 19%.

These clustering results (and their close relationship to the lab study's results) indicate that the points chosen from the lab study should provide a reasonably close approximation of those chosen in the field. This motivates our attacks based on the click-points harvested from the lab study.

4.2 Harvesting Attacks - Method and Results

We hypothesized that due to the clustering effect we observed in the lab study, human-seeded attacks based on data harvested from other users might prove a successful attack strategy against click-based graphical passwords. Here we describe our method of creating these attacks, and our results are presented below.

Table 4 provides the results of applying various attack dictionaries based on our harvested data, and their success rates when applied to our field study's password database.

C_u^R is a dictionary composed of all 5-permutations of click-points collected from u users.[11] In our lab study, $u = 33$ for *cars*, and $u = 35$ for *pool*. Thus, the size of C_u^R for *cars* is $P(165, 5) = 2^{36.7}$ entries, and for *pool* is $P(175, 5) = 2^{37.1}$ entries. C_u^V is a dictionary composed of all 5-permutations of the *clusters* calculated (using the method described in Section 3.1) from the click-points from u users. Thus, the alphabet size (and overall size) for C_u^V is smaller under the same number of users than in a corresponding C_u^R dictionary. Note that all of these dictionary sets can be computed on-the-fly from base data as necessary, and thus need not be stored.

Set	*cars* ($u = 33$)					*pool* ($u = 35$)				
	m	bitsize	# passwords guessed out of 109			m	bitsize	# passwords guessed out of 114		
			avg	min	max			avg	min	max
C_u^R	165	36.7	37(34%)	†	†	175	37.1	59(52%)	†	†
C_u^V	104	33.4	22(20%)	†	†	77	31.1	41(36%)	†	†
C_{25}^R	125	34.7	25(23%)	11(10%)	30(28%)	125	34.7	37(34%)	25(22%)	59(52%)
C_{25}^V	86	31.9	12(11%)	2(2%)	24(22%)	63	29.5	21(19%)	12(11%)	39(34%)
C_{20}^R	100	33.1	13(12%)	2(2%)	26(24%)	100	33.1	36(32%)	25(22%)	49(43%)
C_{20}^V	72	30.6	6(6%)	0(0%)	21(19%)	52	28.2	20(18%)	16(14%)	24(21%)
C_{15}^R	‡	‡	‡	‡	‡	75	30.9	26(23%)	19(16%)	45(39%)
C_{15}^V	‡	‡	‡	‡	‡	42	26.6	16(14%)	9(8%)	25(22%)

Table 4: Dictionary attacks using different sets. All subsets of users (after the first two rows) are the result of 10 randomly selected subsets of u short-term study user passwords. For rows 1 and 2, note that $u = 33$ and 35. m is the alphabet size, which defines the dictionary bitsize. See text for descriptions of C^V and C^R. †The first two rows use all data from the short-term study to seed a single dictionary, and as such, there are no average, max, or min values to report. ‡The *cars* image showed wide variability for random sets of 20 (as shown in the min-max variation for e.g., C_{20}^R), thus there would be little motivation for an attacker to seed a dictionary with data from as few as 15 users.

Table 4 illustrates the efficacy of seeding a dictionary with a small number of user's click-points. The most striking result shown is that initial password choices harvested from 15 users, in a setting

[11] Note C_u^R bitsize is a slight overestimate, as there are some combinations of points that would not constitute a valid password, due to two or more points being within $t = 9$ pixels of each other. If this were taken into account, our attacks would be slightly better.

Dictionary	*cars*			*pool*		
	m	bitsize	# passwords guessed	m	bitsize	# passwords guessed
$C^R_{20,longterm}$	100	33.1	29/89 (33%)	100	33.1	52/94 (55%)
$C^R_{10,longterm}$	50	27.9	23/99 (23%)	50	27.9	22/104 (21%)

Table 5: Dictionary attack results, using the first 20 and 10 users from the long term study to seed an attack against the others. m is the alphabet size. See text for descriptions of C^V and C^R.

