Inkblot Authentication

Adam Stubblefield

Daniel R. Simon

astubble@cs.jhu.edu, dansimon@microsoft.com

August, 2004

Technical Report
MSR-TR-2004-85

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Inkblot Authentication

Adam Stubblefield[*] Johns Hopkins University

Daniel R. Simon Microsoft Research

Abstract

In this paper we present a system that helps users select, remember, and differentiate strong passwords. Similar in nature to a Rorchach inkblot test, the system asks the user to form semantic associations with a set of randomly generated inkblot-like images. These image associations can then be used to authenticate the user. This approach is based on evidence from the psychological literature that suggests users will both choose dissimilar associations and retain their associations for long periods of time. We report on two user studies which show that inkblot based passwords are both memorable and high in entropy.

1 Introduction

The weakest links in many secure computing environments are the users themselves. Trading off security and usability, users often choose to subvert policies intended to protect their own (or their organization's) data. One example of such a tradeoff involves the selection of passwords. Despite a security administrator's best intentions, users are likely to choose weak passwords that they believe they can easily remember.

In this work, we have attempted to design a scheme that does not coerce users to choose difficult but secure passwords, but instead ensures that nearly every easily remembered password is secure. At its core is an image association task, partially borrowed from the Rorschach inkblot test [15], a personality test used by psychologists for more than half a century. In a Rorschach inkblot test, subjects are shown a series of ten random looking images produced by blotting pen ink on a peice of paper and folding it (see figure 1 for examples of the original ten blots). For each blot, the subject is asked to tell the examiner what he or she ”sees” in the blot, freely associating abstract or concrete representations with the random image. These responses are then used by the examiner to characterize the personality of the subject and diagnose any personality disorders. The Rorschach test suggests the following password scheme: when selecting a password each user is shown a series of inkblot-like random images. For each image, the user is asked to enter a sequence of characters derived from the user's association with the image. Then, on each subsequent login attempt, the user is shown the same images, recalls the same associations, and enters the same sequence of characters. This process will be discussed more formally in section 3.

In recent years, the usefulness of the Rorschach test as a diagnostic tool has been cast into doubt [16]. Fortunately, even if the personality conclusions drawn by Rorschach researchers are incorrect, the raw data for the association task is still valid. Because of the difficulty of conducting large user trials, we choose to base some of our initial assumptions about the password scheme on large previously published Rorschach data sets. In addition, we conducted small user trials to verify these assumptions about our password scheme.

[*]Partially based on work done while a summer student at Microsoft Research

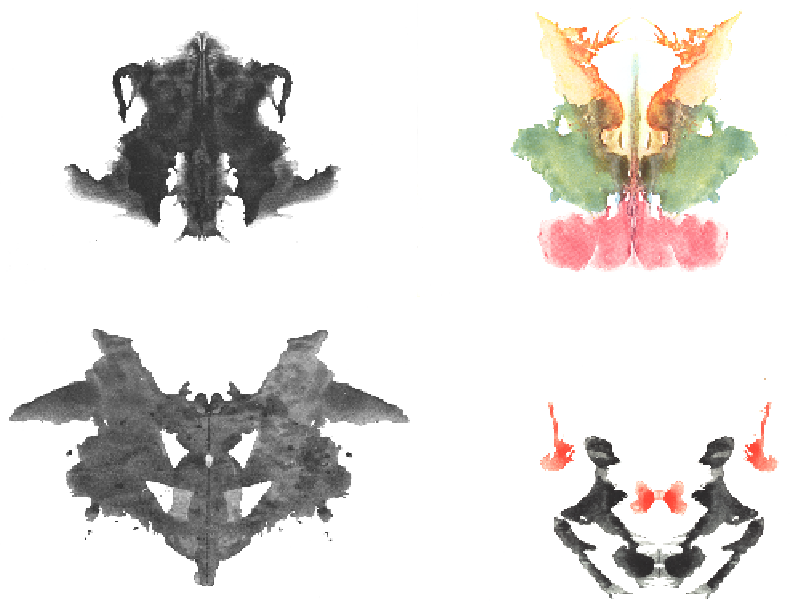

Figure 1: Four examples of the original Rorschach inkblots.

The Rorschach data cannot be used to directly verify properties of our password scheme since the actual association task had to be modified for the password setting. In the original test, subjects were allowed (and even encouraged) to come up with as many associations as they could for each blot[1]. In a password scheme multiple associations would only reduce the search space an attacker would need to cover. Therefore, in the password scheme only a single association per image is allowed for each user. We used small user trials to verify that single associations are as memorable and random as the original multiple responses.

Besides limiting users to a single response, the scheme also requires users to "hash" their associations into a small constant number of characters that can be typed quickly. For example, the simplest hash would be for the user to enter the first character of a verbal description of the association for each blot.

