On Predictive Models and User-Drawn Graphical Passwords[1]

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

Abstract

In commonplace text-based password schemes, users typically choose passwords that are easy to recall, exhibit patterns, and are thus vulnerable to brute-force dictionary attacks. This leads us to ask whether other types of passwords (e.g., graphical) are also vulnerable to dictionary attack due to users tending to choose memorable passwords. We suggest a method to predict and model a number of such classes for systems where passwords are created solely from a user's memory. We hypothesize that these classes define weak password subspaces suitable for an attack dictionary. For user-drawn graphical passwords, we apply this method with cognitive studies on visual recall. These cognitive studies motivate us to define a set of *password complexity factors* (e.g., reflective symmetry and stroke-count), which define a set of classes. To better understand the size of these classes, and thus how weak the password subspaces they define might be, we use the "Draw-A-Secret" (DAS) graphical password scheme of Jermyn et al. (1999) as an example. We analyze the size of these classes for DAS under convenient parameter choices, and show that they can be combined to define apparently popular subspaces that have bit-sizes ranging from 31 to 41 – a surprisingly small proportion of the full password space (58 bits). Our results quantitatively support suggestions that user-drawn graphical password systems employ measures such as graphical password rules or guidelines, and proactive password checking.

1. INTRODUCTION

The ubiquitous use of text-based passwords for user authentication has a well-known weakness: users tend to choose passwords with predictable characteristics, related to how easy they are to remember. This often means passwords which have "meaning" to the user. Unfortunately, many of these "higher probability" passwords fall into a tiny subset of the full password space. We refer to such subsets as *weak password subspaces* (we define this more formally in Section 2.1).

Ideally, users would choose passwords equi-probably from a large subset of the overall password space, to increase the cost of a *dictionary attack*, i.e., a brute-force guessing attack involving candidate guesses from a prioritized list of "likely passwords". If a password scheme's probability distribution is non-uniform, its entropy is reduced. In Klein's case study [1990], 25% of 14000 user passwords were found in a dictionary of only 3×10^6 words; the Morris Worm [Spafford 1989] used a dictionary of only 432 words in addition to the 1988 UNIX online dictionary (about 25000 words [Spafford 1992]) with remarkable success: some sites reported that 50% of passwords were correctly guessed. This suggests that a password scheme's security is linked more closely to the size of its weak password subspaces than that of the full password space (which, e.g., for 8-character passwords of digits and mixed-case letters, is about 2×10^{14}).

Graphical password schemes (e.g., [Jermyn et al. 1999; Wiedenbeck et al. 2005; Dhamija and Perrig 2000; Real User Corporation 2004]) require users to remember picture-based information instead of text, motivated in part by the fact that humans have a remarkable capability to remember pictures. If the number of possible pictures is sufficiently large, and the diversity of picture-based passwords can be captured, graphical passwords may be less susceptible to having weak password subspaces and offer better security. Since graphical password systems have not been widely deployed to date, we lack knowledge of the distribution of the sort of pictures people are likely to *select* as graphical passwords. To this end, we propose models (which turn out to be supported by other user studies) to predict and characterize user choice in graphical password systems in which passwords are created solely from the user's memory without any other visual cues. Examples of such graphical passwords include free-form user-drawn graphical password schemes (e.g., DAS [Jermyn et al. 1999], variations such as Pass-Go [Tao 2006]), and schemes whereby a user might create

[1]Version: June 2, 2007. Preliminary versions of parts of this paper appeared as [Thorpe and van Oorschot 2004a] and [Thorpe and van Oorschot 2004b].

a graphical password by dragging and dropping basic shapes. We thus refer to such systems as *user-drawn* graphical password systems.

As mentioned, the high success rate of brute-force dictionary attacks against text-based passwords is believed to be strongly related to the recall capabilities of humans and how this affects password selection: meaningful and thus more easily remembered strings are frequently chosen. This leads one to ask whether other types of passwords (e.g., graphical) are also vulnerable to dictionary attack due to users' tendencies to choose memorable passwords. For relatively new password schemes where there is an absence of large datasets from diverse populations, we are motivated by the questions: (1) How might an attacker build a dictionary? (2) How successful would a brute-force dictionary attack using such a dictionary be?

Under conjecture that available studies from research on human memory might reveal higher-probability password choices, we provide a general predictive method for modelling and defining *weak password subspaces* in Section 2.1. We apply this method to user-drawn graphical passwords, and use these subspaces to build a *graphical dictionary* (i.e., an attack dictionary against a graphical password scheme). We expect that a clever attacker would prioritize a graphical dictionary according to how easy pictures or picture elements in the password are to recall or recognize, based on evidence from similar or related contexts. To find complexity properties that an attacker might use to define weak password subspaces for a graphical dictionary, and since our focus is on user-drawn graphical password schemes that require pure recall on the part of the user, we review cognitive studies indicating the types of images people are most likely to recall. We introduce a set of user-drawn graphical password complexity properties, including: password length, number of components, and symmetry. We model what we conjecture to be classes of higher-probability user-drawn graphical passwords based on these complexity properties.

To apply our classes to a real graphical password scheme, we use "Draw-A-Secret" (DAS) [Jermyn et al. 1999] as an example, as it has a large full password space, implicitly suggesting it is free of weak password subspaces. We apply two of our conjectured classes of higher-probability user-drawn graphical passwords (individually and combined) to DAS as candidate weak password subspaces. We find that these classes for DAS are small enough to be computationally exhausted, thus if their probability is high as implied by the motivating visual memory studies (and supported by user studies[2]), they would appear to be weak password subspaces. Under convenient parameter choices, where the size of the full DAS password space is 58 bits, the size of these combined classes is surprisingly small - 31 to 41 bits.

Our Contributions. For new password schemes, we introduce a method to predictively model weak password subspaces. For user-drawn graphical password schemes, our contributions include: the application of our predictive modelling method to user-drawn graphical passwords, the identification of user-drawn graphical password complexity properties, the definition of two broad subspaces of weak user-drawn graphical passwords, and the introduction of graphical dictionaries. We analyze the size of two weak password subspaces, and discuss what counter-measures can increase security. Our results naturally lead to password rules for user-drawn graphical passwords and motivate the use of proactive graphical password checkers. We propose a preliminary set of user-drawn graphical password rules motivated by our analysis, the most significant of which is to increase the stroke-count (in schemes where the input order matters).

Organization. The remainder of this paper is organized as follows. Section 2 introduces a predictive method to model weak password subspaces, presents a complexity model for user-drawn graphical passwords, provides an overview of attacker strategies, and defines classes of weak user-drawn graphical passwords. As an example, Section 3 explains applying our graphical dictionaries to DAS. Section 4 presents our results and analysis of the security of DAS; the most compelling results are given in Sections 4.2 and 4.3. Section 5 summarizes user studies (by others) whose results support that our classes are indeed weak password subspaces. Section 6 discusses methods to increase user-drawn graphical password security (including practical considerations). Section 7 briefly discusses related work. Section 8 provides concluding remarks.

2. PREDICTIVE METHOD AND PASSWORD COMPLEXITY MODEL

Since text-based password dictionaries focus on words people recall better, we are lead to consider how this might apply to dictionaries for other password schemes (e.g., graphical) for which we lack knowledge of the distribution of user choice. In Section 2.1, we introduce a predictive method as a starting point for

[2]Our conjectures that studies of visual memory from other contexts might predict higher-probability password subspaces are supported by two user studies [Nali and Thorpe 2004; Tao 2006], as discussed in Section 5.

modelling user choice. Here we apply this predictive method to *user-drawn* graphical password systems (*g_{ud}-passwords*).

We assume that users will choose g_{ud}-passwords that minimize their complexity, therefore we model memorable (and thus higher-probability) g_{ud}-passwords as those that have low complexity. Motivated by cognitive studies on visual recall (see Section 2.2), we introduce a set of g_{ud}-password complexity properties: password length, number of components (i.e., the visually distinct parts of the password), symmetry, and number of turns in each component. In this paper, we focus on all but the latter. Section 2.3 introduces a user-drawn graphical password complexity model, and Section 2.4 discusses attacker strategy to build a graphical dictionary using this model.

2.1 Predictive Method

We suggest the following general predictive method for modelling user choice in memory-based authentication schemes:

(1) Identify the tasks required from users during login, and what type of demand this places on their memory (e.g., verbal recall, visual recall, recall of input order).
(2) Determine what relevant information is available about user's memory (possibly from other contexts) regarding these identified demands.
(3) Identify password *complexity properties* based on this information. We informally define a password *complexity property* to be a characteristic that affects password memorability (and by conjecture, the chance of selection by users).
(4) Use these properties to model classes of memorable passwords.
(5) Estimate the size of these classes; any computationally exhaustible subset of the password space is a candidate weak password subspace (see Definition 1).

Definition 1 ***(Weak password subspace).*** *A weak password subspace is a subset W of a password space P that exhibits the following three properties:*

(1) (Smallness of W). $|W| \leq t_e$, where t_e is a threshold number of passwords that may be tested within an adversary's computational resources and environment.

(2) (Significance of W). $\sum_{w_i \in W} p(w_i) \geq \alpha$, where α is an upper bound on the "tolerable" probability of a password compromise (over a period of time commensurate with the resource expenditure in 1).[3]

(3) (Generability of W). W has defining characteristics that allow generation of all of its passwords.

Informally, a *weak password* is any password that falls within a weak password subspace. Weak passwords are similarly informally defined by Spafford [1992].

Weak password subspaces are a general concept which extend beyond graphical passwords; informally, they are subsets (of a space of secrets) whose elements are more easily guessed. Other examples of this concept include weak keys in symmetric cryptosystems [Daemen et al. 1993], weak RSA primes (e.g., see [Menezes et al. 2001]), and text password dictionaries [Openwall Project 2004a].

It follows from Definition 1 that for a scheme to be free of weak password subspaces, it is necessary for it to have a large total space; however, this is not sufficient due to what an attacker might be able to predict about user choice. Different systems can tolerate different levels of risk; this is captured by the parameter α which represents the tolerable probability for an attacker's success. For example, a user might feel that it is tolerable for an attacker to have a probability 2^{-10} of compromising their password, whereas government and banking servers might require a probability of at most 2^{-30}.

2.2 Relevant Memory Studies

We focus on user-drawn graphical passwords, where a user's login task involves pure visual recall of a drawn image, and recall of the temporal order (i.e., how the image was drawn). Following step (2) of the predictive method in Section 2.1, we examine and discuss a collection of relevant cognitive studies on visual recall.

[3]This condition is naturally related to that of entropy [Shannon 1948].