Click-order pattern	*cars* image ($u = 25$)		*pool* image($u = 15$)	
	# passwords guessed of 109	dictionary size (bits)	# passwords guessed of 114	dictionary size (bits)
C^V_u (with no pattern)	9 (8%)	31.7	18 (16%)	26.4
LR, RL, CW, CCW, TB, BT	7 (6%)	28.3	18 (16%)	23.0
LR, RL	7 (6%)	26.5	16 (14%)	21.7
TB,BT	3 (3%)	21.3	3 (3%)	16.3
CW, CCW	0 (0%)	26.5	2 (2%)	23.1
DIAG	7 (6%)	22.3	12 (11%)	17.3

Table 6: Effect of incorporating click-order patterns on dictionary size and success, as applied to a representative dictionary of clusters gathered from u users. Results indicate that the DIAG pattern produces the smallest dictionary, and still guesses a relatively large number of passwords.

where long term recall is not required, can be used to predict (on average) 23% of user passwords for *pool* (see C^R_{15}) . As we expected, *cars* was not as easily attacked as *pool*; more user passwords are required to seed a dictionary that achieves similar success rates (see C^R_{25}). Table 5 provides the results of using a small set of field-study user passwords to seed an attack against the passwords of other users from the same population. This result shows that there is a difference between the passwords selected by the lab study and field study user's final passwords; however, there is enough similarity between the two groups to launch effective attacks using the lab-harvested data.

Next we examined the effect of click-order patterns to reduce our dictionary sizes. For each image, we select one dictionary to optimize with click-order patterns. This dictionary is one of the ten randomly selected C^V subsets that were averaged (results of this average are in Table 4). We selected the dictionary whose guessing success was closest to the average reported in Table 4. The success rate that these dictionaries achieve (before applying click-order patterns) is provided in the first row of Table 6.

We hypothesized that many users will choose passwords in one (or a combination) of six general patterns: right to left (RL), left to right (LR), top to bottom (TB), bottom to top (BT), clockwise (CW), and counter-clockwise (CCW). Diagonal (DIAG) is a combination of a consistent vertical and horizontal direction (e.g., both LR and TB).[12] We apply our base attack dictionaries (one for each image), under various sets of these click-order pattern constraints to determine their success rates and dictionary sizes.

The results shown in Table 6 indicate that, on average for the *pool* image, using only the diagonal constraint will reduce the dictionary size to 17 bits, while still cracking 11% of passwords. Similarly, for the *cars* image, using only this constraint will reduce the dictionary to 22 bits, while

[12]Straight lines also fall into this category; for example, when (x_i, y_i) is a horizontal and vertical pixel coordinate, the rule for LR is $(x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5)$, so a vertical line of points would satisfy this constraint.

still cracking 6% of passwords. The success rate of our harvesting attack is comparable to recent results on cracking text-based passwords [22], where 6% of passwords were cracked with a 1.2 million entry dictionary (2 bits less than our DIAG dictionary based on harvested points of 25 users for *cars*, and 3 bits more for DIAG based on 15 users for *pool*). Furthermore, unlike most text dictionaries, we do not need to store the entire dictionary as it is generated on-the-fly from the alphabet. At best, this indicates that these graphical passwords are slightly less secure than the text-based passwords they have been proposed to replace. However, the reality is likely worse. The analogy to our attack is collecting text passwords from 15-25 users, and generating a dictionary based on all permutations of the characters harvested, and finding it generated a successful attack. The reason most text password dictionaries succeed is due to known dependent patterns in language (e.g., using di or tri-grams in a Markov model [28]). The obvious analogy to this method has not been yet attempted, but would be another method of further reducing the dictionary size.

5 Purely Automated Attacks Using Image Processing Tools

Here we investigate the feasibility of creating an attack dictionary for click-based graphical passwords by purely automated means. Pure automation would side-step the need for human-seeding (in the form of harvesting points), and thus should be easier for an attacker to launch than the harvesting attacks presented in Section 4. We create this attack dictionary[13] by modelling user choice using a set of image processing methods and tools. The idea is that these methods may help predict hot-spots by automated means, leading to more efficient search orderings for exhaustive attacks. This could be used for modeling attackers constructing attack dictionaries, and proactive password checking.

5.1 Identifying Candidate Click-Points

We begin by identifying details of the user task in creating a click-based graphical password. The user must choose a set of points (in a specific order) that can be remembered in the future. We do not focus on mnemonic strategies for these automated dictionaries (although they could likely be improved using the click-order patterns from Section 4.2), but rather the basic features of a point that define candidate click-points. To this end, we identify a *candidate click-point* to be a point which is: (1) *identifiable* with precision within the system's error tolerance; and (2) *distinguishable* from its surroundings, i.e., easily picked out from the background. Regarding (1), as an example, the *pool* image has a red garbage can that is larger than the 19×19 error tolerance; to choose the red garbage can, a user must pick a *specific* part of it that can be navigated to again (on a later occasion) with precision, such as the left handle. Regarding (2), as an example, it is much easier to find a white logo on a black hat than a brown logo on a green camouflage hat.

