A VISUAL DICTIONARY ATTACK ON PICTURE PASSWORDS

Amir Sadovnik and Tsuhan Chen

Department of Electrical and Computer Engineering, Cornell University

ABSTRACT

Microsoft's Picture Password provides a method to authenticate a user without the need of typing a character based password. The password consists of a set of gestures drawn on an image. The position, direction and order of these gestures constitute the password. Besides being more convenient to use on touch screen devices, this authentication method promises improved memorability in addition to improving the password strength against guessing attacks. However, how unpredictable is the picture password? In this paper we exploit the fact that different users are drawn to similar image regions, and therefore these passwords are vulnerable to guessing attacks. More specifically, we show that for portrait pictures users are strongly drawn to use facial features as gesture locations. We collect a set of Picture Passwords and, using computer vision techniques, derive a list of password guesses in decreasing probability order. We show that guessing in this order we are able to improve the likelihood of cracking a password within a limited number of guesses.

Index Terms— Graphical Password, Picture Password

1. INTRODUCTION

Graphical passwords have been proposed as alternatives to text based passwords starting in 1999. The basic idea was that a visual password would be more memorable and more secure than a text based password. The increased memorability comes from the fact that research has shown that people have an easier time remembering visual information versus verbal information. For example, the *dual-coding theory* [1] describes how visual and verbal information are processed in different parts of the brain. Whereas visual memories retain similar perceptual features to the physical observation, verbal memories are stored in symbolic form. This conversion process is what makes verbal memory more challenging.

The increased security comes from the fact that the symbol space can be larger than for character-based passwords. For example, there are 95 printable ascii characters. However, if we allow a user to select a point on a 15×20 grid, we have tripled the possible options. This grows exponentially with the number of characters allowed, and additional gestures provides for an even larger password space. However, the true strength of a password cannot be measured simply by looking at the space of all possible passwords (as has been known for character based passwords [2]). This is an upper-bound since people tend to choose easily guessable passwords, since they tend to be memorable [3]. In this paper, we extend the idea from character-based passwords to graphical passwords by using computer vision techniques to predict easily guessable graphical passwords.

Many different types of graphical passwords have been proposed (for a survey see [4]), each one dealing with the issues of memorability and security in different ways. In this paper we focus specifically on a variant of PassPoints [5] proposed by Microsoft for their new operating system "Windows 8" [6] . While PassPoints allows a series of taps on different locations of a given image, the Microsoft Picture Password (MPP) adds two additional possible gestures: the line and the circle. A user selects an image from his library and creates a password by drawing a combination of three gestures. Each gesture can be of a different type: tap, line or circle (See Fig. 1).

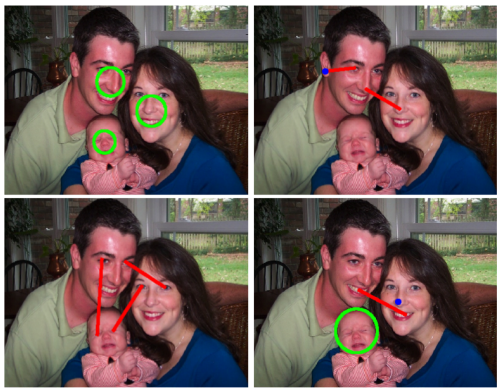

Fig. 1. An example of four picture passwords given by four different people over the same image. The password consists of a series of three gestures. These can be any combination of taps (blue), lines (red) and circles (green). The user is allowed to draw the gestures anywhere in the image. However, it is clear from the examples that users tend to choose similar locations. In this paper we use computer vision techniques to exploit this weakness, and measure how quickly a picture password can be guessed.

We focus specifically on the MPP since it is part of a new major operating system which is already used by 2.25% of web users. However, our method and results shed light on the broader subject of graphical passwords and raise important questions about their strength. Specifically, the problem of finding the balance between choosing a memorable password and a strong one exists for graphical passwords just as it exists for character-based ones. In addition, analyzing the graphical passwords gives additional insight in to more general subjects such as image saliency and memorability.

