

we will find a lower bound on the second sum. To do so, we first derive an upper bound on  $c_l$  for  $l$  satisfying  $l < (\rho^* - \epsilon)n$ . We start with

$$c_l \leq \binom{n}{l} \leq 2^{nH(l/n)}. \quad (14)$$

The second inequality follows from Lemma 2.4.2 in [2] and holds for any  $l < n/2$  for sufficiently large  $n$  (e.g.,  $n > n_1$ ). Using the fact that  $H(x)$  is increasing on  $[0, 1/2]$ , from Taylor expansion of  $H(x)$  at  $\rho^*$

$$2^{nH(l/n)} \leq 2^{nH(\rho^* - \epsilon)} = 2^{n(\alpha - \epsilon H'(\xi))} \quad (15)$$

where  $\rho^* - \epsilon < \xi < \rho^*$ . Finally, because  $H'$  is decreasing on the same interval

$$c_l \leq 2^{\alpha n} 2^{-n\epsilon H'(\xi)} < 2^{\alpha n} 2^{-n\epsilon H'(\rho^*)} \quad (16)$$

for any  $l < (\rho^* - \epsilon)n$ .

We now obtain a lower bound for  $R_a$ . Writing  $l_0 = \lfloor (\rho^* - \epsilon)n \rfloor$ , from (13)

$$\begin{aligned} R_a &\geq \sum_{l=l_0+1}^{\rho n} \frac{l c_l}{2^{\alpha n}} \geq (\rho^* - \epsilon)n \sum_{l=l_0+1}^{\rho n} \frac{c_l}{2^{\alpha n}} \\ &= (\rho^* - \epsilon)n \left( 1 - \sum_{l=0}^{l_0} \frac{c_l}{2^{\alpha n}} \right) \end{aligned} \quad (17)$$

because  $\sum_{l=0}^R c_l = 2^{\alpha n}$ . Using (16)

$$\begin{aligned} R_a &\geq (\rho^* - \epsilon)n \left( 1 - (\rho^* - \epsilon)n \cdot 2^{-n\epsilon H'(\rho^*)} \right) \\ &= (\rho^* - \epsilon)n (1 - \delta(n)) \end{aligned} \quad (18)$$

where  $\delta(n) \rightarrow 0$  exponentially fast with  $n \rightarrow \infty$ . Combining this result with  $\rho_a \leq \rho \leq \rho^* + \epsilon$ , we obtain the following bounds for the average distance to code in terms of the relative quantities (for  $n > \max(n_0, n_1)$ ):

$$(\rho^* - \epsilon)(1 - \delta(n)) \leq \rho_a \leq \rho \leq \rho^* + \epsilon \quad (19)$$

which proves the claim because  $\epsilon > 0$  was arbitrary and  $\delta(n) \rightarrow 0$  for  $n \rightarrow \infty$ .

#### ACKNOWLEDGMENT

The authors would like to thank J. Bierbrauer, P. Lisonvěk, and M. Goljan for many useful discussions.

#### REFERENCES

- [1] J. Bierbrauer, *On Crandall's Problem*. 1998 [Online]. Available: <http://www.ws.binghamton.edu/fridrich/covcodes.pdf>, Personal Communication.
- [2] G. D. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*. Elsevier, North-Holland Mathematical Library, 1997, vol. 54.
- [3] R. Crandall, "Some notes on steganography," *Steganography Mailing List* [Online]. Available: <http://os.inf.tu-dresden.de/westfeld/crandall.pdf>, 1998.

- [4] J. Fridrich, M. Goljan, and D. Soukal, "Wet paper codes with improved embedding efficiency," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 1, pp. 102–110, Mar. 2006.
- [5] F. Galand and G. Kabatiansky, "Information hiding by coverings," in *Proc. ITW*, Paris, France, 2003, pp. 151–154.
- [6] M. Goljan, J. Fridrich, and T. Holotyak, "New blind steganalysis and its implications," in *Proc. SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII*, E. Delp, III and P. W. Wong, Eds., San Jose, CA, Jan. 16–19, 2006, vol. 6072, pp. 1–13.
- [7] A. D. Ker, "Steganalysis of LSB matching in grayscale images," *IEEE Signal Process. Lett.*, vol. 12, no. 6, pp. 441–444, Jun. 2005.
- [8] M. van Dijk and F. Willems, "Embedding information in grayscale images," in *Proc. 22nd Symp. Information and Communication Theory Benelux*, Enschede, The Netherlands, May 15–16, 2001, pp. 147–154.
- [9] A. Westfeld, "High capacity despite better steganalysis (F5—a steganographic algorithm)," *Information Hiding*, 4th International Workshop I. S. Moskowitz, Ed. New York, Springer-Verlag, 2001, vol. 2137, Lecture Notes Comput. Sci., pp. 289–302.
- [10] F. J. M. Williams and N. J. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North-Holland, 1977.