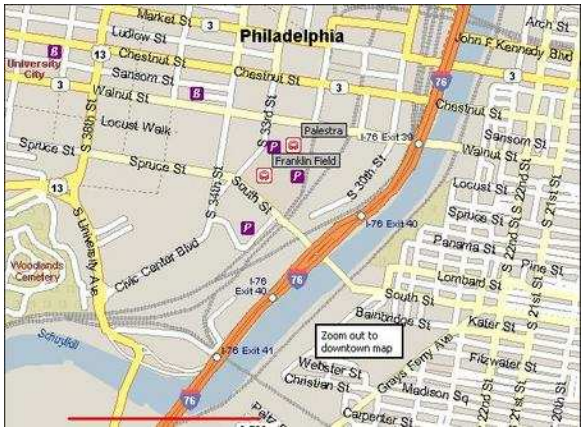
For modelling purposes, we hypothesize that the fewer candidate click-points (as defined above) that an image has, the easier it is to attack. We estimate candidate click-points by implementing a variation of Itti et al.'s bottom-up model of visual attention (VA) [16], and combining it with Harris corner detection [15].

Corner detection picks out the areas of an image that have variations of intensity in horizontal and vertical directions; thus we expect it should provide a reasonable measure of whether a point is identifiable. Itti et al.'s VA determines areas that stand out from their surroundings, and thus we expect it should provide a reasonable measure of a point's distinguishability. Briefly, VA calculates a saliency map of the image based on 3 channels (color, intensity, and orientation) over multiple scales.

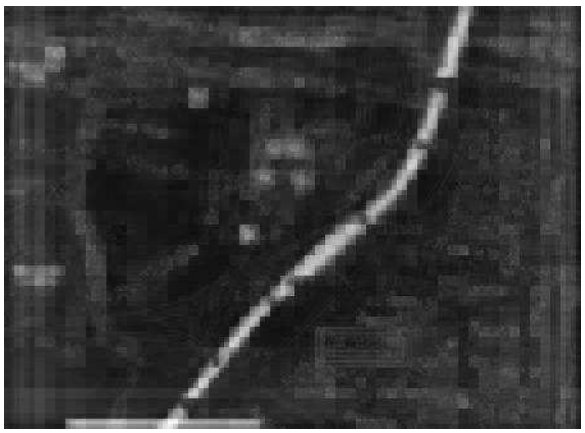
[13] As in Section 4.2, such a dictionary is created on-the-fly from base data, and need not be stored.

The saliency map is a grayscale image whose brighter areas (i.e., those with higher intensity values) represent more conspicuous locations. A viewer's focus of attention should theoretically move from the most conspicuous locations (represented by the highest intensity areas on the saliency map) to the least. We assume that users are more likely to choose click-points from areas which draw their visual attention.

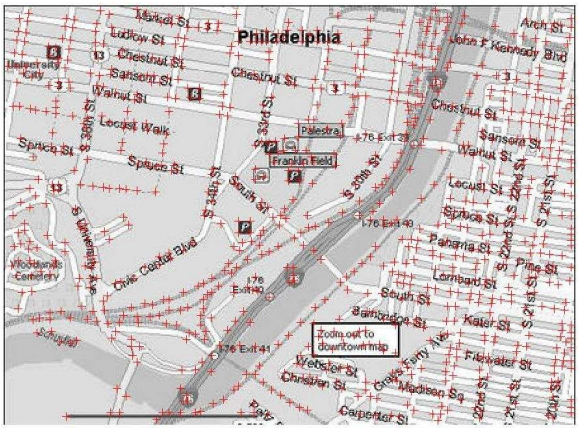
We implemented a variation of VA and combined it with Harris corner detection to obtain a prioritized list of candidate click-points *(CCP-list)* as follows. (1) Calculate a VA saliency map (see Fig. 4(b)) using slightly smaller scales than Itti et al. [16] (to reflect our interest in smaller image details). The higher-intensity pixel values of the saliency map reflect the most "conspicuous" (and distinguishable) areas. (2) Calculate the corner locations using the Harris corner detection function as implemented by Kovesi [21][14] (see Fig. 4(c)). (3) Use the corner locations as a bitmask for the saliency map, producing what we call a *cornered saliency map* (CSM). (4) Compute an ordered CCP-list of the highest to lowest intensity-valued CSM points. Similar to the focus-of-attention inhibitors used by Itti et al., we inhibit a CSM point (and its surrounding tolerance) once it has been added to the CCP-list so it is not chosen again (see Fig. 4(d)). The CCP-list is at least as long as the alphabet size (414), but is a prioritized list, ranking points from (the hypothesized) most to least likely.