2 Related Work

The insecurity of ordinary, user-selected passwords is well-documented in the literature [11, 7, 12]. These works show that the space of likely passwords is so small that it can be exhaustively searched by an attacker aided by a dictionary of common words and phrases. Many techniques have been proposed to thwart such "dictionary attacks." A simple solution is to attempt to attack each new password that a user selects before it is accepted. However, there is no guarantee that the system will be using the same dictionary as an attacker. An alternative is to ensure that each new password follows a set of guidelines, such as including at least one number and both lowercase and capital letters [2]. Instead of enforcing such rules, users can be given a suggested "algorithm" for choosing a strong password. One popular and effective "algorithm" takes the first character of each word in a memorable phrase or song [17].

[1]An alternative test, the Holtzman Inkbot Test [9], only allows one response per blot, however there is less extensive data available to support it

Many schemes not based on traditional text-based passwords have also been proposed. These proposals are often based on psychological studies showing that humans can remember pictorial representations more readily than textual or verbal representations.The schemes are based on such tasks as sub-image recognition [3], drawn image recall [10], face recognition [4], and random image recognition [6]. In the recall-based schemes, users are required to remember a specific image, whereas in the recognition-based schemes users are required to remember whether they have previously selected a given image.

The scheme presented in this paper resembles both the "algorithmic" and graphical approaches to choosing a password. It is algorithmic in the sense that users are asked to hash their associations down to a few characters that contain most of the entropy of the association and graphical in the sense that the inkblot images are used to spur the associations. However, the scheme in this paper provides benefits that neither approach provides alone. The algorithmic schemes do not provide prompts to help jog the user's memory, while the graphical schemes generally use the output device to gather input (allowing shoulder surfing), provide only limited entropy per user entry, and require a different entry on each log-in.

3 Overview

There are a wide variety of possible inkblot authentication schemes rooted in the image association task. In this section, we outline a very simple (and not necessarily secure) inkblot authentication scheme. The constructions used in our experiments are discussed in section 4.

3.1 Goals

Before discussing the actual schemes we formally state our goals for the system. First and foremost, the system should ensure that the space of likely passwords is large enough that dictionary-style attacks are unlikely to succeed. More specifically, in response to a given image, the distribution on users' textual responses should have a high entropy. Second, when shown an inkblot, users should be able to reliably form a single, memorable textual association. In the best case, users should simply be able to see the inkblot and re-form the association without having to remember the association explicitly from session to session.

Besides strength and memorability, a number of other factors influenced our design. One common complaint about image-based schemes is their reliance on graphical input methods. In other words, many image-based schemes require users to expose their passwords on an *output device*, for example by clicking certain portions of an image with a mouse. While this can be appropriate in some situations (e.g. authenticating to a PDA), in other situations it can expose a user's password to an attacker looking over the user's shoulder. For this reason we decided that only keyboard entry should be used. Fortunately, using the keyboard also has other advantages. For example, each entry the user makes comes from a large space of possible choices (the keys). Assuming a reasonable distribution on the choices, this can reduce the number of entries required before the user is authenticated. In order to further reduce the time necessary for authentication, we decided against using a challenge-response-style system. Instead, users type exactly the same password at every login; thus, they can eventually bypass the image association task entirely and rely on muscle memory alone for typing the correct sequence of keys.

While not a goal per se, it is important to remember that the system was designed as a method for helping users to choose and remember strong conventional passwords rather than as an "alternative to passwords." Instead of changing the way that users authenticate, the system is intended to make authentication easier and more secure without altering the basic password model.

3.2 A Simple Inkblot Authentication Scheme

As an illustrative example, assume that each user is shown the original Rorschach inkblots, one at a time, during the password selection dialog. For each image, the user is asked to form an association and type the first character of that association. As each character is entered, a new blot is displayed. Once all ten associations are made, the password is accepted.

On each subsequent authentication attempt, users are presented with the same ten blots, in the same order. By looking at each blot, they can form the same association and thus type the same password. Since each character of such a password is likely drawn from a distribution of first letters of a set of English words, even an attacker who knows the precise distribution will have to guess from a large space of possible passwords, assuming that few users form the same textual association.

Unfortunately, even if such an assumption holds and the associations are easy for users to remember, such a system would not be secure. Besides allowing an attacker to build a very good model of the responses for each blot, the use of a small set of static blots prevents the system from being used in multiple environments and prevents users from ever changing their passwords. Because each user cannot be expected to make (or keep track of) more than one association with each blot, using the system in multiple environments would require that the user use the same password in every environment, including environments under different administrative domains. Even if each administrative domain were to choose a different set of static blots, each time that users wanted to change their passwords a new set of blots would need to be introduced. Both of these issues can be resolved by replacing the static images with an algorithm that can produce a (nearly) unlimited supply of images that resemble inkblots. The design of such a generator is discussed in section 4.1.

The choices to use ten inkblots and to have users enter the first character of each association were made somewhat arbitrarily. For example, it might be easier for users to enter the first two characters of five associations. This might create an opportunity for the attacker however, as there would only be five distributions of english word-beginning digraphs to model. This tradeoff among memorability, speed of entry, number of blots, and simplicitly of the user computable hash function is discussed in section 4.2.