## Graphical Passwords Based on Robust Discretization

Jean-Camille Birget, Dawei Hong, and Nasir Memon

**Abstract**—This paper generalizes Blonder's graphical passwords to arbitrary images and solves a robustness problem that this generalization entails. The password consists of user-chosen click points in a displayed image. In order to store passwords in cryptographically hashed form, we need to prevent small uncertainties in the click points from having any effect on the password. We achieve this by introducing a robust discretization, based on multigrid discretization.

**Index Terms**—Images, passwords, robust discretization.

### I. INTRODUCTION

Graphical passwords were first proposed by Blonder [1]. In his scheme, a password uses an image in which many small regions have been delineated. The user has to choose some of these regions as a password and in order to log in later, the user must click in each of the chosen regions (with a mouse or a stylus). The user must remember the chosen click regions and keep them secret. Several implementations of this idea were given by [9]. Another version of click regions, this time with movement, was carried out in [4]. Somewhat different graphical password schemes (based on drawings in a grid) were introduced and

Manuscript received September 14, 2005; revised May 29, 2006. The work of J.-C. Birget was supported by the National Science Foundation (NSF) under Grant DMS-9970471, in part by the National Science Foundation (NSF) under Grant CCR-0310793, and in part by a Rutgers University ISATC Pilot Grant. The work of D. Hong was supported in part by the National Science Foundation (NSF) under Grant CCR-0310793 and in part by a Rutgers University ISATC Pilot Grant. The work of N. Memon was supported by National Science Foundation under grants. An earlier version of this paper appears in the *Cryptology ePrint Archive*, report 2003/168 (Aug. 2003). Preliminary work also appeared in [11]. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Roy A. Moxon.

J.-C. Birget and D. Hong are with the Department of Computer Science, Rutgers University, Camden, NJ 08102 USA (e-mail: birget@camden.rutgers.edu; dhong@camden.rutgers.edu).

N. Memon is with the Department of Computer and Information Science, Polytechnic University, Brooklyn, NY 11201 USA (e-mail: memon@poly.edu). Digital Object Identifier 10.1109/TIFS.2006.879305

analyzed in [5]. Yet, other graphical password schemes have been invented, based on image recognition [2], [8], [10].

The click region passwords of Blonder have a limitation in that the set of possible click regions must be predefined; they are part of the design of the image. This implies that the users cannot provide images of their own for making passwords, and that users cannot choose click places that are not among the preselected ones. Moreover, a complex “natural” image (landscape, cityscape, art, photo of individual people or groups, etc.) is not easy to subdivide into fixed and recognizable click regions.

On the other hand, allowing arbitrary click points leads to a robustness problem: Usually a person will not be able to click repeatedly on exactly the same places, which means that the password clicked on by the user is “a little” different from the password that was originally chosen. Allowing approximately correct passwords, however, prevents the use of cryptographic password hashing (also known as “password encryption”) since passwords that are approximately (but not exactly) the same will usually have very different hash values. Cryptographic password hashing is important because it enables secure storage of passwords in an insecure storage (and backup) environment.

Thus, our passwords need to be discretized. But this is not good enough by itself because of the edge problem of discretization: If the place pointed to in the image is near a grid line, then slight perturbations in the pointing can lead to significant changes in the output of the discretization. In Blonder-type graphical passwords, this would often prevent a legitimate user from logging in. So, we need robust discretization in which outputs are not influenced by small uncertainties in the input. We will achieve such a discretization by using a small number of discretization grids simultaneously.

Our graphical password scheme, described in detail in Section III, allows arbitrary images and has three components: image handling (which allows a user to choose an image from a data base, or update the image data base); password selection (during which a user chooses click points in a chosen image); and login (during which a user clicks close to the previously chosen click points in the previously chosen image).

*Overview:* A robust discretization method is described in Section II. The main motivation of the paper is the application of robust discretization to graphical passwords, as described in Section III. Security is briefly discussed in Section IV.

## II. ROBUST DISCRETIZATION

We will start with a discussion of discretization, and then develop a discretization which is robust. We define the concepts of robustness and of safety within a discretization grid, and give a rigorous proof of robustness for our method.

Discretization (also called quantization) of data consists of approximating a continuum, or a very large discrete set, by a discrete set of limited size. To fix the terminology, we describe a discretization of a two-dimensional (2-D) rectangular gray picture. The picture is given by a function  $[0, a] \times [0, b] \rightarrow [0, 1]$ , where  $[0, a], [0, b]$  are intervals in the reals  $\mathbb{R}$ , or in the integers  $\mathbb{Z}$ . In this paper, we are only interested in the discretization of the domain of the picture (i.e., the rectangle  $R_2 = [0, a] \times [0, b]$ ). To discretize  $R_2$ , we choose a positive number  $q$  (called the quantum) and an offset  $(\varphi, \psi)$  (where  $|\varphi|, |\psi| < q$ ), and we superimpose a square grid on the rectangle. The grid has  $\lfloor a/q \rfloor + 1$  vertical lines

$$(V_m) \quad x = qm + \varphi, \quad \text{where } m = 0, \dots, \lfloor a/q \rfloor$$

and  $\lfloor b/q \rfloor + 1$  horizontal lines

$$(H_n) \quad y = qn + \psi, \quad \text{where } n = 0, \dots, \lfloor b/q \rfloor.$$