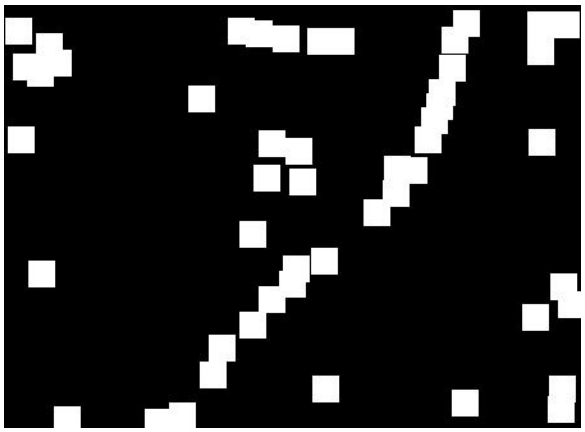
(a) Original image [42].



(b) Saliency map.



(c) Corner detection output.



(d) Cornered saliency map (CSM) after top 51 CCP-list points have been inhibited.

Figure 4: Illustration of our method of creating a CCP-list (best viewed electronically).

[14] As *harris(image, 1, 1000, 3)*

5.2 Model Results

We evaluated the performance of the CCP-list as a model of user choice using the data from both the lab and field user studies. We first examined how well the first half (top 207) of the CCP-list overlaps with the observed high-probability clusters from our lab user study (i.e., those clusters of size at least 5). We found that it works very well on the *icons*, *faces*, and *cars* images; by automated means, this half-alphabet found all high-probability clusters. It found most of the high-probability clusters on 11 of the 17 images. Most of the images that our model performed poorly on appeared to be due to the saliency map algorithm being overloaded with too much detail (*pcb*, *citymap-gr*, *paperclips*, *smarties*, and *truck* images). The other image on which this approach did not perform well (*mural*) appears to be due to the corner masking in step (3); the high probability points were centroids of circles.

To evaluate how well the CCP-list works at modelling users' *entire* passwords (rather than just a subset of click-points within a password), we used the top ranked one-third of the CCP-list values (i.e., the top 138 points for each image) to build a graphical dictionary and carry out a dictionary attack against the observed passwords from both user studies (i.e., on all 17 images in the lab study, and the *cars* and *pool* images again in the field study). We found that for some images, this 35-bit dictionary was able to guess a large number of user passwords (30% for the *icons* image and 29% for the *philadelphia* map image). For both short and long-term studies, our tool guessed 9.1% of passwords for the *cars* image. A 28-bit computer-generated dictionary (built from the top 51 ranked CCP-list alphabet) correctly guessed 8 passwords (22%) from the *icons* image and 6 passwords (17%) from the *philadelphia* image. Results of this automated graphical dictionary attack are summarized in Table 5.

Image	passwords guessed (lab study)	passwords guessed (field study)
1. *paperclips*	2/36 (5.5%)	–
2. *cdcovers*	2/35 (5.7%)	–
3. *philadelphia*	10/35 (28.6%)	–
4. *toys*	2/39 (5.1%)	–
5. *bee*	1/40 (2.5%)	–
6. *faces*	0/32 (0.0%)	–
7. *citymap-nl*	1/34 (2.9%)	–
8. *icons*	11/37 (29.7%)	–
9. *smarties*	5/37 (13.5%)	–
10. *cars*	3/33 (9.1%)	10/109 (9.1%)
11. *pcb*	3/36 (8.3%)	–
12. *citymap-gr*	0/34 (0.0%)	–
13. *pool*	1/35 (2.9%)	2/114 (0.9%)
14. *mural*	1/36 (2.8%)	–
15. *corinthian*	3/35 (8.6%)	–
16. *truck*	1/35 (2.9%)	–
17. *tea*	2/38 (5.3%)	–

Figure 5: Passwords correctly guessed (using a 35-bit dictionary based on a CCP-list). The number of target passwords is different for most images (32 to 40 for the lab study).

Figure 5 shows that the CCP-list does a good job of modelling observed user choices for some images, but not all images. This implies that on some images, an attacker performing an automated attack is likely to be able to significantly cut down his search space. This method also seems to perform well on the images for which the visual attention model made more definite decisions – the saliency map shows a smaller number of areas standing out, as indicated visually by a generally darker saliency map with a few high-intensity (white) areas. An attacker interested in any one of a set of accounts could go after accounts using a background image that the visual attention model performed well on.