Generally, free recall is ordered along the concreteness continuum: concrete words are recalled more easily than abstract words, pictures more easily than concrete words, and objects better than pictures [Madigan 1983]. Various studies support this result (e.g., [Kirkpatrick 1894; Calkins 1898; Madigan and Lawrence 1980]). Another study [Bower et al. 1975] found that a series of line drawings is poorly remembered if the subject is unable to interpret the drawings in a meaningful way. The more concrete a drawing, the more meaningful it will be to the viewer.

Patterns in what *types* of images people recall better than others could be used to create classes of memorable and thus weak passwords, if such classes are sufficiently small.

There appears to be little existing research that examines the *types* of pictures people recall better. However, one cognitive study with interesting implications showed experimentally how visual recall progressively changed over time toward a symmetric version of the image [Perkins 1932]. Given a set of asymmetrical, geometric images, when test subjects were asked to draw the image from recall, all changes made from the originals were in the direction of some balanced or symmetrical pattern. This change was progressive over time toward a symmetric pattern. That people recall images as increasingly symmetric with time suggests to us that people prefer images that are symmetric.

A representative overview of literature for human symmetry perception [Tyler 1996] notes that many objects in our environment are symmetric. There is also significant evidence [Wagemans 1996] that mirror symmetry has a special status in human perception over other symmetry types such as repetition, translation or rotational symmetry, which were found to require scrutiny; in contrast, mirror symmetry is “effortless, rapid, and spontaneous” [Tyler 1996].

The classical studies mentioned above found better recall for pictures than words, and better recall for objects than pictures. If people recall objects best, and most objects are mirror symmetric, this suggests that people may recall mirror symmetric patterns best. This is supported by an observation by Attneave [1955]: when subjects were given random patterns and symmetric patterns of dots, the symmetric ones were more accurately reproduced than random patterns with the same number of dots. Attneave theorized that this may indicate that some perceptual mechanism is capable of organizing or encoding the redundant pattern into a simpler, more compact, less redundant form. In a separate study, French [1954] observed that dot patterns that were symmetric were more easily remembered. Intuitively, this is no surprise – in the case of mirror symmetry, a subject must only recall half of the image and its reflection axis in order to reconstruct the entire image.

Mirror symmetry has a special meaning to human visual perception, particularly when the axis is about the vertical and horizontal planes. Mirror symmetry has been found to be more easily perceived as having meaning when it is about the vertical axis, followed by when it is about the horizontal axis [Wagemans 1996]. Note that most living organisms and plants, as well as almost all forms of human construction are mirror symmetric (reflective) about a vertical axis.

Attneave’s [1957] findings of shape complexity also imply that people are better at recalling a low number of components. The following studies imply that values of “low” might lie between 3 and 8. Vogel and Machizawa [2004] found neurophysiological evidence that the human visual short term memory is limited to 3-4 symbols. Similar values were obtained for the number of dots recalled in grids of different sizes (recall decreased significantly after 3 or 4 dots) [Ichikawa 1982]. Alternately, French [1954] found that people have optimal memory for dot patterns containing 6 to 8 dots.

2.3 Model Motivated by Studies

Motivated by these collective studies, we propose the following.

Conjecture 1 *Since people are more likely to recall symmetric images and patterns, and people perceive mirror symmetry as having a special status, a significant subset of users are likely to choose mirror symmetric patterns as g_{ud}-passwords.*

More specifically, we propose that the mirror symmetric patterns chosen are more likely to be about vertical or horizontal axes. For g_{ud}-passwords, this leads us to define a *Class 1 password* (Definition 3). Findings that people are more likely to recall a low number of *components* (Definition 2) leads us to Conjecture 2 and to define a *Class 2 password* (Definition 4).

Definition 2 ***(component).*** A *component* is a visually distinct part of an image.

For example, a component in DAS is a drawn stroke. As another example, for a scheme wherein the user creates a password by dragging and dropping basic shapes, a component is a dragged/dropped shape.

Definition 3 ***(Class 1 password).*** A *Class 1 password* (C_1-password) is a g_{ud}-password that exhibits mirror symmetry about a vertical or horizontal axis in its components. Thus each component is either mirror symmetric in its own right, or is part of a pair of components that are mirror symmetric images of each other.

Conjecture 2 *Since people are likely to only recall a small number (between 3 and 8) of symbols, a significant subset of users are likely to choose g_{ud}-passwordswith a small number of components.*

Definition 4 ***(Class 2 password).*** A *Class 2 password* (C_2-password) is a g_{ud}-password with a small number $3 \leq c \leq 8$ of components.

The *Class 1 password space* is composed of the set of encoded representations of C_1-passwords; these form a graphical dictionary (i.e., a *Class 1 graphical dictionary*). *Class 2 password space* and *Class 2 graphical dictionary* are defined analogously.

2.4 Attack Strategy and Defining Graphical Dictionaries

There are two main attack strategies that an adversary can follow when trying to crack passwords: attacking a specific user account, or attacking a system by guessing the passwords for some larger set of (e.g., random) accounts. If the probability distribution for user choice within the password space is known, one can estimate the expected number of guesses for a specific user password using statistical expectation, or estimate the number of guesses required to guess a password on a system by using entropy (see [Massey 1994]). We use neither approach since in general, we do not know these probabilities.[4]

If C_1-passwords and C_2-passwords are significantly more probable in practice as we conjecture, the probability distribution of the graphical password space of g_{ud}-password schemes is highly non-uniform, significantly reducing the entropy of the password space. Similarly, the entropy is adversely affected if users favor passwords with characteristics that define other relatively small subsets of the full password space. We define graphical dictionaries for guessing attacks based on various subsets of passwords later in this section.

A clever attacker doing a brute-force guessing attack would prioritize candidate password guesses according to their probability of being chosen. We believe our graphical dictionaries would be the basis of such an ordering. We suggest that a clever attacker may prioritize a multi-class graphical dictionary according to passwords with an increasing number of components, and belonging to our classes as follows:

(1) Class 1 $\cap$ Class 2,
(2) Class 2 $-$ (Class 1 $\cap$ Class 2),
(3) Class 1 $-$ (Class 1 $\cap$ Class 2),
(4) Full password space $-$ (Class 1 $\cup$ Class 2)

Each class might also be internally prioritized as discussed in their respective sections, below.

2.4.1 *Class 1 Dictionaries.* A logical way to prioritize the Class 1 dictionary is to assume that it is more likely for a user to choose a single reflection axis, or reflection axes that are close together. If an image's components are symmetric about axes that are far apart (e.g., Fig. 1a), the image does not appear to be symmetric as a whole; we say such images are *locally symmetric.* When an image's components are symmetric about axes that are close together (e.g., Fig. 1b), it is increasingly symmetric as a whole, producing an image that is *pseudo-symmetric.* When all components of an image are symmetric about the same axis (e.g., Fig. 1c), it produces an image that is symmetric as a whole (i.e., *globally symmetric*). The set of globally symmetric passwords best captures the symmetry discussed in Section 2.2, and our intuition suggests that global symmetry (Fig. 1c) is more likely than pseudo-symmetry (Fig. 1b), which is more likely than local symmetry (Fig. 1a).

This leads us to define sub-classes of Class 1, based on the number of axes that the image's components are symmetric about. If it turns out that global symmetry is more likely than pseudo-symmetry, an attacker may

[4]However, see Section 5 for some data points from one large user study.

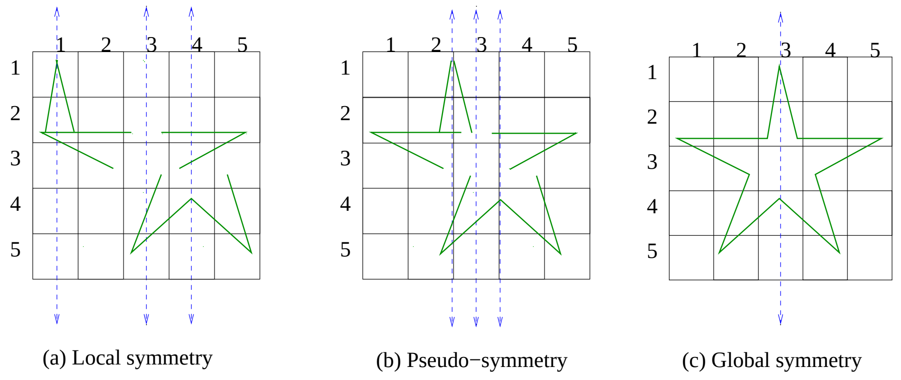

Fig. 1. Example Class 1 DAS passwords containing the same components, symmetric about different patterns of axes: (a) 3 different, scattered axes, (b) 3 different, nearby axes, and (c) a single axis.

place passwords that are composed of components symmetric about the center-most axes at a higher priority in the graphical dictionary. Additionally, for user-drawn schemes (e.g., DAS), if the user subconsciously uses the input area to frame the drawing (i.e., using the grid as part of the drawing's overall symmetry), the resulting drawings would be symmetric about the center-most axes.

Definition 5 ***(Class 1a).*** *Class 1a* is the subset of passwords in Class 1 that use only the center 3 of each set of horizontal or vertical axes (e.g., the marked axes in Fig. 2), producing pseudo-symmetric images.

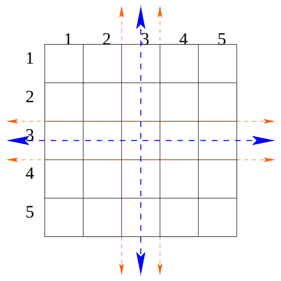

Fig. 2. Class 1a reflection axes for the DAS scheme. The thickest axes are the vertical and horizontal center axes. Adjacent axes are marked as thinner.

Definition 6 ***(Class 1b).*** *Class 1b* is the subset of passwords in Class 1 that use only the center of each set of horizontal or vertical axes, producing globally symmetric images.

Class 1b captures all passwords that are globally symmetric and centered about the grid (vertically and/or horizontally), plus those that have components symmetric about the center vertical and horizontal axes (e.g., the coffee cup in Fig. 3).

Class 1b is a subset of Class 1a, which is a subset of Class 1. We expect that an attacker would order Class 1b passwords first in a Class 1 graphical dictionary, followed by the remaining Class 1a passwords, and finally the remaining passwords in Class 1.

2.4.2 *Class 2 Dictionaries.* An obvious attack strategy for Class 2 graphical dictionaries is to prioritize based on the number of components, in increasing order. A dictionary attack would thus try all entries with one component, then two, etc. Given the evidence to show a threshold number of 3 or 4 components is more memorable (recall Section 2.2), we focus on those C_2-passwords with $c = 4$ for our model. An attacker with a particular account in mind might expand a dictionary to also consider larger values of c.