This subdivides  $R_2$  into grid squares of side-length  $q$ ; near the borders of  $R_2$ , the grid squares are truncated. The discretization yields a grid map, which tells us which points of the rectangle  $R_2$  are mapped to which grid vertices

$$g : (x, y) \in [0, a] \times [0, b] \mapsto \left( \left\lfloor \frac{x - \varphi}{q} \right\rfloor, \left\lfloor \frac{y - \psi}{q} \right\rfloor \right).$$

The set of points of  $R_2$  that are mapped to a given grid point  $(m, n)$  is

$$g^{-1}(m, n) = \{(x, y) \in R_2 : qm + \varphi \leq x < q(m + 1) + \varphi, \\ qn + \psi \leq y < q(n + 1) + \psi\}.$$

The set  $g^{-1}(m, n)$  is called a grid square; the grid map  $g$  maps this entire grid square, namely  $[qm, q(m + 1)) \times [qn, q(n + 1))$ , to the grid point  $(m, n)$ .

Notation for intervals: In order to avoid confusion with the point  $(a, b)$ , the open interval  $\{x : a < x < b\}$  is denoted by  $\langle a, b \rangle$ . The closed interval is denoted by  $[a, b]$ , and the half-open intervals are denoted by  $[a, b)$  and  $\langle a, b]$ .

Usually, the goal in the design of a good discretization is to minimize the loss of precision, or “quantization error” (i.e., the distance between data points  $(x, y)$  and their discretizations). Our goal is quite different: We want the discretization to keep outputs unchanged when inputs are slightly perturbed, and we are not so much concerned with the loss of precision. Indeed, in our applications, one of the main problems with discretization is the edge problem, mentioned in the Introduction: Important features of an image could be near grid lines  $V_m$  or  $H_n$ , and slight perturbations of a chosen location could then lead to significant changes in the output of the discretization. By definition, a discretization is robust if and only if it has two properties which are: 1) if a location pointed to is close to an originally chosen location (within a specified tolerance distance  $< r_1$ ), then the output is the same as for the originally chosen location and 2) if a location pointed to is at distance greater than  $r_2$  from the originally chosen location (for some specified tolerance distance  $r_2$  with  $r_2 > r_1$ ), the output is guaranteed to be different than for the originally chosen location. (Here, “pointing to” a location means, for example, pointing with a stylus, or clicking with a mouse.) In our application to graphical passwords, condition 1) guarantees the acceptance of approximately correct passwords, and condition 2) guarantees the rejection of significantly wrong passwords.

In order to make sure that all features of an image are at a safe distance from grid edges, we use several grids at the same time. It is fairly intuitive that in 2-D images, three grids are necessary and sufficient; then we can lay out the grids so that every point in the image is at a safe distance from the edges in at least one of the three grids (Fig. 1). In a  $d$ -dimensional “image,” we will show that  $d + 1$  grids with appropriate offsets are necessary and sufficient.

The “safe distance from the edges” is a parameter  $r > 0$ . The open  $r$ -disk around a point  $(x_1, \dots, x_d)$  is  $D_r(x_1, \dots, x_d) = \{(z_1, \dots, z_d) : \|(x_1, \dots, x_d) - (z_1, \dots, z_d)\| < r\}$ , where  $\|\cdot\|$  denotes the Euclidean norm in  $d$ -dimensional space. The following definition makes the phrase “a point is at a safe distance from the edges” precise.

*Definition 2.1:* Let  $r$  be any positive real number. A point  $(x_1, \dots, x_d)$  is  $r$ -safe in a  $d$ -dimensional square grid  $G$  if the open  $d$ -dimensional  $r$ -disk around this point  $(x_1, \dots, x_d)$  is entirely contained in one grid hypercube of  $G$ .

Just as for the integers, we define the mod operation for reals  $t$  and  $q$  (with  $q \neq 0$ ) by  $t \bmod q =_{\text{def}} t - \lfloor t/q \rfloor \cdot q$ .