In this paper we show that people are drawn to use facial features when selecting the location of their gestures in portrait pictures. People tend to use these features (i.e. eyes, nose, mouth) as locations for taps, line beginnings and ends, and circle centers. We exploit this fact to perform a guessing attack on the Picture Password. We show that by learning the probabilities of using these facial features in a password, and creating a list of password guesses in decreasing probability order we can perform an effective attack on the MPP.

2. PREVIOUS WORK

As far as we know, there has not been any previous work attempting to attack the MPP. However, there have been attempts to construct dictionary attacks for PassPoints which is similar to the MPP, but only allows the tap gesture. In PassPoints, the user taps n points on the image in a successive order, which constitute his passwords.

There are two main categories for these attacks: human-seeded and purely automatic. Both rely on the fact that users tend to select similar locations in images. The human-seeded attacks [7, 8], need to have a small set of click points initialized by humans for the same images they are attempting to predict the password on. By clustering these points in a smart way they are able to identify regions which have a higher probability of being used in a password. Although this yields state-of-the-art results it is not applicable in the case of the MPP since each user will select their own personal image, and no prior information about the image exists.

The purely automated attacks try to use image processing techniques to identify regions which have a higher probability of being used in a password. For example, Thorpe et al. [8] use corner detection and a low-level saliency model to predict the probable regions. Salehi-Abari et al. [9] segments the image and uses regions centers as additional interest points. Using the corners and region centers as interest point maps, and scoring each point by its saliency provides an order to guess the passwords. Because this is a purely automated method, it relies on the assumption that the saliency map correlates with the password selection. This is not necessarily true.

We extend the previous work in a few ways. First, our work combines the advantages given by these two main lines of work. We learn from training passwords and guess passwords on unseen images. We do this by detecting points which are semantically similar between our training data and the new image, and use them to transfer our learned knowledge.

This also allows us to deal better with the different gestures allowed in the MPP. Although it makes sense to assume that there is a correlation between saliency and point selection, it is harder to make assumptions about where people would use different gestures (for example, drawing a circle vs. a line). Training data can help us deal with that problem. Finally, the fact that we are using higher-level semantic data (in our case: face detection) is also a novelty for this task. It has been shown [10] that semantic data can better predict saliency and we show that higher-level information provides better guesses for interest points in the password.

We deal with images which contain groups of people, and which contain large faces (each face's size is at least $\frac{1}{10}$ of the image). Although people can select any image as their background for their MPP, we assume that most images in many personal libraries contain faces, and that there is a high probability that a user would select one of these images. Although we deal specifically with portrait images, our general idea can be extended to other types of images as well. For example, although we only use face detection, an easy extension to this work would be to add a body pose detection algorithm for images which contain full views of people.

3. METHOD

We consider the set of three gestures needed to be guessed as three unknown words. The tap word includes a tap location (x, y), the line word includes the beginning and end locations (x_1, y_1, x_2, y_2), and the circle word includes the center location, the radius and direction of drawing (clockwise/counter clockwise) (x, y, r, d), similar to [11]. Given a new image, our guessing algorithm proposes passwords such that the more probable ones are proposed earlier. The goal is to guess a password in as few guesses as possible.

As in the MPP we allow an error tolerance when verifying the password. In our experiments we allow an error within a 40×40 pixel rectangle centered on the signified location. That is, when verifying a password, if the correct location is inside the 40×40 rectangle we consider it a match. For circle directions we do not allow any error, and for radius we allow an error of up to 20 pixels. This again is similar to the details given in [11] , with the slight difference that when verifying locations we place a rectangular error window while [11] place a circular one.

As a nominal evaluation of the size of the password space, we divide the image into a grid where each cell is 40×40 pixels. Guessing by iterating over all the centers of these grid squares, would cover the entire image. For example, for an image of size 600×800 pixels we divide the image into a 15×20 gird and our vocabulary size is:

- $15 \times 20 = 300$ for taps
- $(15 \times 20)^2 = 90000$ for lines (beginning and end point)
- $15 \times 20 \times 2 \times 10 = 6000$ for circles (2 directions, bin the radius into 10 bins).
- $(90000+6000+300)^3 = 8.12 \times 10^{14}$ for the entire password space