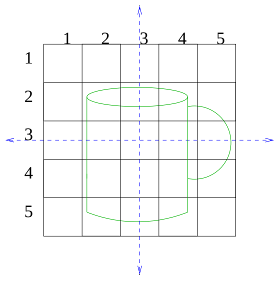

Fig. 3. Example of a Class 1b DAS drawing. One component (the handle) is symmetric about the center horizontal axis, and another (the cup), is symmetric about the center vertical axis.

2.4.3 *Class 3, 4, and 5 Dictionaries.* Here we mention three additional classes of graphical dictionaries. User-drawn passwords in the form of alphanumeric symbols are considered by Tao [2006]; we suggest calling this Class 3. We identify *Class 4 passwords* and *Class 5 passwords* based on repetitive and rotational symmetry respectively. These are common types of symmetry, although according to cognitive studies they do not hold the same special status as mirror symmetry. We note that these classes (and possibly many others) might also be placed in a graphical dictionary. We do not pursue further details in this paper.

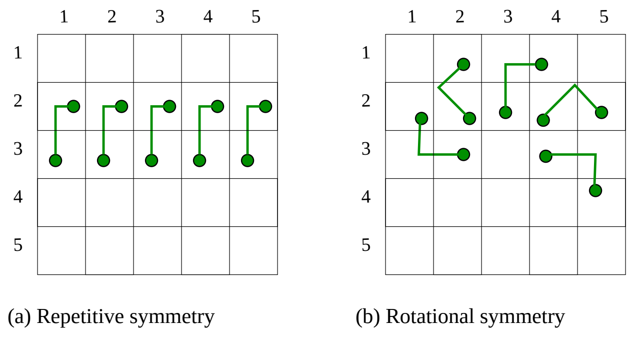

Fig. 4. Example Class 4 and Class 5 DAS passwords.

3. APPLYING GRAPHICAL DICTIONARIES TO DAS

As an example of applying our methods, and to augment the original (and to our knowledge the only previous) security evaluation of DAS [Jermyn et al. 1999], we determine the size of the more probable subsets of the DAS Class 1 and Class 2 password spaces of Section 2.3, i.e., the number of encoded DAS passwords representing at least one Class 1 (respectively Class 2) password. This is based on the reasoning that the number of entries in a "successful" attack dictionary provides a measure of (in)security.

The DAS graphical password scheme relies on a user's ability to recall their DAS password "exactly" (as defined by the resolution of the encoding scheme). For this analysis, we only consider passwords that are allowed within DAS (see Section 3.1). What users must recall can be divided into two parts: the temporal order and number of strokes used in the drawing, and the final appearance of the drawing. C_1-passwords consider the latter, and C_2-passwords partially consider the former (a user must recall an increasing amount of temporal order information with an increasing number of components).

Assumptions concerning the temporal order of C_1-passwords in DAS (i.e., the order of the input of cells) are made in Section 3.2 for our analysis, leading us to define the set S_1 (Definition 12). In order to map C_1-passwords to DAS, we must discuss these assumptions about temporal order: Section 3.2.1 discusses our terminology and general approach, and Section 3.2.2 discusses additional cases. Section 3.3 discusses the mapping of C_2-passwords to DAS, leading us to define the set S_2 (Definition 14).

3.1 Review of DAS

DAS encompasses both a general idea – user drawings as passwords – and a specific grid-based method to implement that idea (i.e., the encoding that maps a user drawing into an exactly repeatable password) [Jermyn et al. 1999; Monrose 1999]. To distinguish these concepts, we will refer to the specific encoding scheme of Jermyn et al. as DAS_J, and hereafter reserve the term DAS for the general idea. DAS$_J$ decouples the position of password input from the temporal order, producing a larger password space than text-based password schemes with keyboard input (where the order in which characters are typed predetermines their position).

A DAS password is a simple picture drawn on a $G \times G$ grid. Each grid cell is denoted by two-dimensional coordinates $(x, y) \in [1 \ldots G] \times [1 \ldots G]$. For DAS$_J$, an *encoded* password is a sequence of coordinate pairs listing the cells through which the drawing passes, in the order in which it passes through them. Each time the pen is lifted from the grid surface, this "pen-up" event is represented by the distinguished coordinate pair $(G + 1, G + 1)$. Two drawings having the same encoding (i.e., crossing the same sequence of grid cells with pen-up events in the same places in the sequence) are considered equivalent.[5] Drawings are divided into equivalence classes in this manner.

DAS$_J$ disallows passwords considered difficult to repeat exactly (e.g., passwords involving user input lying close to a grid boundary). The definition of "close to a grid boundary" is imprecise [Jermyn et al. 1999]; we define it as any part of a stroke for which the cell(s) it lies within is indiscernible, meaning it lies within the *fuzzy region* surrounding a grid line. Any stroke is invalid if it starts or ends in a fuzzy region, or if it crosses through the fuzzy region near the intersection of grid lines. We reuse the following terminology.

—The *neighbors* $N_{(x,y)}$ of cell (x, y) are $(x - 1, y), (x + 1, y), (x, y - 1)$ and $(x, y + 1)$.

—A *stroke* is a sequence of cells $\{c_i\}$, in which $c_i \in N_{c_{i-1}}$ and which is void of a pen-up.

—A DAS$_J$ *password* is a sequence of strokes separated by pen-ups.

—The *length of a stroke* is the number of coordinate pairs it contains.

—The *length of a DAS$_J$ password* is the sum of the lengths of its strokes (excluding pen-ups).

Jermyn et al. [1999] recursively compute the (full) password space size, i.e., the number of distinct encoded graphical passwords in DAS$_J$. This gives an upper bound on the size of weak password subspaces and thus on the security of the scheme. It is assumed that all passwords of total length greater than some fixed value have probability zero. They compute the full password space size for passwords of total length at most L_{max}. For $L_{max} = 12$ and a 5×5 grid, this is 2^{58}, exceeding the number of text-based passwords of 8 characters or fewer constructed from the printable ASCII codes ($\sum_{i=1}^{8} 95^i < 2^{53}$).

3.2 Class 1 DAS$_J$ Graphical Dictionaries

In this section we describe how we map the visual mirror symmetry from Class 1 to DAS$_J$ passwords. Our general approach and additional cases are described in Section 3.2.1 and Section 3.2.2 respectively, leading us to define our mapping from Class 1 to DAS$_J$ in Section 3.2.3.

3.2.1 *Basic Terminology and General Approach.* Our approach is to model each Class 1 password in DAS$_J$ as a series of strokes (each representing a single component or pair of components; recall Definition 3) drawn using only symmetric strokes (Definition 8. Each such stroke is modelled by a *defining stroke* from virtual start point $s = (x, y)$ to virtual end point $e = (x, y)$. We enumerate all possible values of s and e for each reflection axis, using these values as a model of the symmetry, and then consider the ways the resulting (user-drawn) stroke might be drawn. We emphasize that s and e are used to model the symmetry, and are not necessarily the start and end points of the user-drawn stroke.

To capture mirror symmetric DAS$_J$ passwords, we first consider which reflection axes to use. We assume that the user references the grid lines for the symmetry in the drawing, since if the reflection axis is a point of reference, the password will be easier to repeat exactly. Therefore, the reflection axes considered are those that cut a set of grid cells (Fig. 5a), or are on a grid line (Fig. 5b). This means that any symmetric password drawn such that its axis is off-center within a set of cells is not considered. For example, the password in Fig. 6a is visually symmetric when the grid is not in place, but we do not consider it part of the set of Class 1 passwords in DAS$_J$ since its reflection axis is not on a grid line or centered in a set of cells as shown in

[5]This implies a many-to-one mapping of user drawings to encoded DAS$_J$ passwords.

Fig. 6b. We justify this assumption as follows: it is more difficult for a user to draw an exactly repeatable symmetric password without a visible point of reference on the grid for the reflection axis.

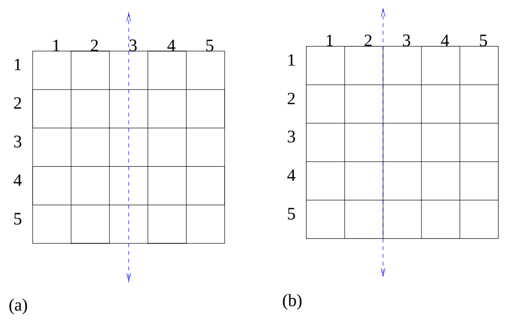

Fig. 5. Possible axes can (a) cut a set of cells; or (b) be on a grid line between sets of cells.

We thus define the set of axes within a $W \times H$ grid (width W, height H): $A = A_h \cup A_v$; $A_h = \{1, 1.5, 2, \ldots, (H-1).5, H\}$; $A_v = \{1, 1.5, 2, \ldots, (W-1).5, W\}$. Here $i.5$ is the grid line separating rows i and $i+1$, or columns i and $i+1$ respectively.

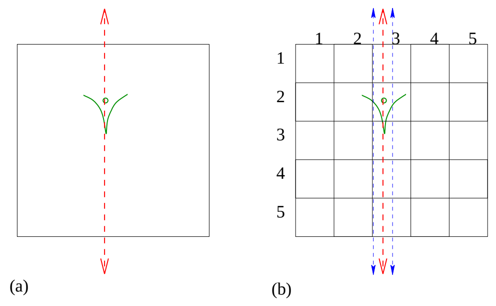

Fig. 6. Drawing that is symmetric about a difficult to reference axis. Assuming the v is drawn before the dot, the encoding of (b) is (2,2), (3,2), (3,3), (3,2), pen-up, (3,2), pen-up. If shifted slightly right to be symmetric about the vertical axis $x = 3$, it has symmetric encoding: (3,2), (3,3), (3,2), pen-up, (3,2), pen-up.

Definition 7 ***(symmetric area).*** The *symmetric area* (given a reflection axis a), is the area between a and the closest grid boundary parallel to a, reflected about a (see Fig. 7).

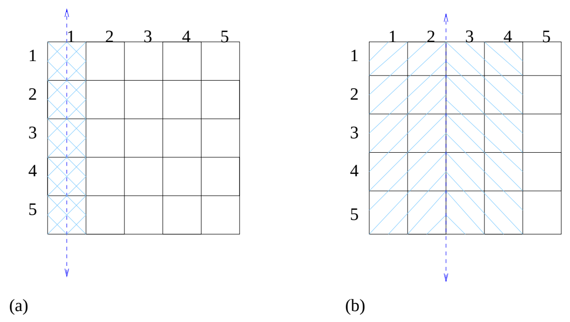

Fig. 7. Example symmetric areas for (a) the axis $x = 1$; and (b) $x = 2.5$

One way to draw a symmetric stroke is to draw a stroke within the symmetric area (possibly crossing over the reflection axis), then draw its reflection about the reflection axis as shown in Fig. 9a. We call the initial stroke from virtual start point s to virtual end point e that the reflection is based upon the *defining stroke*, and the reflection the *reflected stroke*, which can be drawn from s^R (the reflection of s) to e^R (the reflection of e) or vice versa. When the defining stroke is drawn from e to s, we consider (and count) it a different defining stroke, since input order is relevant in DAS$_J$.