*Lemma 2.2:* A point  $(x_1, \dots, x_d)$  is  $r$ -safe in a  $d$ -dimensional grid  $G$  with quantum  $q$  and offset  $(\psi_1, \dots, \psi_d)$  if for all  $i = 1, \dots, d$

$$r < (x_i - \psi_i) \bmod q < q - r.$$

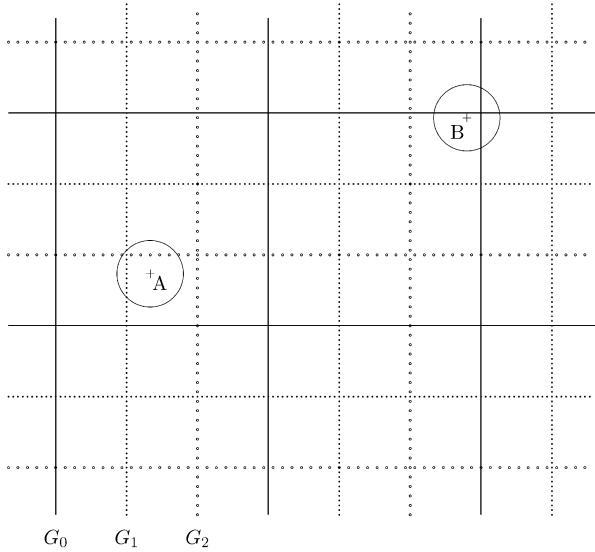


Fig. 1. Three grids  $G_0$ ,  $G_1$ , and  $G_2$  (A is safe in  $G_0$ , B is safe in  $G_1$  and in  $G_2$ ).

*Proof:* Suppose that the grid spacing (i.e., the discretization quantum) is  $q$ , and the offset of the grid hyperplanes in dimension  $i$  is  $\psi_i$  (for  $i = 1, \dots, d$ ). Let  $H_{m_1}^{(1)}, \dots, H_{m_d}^{(d)}$  be the grid hyperplanes that border the grid hypercube containing the point  $(x_1, \dots, x_d)$ . The point is  $r$ -safe if it is at distance  $> r$  from each of these hyperplanes (i.e.,  $(x_1, \dots, x_d)$  is  $r$ -safe if for some integers  $m_1, \dots, m_d$  we have (for  $i = 1, \dots, d$ ):  $\psi_i + qm_i < x_i < \psi_i + q(m_i + 1)$ ). By the definition of “mod,” this is equivalent to the statement of the Lemma.  $\square$

In two dimensions, we now introduce three grids  $G_0, G_1, G_2$ . All three have quantum  $q = 6r$ , and they are “staggered”:  $G_k$  has offset  $(-2rk, -2rk)$ , for  $k = 0, 1, 2$ . We will see (as a consequence of the Theorem below), that every point is  $r$ -safe in at least one of these three grids.

More generally, in a  $d$ -dimensional space, we consider a rectangle  $R_d = [0, a_1] \times \dots \times [0, a_d]$ , and we generalize all of the definitions above in an obvious way. We introduce  $d + 1$  grids  $G_k$  (with  $k = 0, 1, \dots, d$ ), all with quantum  $q = 2r(d + 1)$ , that are staggered:  $G_k$  has offset  $(-2rk, \dots, -2rk)$ .

**Lemma 2.3:** A point  $(x_1, \dots, x_d)$  is  $r$ -safe in grid  $G_k$  ( $k = 0, 1, \dots, d$ ) if for all  $i = 1, \dots, d$

$$r < (x_i + 2rk) \bmod (2r(d + 1)) < r(2d + 1).$$

*Proof:* In Lemma 2.2, just plug in  $6r$  for the quantum  $q$ , and plug in  $2rk$  for each offset  $\psi_i$ .  $\square$

The following theorem shows that robust discretization is possible. When the dimension  $d$  is 1 or 2, the proof is intuitive from the picture of the grids.

**Theorem 2.4:** Let  $r$  be any positive real number. For every point  $(x_1, \dots, x_d)$  in  $d$ -dimensional space, there is at least one grid  $G_k$  ( $k = 0, 1, \dots, d$ ) such that  $(x_1, \dots, x_d)$  is  $r$ -safe in that grid.

The proof is given in the Appendix.

The number  $d + 1$  is the minimum number of grids that gives us  $r$ -safety. Indeed, for  $d$  grids  $G_k$  ( $k = 1, \dots, d$ ), let  $x_i = c_{i,k}$  be a grid hyperplane of  $G_k$  perpendicular to coordinate axis  $x_i$ . Then, the point  $(x_i = c_{i,i})_{i=1, \dots, d}$  belongs to a grid hyperplane for each grid. Hence, this point cannot be  $r$ -safe, no matter how small a positive number  $r$  is. So  $d$  grids will not be sufficient, no matter what the quantum and the offsets may be.

**Robust discretization:** Since we now know that every point  $(x_1, \dots, x_d)$  is  $r$ -safe in at least one of the  $d + 1$  grids  $G_k$  ( $k =$