Since we have three different types of gestures with different types of words, we look at two sets of probabilities. First, for each gesture type we calculate $P(word|type)$. Second, for the entire password we calculate the probabilities of the different gesture type orders. Once we have these two sets of probabilities, we can rank all possible passwords by the total probability:

$$\begin{aligned} P(w_{ij}, w_{kl}, w_{mn}) =& P(t_1 = i, t_2 = k, t_3 = m) \\ & \times P(w_{ij}|i) P(w_{kl}|k) P(w_{mn}|m) \end{aligned} \quad (1)$$

Where $t_{(1,2,3)}$ are the types of the first, second and third gestures respectively, $i, k, m \in \{tap, line, circle\}$ and w_{ij} is the j^{th} word of type i.

First in Sec. 3.1 we describe how we calculate the probability for all possible words $p(w|t)$. Then in Sec. 3.2 we provide an algorithm to use this ranked list to guess an entire password.

3.1. Words-Probabilities

We would like to learn from a dataset of passwords which words users tend to use. For word location, we cannot use the absolute location in the image, as has been done previously in [7, 8], since we are attempting to guess a password on an unknown image. Instead, we do this by labeling the training images with higher-level semantic data, and then learning how often people use these regions in their passwords. For example, we may learn that people tend to circle a nose or draw lines between eyes. When presented with a new image, we simply have to automatically detect the regions which contain these semantic features, and rank these region types by how often they were used in the training data.

We theorize that since we are dealing with images with large faces, people will be drawn to the different facial features. Therefore, we use the algorithm given in [12] for face and pose detection. This algorithm provides 68 key points across each face in the image which correspond to certain areas of the face. For example, point 6 corresponds to the tip of the nose, point 67 corresponds to part of the left ear, etc. We manually cluster these 68 points to 8 main facial features: left eye, right eye, left eyebrow, right eyebrow, nose, mouth, chin, and the rest of the face.

Once we detect these regions on all training images, we count how many times each one was used in our training set. We do this by looking at a 40×40 rectangle centered at the training point, and adding a vote for all facial features which appear in this rectangle for the specific gesture type. Once we have counted all the votes we

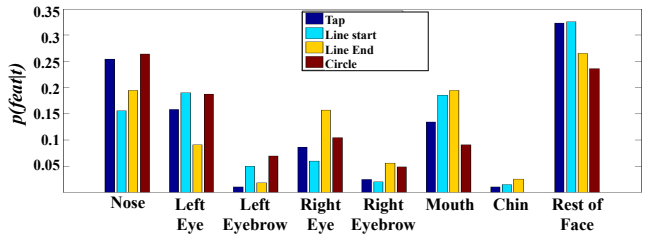

Fig. 2. The probability of each facial feature used given the gesture type: $p(feat|t)$. These are results from our collected passwords, normalized by the total number of gestures of each type.

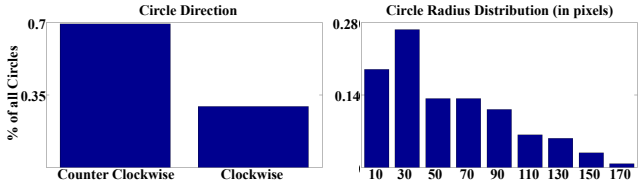

Fig. 3. The frequency of the two extra parameters for the circle gesture. The circle direction graph $p(direction)$ shows a clear preference for counterclockwise circles, while the radius distribution $p(radius)$ shows a preference for small circles.

normalize by the number of total gestures of that type to get $p(feat|t)$ where *feat* is one of our facial features. Fig. 2 shows this analysis on our collected training data (our password collection method is described in Sec. 4) . Some interesting trends appear in this analysis. For example, users tend to start lines at the left (eye/eyebrow) end end at the right (eye/eyebrow). In addition, the nose was the most popular region for circles.

In addition to location statistics we also needed information regarding the circle's radius and direction in order to construct a probability for words of that type. We can easily determine these probabilities by calculating the empirical probabilities from our training data. Fig. 3 presents the results from our password collection. As can be seen, about 70% of circles were drawn counterclockwise, which is consistent with research regarding the general population [13]. In addition, users tend to draw smaller circles (around 30 pixels in radius), which show that people tended to circle smaller features in the images vs. whole faces (face radii were on the order of 100 pixels).