In essence, this method achieves a reduction (by leaving out some "unlikely" points) from a 43-bit full password space to a 35-bit dictionary. The 43-bit full password space is the

proper base for comparison here, since an actual attacker with no a priori knowledge must consider all T-regions in an image. However, we believe this model of candidate click-points could be improved through a few methods. The images that the model performed poorly on appeared to be due to failure in creating a useful visual attention model saliency map. The saliency maps seem to fail when there are no areas that stand out from their surroundings in the channels used in saliency map construction (color, intensity, and orientation). Further, centroids of objects that "stand out" to a user will not be included in this model (as only corners are included); adding object centroids to the bitmask is thus an avenue for improvement.

6 Related Work

In the absence of password rules, practical text password security is understood to be weak due to common patterns in user choice. In a dated but still often cited study, Klein [20] determined a dictionary of 3 million words (less than 1 billionth of the entire 8-character password space) correctly guessed over 25% of passwords. Automated password cracking tools and dictionaries that exploit common patterns in user choice include *Crack* [27] and *John the Ripper* [29]. More recently, Kuo et al. [22] found John the Ripper's English dictionary of 1.2 million words correctly guessed 6% of user passwords, and an additional 5% by also including simple permutations. In response to this well-known threat, methods to create less predictable passwords have emerged. Yan [44] explores the use of passphrases to avoid password dictionary attacks. Jeyaraman et al. [19] suggest basing a passphrase upon an automated newspaper headline. In theory, creating passwords using these techniques should leave passwords less vulnerable to automated password cracking dictionaries and tools, although Kuo et al. [22] show this may not be the case. Proactive password checking techniques (e.g., [35, 6, 2]) are commonly used to help prevent users from choosing weak passwords.

Many variations of graphical passwords are discussed in surveys by Suo et al. [36] and Monrose et al. [26]. We discuss two general categories of graphical passwords: recognition-based and recall-based. In the interest of brevity, we focus on the areas closest to our work: click-based graphical passwords, and practical security analyses of user authentication methods.

Typical recognition-based graphical passwords require the user to recognize a set (or subset) of K previously memorized images. For example, the user is presented a set of N ($> K$) images from which they must distinguish a subset of their K images. The user may be presented many panels of images before providing enough information to login. Examples are Déjà Vu [9], which uses random art images created by hash visualization [31]; *Passfaces* [34], whereby the set of images are all human faces; and *Story* [7], whereby the images are from a variety of photo categories (e.g., everyday objects, locations, food, and people), with users encouraged to create a story as a mnemonic strategy. In the cognitive authentication scheme of Weinshall [40], a user computes a path through a grid of images based on the locations of those from K. The end of the path provides a number for the user to type, which was thought to protect the values of K from observers; Golle et al. [13] show otherwise.

Recall-based schemes can be further described as cued or uncued. An uncued scheme does not provide the user any information from which to create their graphical password; e.g., DAS (Draw-A-Secret) [18] asks users to draw a password on a background grid. Cued schemes show the user something that they can base their graphical password upon. A click-based password using a single background image is an example of a cued graphical password scheme where the user password is a sequence of clicks on a background image. Blonder [4] originally proposed the idea of a graphical password with a click-based scheme where the password is one or more clicks on predefined image regions. In the Picture Password variation by Jansen et al. [17], the entire image is overlayed by a

visible grid; the user must click on the same grid squares on each login.

Birget et al. [3] allow clicking anywhere on an image with no visible grid, tolerating error through "robust discretization". Wiedenbeck et al. [41, 42, 43] implement this method as PassPoints, and study its usability including: general perception, error rates and the effect of allowed error tolerance, the effect of image choice on usability, and whether people can create passwords in a reasonable period of time. They report the usability of PassPoints to be quite good.

Regarding explorations of the effect of user choice, Davis et al. [7] examine this in a variation of Passfaces and Story (see above), two recognition-based schemes which essentially involve choosing an image from one or more panels of many different images. Their user study found very strong patterns in user choice, e.g., the tendency to select images of attractive people, and those of the same racial background. The high-level idea of finding and exploiting patterns in user choice also motivated our current work, although these earlier results do not appear directly extendable to (cued recall) click-based schemes that select unrestricted areas from a single background image. Thorpe et al. [38, 39] discussed likely patterns in user choice for DAS (mirror symmetry and small stroke count), later corroborated through Tao's user study [37]. These results also do not appear to directly extend to our present work, aside from the common general idea of attack dictionaries.