$0, 1, \dots, d$ ), we simply map the point into one of the grids in which it is  $r$ -safe. We have to make a choice here, since there often will be more than one grid in which  $(x_1, \dots, x_d)$  is  $r$ -safe. A safe grid choice map  $\gamma: \mathbb{R}^d \mapsto \{0, 1, \dots, d\}$  from points to grids, is any map such that  $(x_1, \dots, x_d)$  is  $r$ -safe in grid  $G_{\gamma(x_1, \dots, x_d)}$ , for all  $(x_1, \dots, x_d)$ . For example,  $\gamma(x_1, \dots, x_d)$  could be defined to be the smallest  $k$  such that  $(x_1, \dots, x_d)$  is  $r$ -safe in grid  $G_k$ ; or  $\gamma(x_1, \dots, x_d)$  could be a  $k$  for which the distance of  $(x_1, \dots, x_d)$  to the grid hyperplanes of  $G_k$  is maximized; or  $\gamma(x_1, \dots, x_d)$  could be chosen randomly, always subject to the  $r$ -safety condition (but kept fixed once chosen). The robust grid map is then defined by

$$g: (x_1, \dots, x_d) \in \mathbb{R}^d \mapsto (\gamma(x_1, \dots, x_d), g_{\gamma(x_1, \dots, x_d)}(x_1, \dots, x_d)).$$

So the grid map provides a grid identifier  $k = \gamma(x_1, \dots, x_d) \in \{0, 1, \dots, d\}$  and a grid-point  $g_{\gamma(x_1, \dots, x_d)}(x_1, \dots, x_d) \in \mathbb{Z}^d$  of the corresponding grid  $G_k$ .

The definition of  $r$ -safe says that if a chosen point  $p$  is  $r$ -safe in the grid  $G_k$ , and if a point  $x$  is at Euclidean distance  $< r$  from  $p$ , then  $x$  is in the same grid square as  $p$ . In other words, once the system has chosen a grid  $G_k$  in which the click point  $p$  is  $r$ -safe (i.e.,  $k = \gamma(p)$ ), then any click within distance  $< r$  from  $p$  will be in the same grid square as  $p$  and, hence, will be recognized by the system as the same as  $p$ . So the user has a guaranteed tolerance  $r$  for click errors.

On the other hand, if the user clicks on a point  $x$  at a distance  $> r(2d + 1)\sqrt{d}$  from the chosen point  $p$ , then the click is guaranteed to be treated as incorrect. Indeed, each grid square has side-length  $q = 2r(d + 1)$ , and a safe point is at distance  $> r$  from the edges; therefore, if  $x$  is in the same grid square as  $p$ , then  $|p_i - x_i| \leq 2r(d + 1) - r = r(2d + 1)$  for all of the coordinates  $(x_1, \dots, x_d) = x, (p_1, \dots, p_d) = p$ . In other words, the max-distance between  $p$  and  $x$  is  $\leq r(2d + 1)$ ; hence, the Euclidean distance (in  $d$ -dimensional space) between  $p$  and  $x$  is  $\leq r(2d + 1)\sqrt{d}$ . In 2-D space, this means that a click at distance  $> r5\sqrt{2} (\approx 7.07r)$  is guaranteed to be treated as incorrect.

If a point  $x$  is at a distance between  $r$  and  $r(2d + 1)\sqrt{d}$  from a chosen point  $p$ ,  $x$  may be in the same grid square as  $p$ , and they may be in different grid squares (depending on the exact locations of  $x$  and  $p$ ). In any case, for each grid, the number of grid squares in an  $a_1 \times \dots \times a_d$  hyper-rectangle is at least  $\lfloor a_1/q \rfloor \cdot \dots \cdot \lfloor a_d/q \rfloor$ , and all of these grid squares will be treated differently by the system.

**Remark on connections with error-correcting codes:** The midpoints of the grid (hyper-)squares of a grid are an example of an error-correcting code (with the decoding map being the mapping of a point to the midpoint of the square that contains the point). All error-correcting codes have an “edge problem” of a similar nature as grids do; if users choose messages that are about equally close to two (or possibly more) code words, the result of decoding becomes “unrobust.”

Previous work on robust discretization: Ideas similar to our robust discretization appear in Wu’s work [15], where the edge problem is discussed as well as the need for  $d + 1$  grids in  $d$ -dimensional space. However, the properties of robust discretization (namely, the definition of  $r$ -safety and our Theorem 2.4) are not explicitly stated in [15], and the sufficiency of  $d + 1$  is not proved (which we prove here for our Theorem 2.4).

### III. GRAPHICAL PASSWORD SCHEME

We now give a detailed description of our graphical password scheme, that was outlined in the Introduction. Our graphical password scheme has three components: image handling, password selection, and login.

- 1) The image-handling component enables users to choose images or to introduce their own; the images are stored together with a

collection of images provided by the system. For this password system to work well, it is important that the images be fairly intricate, with hundreds of interesting details that could be chosen as click regions (e.g., topographic maps, architectural images, cityscapes, certain landscapes, and renaissance paintings).

- 2) The password selection component allows the user to select a new password. Assuming the user has already logged in (by using either a graphical or a conventional password), the user enters the “password” command. The system then prompts the user for a user name and current password. If the system accepts the current password, it lets the user specify a new image (or keep the current image), and set the safety parameter  $r$  for robust discretization (or keep a default value).