Definition 8 ***(symmetric stroke).*** A *symmetric stroke* is a stroke (or pair of strokes) drawn such that it follows one of the *disjoint case*, *continuous case*, or *closed case* (per Definitions 9, 10, and 11). For context, see Fig. 8. Whether actually drawn as a single stroke or pair of strokes, it is modelled by the combined result of a defining stroke and a reflected stroke about an axis a, remaining within the bounds of the symmetric area defined by a.

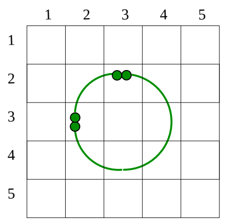

Fig. 8. Example DAS$_J$ password that is in Class 1, but is *not* drawn in symmetric strokes.

Definition 9 ***(disjoint case).*** The *disjoint case* consists of two user-drawn strokes holding the property of exact reflection. Given a defining stroke z, its reflected stroke z^R (relative to an axis a) is said to be an *exact reflection* if z^R is z's mirror image about a and they are separated by a pen-up.

As a stroke pair that falls within the disjoint case has the property of exact reflection, its length will always be even. The product of the number of ways to draw a defining stroke and the number of ways to draw its reflected stroke provides the number of ways to draw that stroke in the disjoint case (effectively counting each way to draw the reflected stroke for each way to draw the defining stroke). The disjoint case is not the only type of symmetric stroke (see Section 3.2.2).

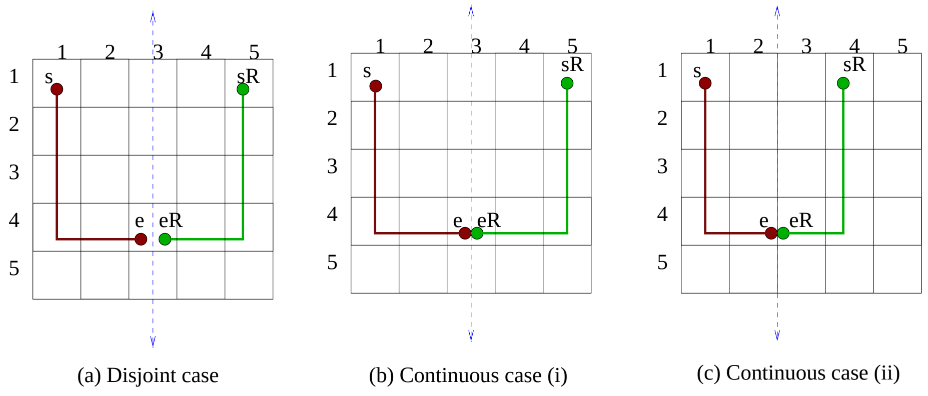

Fig. 9. Disjoint and Continuous Cases. Symmetric strokes consist of a defining stroke (solid line from s to e) and reflected stroke (solid line from s^R to e^R). The last two, visually representing the letter 'U', show continuous cases where: (b) the axis cuts a set of cells; and (c) the axis is on a grid line.

3.2.2 *Continuous and Closed Cases.* A point $p = (x, y)$ in an encoded defining stroke is *potentially continuous* if it denotes a cell that is either cut by the reflection axis a in question, or adjacent to a when a is on a grid line. If p is potentially continuous, its reflection p^R is in the same cell as p or in a neighboring cell, and thus the stroke can be drawn directly from p to p^R without a pen-up. When the start and end points of a defining stroke are potentially continuous, the three most apparently straightforward ways to draw the resulting symmetric stroke are as follows: disjointly, as one continuous stroke, or as one continuous closed stroke (see Definitions 9, 10, and 11).

A symmetric stroke can be drawn as a continuous case when the defining stroke's end point is potentially continuous.

Definition 10 ***(continuous case).*** The *continuous case* consists of one user-drawn stroke, whereby the defining stroke continues through the axis to the reflected stroke, in a single, continuous stroke.

For example, the encoding for Fig. 9b would be: (1,1), (1,2), (1,3), (1,4), (2,4), (3,4), (4,4), (5,4), (5,3), (5,2), (5,1), ending with a pen-up. The stroke could also be drawn in the reverse order. Examples of the same visual representation of a 'U', with one disjoint and the other continuous, are shown in Figures 9a and b. Note that the continuous case's encoding is different, depending on whether the axis a cuts a set of cells or is on a grid line. If a cuts a set of cells as in Fig. 9b, the defining stroke's endpoint e is the same as its reflection e^R. Since there is no pen-up to separate e from e^R, it cannot appear in the encoding twice, thus e^R does not appear in the resulting encoding. If a is on a grid line (Fig. 9c), e and e^R reside in different cells, and both e and e^R appear in the resulting encoding.

A symmetric stroke can be drawn as a closed case when both the defining stroke's start and end points are potentially continuous (e.g., Fig. 10).

Definition 11 ***(closed case).*** The *closed case* consists of one user-drawn stroke, whereby the defining stroke continues through the reflection axis to the reflected stroke, and then ends up back in the same cell as the start of the defining stroke, essentially creating a closed shape. When a drawing is closed, the user-drawn stroke may start and end at any point in the shape (e.g., Fig. 10a).

As with the continuous case, the closed case's encoding is different, depending on whether the axis a cuts a set of cells or is on a grid line. The continuation of the defining stroke into the reflected stroke will be encoded as in the continuous case; the difference between these two cases is the encoding to join the reflected stroke back into the defining stroke. When a is on a grid line, the start point of the defining stroke is repeated as the last point of the user-drawn stroke (e.g., Fig. 10b). When a cuts a set of cells (e.g., Fig. 10a), it is the same as the continuous case since $s = s^R$, enclosing the shape. Thus, to avoid double-counting, we must (and do) exclude the cases where s is potentially continuous from the continuous case. Note that when the defining stroke (completely) repeats over itself is counted by this case; this is included when the defining stroke is a closed case itself. Cases where the closed case only partially repeats over itself (e.g., only a few cells) are not considered a symmetric stroke.

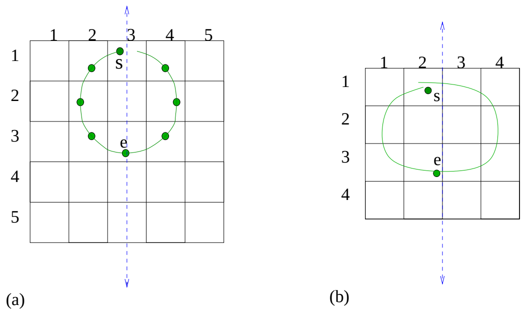

Fig. 10. Different types of the closed case. The reflection axis in (a) cuts a set of cells, and (b) is on a grid line. Case (a) shows all possible user-drawn start/end points in the symmetric stroke modelled by s and e.

3.2.3 *Classes S_1, S_{1a}, and S_{Ib}* . The definition of C_1-passwords takes into account only their final visual appearance. There is a one-to-many relationship between a given C_1-password and the number of ways it can be drawn in the DAS$_J$ scheme (which are then mapped to possibly fewer unique DAS$_J$ encodings). We believe there are some more likely *ways* that users will draw mirror symmetric components in their DAS passwords; we use S_1 to denote this "more probable" subset of unique DAS$_J$ encodings of C_1-passwords defined as follows.

Definition 12 *(S_1).* S_1 is the DAS$_J$-related subset of Class 1 passwords, containing only those passwords whose components are drawn in symmetric strokes.

Definition 13 *(S_{1a} **and** S_{1b}).* S_{1a} and S_{1b} are subsets of S_1 that belong to Class 1a and Class 1b respectively. More formally, $S_{1a} = S_1 \cap$ Class 1a; $S_{1b} = S_1 \cap$ Class 1b.

Preliminary user studies have shown that the temporal order has an adverse effect on user's ability to recall a DAS password [Goldberg et al. 2002]. If so, then we expect that users will choose DAS passwords with less complexity (e.g., fewer strokes). We believe that S_1 captures the easiest (and thus most likely to be chosen) ways to draw C_1-passwords, although not all possible ways (see Section 4 for discussion of the implications of this approximation). The details of how we enumerated the DAS$_J$ Class 1 space (or equivalently, graphical dictionaries) can be found in Thorpe et al. [2005].

3.3 Class 2 DAS$_J$ Graphical Dictionaries

In Definition 4, we chose to define a C_2-password to have at most c components; for our DAS$_J$ analysis, we use $c = 4$, since this value has support from 2 cognitive studies (recall Section 2.2). We assume that users will try to minimize the amount of temporal information to recall by drawing each component with the smallest number of strokes possible; we make the simplifying assumption that this is 1 (one). Thus for DAS$_J$, we characterize a C_2-password by the number of composite strokes. This leads us to define S_2 below. We quantify the relationship between the DAS$_J$ password space and the stroke-count in Section 4.2.

Definition 14 *(S_2).* S_2 is the DAS$_J$-related subset of Class 2 passwords, containing only those passwords having a stroke-count ≤ 4.

Our general approach is to determine how many DAS$_J$ passwords are of length at most a given maximum password length L_{max}, with a maximum stroke-count of X. Counting all passwords of length at most L_{max} follows Jermyn et al. [1999]. We modify their function *P(L,G)* that counts the number of passwords of length $\leq L$ (where $1 \leq L \leq L_{max}$ is the password length and G is the grid dimension), to limit the stroke-count in each password to at most X.

4. SECURITY ANALYSIS OF DAS$_J$

Following the relevant parts of the attack strategy from Section 2.4, and using subsets of Class 1, we expect that a DAS$_J$ graphical dictionary would be prioritized in the following order (see Fig. 11):

(1) $S_{1b} \cap S_2$
(2) $(S_{1a} - S_{1b}) \cap S_2$
(3) $(S_1 - S_{1a}) \cap S_2$
(4) $S_2 - S_1$
(5) $S_1 - S_2$
(6) the remainder of the DAS$_J$ password space that does not fall into any of S_1 or S_2.

While we expect (as mentioned in Section 3.2.3) that our Class 1 graphical dictionary for DAS$_J$ will include most C_1-passwords, we recognize that some C_1-passwords will not be included due to our definition of S_1. However, even if our dictionary only includes as few as e.g., $\frac{1}{8}$ of the ways users would typically draw C_1-passwords, this would imply our approximated bit-sizes are off by at most three bits – not significantly affecting our results. Furthermore, there is strong statistical support that our assumptions are indeed quite realistic (see Section 5), and thus the above error estimate appears to be conservative.