Lopresti et al. [23] introduce the concept of generative attacks to behavioral biometrics. Ballard et al. [1] generate and successfully apply a generative handwriting-recognition attack based on population statistics of handwriting, collected from a random sample of 15 users with the same writing style. In arguably the most realistic study to date of the threats faced by behavioral biometrics, they found their generative attacks to be more effective than attacks by skilled and motivated forgers [1]. Our most successful attack from Section 4.2 may also be viewed as generative in nature; it uses click-points harvested from a small population of users from another context (the lab study), performs some additional processing (clustering), and recombines subsets of them as guesses. Our work differs in its application (click-based graphical passwords), and in the required processing to generate a login attempt.

7 Further Discussion and Concluding Remarks

Harvesting data from a small number of human users allows quite effective guessing attacks against click-based graphical passwords. This makes individual users vulnerable to targeted (spear) attacks, as one should assume that an attacker could find out the background image associated with a target victim, and easily gather a small set of human-generated data for that image by any number of means. For instance, an attacker could collect points by protecting an attractive web service or contest site with a graphical password. Alternatively, an attacker could pay a small group of people or use friends. This at least partially defeats the hope to improve one's security in a click-based scheme through a customized image.

It appears that if these harvesting attacks are seeded with passwords created from a similar population, they are more likely to succeed with less computational effort. We found that seeding the attack using the first 20 user passwords from the long-term field study resulted in a noticeably stronger attack. On the *pool* image, the lab study seeding of 20 user's click-points (C_{20}^R) guessed an average of 32% of passwords, where the field study seeding guessed 55%. Similarly, for the *cars* image, the lab study seeding of 20 user's click-points (C_{20}^R) guessed an average of 12% of passwords, where the field study seeding guessed 33%. Each of these dictionaries contains 2^{33} entries.

Our purely automated attack using a combination of image processing measures – and which likely can be considerably improved – already gives cause for concern. For images on which Itti et al.'s [16] visual attention model worked well, our model appeared to do a reasonable job of

predicting user choice. For example, an automatically-generated 28-bit dictionary from our tools cracked 8 out of 37 (22%) observed passwords for the *icons* image, and 6 out of 35 (17%) for the *philadelphia* image. Our tools cracked 9.1% of passwords for the *cars* image in both the short-term lab and long-term field studies. Improvements to pursue include adding object centroids to the bitmask used in creating the cornered saliency map.

Naturally, our attack dictionaries could be used defensively, as part of proactive password checking [35, 6, 2]. Additionally, it may be best to avoid background images on which the visual attention model performs well (e.g., identifies some areas as being much more interesting than others). For example, the visual attention model performs better for Fig. 4 than on the *truck* image.

The success of our harvesting attack dictionaries appears to be related to the amount of hot-spotting on an image. The prevalence and impact of hot-spots contrasts earlier views which underplayed their potential impact, and suggestions [43] that any highly detailed image may be a good candidate. Our studies allow us to update previous assumptions that half of all click-regions on an image will be chosen by users. After collecting 570 and 545 points, we only observed 111 and 133 points (for *pool* and *cars* respectively); thus, one quarter to one third of all points (or more precisely, regions) would be a more reasonable estimate even from highly detailed images, and the relative probabilities of these regions should be expected to vary quite considerably.

Although our presented attacks are successful, there are many avenues for improvement. A natural step is to combine or refine a purely automated attack dictionary by clusters harvested from sets of users, as per our harvesting attacks. The two strategies appear to complement each other; where the purely automated attack had poor results for *pool*, the harvesting attack worked very well.

Our work initiates the exploration of other ways that click-based graphical passwords could be analyzed for patterns in user choice, considering simple patterns in click-order (e.g., sweeps from left to right). We expect this general direction will yield other results, including patterns due to mnemonic strategies (e.g., clicking all red objects).

Overall, the degree of hot-spotting confirmed by our studies, and the successes of the various attack strategies herein, seriously call into question the viability of click-based schemes like PassPoints in environments where off-line attacks are possible. Indeed in such environments, a 43-bit full password space is clearly insufficient to start with, so one would assume some tolerable level of password stretching (e.g., [14, 33]) would be implemented to increase the difficulty of attack. Thus an interesting remaining question is whether altering parameters (e.g., pixel sizes of images, tolerance settings, number of click-points) in an attempt to improve security can result in a system with acceptable security *and* usability simultaneously. This is unclear, and any proposal with significantly varied parameters would require new user studies exploring hot-spotting and usability.