Next, the image is displayed, and the user has to click on a few places (of the user’s choice); for security’s sake, at least five places should be clicked. Let  $c$  be the number of clicks, and let  $(x_1, y_1), \dots, (x_c, y_c)$  be the sequence of click places. The system takes the pixel coordinates of the click places, and for each click place  $(x_i, y_i)$ , it computes a grid identifier  $k_i = \gamma(x_i, y_i) \in \{0, 1, 2\}$  such that  $(x_i, y_i)$  is  $r$ -safe in grid  $G_{k_i}$  (for  $i = 1, \dots, c$ ). For each  $(x_i, y_i)$ , the system remembers the grid identifier  $k_i$ ; in our scheme,  $k_i$  is stored in the clear (not cryptographically hashed).

The system also computes the grid point  $g_{k_i}(x_i, y_i)$  of the click place  $(x_i, y_i)$  with respect to the grid  $G_{k_i}$ , and it remembers the secure hash value of the sequence of grid points of the click places. In summary, the user provides a sequence of click places  $((x_1, y_1), \dots, (x_c, y_c))$ , terminated by a “return.” The system remembers

$$\begin{aligned} & (\gamma(x_1, y_1), \dots, \gamma(x_c, y_c)) \quad \text{and} \\ \text{HASH} & (g_{\gamma(x_1, y_1)}(x_1, y_1), \dots, g_{\gamma(x_c, y_c)}(x_c, y_c)) \end{aligned}$$

in the user’s password record. The secret consists of the sequence of grid points.

As usual for password systems, before putting the new password in operation, the system should ask the user to confirm the password (by repeating the choice of clicks, but this time, it tolerates errors within the safety parameter  $r$ ).

- 3) The login component presents the user with a window into which the user types the user name. The system then retrieves the user’s password record (which contains the sequence of grids to be used), and displays the user’s password image. (If the user is not valid, the system will display a default image, and will eventually reject the user.)

The user then makes a sequence of clicks in the image. For the  $i$ th click, the system uses the  $i$ th grid ( $1 \leq i \leq c$ ) in the stored sequence of grid identifiers, and computes the grid point of that click place. (The system will not check whether the clicked point is  $r$ -safe in this grid, because we want to tolerate errors up to  $r$ .) When the user types “return,” the system computes the secure hash value of the sequence of grid points and compares this with the hash value stored in the user’s password record. If the two are identical, the user is accepted; otherwise, the user is rejected.

*Improved Implementation:* In the graphical password scheme described above, the  $c$  clicks were implemented as  $c$  2-D points. Instead, we could represent the click points  $(x_1, y_1), \dots, (x_c, y_c)$  as one  $2c$ -dimensional point  $(x_1, y_1, \dots, x_c, y_c)$ . This does not change anything from the user’s point of view, but it decreases the amount of information contained in the chosen grids. Indeed, when the  $c$  clicks are viewed as  $c$  2-D points, a sequence of  $c$  grids is stored in the system (in the clear). At each click, there are three grids that are possible, so for  $c$  clicks,  $3^c$  grid sequences are possible. One out of  $3^c$  possible grid sequences is

stored in the system files. On the other hand, for a  $2c$ -dimensional point, there are  $2c + 1$  grids. One grid out of  $2c + 1$  possible grids is stored. Of course, a source of information in which each event has probability  $(1/2c + 1)$  has much less entropy than a source in which each event has probability  $(1/3^c)$  (when  $c > 1$ ). For example, for  $c = 5$ , the grid information is  $\log_2(2c + 1) = \log_2 11 \approx 3.5$  for the improved method. For the first implementation, the grid information is  $\log_2 3^c \approx 7.9$ . The advantage of the improved implementation is thus that it represents the chosen grids slightly more compactly.

*Zoom:* Zooming in magnifies the area of the screen close to the cursor. The magnification allows the user to choose finer features of the image as click places. This may be a convenience for the user; it also increases the password space significantly by, in effect, decreasing the parameter  $r$ .

One can imagine several options, which are: 1) the user could click with the second mouse button to activate or deactivate the zoom-in; 2) there is automatic zoom in in the vicinity of the cursor; and 3) The automatic zoom in around the cursor depends on the speed of movement of the cursor; as the cursor slows down (near a possible click target), the magnification increases.

#### IV. SECURITY

The security of a password scheme depends, roughly, on the size of the password space (i.e., the total number of possible passwords for any given setting of the password parameters), and also on the way humans tend to use the password scheme. We are not able to give a serious security analysis of our graphical password scheme, for lack of data on human behavior. We limit ourselves to a brief discussion based on plausibility.