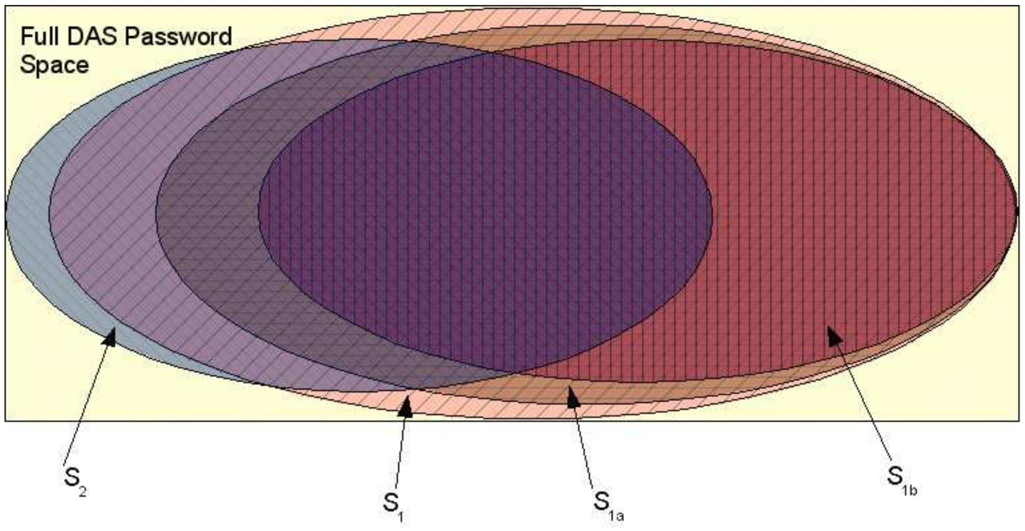

Fig. 11. Illustrative (bit-size) relationship between S_{1b}, S_{1a}, S_1, and S_2.

We approximate the size of password spaces S_{1b}, S_{1a}, S_1, and S_2, individually (Section 4.1 and Section 4.2) and their intersection (Section 4.3). For illustrative purposes, we estimate example attack times for exhausting each of these as potential graphical dictionaries in Section 4.4.

4.1 Approximate Size of Class 1 Graphical Dictionaries

To estimate the size of various subsets of the password space, many (equivalent) counting methods are possible. Our definitions in Sections 2.3 and 2.4 define what DAS$_J$ passwords are in the Class 1 password space. Section 3.2 defines which of these Class 1 passwords belong to S_1. We detail counting methods for generating these results in Appendix C of [Thorpe and van Oorschot 2005].

Table I gives sample results for S_1 and the subsets of S_{1a} and S_{1b} (recall Definitions 12 and 13). Values given are log_2(number of passwords). S_{1a} and S_{1b} both show an exponential reduction from the full DAS$_J$ space: S_{1b} grows at an exponential rate of approximately 3.6 bits per unit increase in password length and S_{1a} grows at a corresponding rate of approximately 4.0, whereas the full DAS$_J$ space and S_1 grow at a corresponding rate of approximately 4.8. For example, when $L_{max} = 12$, the size of the full space is 57.7 bits, S_1 is 56.7 bits, S_{1a} is 48.1 bits, and S_{1b} is 42.7 bits. The size of the full DAS$_J$ password space was cross-checked using a variation of our method (for full details, see Appendix C of [Thorpe and van Oorschot 2005]), closely matching the results given by Jermyn et al. [1999].

L_{max}	1	2	3	4	5	6	7	8	9	10
Full DAS$_J$ space	4.7	9.5	14.3	19.2	24.0	28.8	33.6	38.4	43.2	48.1
S_1	4.7	9.5	14.3	19.1	23.9	28.7	33.6	38.4	43.2	48.0
S_{1a}	3.3	7.7	11.6	15.7	19.8	23.8	27.9	31.9	36.0	40.0
S_{1b}	3.3	6.9	10.5	14.1	17.7	21.2	24.8	28.4	32.0	35.6

L_{max}	11	12	13	14	15	16	17	18	19	20
Full DAS$_J$ space	52.9	57.7	62.5	67.3	72.2	77.0	81.8	86.6	91.4	96.2
S_1	52.8	57.6	62.4	67.2	72.0	76.8	81.7	86.5	91.3	96.1
S_{1a}	44.1	48.1	52.1	56.2	60.2	64.3	68.3	72.4	76.4	80.4
S_{1b}	39.1	42.7	46.3	49.9	53.4	57.0	60.6	64.2	67.8	71.4

Table I. Bit-size of DAS$_J$ space, for total length at most L_{max} on a 5×5 grid.

Each of the three subclasses of C_1-passwords presented in Table I allow perceptually quite distinct classes of drawings (recall Fig. 1). We initially found the size of S_1 to be surprisingly close to that of the full DAS$_J$ space; however, upon reflection this is sensible, as the only requirement for a stroke to be symmetric is that it is locally symmetric about any axis in A (e.g., Fig. 1a), which includes the combinatorially large set of all permutations of dots and lines of length two.

The smaller the set of reflection axes used, the smaller the corresponding graphical sub-dictionary becomes. As discussed earlier, a reasonable attack strategy is to narrow down the graphical dictionary to a small number of axes, or prioritize a search such that globally symmetric passwords (e.g., Fig. 1c) are considered first. When any single axis (or two) are considered at a time to produce globally symmetric passwords, each result will never be larger than that for the two center axes, as the latter maximizes the symmetric area in which the passwords can reside. Thus, the maximum dictionary size of such a variation would be at most a small constant factor, proportional to the number of axes considered, of that using only the center axes.

4.2 Approximate Size of Class 2 Graphical Dictionaries

For Class 2 graphical dictionaries, following Jermyn et al. [1999] we focus our discussion on a set of results for a 5×5 grid size, giving the bit-size of the password space for passwords of length at most L_{max} (from 1 to 20) and each possible maximum stroke-count X. The full set is provided in Table III (Appendix A).

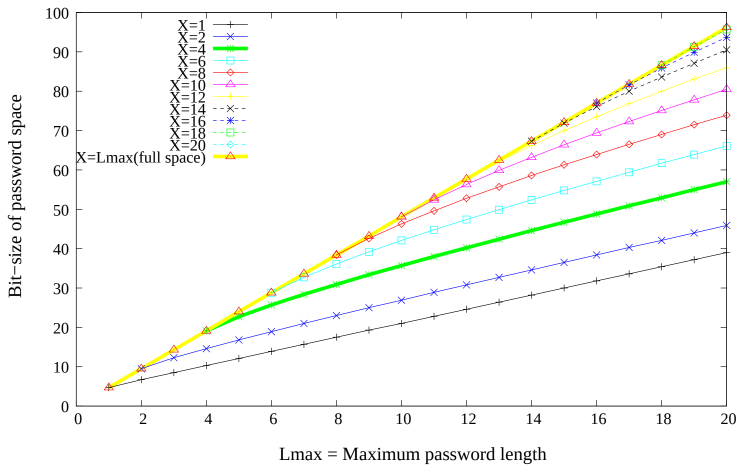

Fig. 12. Size of DAS$_J$ space for passwords of at most X strokes (for a 5×5 grid and a fixed maximum password length L_{max}). S_2 (for $c = 4$) is represented by the thick line where $X = 4$.

Fig. 12 shows the effect (log_2) of increasing L_{max} for a given X: the password space's size increases exponentially, illustrating the roles of both L_{max} and X in the DAS$_J$ password space. Note that the left ends of all but the line representing the full password space ($X = L_{max}$) have been omitted for simplicity – we know that the maximum stroke-count for a password of length L_{max} is L_{max}, thus any line where $X > L_{max}$ will have the same value as when $X = L_{max}$.

Increasing X accounts for at least one half of the bit-size (see the difference between the $X = 1$ line and the full space line, when $L_{max} \geq 5$). The top line, where $X = L_{max}$, in Fig. 12 shows what one would likely expect from reading the original DAS paper [Jermyn et al. 1999] (i.e., "58 bits of security" against guessing attacks when $L_{max} = 12$). The other thick line, where $X = 4$ represents the size of S_2. We suggest that S_2 is more representative of the "effective security" of unconstrained user-selected DAS passwords, taking into account the entropy reduction due to user choice in DAS passwords, and assuming all passwords composed of 4 or fewer strokes are equi-probable. This graph highlights the impact of the number of strokes on the DAS$_J$ password space; the size of the password space is significantly smaller (40 bits) if users choose a password of length at most 12, composed of 4 or less strokes. The password space still increases with longer password lengths (as shown by the rise in each curve), but at a slower rate for smaller stroke-counts (as shown by the gaps between curves). Note that for a fixed L_{max}, a smaller maximum stroke-count X implies a longer average stroke length.

Much of the strength of DAS$_J$ arises from the temporal order in terms of the direction of strokes, and more importantly, the order in which these strokes are drawn. This explains why increasing the stroke-count results in large increases in the size of the password space: there are many more possible permutations of these strokes.

A high stroke count for a fixed password length implies a short average stroke length. This leads us to ask: how much of the total password space consists of passwords composed entirely from seemingly unlikely combinations of very short strokes, i.e., entirely of strokes of length 1 and/or 2? This is easily computed by discarding those strokes of length > 1 (or > 2), and the results are interesting: passwords composed entirely of strokes of length 1 comprise approximately $\frac{1}{4}$ of the total password space, and passwords composed of only strokes of length ≤ 2 comprise approximately $\frac{1}{2}$ of the password space. Thus $\frac{1}{2}$ of the password space is already accounted for by passwords that appear to be very unlikely user choices.

This might be examined from another angle: how much of the total password space consists of passwords *without* any strokes of length 1? Again this is easily counted, with the result that if users do not draw any strokes of length 1 (e.g., dots) in their DAS password, the size of the password space when $L_{max} = 12$ on a 5×5 grid is effectively reduced from 58 to 40 bits, very dramatically increasing susceptibility to dictionary attacks. We expect that many user-chosen passwords will not contain any length-1 strokes – and note with interest that 51.5% of passwords in one large study did not contain any length-1 strokes (see Section 5).

4.3 Approximate Size of Combined Class 1 and 2 Dictionary

Figure 13 shows how restricting the maximum number of strokes, while also staying within different subclasses of C_1-passwords affects the size of the DAS$_J$ password space. All data shown is for $L_{max} = 12$ on a 5×5 grid.