Given a new image, we can now calculate the probability of each word. We first divide the image into a grid and for each grid cell we look at the facial features which it overlaps (regardles of which face it comes from). Once we label every grid cell with a facial feature, we can simply calculate the probabilities of all possible words for each type.

- Tap: $\frac{p(feat_{xy}|tap)}{size(feat_{xy})}$
- Line: $\frac{p(feat_{x_1y_1}|line\_start)}{size(feat_{x_1y_1})} \times \frac{p(feat_{x_2y_2}|line\_end)}{size(feat_{x_2y_2})}$
- Circle: $\frac{p(feat_{xy}|circle)}{size(feat_{xy})} \times p(direction) \times p(radius)$

Where $feat_{xy}$ is the facial feature at (x, y) and $size(feat_{xy})$ is the number of grid cells this feature occupies. Since we assume a uniform probability among all grid cells within a certain feature, we need to divide by the feature size. It is important to remember that since we are only looking at facial features, our word space is much smaller than the entire possible word space (for cells that do not contain any features we set the probability to zero). Therefore calculating probabilities for all facial feature words is possible.

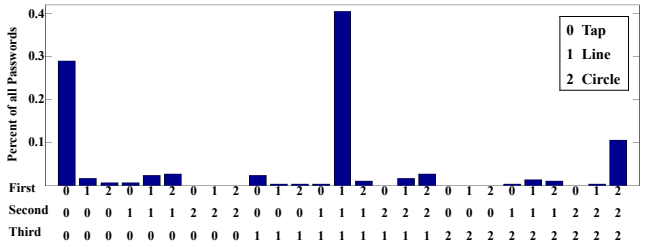

Fig. 4. Probability of the different gesture orders for our collected password database: $P(t_1, t_2, t_3)$. The x-axis represents all possible gesture orders ($3^3 = 27$).

3.2. Full Password Guessing

In order to guess an entire password we calculate the probabilities of the different gesture orders: $P(t_1 = i, t_2 = k, t_3 = m)$. We learn these probabilities from our training data and results are shown in Fig. 4. Surprisingly, users often use the same gesture type for all the gestures in the password. Although people did not use circles very often, when they did it was usually in a series of 3 circles. In fact, this trend was so strong (only 20% of the passwords have more than one type of gesture in their order), that we only consider the three dominant cases for our guessing algorithm. This significantly reduces our search space, even before we analyze the different locations.

Once we calculate the probabilities of all the words given the types in addition to the order probability we can simply guess passwords in decreasing order of probability (Eq. 1).

4. EXPERIMENTS AND RESULTS

We use Amazon Mechanical Turk to collect our training passwords. We created a web interface in which a user is asked to create a picture password. We use the $1 recognizer [14] to recognize the circle and line gestures. The directions and setup are very similar to the ones given by the original MPP. Once a user has created a password he is asked to confirm it by drawing it again. In order to ensure that users are truly selecting passwords which they can memorize we introduce a distractor before the third confirmation. After the second confirmation, a pop-up appears that covers the image and the user is asked a set of random trivia questions. The goal of this step is too distract the user from the image. Once the user has answered all the questions he is asked to confirm the password once more. If any of the confirmations are wrong the user is requested to start from the beginning (set up a new password). We collect 300 passwords from 300 separate users, over 30 images. We use images from the Images Of Groups Dataset [15] that contain 2 to 3 people.

It is important to note two main differences between our password collection algorithm and the actual MPP. First, our users do not use touch screens (which the MPP is mostly geared towards). We believe this is one of the reasons circles were used less in our password collection (circling with a mouse is much harder than circling with your finger). This is not a problem; by collecting additional touch-screen data, we can simply update the probabilities that our algorithm uses. The algorithm itself would not need to be altered. In addition, the MPP can also be used on non-touch screen computers and so this collection is valid as well.