Acknowledgments

We thank Sonia Chiasson and Robert Biddle for their cooperative effort with us in running the user studies. We are grateful to Prosenjit Bose, Louis D. Nel, Weixuan Li, and their Fall 2006 classes for participating in our field study. We also thank Anthony Whitehead for recommending relevant work on visual attention and image segmentation. The first author acknowledges NSERC for funding a Canada Graduate Scholarship. The second author acknowledges NSERC for funding a NSERC Discovery Grant and his Canada Research Chair in Network and Software Security.

References

[1] L. Ballard, F. Monrose, and D. Lopresti. Biometric Authentication Revisited: Understanding the Impact of Wolves in Sheep's Clothing. In *15th Annual USENIX Security Symposium*, pages 29–41, 2006.

[2] F. Bergadano, B. Crispo, and G. Ruffo. High dictionary compression for proactive password checking. *ACM Trans. Inf. Syst. Secur.*, 1(1):3–25, 1998.

[3] J.C. Birget, D. Hong, and N. Memon. Robust Discretization, with an Application to Graphical Passwords. *IEEE Transactions on Information Forensics and Security*, 1:395–399, 2006.

[4] G. Blonder. Graphical passwords. United States Patent 5,559,961, 1996.

[5] Ian Britton. `http://www.freefoto.com`, site accessed Feb. 2, 2007.

[6] C. Davies and R. Ganesan. BApasswd: A New Proactive Password Checker. In *16th National Computer Security Conference*, pages 1–15, 1993.

[7] D. Davis, F. Monrose, and M.K. Reiter. On User Choice in Graphical Password Schemes. In *13th USENIX Security Symposium*, 2004.

[8] J.L. Devore. *Probability and Statistics for Engineering and the Sciences*. Brooks/Cole Publishing Company, Pacific Grove, CA, USA, 4th edition, 1995.

[9] R. Dhamija and A. Perrig. Déjà Vu: A User Study Using Images for Authentication. In *9th USENIX Security Symposium*, 2000.

[10] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient Graph-Based Image Segmentation. *Int. J. Computer Vision*, 59(2), 2004. Code available from: `http://people.cs.uchicago.edu/~pff/segment/`.

[11] FreeImages.com. `http://www.freeimages.com`, site accessed Feb. 2, 2007.

[12] Freeimages.co.uk. `http://www.freeimages.co.uk`, site accessed Feb. 2, 2007.

[13] P. Golle and D. Wagner. Cryptanalysis of a Cognitive Authentication Scheme. Cryptology ePrint Archive, Report 2006/258, 2006. `http://eprint.iacr.org/`.

[14] J. A. Halderman, B. Waters, and E. W. Felten. A convenient method for securely managing passwords. In *Proceedings of the 14th International World Wide Web Conference*, pages 471–479. ACM Press, 2005.

[15] C.G. Harris and M.J. Stephens. A combined corner and edge detector. In *Proceedings Fourth Alvey Vision Conference*, pages 147–151, 1988.

[16] L. Itti, C. Koch, and E. Niebur. A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

[17] W. Jansen, S. Gavrilla, V. Korolev, R. Ayers, and Swanstrom R. Picture password: A visual login technique for mobile devices. NIST Report: NISTIR 7030, 2003.

[18] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin. The Design and Analysis of Graphical Passwords. In *8th USENIX Security Symposium*, 1999.

[19] S. Jeyaraman and U. Topkara. Have the cake and eat it too - infusing usability into text-password based authentication systems. In *21st ACSAC*, pages 473–482, 2005.

[20] D. Klein. Foiling the Cracker: A Survey of, and Improvements to, Password Security. In *The 2nd USENIX Security Workshop*, pages 5–14, 1990.

[21] P. D. Kovesi. MATLAB and Octave Functions for Computer Vision and Image Processing. Univ. Western Australia. Available from: `http://www.csse.uwa.edu.au/~pk/research/matlabfns/`.

[22] C. Kuo, S. Romanosky, and L.F. Cranor. Human Selection of Mnemonic Phrase-based Passwords. In *2nd Symp. Usable Privacy and Security (SOUPS)*, pages 67–78, New York, NY, 2006. ACM Press.

[23] Daniel P. Lopresti and Jarret D. Raim. The Effectiveness of Generative Attacks on an Online Handwriting Biometric. In *AVBPA*, pages 1090–1099, 2005.

[24] S. Madigan. Picture Memory. In John C. Yuille, editor, *Imagery, Memory and Cognition*, pages 65–89. Lawrence Erlbaum Associates Inc., N.J., U.S.A., 1983.

[25] J.L. Massey. Guessing and Entropy. In *ISIT: Proceedings IEEE International Symposium on Information Theory*, page 204, 1994.