The triangle point on the upper-right corner is the total number of DAS$_J$ passwords of length ≤ 12 on a 5×5 grid. Again, this 2^{58} value is the "security" measure one might originally have expected from DAS$_J$. Notice the triangle point where $X \leq 4$ – this value of 2^{40} shows the affect of number of strokes on the full DAS$_J$ password space. The S_{1b} bar directly below is the intersection of $X \leq 4$ with S_{1b}; we believe that this value of 2^{31} is a better "ball park" measure of the amount of security DAS$_J$ provides.[6]

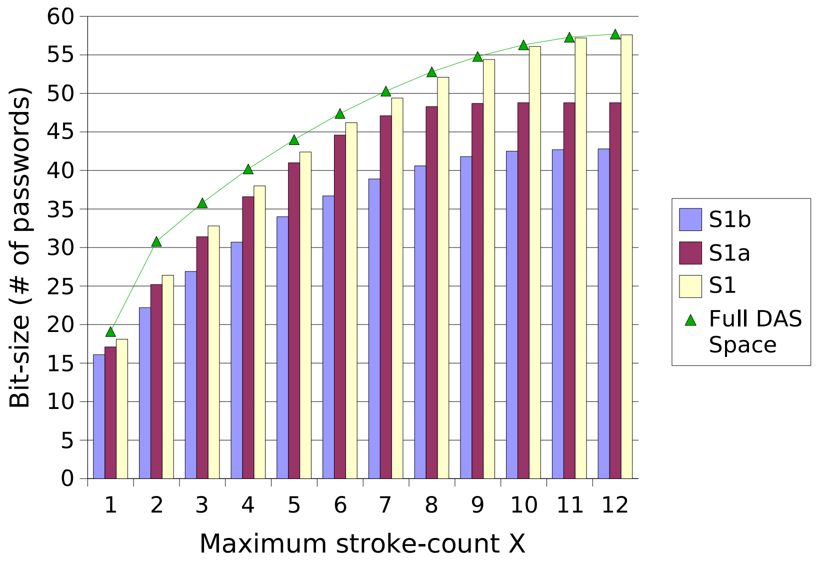

Fig. 13. Bit-size of DAS$_J$ graphical password space. Values are given for each dictionary, with a fixed total password length at most 12, with at most X strokes on a 5×5 grid (see Table IV in Appendix A for actual data points).

[6]In fact, this estimate itself very likely over-estimates security, as (a) it does not take into account other highly-probable (but not yet known) "Classes" of passwords; and (b) within Classes 1 and 2, our assumption of passwords being equi-probable is unrealistic, thus over-estimating password entropy.

4.4 Illustrative Attack Times

For illustrative purposes only, we estimate how long it might take to perform a dictionary attack against DAS$_J$ using our predictive method and related graphical dictionaries. The exact time required depends on implementation details. For this illustration, we assume that password verification involves at least hashing the entered password using the MD5 hash algorithm, then comparing the result to a stored value.[7]

Here the attack time is at least the time to hash each candidate password. We calculate two sets of times: one for an attacker with one *Pentium 4* 3.2GHz machine, and another for an attacker with one thousand such machines. It is reasonable to consider that a determined attacker could exploit one thousand, or even one hundred thousand machines using a worm, to distribute the password-cracking load. Using an MD5 performance result of 3.66 cycles/byte for a *Pentium 3* 800MHz machine [Nakajima and Matsui 2002] (scaled to 3.2GHz), and a 512 bit block size, approximately 1.37×10^7 hashes can be performed per second per machine. Given the assumed resources, the time to generate the password hashes is given in Table II for various sets.

Dictionary ($L_{max} = 12$)	Time to exhaust (1 machine)	Time to exhaust (1000 machines)
Full Space	541.8 yrs	197.8 days
S_1	505.6 yrs	184.5 days
S_{1a}	255 days	6.1 hours
S_{1b}	6 days	8.7 mins
S_2	1.1 days	1.5 mins
$S_{1b} \cap S_2$	2.1 mins	0.1 secs

Table II. Illustrative times to exhaust various entire DAS$_J$ graphical dictionaries (3.2GHz machines, 5×5 grid). The time (in seconds) is calculated by: $\frac{\text{dictionary size}}{(1.37 \times 10^7) \times \text{no. of machines}}$

The tabulated time is the worst case attack time if the password(s) under attack belong to the dictionary in use. An attacker may achieve success substantially faster if dictionary entries are ordered according to their probability of occurring. Note also that if the target passwords are not in any of the dictionaries, the attack fails.

Although entirely illustrative and heavily dependent on the assumed parameters, Table II highlights the practical implications of the graphical dictionary size. Assuming that we want an attacker with 1000 computers at 3.2GHz to require an average of 10 years to exhaust these dictionaries, the dictionary size must be approximately 2^{63}. Referring to our S_2 dictionary (stroke-count ≤ 4), if we extrapolate our results, this requires $L_{max} = 23$. This implies that for this level of security (and a 5×5 grid) in the absence of other measures, DAS$_J$ users should choose passwords of length at least 23 to resist dictionary attacks based solely on S_2 – which would appear to detract from usability.

On a more positive note, some of the larger text-based password dictionaries (which are effective in practice) contain approximately 4×10^7 entries [Openwall Project 2004b]. One of our smallest graphical dictionaries (S_2) exceeds this number of entries for $L_{max} \geq 9$. This implies that even if users choose passwords in S_2 (and if these are equiprobable – which admittedly is unlikely), provided the password length is at least 9, DAS$_J$ may still offer greater security than text-based passwords against dictionary attacks.

5. SUPPORTING EVIDENCE FOR S_1 AND S_2

Our proposed classes of weak g$_{ud}$-passwords were originally motivated by available information about what types of pictures people find easier to recall. In this section, we discuss two reported studies supporting the hypothesis that these classes are indeed weak password subspaces, and thus that the proposed graphical dictionary attack strategy is likely to perform quite well from an attacker's perspective, at least for the implementation discussed. We discuss limitations of these user studies in Section 5.2.

A user study of 167 students was performed on an implementation of a variation of DAS called Pass-Go [Tao 2006] (see Section 5.1). University students used the system to access their grades for one course over

[7]It should be clear that this attack time could be significantly increased by various implementation enhancements, e.g., see Section 6.

a 4-month semester. The results were analyzed using S_1 and S_2 as proposed herein (based on preliminary publications [Thorpe and van Oorschot 2004a; 2004b]), and a third class (recall Section 2.4.3), which Tao defines as a subset of alphanumeric characters and well-known symbols. The study found that 40% of users chose g_{ud}-passwords that fell in our S_{1b}, and when no stroke-count restrictions were applied, 72% of users chose g_{ud}-passwords that fell in our S_2. Other findings of interest include that password creation was increasingly difficult (in terms of the password creation success rate) when more restrictive stroke-count policies were applied (up to a requirement of 4 strokes), and that 41% of users created passwords that fell into our class S_{1a} (only 1% more than the 40% already in S_{1b}), implying that global symmetry is significantly more probable than pseudo-symmetry (recall Section 2.4). This study also supports our expectation that many users will not choose strokes of length 1 in their passwords; 51.5% of users had passwords with no length-1 strokes. Tao's subset of alphanumeric characters and symbols (under specified temporal orders and lengths) was 35.9 bits in size, and 19% of users chose passwords from this subset.

Of related interest, Tao also reported having posted a web site where anyone could create and practice Pass-Go passwords, and found that the results for this site (although yielding a study of smaller size and less controlled) were in line with that of the longitudinal study of 167 students. Of the 57 practice passwords created on this site, 37.5% fell into S_{1b} and 67% fell into S_2 [Tao 2006]. Also, no additional passwords fell into S_{1a} beyond those in S_{1b}, again showing a preference for global symmetry.

Tao's results are similar to that of an informal paper-based user study of 16 students [Nali and Thorpe 2004]. This study found that 45% of users chose symmetric passwords, two-thirds of which were mirror symmetric (and thus would fall into class S_{1b}). It also found that 80% of users chose passwords composed of 1-3 strokes, and with a definite tendency towards centering the password on the grid provided (56% were perfectly centered; an additional 30% were centered about about a set of cells on either side of the center grid lines).

While it would be wrong to assert that these studies are indicative of all user-drawn password systems, their results support that our classes (and predictive method used for generating them) should be seriously considered as a viable attack strategy. We now discuss how Pass-Go maps to DAS$_J$.

5.1 The Pass-Go to DAS$_J$ Mapping

In terms of the user drawing, Pass-Go may be viewed as an implementation of DAS, wherein the feedback to the user is normalized to better reflect the underlying encoding. To compensate for the rigidity of this normalization, Pass-Go allows for strokes to also connect to diagonal neighbors. The user drawing is normalized to points on cell corners, as opposed to DAS$_J$'s normalizing to points on grid cell centers. This implementation design arguably addresses the potential repeatability issue in DAS$_J$ when users draw passwords that are too close to grid lines and corners.

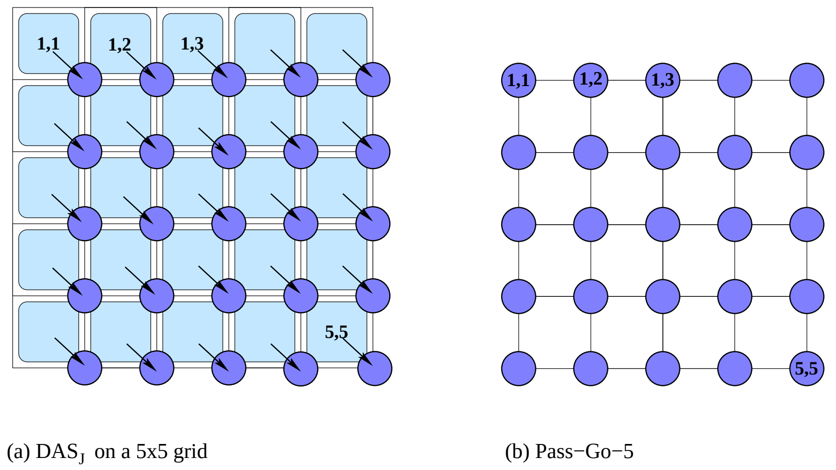

Fig. 14. Mapping from (a) DAS$_J$ to (b) Pass-Go.

There is a one-to-one mapping between the corners of a Pass-Go-5 grid and the cells of a 5×5 DAS grid as shown in Figure 14. The difference is in what strokes are valid; DAS$_J$ permits strokes between horizontal and vertical neighbors, whereas Pass-Go also permits strokes between diagonal neighbors. Thus in Pass-Go every point has up to 8 neighbors versus at most 4 in DAS$_J$. This results in a full password space increase of approximately one bit when the maximum password length is 12. The reason for this surprisingly small increase is that the password space is dominated by all permutations of length-1 strokes (which do not connect to any neighbors).[8]

5.2 Limitations of User Studies

It appears that the best (most reliable, and perhaps only) method to determine user choice and behavior for a given system is to deploy the system under investigation in the field, and study the results from a variety of populations. However, as for most user studies, those discussed in Section 5 only apply to a particular deployment environment, for a single user population. Thus, we discuss potential limitations of these studies when applying their results to other deployment environments and/or user populations.