The size of the password space of our graphical passwords is parameterized by the safety parameter  $r$  (or, equivalently, the quantum  $q = 6r$ ) and the number of click points  $c$ . Moreover, the image size  $a \times b$  (or the screen size), and the resolution (number of pixels per square centimeter) are parameters, but we usually have little control over them. In general, the number of possible passwords is  $(\lfloor a/q \rfloor \cdot \lfloor b/q \rfloor)^c$ . For example, for an image of size  $330 \times 260 \text{ mm}^2$ , with safety parameter  $r = 1 \text{ mm}$  (so,  $q = 6r = 6 \text{ mm}$ ), each grid has at least  $2365$  ( $\approx 330/6 \times 260/6$ ) grid points. For graphical passwords with  $c = 5$  clicks, the number of possible passwords is therefore  $2365^5 \approx 7 \times 10^{16}$ . With six clicks, the number of possible passwords is  $2365^6 \approx 1.7 \times 10^{20}$ , and with seven clicks, it is  $2365^7 \approx 4 \times 10^{23}$ .

For the value  $r = 1 \text{ mm}$ , the grid squares have side-length  $q = 6 \text{ mm}$ ; this could be inconveniently small for many users, and for present-day computer screens (due to poor resolution). However, when the zoom feature is used,  $r = 1 \text{ mm}$  will not be small. When  $r = 2 \text{ mm}$ , there will be about  $560$  grid points; with six clicks, the number of possible passwords is then  $560^6 \approx 3 \times 10^{16}$ .

The human use of a password scheme determines the effective password space which, intuitively, consists of the passwords that users are likely to use. Although it is difficult to define the effective password space rigorously in general, for our graphical password system, we can find an operational definition as follows: The chosen image itself is an additional parameter now, and instead of the whole image size  $a \times b$ , we now count the area of the image that is occupied by “memorable” features. We cannot define rigorously what a memorable feature is, but by human experiments, we can find out which grid squares contain features that humans will accept as being possible to remember; we can also double-check by determining the nonmemorable regions—intuitively, those are large “empty” areas that do not contain edges or contrasts. Thus, for our graphical password system to be effective, we have to use images that are intricate enough to offer hundreds of attractive click places (e.g., maps, architectural graphics, renaissance paintings, city scapes, and complicated landscapes).

Some human aspects of graphical passwords are analyzed in [5] and in [8], but the graphical passwords considered in those papers are very different from ours. Our graphical password system has been implemented and some human factors experiments have been carried out on it, mainly from the point of view of ease of use; we refer to [12]–[14] for details.

Extensive human factors experiments will be needed to explore the size of the effective password space as well as possible unsafe usage practices by human users which could then be exploited by attacks. Moreover, in order to obtain a more general assessment of the human security of our graphical passwords, real-world experience will be needed, since in an experimental settings, it is difficult to replicate the habits and practices that users develop as a result of long-term everyday experience. Such experiments and experience are beyond the scope of this paper, and will be addressed in future research.

## V. CONCLUSION

We generalized Blonder's graphical passwords to arbitrary images, which makes this type of password more usable. We introduced a robust discretization of images, based on a multigrid discretization; this enables us to produce a unique output of the password, despite the fact that users are not able to input exactly the same graphical password at each login. This enables the system to store our graphical passwords in cryptographically hashed form.

## APPENDIX PROOF OF THEOREM 2.4

Consider any point  $(x_1, \dots, x_d)$ . Since  $r$ -safety only depends on the value of the coordinates mod  $(2r(d+1))$ , we can assume  $0 \leq x_i < 2r(d+1)$  for each  $i = 1, \dots, d$ . Also, by renaming the coordinates if necessary, we can assume that  $0 \leq x_1 \leq x_2 \leq \dots \leq x_d < 2r(d+1)$ .

*Claim:* For some  $i_o \in \{1, \dots, d\}$  and some  $h_o \in \{0, 1, \dots, d\}$

$$\begin{aligned} & \{2rh_o + r, 2r(h_o + 1) + r\} \\ & \subset \langle x_{i_o}, x_{i_o+1} \rangle, \quad \text{with } i_o < d, h_o < d, \text{ or} \\ & \{2rd + r, 2r(d+1)\} \cup [0, r) \\ & \subset \langle x_d, 2r(d+1) \rangle \cup [0, x_1), \quad \text{when } i_o = h_o = d. \end{aligned}$$

*Proof of the Claim:* There are  $d$  numbers  $x_j$  (for  $j = 1, \dots, d$ ), and there are  $d+1$  disjoint sets

$$S_h = [2rh + r, 2r(h+1) + r), \quad \text{for } h = 0, 1, \dots, d-1$$

and

$$S_d = [2rd + r, 2r(d+1)) \cup [0, r).$$

Hence, by the pigeonhole principle, at least one of these sets, say  $S_{h_o}$ , does not contain any  $x_j$ . We take  $x_{i_o}$  to be the largest  $x_j$  that is less than all of the elements of the set  $S_{h_o}$ . This proves the claim.