The second difference is the fact that our users are not setting actual passwords, but only completing a task on Mechanical Turk. This means that they may be less inclined to create strong passwords. While we recognize this, we note that this type of password collection has been used previously for character based passwords [16]. In

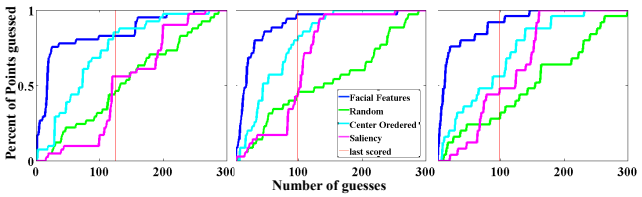

Fig. 5. Results on three images for our first experiment (interest point detection). There are a total of 300 grid-cells so all methods reach 100% by 300 guesses. The red vertical line shows how many grid cells have facial features (after that we guess randomly).

order to guess real word passwords, real passwords might need to be collected from real users.

In order to test our prediction performance we perform a leave-one-out cross validation. That is, we learn the probabilities from a set of 29 images (290 passwords) and then try to guess the passwords on the remaining image. In our first experiment we show that using facial features for this type of image is superior to using corner detection and low-level saliency (similar to [9]). In this experiment, we simply wish to show that most password words' locations fall within our facial feature regions. We examine all the points used for each image (taps, line beginnings and ends, centers of circles) and iteratively guess each grid cell on the image. At each step we count how many points we have guessed correctly. We compare using facial features with learned probability to 3 other guessing methods:

- Random: Randomly moving throughout the grid.
- Centered: Start guessing from the center and then move out.
- Saliency: Corner +regions centers ranked by saliency ([9])

Fig. 5 shows the results for three different test images. It is clear to see that our ranking considers more important points earlier on. Surprisingly, going out from the center outperformed using saliency. This is because most of the faces in the image tend to be closer to the center of the image, but the low level saliency model detects many points in the background and on the faces edges which tend to be used less in our password dataset.

In our second experiment we attempt to guess entire passwords using the ranking from Sec. 3. We compare to the following methods:

- Random: Guessing passwords from entire image in random order.
- All Image: Guessing passwords from entire image uniformly, using collected order statistics ($P(t_1, t_2, t_3)$).
- Full Face: Guessing passwords from the entire face uniformly, using collected order statistics ($P(t_1, t_2, t_3)$).
- All Features: Guessing passwords in decreasing score order (Eq. 1) with all facial features.

Fig. 6 presents our results for all 300 passwords (using leave-one-image-out cross validation). The *All Image* results show already how much is gained simply from using the order statistics, where 25% of the passwords are guessed within 3.4×10^{-6}% of the entire space. This method only guesses 80% of all the passwords, since the other 20% are passwords with different gestures (see Sec. 3.2). When we focus solely on the face region (without looking at the individual features), we are able to get an additional improvement. Although using this method we only guess 60% of the passwords correctly (the extra 20% include non facial regions) we begin guessing passwords earlier. Finally, using all the facial features including

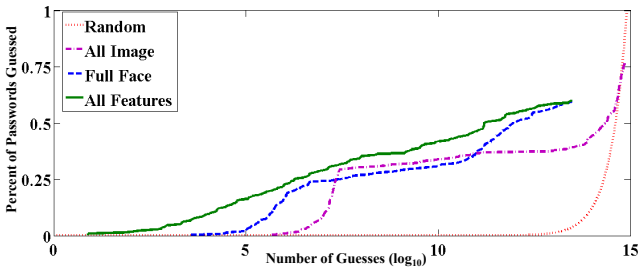

Fig. 6. The results of our guessing algorithm. For further details about the different methods see Sec. 4.

Fig. 7. Examples of two images which the algorithm guessed the most passwords (green background) vs. two images were the algorithm performed the worst (red background). The numbers represent the number of passwords guessed correctly.

the probabilities leads to the best results. We are able to start guessing passwords correctly very early on. In fact, 3 of the passwords (3 taps on the left eye) are guessed with 8 guesses. We also perform better than all other methods throughout the entire guessing algorithm.

The results shown in Fig. 6 are in log scale and so the time gains are very large. For example, if each guess takes 1 millisecond, it would take on the order of 1000 years to guess 25% of the passwords using a brute force method. However, using our ranked guess list we are able to guess 25% of the passwords in about 16 minutes.