The Pass-Go system of Tao [2006] was used to protect course materials, including course marks. Tao reports, based on responses from a post-study questionnaire, that 80% of users did not perceive this information to be sensitive and stated that for this reason, they picked passwords that were easy or simple. Thus, it is possible that if the system had protected information that was perceived to be sensitive, some of these users might have created passwords they perceived to be "more complex". Without a secondary study of the same user population, it is unknown whether these users would have created different passwords, and if so, in what way they would differ (e.g., more strokes, less symmetry, longer passwords, and by what degree). How users' choice in user-drawn graphical passwords is affected by their perception of the need for heightened security remains unanswered. Also, there are differences in implementation detail between DAS and Pass-Go, such as the use of indicators on the grid (in the form of stars and shaded cells) to help users navigate. We note that these indicators were placed in a globally symmetric pattern on the grid, which may have encouraged symmetric drawings. However, the star indicators alone did not appear to increase the number of symmetric passwords; in Tao's web-based practice study, which did not use star indicators, the number of passwords starting at either stars or corners was reduced (from 68% in the longitudinal study to 39% in the web-based practice study), but the percentage of passwords in S_1 remained approximately the same.

The study by Nali and Thorpe [2004] examined initial user choice on a single occasion, and as such did not account for the effect of password resets over time. Thus, it is possible that if users had to remember their passwords, they would have created (or reset) them differently. It is unknown how user-drawn graphical passwords change in complexity over password resets. Also, as this study was performed on paper, it may have resulted in different choices relating to the input device; for example, in mouse-based systems, users might be less likely to choose passwords with fine detail due to increased difficulty to draw such passwords with a mouse. The differences between user-drawn graphical passwords created on different input devices also remains a question worth independent study. Finally, the sample size of this study was small (16 users), and thus may not have been enough to obtain a representative distribution of user choice, although we note the percentages of users choosing passwords in S_1 and S_2 are quite similar to those in Tao's study.

6. TOWARDS INCREASING GRAPHICAL PASSWORD SECURITY

There are many potential methods to increase the security of graphical passwords in practice. These include password rules (see Section 6.1) and mnemonic strategies to aid users in choosing stronger passwords. Pass-phrases, i.e., sentences that help users recall a password, are a text-based password mnemonic. An analogous mnemonic proposed for graphical passwords is to create a story based on the picture(s) [Davis et al. 2004]. Any mnemonic strategy should be analyzed for new user choice patterns it may encourage; Kuo et al. [2006] found that a 400,000 entry dictionary guessed 4% of text passwords created using pass-phrases.

Implementation enhancements for text-based passwords can also be applied to graphical passwords. Graphical passwords that are exactly repeatable (such as DAS$_J$), can be stored using a one-way hash. Hashing algorithms with an adaptable cost (e.g., see [Provos and Mazieres 1999]) and that use *password stretching* or repeated hashing of passwords (e.g., see [Halderman et al. 2005]) increase the computational cost of guessing attacks. "Salting" adds random data to the computation of each user's password hash, and thus if any users

[8]The difference is more noticeable in the space of passwords without any length-1 strokes; here there is an increase of 6.2 bits for a maximum password length of 12 over original 5×5 DAS$_J$.

have the same password, the hashes will be different. Salting thus forces an attacker to compute a new hash for each password guess/user combination, increasing the computational cost of guessing attacks against a set of users.

Also, minor enhancements of a g_{ud}-password implementation can increase security (albeit typically at some cost in usability), e.g., additional user-selected characteristics of drawings such as color, backgrounds, and textures. The g_{ud}-password space could also be increased by increasing the area from which the user can select a graphical password – in DAS, this could be achieved by increasing the grid size. Unfortunately, in addition to possibly having a negative effect on the memorability of DAS passwords, increasing the grid size from 5×5 to 10×10 only provides 5-20 extra bits for convenient parameter choices [Thorpe and van Oorschot 2004b]. Another method that might increase security is to use a form of "zooming in", originally proposed for image-click schemes [Birget et al. 2003; Jansen et al. 2003]. One grid-based analogy to zooming in, *grid selection* [Thorpe and van Oorschot 2004b], involves a user selecting a smaller "drawing grid" (on which the password is later drawn) from a larger "selection grid". However, to have any confidence in this method, a (somewhat elaborate) user study involving a variety of implementation designs would be necessary to determine how much additional entropy it might add.

Finally, our work can naturally be applied to create a set of g_{ud}-password rules as guidelines for users and for use in proactive checkers. This rule-set is provided in Section 6.1.

6.1 Password Rules for User-Drawn Graphical Passwords

The impact of symmetry, a small stroke-count, and/or no strokes of length 1 as illustrated on the DAS_J password space naturally motivate the use and enforcement of g_{ud}-password rules. Given our knowledge to date, we suggest the following as an initial set of g_{ud}-passwords rules. For other variations on g_{ud}-passwords, stroke-count could be generalized to the smallest user-created units whose input order matters. We expect this list will grow over time, as more hypotheses of password complexity properties are developed and observed to hold in practice.

(1) Require a stroke-count of at least $\lfloor \frac{L_{max}}{2} \rfloor$.
(2) Disallow passwords having global reflective (mirror) symmetry (e.g., Class 1b).
(3) Require at least one stroke of length 1.

7. RELATED WORK

The security for a password scheme depends at least in part on its resistance to dictionary attack. To prevent on-line dictionary attacks, Pinkas and Sander [2002] discuss human-in-the-loop methods; see also Stubblebine and van Oorschot [2004]. One defense against off-line dictionary attacks is to reduce the probability of cracking through enforcing password policies and proactive password checking. Yan [2001] discusses some popular proactive text-based password checkers such as *cracklib.* To perform effective proactive text-based password checking, it is important to understand available text-based password cracking dictionaries and tools (e.g., *Crack* [Muffett 2004] and *John the Ripper* [Openwall Project 2004a]).

Graphical password schemes proposed to date can be generally categorized as recognition-based or recall-based. This categorization is also used by a recent survey of graphical passwords [Suo et al. 2005]. Monrose and Reiter's [2005] overview of graphical passwords provides a different (yet similar) categorization.

One recognition based scheme using hash visualization [Perrig and Song 1999] was implemented in a program called Déjà Vu [Dhamija and Perrig 2000]. Generally, in this scheme a user has a portfolio of pictures of cardinality F that they must be able to distinguish within a group of presented pictures of cardinality T. Another recognition-based scheme called $Passfaces$ [Real User Corporation 2004] requires that a user select a set of human faces as their password. Similar to Déjà Vu, the user is expected to correctly select each of the faces in their password from a set (or sets) of presented faces. *Story* [Davis et al. 2004] is a recognition-based scheme similar to $Passfaces$, but uses a variety of photo categories (e.g., everyday objects, locations, food, and people), and the user is asked to create a story to help them remember their portfolio.

Recall-based schemes might provide the user with some visual information which they may use during login. Blonder [1996] originally proposed a scheme whereby a user is presented with an image; the password is one or more clicks on predefined image regions. Birget et al. [2003] propose a similar scheme in that the user is presented an image and is expected to click on several points; however, the user may click anywhere on an image and error tolerance is achieved through their "robust discretization" technique. Robust discretization for this image-click scheme was implemented and studied by Weidenbeck et al. [2005]. Other schemes do not provide the user additional visual information, such as those based on user-defined drawings (the DAS scheme [Jermyn et al. 1999]; see Section 3.1 and Pass-Go [Tao 2006]; see Section 5.1). All of these recall-based schemes require exactly repeatable passwords (as defined within each scheme), allowing the password to be stored as the output of a one-way function, or used to generate cryptographic keys.

Regarding memorability issues for graphical passwords, Davis et al. [2004] examine user choice in two recognition-based schemes, showing a strong bias in user choice. Jermyn et al. [1999] argue that the DAS scheme has a large memorable password space by modelling user choice as those passwords describable by a short algorithm. They also examine the size of the password space for combinations of one or two rectangles, and show that this is comparable to the size of many text-based password dictionaries.[9]

Jermyn et al. [1999] suggest that the security of graphical password schemes benefit from the current lack of knowledge of their probability distribution; this motivates our present work, which apparently weakens this argument. Tao [2006] performed a user study on his Pass-Go (an implementation of DAS); his results support that our classes are weak password subspaces.

Relevant motivating literature on human memory is discussed in Section 2.2.

8. CONCLUDING REMARKS

This work extends and complements existing analysis and understanding of user-drawn graphical passwords (g_{ud}-passwords). Our work demonstrates methods to create graphical dictionaries, leading to a viable g_{ud}-password dictionary attack strategy. Although the focus of our work is the application of our predictive method (Section 2.1) to create graphical dictionaries for g_{ud}-passwords, we believe that graphical dictionaries could be applied to other types of graphical passwords guided by our predictive method and the question: "What applicable information is available about human memory and preferences that is related to the user's task?".

While one may question the likelihood of users choosing passwords within the identified classes of g_{ud}-passwords, motivated purely by cognitive studies on visual recall, at least two g_{ud}-password studies support that a notable proportion of users chose passwords that belong to these classes (recall Section 5). Combined with our results, this supports that the proposed classes are weak password subspaces; in the largest study of 167 users by Tao [2006], 40% of passwords fell within Class 1b, and 72% fell within Class 2. This provides strong statistical support that Classes 1 and 2 form a significant portion of the passwords that users are likely to choose (in the absence of password rules), and are thus weak password subspaces.

Class 2 passwords are (to date) the weakest identified password subspace for DAS_J, the original implementation and encoding of DAS. The space of DAS_J passwords when $L_{max} = 12$, restricted to at most 4 strokes (or alternatively with no strokes of length 1) is only approximately 40 bits. If users often choose passwords with a small stroke-count, or with no strokes of length 1, DAS_J falls easily to a dictionary attack. Indeed, Tao [2006] shows that a high percentage of users chose passwords in our Class 2, and that over one half of users chose passwords that did not have any strokes of length one. An attacker could also use this knowledge to further prioritize a dictionary (e.g., within Class 1; see Section 4).

Our work quantitatively supports earlier suggestions [Jermyn et al. 1999] that in order for g_{ud}-passwords to be secure against off-line dictionary attacks, password rules and proactive checking should be employed. Under the proposed password rules, users must be able to recall asymmetric passwords with a larger number of short strokes. Usability might suffer with password rules (in both memorability and repeatability), which might in turn lead to other predictable patterns in user choice.