*Proof of the Theorem:* With the claim, we only need to consider the two cases  $A$  and  $B$  below. Case  $A$

$$\begin{aligned} & \{2rh_o + r, 2r(h_o + 1) + r\} \subset \\ & \langle x_{i_o}, x_{i_o+1} \rangle, \quad \text{with } h_o < d, i_o < d. \end{aligned}$$

We claim that in this case, the point  $(x_1, \dots, x_d)$  is  $r$ -safe in grid  $G_k$  with  $k = d - h_o (> 0)$ . Indeed, for all  $x_j$  with  $j \leq i_o$ , we have

$$0 \leq x_j \leq x_{i_o} < 2rh_o + r.$$

Hence, by adding  $2r(d - h_o) (= 2rk)$

$$2r(d - h_o) \leq x_j + 2rk < 2rd + r$$

hence, since  $r < 2r(d - h_o)$  when  $h_o \leq d - 1$

$$r < x_j + 2rk < 2rd + r.$$

So, for all  $x_j$  with  $j \leq i_o$ , the condition of Lemma 2.3 is satisfied. For all  $x_j$  with  $j \geq i_o + 1$ , we have

$$2r(h_o + 1) + r < x_{i_o} \leq x_j < 2r(d + 1)$$

hence, by adding  $2r(d - h_o) (= 2rk)$  and then subtracting  $2r(d + 1)$

$$r < (x_j + 2rk) \bmod (2r(d + 1)) < 2r(d - h_o) (< 2rd + r)$$

hence, the condition of Lemma 2.3 is satisfied for all  $x_j$  with  $j \geq i_o + 1$ . Case  $B$

$$\{2rd + r, 2r(d + 1)\} \cup [0, r) \subset \langle x_d, 2r(d + 1) \rangle \cup [0, x_1).$$

We claim that in this case, the point  $(x_1, \dots, x_d)$  is  $r$ -safe in grid  $G_0$ . Indeed, in this case,  $x_d < 2r(d + 1)$  and  $r < x_1$ ; hence,  $r < x_1 \leq \dots \leq x_j \leq \dots \leq x_d < 2r(d + 1)$ . So the condition of Lemma 2.3 is satisfied (with  $k = 0$ ) for all  $x_j$ .  $\square$

## ACKNOWLEDGMENT

The first author would like to thank students B. Isaacson and L. Sobrado, whose work benefited this paper. The authors thank the referees for their recommendations which improved this paper.

## REFERENCES

- [1] G. Blonder, "Graphical Password," U.S. Patent 5 559 961, 1996.
- [2] R. Dhamija and A. Perrig, "Déjà Vu—A user study using images for authentication," in *Proc. 9th Usenix Security Symp.*, Denver, CO, 2000, pp. 45–58.
- [3] D. C. Feldmeier and P. R. Karn, "UNIX Password security—ten years later," in *Lecture Notes Comput. Sci. 435: Advances Cryptology Conf.*, 1990, pp. 44–63.
- [4] B. Isaacson, "The password problem," Hons. thesis, Rutgers Univ., Camden, NJ, Jun. 2001.
- [5] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin, "The design and analysis of graphical passwords," in *Proc. 8th Usenix Security Symp.*, Washington, D.C., 1999, pp. 1–14.
- [6] D. Klein, "A survey of, and improvements to, password security," in *Proc. UNIX Security Workshop II*, Portland, OR, 1990, pp. 5–14, Usenix Assoc.
- [7] R. Morris and K. Thompson, "Password security: A case history," *Commun. ACM*, vol. 22, pp. 594–597, 1979.
- [8] "The science behind Passfaces," *Real User Corporation*, Sep. 2001 [Online]. Available: <http://www.realuser.com>.
- [9] M. Boroditsky, Passlogix Password Schemes, 2001/2002 [Online]. Available: <http://www.passlogix.com>.
- [10] A. Perrig and D. Song, "Hash visualization: A new technique to improve real-world security," in *Proc. Workshop Cryptographic Techniques and E-Commerce*, Hong Kong, 1999, pp. 131–138.
- [11] L. Sobrado and J. C. Birget, "Graphical passwords," *Rutgers Scholar*, vol. 4, 2002 [Online]. Available: <http://RutgersScholar.rutgers.edu/volume04/contents.htm>.
- [12] S. Wiedenbeck, J. Waters, J. C. Birget, A. Brodskiy, and N. Memon, "PassPoints: Design and longitudinal evaluation of a graphical password system," *Int. J. Human-Comput. Studies*, vol. 63, pp. 102–127, 2005.
- [13] S. Wiedenbeck, J. Waters, J. C. Birget, A. Brodskiy, and N. Memon, "Authentication using graphical passwords: Effects of tolerance and image choice," presented at the Symp. Usable Privacy and Security, Carnegie-Mellon Univ., Pittsburgh, PA, Jul. 6–8, 2005.
- [14] S. Wiedenbeck, J. Waters, J. C. Birget, A. Brodskiy, and N. Memon, "Authentication using graphical passwords: Basic results," presented at the Human-Comput. Interaction Int., Las Vegas, NV, Jul. 25–27, 2005.
- [15] C.-W. Wu, "On the design of content-based multimedia authentication systems," *IEEE Trans. Multimedia*, vol. 4, no. 3, pp. 385–393, Sep. 2002.