[26] F. Monrose and M. K. Reiter. Graphical passwords. In L. Cranor and S. Garfinkel, editors, *Security and Usability*, chapter 9, pages 147–164. O'Reilly, 2005.

[27] A. Muffett. Crack password cracker, 2006. `http://www.crypticide.com/users/alecm/security/c50-faq.html`, site accessed Nov. 9, 2006.

[28] Arvind Narayanan and Vitaly Shmatikov. Fast Dictionary Attacks on Passwords Using Time-space Tradeoff. In *CCS '05: Proceedings of the 12th ACM Conference on Computer and Communications Security*, pages 364–372, 2005.

[29] Openwall Project. John the Ripper password cracker, 2006. `http://www.openwall.com/john/`, site accessed Nov. 9, 2006.

[30] Passlogix. `http://www.passlogix.com`, site accessed Feb. 2, 2007.

[31] A. Perrig and D. Song. Hash Visualization: A New Technique to Improve Real-World Security. In *International Workshop on Cryptographic Techniques and E-Commerce*, pages 131–138, 1999.

[32] M. Peters, B. Laeng, K. Latham, M. Jackson, R. Zaiyouna, and C. Richardson. A Redrawn Vandenberg and Kuse Mental Rotations Test: Different Versions and Factors That Affect Performance. *Brain and Cognition*, 28:39–58, 1995.

[33] N. Provos and D. Mazieres. A Future-Adaptable Password Scheme. In *Proceedings of the USENIX Annual Technical Conference*, 1999.

[34] Real User Corporation. About Passfaces, 2006. `http://www.realuser.com/about/aboutpassfaces.htm`, site accessed Nov. 9, 2006.

[35] E.H. Spafford. OPUS: Preventing Weak Password Choices. *Comput. Secur.*, 11(3):273–278, 1992.

[36] X. Suo, Y. Zhu, and G.S. Owen. Graphical Passwords: A Survey. In *21st Annual Computer Security Applications Conference (ACSAC)*, 2005.

[37] H. Tao. Pass-Go, a New Graphical Password Scheme. Master's thesis, University of Ottawa, 2006.

[38] J. Thorpe and P.C. van Oorschot. Graphical Dictionaries and the Memorable Space of Graphical Passwords. In *13th USENIX Security Symposium*, 2004.

[39] J. Thorpe and P.C. van Oorschot. Towards Secure Design Choices for Implementing Graphical Passwords. In *20th Annual Computer Security Applications Conference (ACSAC 2004)*. IEEE, 2004.

[40] D. Weinshall. Cognitive Authentication Schemes Safe Against Spyware (short paper). In *IEEE Symposium on Security and Privacy*, pages 295–300, 2006.

[41] S. Wiedenbeck, J. Waters, J.C. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Basic results. In *Human-Computer Interaction International (HCII 2005)*, 2005.

[42] S. Wiedenbeck, J. Waters, J.C. Birget, A. Brodskiy, and N. Memon. Authentication Using Graphical Passwords: Effects of Tolerance and Image Choice. In *Symp. Usable Priv. & Security (SOUPS)*, 2005.

[43] S. Wiedenbeck, J. Waters, J.C. Birget, A. Brodskiy, and N. Memon. PassPoints: Design and longitudinal evaluation of a graphical password system. *International J. of Human-Computer Studies (Special Issue on HCI Research in Privacy and Security)*, 63:102–127, 2005.

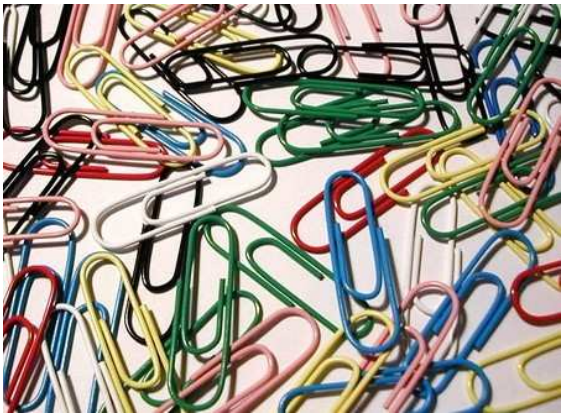
[44] J. Yan, A. Blackwell, R. Anderson, and A. Grant. Password Memorability and Security: Empirical Results. *IEEE Security and Privacy*, 2(5):25–31, 2004.

Appendix A - Subset of Images Used in Study

(a) *cars* [5]

(b) *pool* [42, 43]

(c) *mural* [42]



(d) *paperclips* [12]

Figure 6: Subset of images used.