Greater effective security might be achieved by g_{ud}-password schemes having larger subspaces of memorable passwords, even if at the expense of a smaller full (theoretical) password space. For example, encouraging users to draw passwords with more strokes might eliminate the Class 2 weak password subspace, if at the same time some other means reduces the difficulty for a user to recall their password. This could be achieved by e.g., disregarding the direction of strokes. A better understanding of the breakdown of what users have the most difficulty recalling (leading to a more formal definition of g_{ud}-password complexity properties) would be beneficial to understanding how to strengthen g_{ud}-password implementations. Attneave [1957; 1955] provides some hints in his examination of memory and complexity factors for visual patterns. It

[9]Note that rectangles are a subclass of Class 1 passwords.

would be interesting to determine empirically how easy passwords from the proposed Class 1 and 2 are to remember. Our work highlights the need for memory and usability studies to understand more about what users recall best, and how to take advantage of the strengths of human memory to create more secure and usable graphical password implementations.

Acknowledgements

We thank the anonymous referees for their comments which helped us significantly improve this paper. The first author acknowledges Canada's Natural Sciences and Engineering Research Council (NSERC) for funding a NSERC Discovery Grant and his Canada Research Chair in Network and Software Security. The second author acknowledges NSERC for funding a Canada Graduate Scholarship.

REFERENCES

Attneave, F. 1955. Symmetry, Information and Memory for Patterns. *American Journal of Psychology 68*, 209–222.

Attneave, F. 1957. Physical Determinants of the Judged Complexity of Shapes. *Journal of Experimental Psychology 53,* 4, 221–227.

Birget, J.-C., Hong, D., and Memon, N. 2003. Robust Discretization, With an Application to Graphical Passwords. Cryptology ePrint Archive, Report 2003/168. `http://eprint.iacr.org/`, site accessed Jan. 12, 2004.

Blonder, G. 1996. Graphical passwords. United States Patent 5559961.

Bower, G. H., Karlin, M. B., and Dueck, A. 1975. Comprehension and Memory For Pictures. *Memory and Cognition 3*, 216–220.

Calkins, M. 1898. Short Studies in Memory and Association from the Wellesley College Laboratory. *Psychological Review 5*, 451–462.

Daemen, J., Govaerts, R., and Vandewalle, J. 1993. Weak Keys for IDEA. In *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology.* Lecture Notes In Computer Science; Vol. 773, 224–231.

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

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

French, R.-S. 1954. Identification of Dot Patterns From Memory as a Function of Complexity. *Journal of Experimental Psychology 47*, 22–26.

Goldberg, J., Hagman, J., and Sazawal, V. 2002. Doodling Our Way to Better Authentication. In *Conference on Human Factors and Computing Systems* (April 20 - 25). ACM Press, 868–869. CHI '02 extended abstracts on Human Factors in Computer Systems.

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

Ichikawa, S.-I. 1982. Measurement of Visual Memory Span by Means of the Recall of Dot-in-Matrix Patterns. *Behavior Research Methods and Instrumentation 14(3)*, 309–313.

Jansen, W., Gavrilla, S., Korolev, V., Ayers, R., and R., S. 2003. Picture password: A visual login technique for mobile devices. NIST Report - NISTIR7030.

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

Kirkpatrick, E. A. 1894. An Experimental Study of Memory. *Psychological Review 1*, 602–609.

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

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

Madigan, S. 1983. Picture Memory. In *Imagery, Memory and Cognition*, J. C. Yuille, Ed. Lawrence Erlbaum Associates Inc., N.J., U.S.A., 65–89.

Madigan, S. and Lawrence, V. 1980. Factors Affecting Item Recovery and Hypermnesia in Free Recall. *American Journal of Psychology 93*, 489–504.

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

Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A. 2001. *Handbook of Applied Cryptography.* CRC Press, 290–291. Note 8.8.

Monrose, F. 1999. Towards Stronger User Authentication. Ph.D. thesis, NY University.

Monrose, F. and Reiter, M. K. 2005. Graphical passwords. In *Security and Usability*, L. Cranor and S. Garfinkel, Eds. O'Reilly, Chapter 9, 147–164.

Muffett, A. 2004. Crack password cracker. `http://ciac.llnl.gov/ciac/ToolsUnixAuth.html`, site accessed Jan. 12, 2004.

Nakajima, J. and Matsui, M. 2002. Performance Analysis and Parallel Implementation of Dedicated Hash Functions. In *Advances in Cryptology – Proceedings of EUROCRYPT 2002.* 165–180.

Nali, D. and Thorpe, J. 2004. Analyzing User Choice in Graphical Passwords. Tech. Report TR-04-01, School of Computer Science, Carleton University, Canada, `http://www.scs.carleton.ca/research/tech_reports/2004/TR-04-01.pdf`.

Openwall Project. 2004a. John the Ripper password cracker. `http://www.openwall.com/john/`, site accessed Jan.7, 2004.

Openwall Project. 2004b. Wordlists. `http://www.openwall.com/passwords/wordlists/`, site accessed Jan.7 2004.

Perkins, F. 1932. Symmetry in Visual Recall. *American Journal of Psychology 44*, 473–490.

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

Pinkas, B. and Sander, T. 2002. Securing Passwords Against Dictionary Attacks. In *9th ACM Conference on Computer and Communications Security*. ACM Press, 161–170.

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

Real User Corporation. 2004. About passfaces. `http://www.realuser.com/cgi-bin/ru.exe/_/homepages/technology/passface.htm`, site accessed May 25, 2004.

Shannon, C. 1948. A Mathematical Theory of Communication. *The Bell System Technical Journal 27*, 379–423.

Spafford, E. 1989. Crisis and Aftermath (The Internet Worm). *Comm. of the ACM 32(6)*, 678–687.

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

Stubblebine, S. and van Oorschot, P. 2004. Addressing Online Dictionary Attacks with Login Histories and Humans-in-the-Loop. In *In Financial Cryptography 8th International Conference* (February 9-12). Springer-Verlag LNCS 3110.

Suo, X., Zhu, Y., and Owen, G. S. 2005. Graphical Passwords: A Survey. In *21st Annual Computer Security Applications Conference (ACSAC)* (December 5-9).

Tao, H. 2006. Pass-Go, a New Graphical Password Scheme. M.S. thesis, School of Information Technology and Engineering, University of Ottawa, Canada.

Thorpe, J. and van Oorschot, P. 2004a. Graphical Dictionaries and the Memorable Space of Graphical Passwords. In *13th USENIX Security Symposium* (August 9-13).

Thorpe, J. and van Oorschot, P. 2004b. Towards Secure Design Choices for Implementing Graphical Passwords. In *20th Annual Computer Security Applications Conference (ACSAC 2004)* (December 6-10). IEEE.

Thorpe, J. and van Oorschot, P. 2005. On the Security of Graphical Password Schemes (Extended Version). Tech. Report TR-05-11, School of Computer Science, Carleton University, Canada, `http://www.scs.carleton.ca/research/tech_reports/2005/download/TR-05-11.pdf`.

Tyler, C. 1996. Human Symmetry Perception. In *Human Symmetry Perception and its Computational Analysis*, C. Tyler, Ed. VSP, The Netherlands, 3–22.

Vogel, E. K. and Machizawa, M. G. 2004. Neural Activity Predicts Individual Differences in Visual Working Memory Capacity. *Nature 428*, 748–751.

Wagemans, J. 1996. Detection of Visual Symmetries. In *Human Symmetry Perception and its Computational Analysis*, C. Tyler, Ed. VSP, The Netherlands, 25–48.

Wiedenbeck, S., Waters, J., Birget, J., Brodskiy, A., and Memon, N. 2005. 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.

Yan, J. 2001. A Note on Proactive Password Checking. ACM New Security Paradigms Workshop, New Mexico, USA.

Appendix A - Data Tables for Figures 12 and 13

X / L_{max}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	4.7																			
2	6.7	9.5																		
3	8.5	12.3	14.3																	
4	10.3	14.6	17.5	19.1																
5	12.1	16.8	20.3	22.7	24.0															
6	13.9	18.9	22.8	25.7	27.7	28.8														
7	15.7	21.0	25.1	28.4	30.9	32.7	33.6													
8	17.5	23.0	27.3	30.9	33.8	36.1	37.6	38.4												
9	19.3	25.0	29.5	33.4	36.6	39.2	41.2	42.6	43.2											
10	21.0	26.9	31.7	35.7	39.1	42.1	44.4	46.3	47.5	48.1										
11	22.8	28.9	33.8	38.0	41.6	44.8	47.4	49.6	51.3	52.4	52.9									
12	24.6	30.8	35.8	40.2	44.0	47.4	50.3	52.8	54.8	56.3	57.3	57.7								
13	26.4	32.7	37.9	42.4	46.4	49.9	53.0	55.7	58.0	59.9	61.3	62.1	62.5							
14	28.2	34.6	39.9	44.6	48.7	52.4	55.7	58.6	61.1	63.2	64.9	66.2	67.0	67.3						
15	30.0	36.5	41.9	46.7	50.9	54.8	58.2	61.3	64.0	66.4	68.4	70.0	71.1	71.9	72.1					
16	31.8	38.4	43.9	48.8	53.2	57.1	60.7	63.9	66.8	69.4	71.6	73.5	75.0	76.1	76.7	77.0				
17	33.6	40.3	45.9	50.9	55.3	59.4	63.1	66.5	69.6	72.3	74.7	76.8	78.6	80.0	81.0	81.6	81.8			
18	35.4	42.1	47.9	52.9	57.5	61.7	65.5	69.0	72.2	75.1	77.7	80.0	82.0	83.6	84.9	85.9	86.4	86.6		
19	37.2	44.0	49.8	55.0	59.6	63.9	67.9	71.5	74.8	77.8	80.6	83.1	85.2	87.1	88.7	89.9	90.8	91.3	91.4	
20	39.0	45.9	51.8	57.0	61.8	66.1	70.2	73.9	77.3	80.5	83.4	86.0	88.4	90.5	92.2	93.7	94.9	95.7	96.1	96.2

Table III. Bit-size of DAS$_J$ with different maximum lengths and stroke-counts as illustrated in Fig. 12 of Section 4.2. Values are given for total length at most L_{max} with at most X strokes on a 5×5 grid. S_2 is shown in the column where $X = 4$.

X / *Dictionary*	1	2	3	4	5	6	7	8	9	10	11	12
Full Space	24.6	30.8	35.8	40.2	44.0	47.4	50.3	52.8	54.8	56.3	57.3	57.7
S_1	18.2	26.4	32.8	38.0	42.4	46.2	49.4	52.1	54.4	56.1	57.2	57.6
S_{1a}	17.6	25.2	31.4	36.6	41.0	44.6	47.1	48.3	48.7	48.8	48.8	48.8
S_{1b}	16.1	22.2	26.9	30.7	34.0	36.7	38.9	40.6	41.8	42.5	42.7	42.8

Table IV. Bit-size of DAS$_J$ space as illustrated in Fig. 13 of Section 4.3. Values are given for each dictionary, with a fixed total password length at most 12, with at most X strokes on a 5×5 grid.