Fig. 7 shows the two images where we performed best (green background) vs. the two images where we performed the worst (red background) and the number of passwords guessed correctly. The images we perform poorly on were usually due to misdetections in our images (i.e. the dog and the baby were not detected).

5. CONCLUSION

In this paper we presented a novel approach for guessing graphical passwords (and specifically the MPP) using computer vision. The general idea is to use high level semantic features to transfer learned knowledge from a training set of passwords to a new image. We show through experiments that this technique can guess passwords in a much shorter time than random guesses (we guess 50% of the passwords in 0.001% of the size of the entire space).

This work does not attempt to provide an exact method for actually cracking the MPP since the MPP only allows 5 wrong password guesses before a character-based password is required. Instead, it emphasizes the idea that the same vulnerabilities which lie in alphanumeric passwords, also exist in graphical passwords. More specifically, the fact that people tend to choose "easy" passwords, can be taken advantage of in graphical passwords as well.

There are many directions in which to extend this work. For example, an analysis of image choice for the MPP would be interesting. That is, an automatic algorithm which would recommend images which are harder to guess. In a similar fashion, different password policies can be investigated. For example perhaps a policy which does not allow a user to perform all of his gestures on a face. Finally, extending this work to other types of images which contain full bodies or no people at all would be interesting as well.

6. REFERENCES

[1] A. Paivio, *Mind and its evolution: A dual coding theoretical approach.* Lawrence Erlbaum Associates Publishers, 2007.

[2] W. Ma, J. Campbell, D. Tran, and D. Kleeman, "Password entropy and password quality," in *Network and System Security (NSS), 2010 4th International Conference on*. IEEE, 2010, pp. 583–587.

[3] J. Yan, "A note on proactive password checking," in *Proceedings of the 2001 workshop on New security paradigms*. ACM, 2001, pp. 127–135.

[4] R. Biddle, S. Chiasson, and P. Van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Computing Surveys (CSUR)*, vol. 44, no. 4, p. 19, 2012.

[5] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, "Passpoints: Design and longitudinal evaluation of a graphical password system," *International Journal of Human-Computer Studies*, vol. 63, no. 12, pp. 102 – 127, 2005.

[6] J. Johnson, S. Seixeiro, Z. Pace, G. V. D. Bogert, S. Gilmour, L. Siebens, and K. Tubbs, "Picture gesture authentication," US Patent 20 120 304 284, November, 2012. [Online]. Available: http://www.freepatentsonline.com/y2012/0304284.html

[7] P. van Oorschot and J. Thorpe, "Exploiting predictability in click-based graphical passwords," *Journal of Computer Security*, vol. 19, no. 4, pp. 669–702, 2011.

[8] J. Thorpe and P. van Oorschot, "Human-seeded attacks and exploiting hot-spots in graphical passwords," in *Proceedings of the 16th USENIX Security Symposium*, 2007.

[9] A. Salehi-Abari, J. Thorpe, and P. Van Oorschot, "On purely automated attacks and click-based graphical passwords," in *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*. IEEE, 2008, pp. 111–120.

[10] W. Einhäuser, M. Spain, and P. Perona, "Objects predict fixations better than early saliency," *Journal of Vision*, vol. 8, no. 14, 2008.

[11] Z. Pace. (2011, December) Signing in with a picture password. [Online]. Available: http://blogs.msdn.com/b/b8/archive/2011/12/16/signing-in-with-a-picture-password.aspx

[12] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2879–2886.

[13] J. Goodnow, S. Friedman, M. Bernbaum, and E. Lehman, "Direction and sequence in copying the effect of learning to write in english and hebrew," *Journal of Cross-Cultural Psychology*, vol. 4, no. 3, pp. 263–282, 1973.

[14] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes," in *User Interface Software and Technology*, 2007, pp. 159–168.

[15] A. Gallagher and T. Chen, "Understanding images of groups of people," in *Proc. CVPR*, 2009.

[16] P. Kelley, S. Komanduri, M. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L. Cranor, and J. Lopez, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 523–537.