4 Design and Implementation

In this section we describe the design and implementation of our final inkblot authentication scheme, as well as the experimental methodology used to verify the properties of the system. The design of the system was guided by data gathered during Rorschach inkblot tests. The assumptions made based on this data turned out to be surprisingly accurate. A full analysis of their merits is delayed until sections 5 and 6.

The final inkblot authentication system is very similar to the simple system of section 3.2. Users go through the same steps of being presented with a series of inkblots, one at a time, and having to enter a series of characters based on their associations with each inkblot. The differences lie in the choice of inkblots and user-computable hash functions.

4.1 Inkblot Generation

A static set of inkblot images is unsatisfactory, for the reasons discussed above. Thus, an algorithm for automatically generating inkblots is required. Such an algorithm should take as input information about the user (e.g. the username), information about the authentication target (e.g. the server's DNS name), and a random seed. Each time that a user decides to change passwords, the system generates and stores a new set of random seeds, one for each inkblot in the password. In the user's subsequent authentication attempts, the

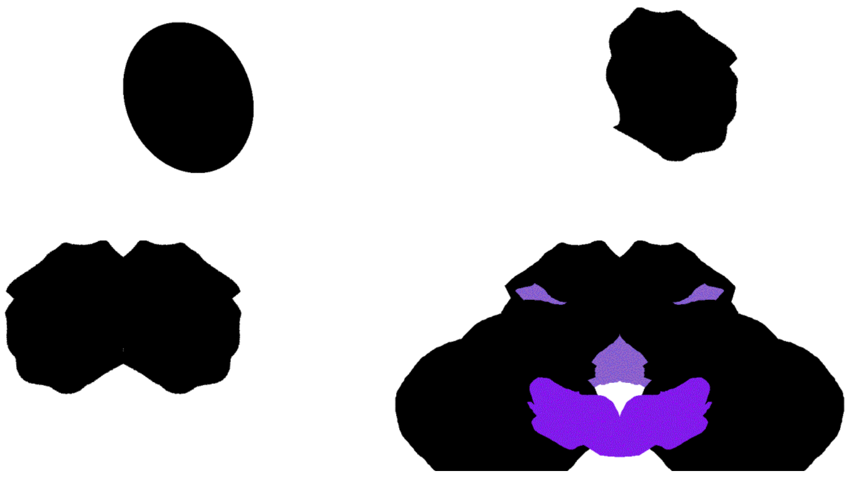

Figure 2: Steps in the automatic generation of inkblots: (a) a randomly rotated, colored oval is placed randomly; (b) the oval's border is perturbed by Perlin noise and a random jitter; (c) the resultant oval is mirrored across the y-axis; (d) the process is repeated for other ovals to complete the final blot

system uses these seeds to generate the (same) inkblots, rather than having to store the actual images. In the same way, if a user is attempting to authenticate to a server over a network, the seeds could be sent across the network and used to render the inkblot on the client's computer rather than wasting network bandwidth by sending the images themselves.

Though there are a number of techniques for simulating watercolors that could be used to closely approximate the original Rorschach inkblots [5], these techniques are too computationally expensive to be completed each time a user wants to authenticate to the system. Instead, a much simpler scheme can be used. Though the resultant blots do not have as much subtle detail as the original Rorschach blots, they do have roughly the same features of form, color, and symmetry.

The blot generation algorithm begins by cryptographically hashing together the user information, server information, and random seed. This hash is then used to seed a random number generator, which determines all of the random choices for the algorithm. Each blot element begins as a single randomly sized oval, randomly placed, colored, and rotated on the canvas (a). Each of these random selections is drawn from a non-uniform distribution that was created by trial and error during the design of the algorithm. For example, only a small set of colors is allowed, and black is the most likely color.

In order to induce a more "blot-like" appearance, the boundary of each oval is perturbed by adding scaled Perlin noise [13] and small amounts of random jitter (b). This "noisy" oval is then reflected across the y-axis to create the image's symmetry (c). This entire process is repeated 4 additional times, to complete the final inkblot (d). The process is shown graphically in figure 2.

4.1.1 Caveats

There are two security concerns to keep in mind when using the blot generator to generate images for the authentication system. The first involves a user who has inkblot-based passwords on multiple servers. Since all inkblot-generating seeds, as well as the inkblot images themselves, are considered public, servers could conceivably launch "inkblot-switch attacks." Because users associate a password with a set of images rather than with an individual server, a server could prompt a user with images normally generated by a different server. The user would then type the associations for the second server's images, giving the first server the ability to authenticate to the second server as the user. This attack can only be prevented by users checking that the correct server information is being used to create each set of images. Note that this problem occurs in all association-based password systems.

The second issue is that the inkblot authentication scheme could be used as an oracle to decide whether a given username exists on the system. Since no random seeds are stored for non-users, the system could be forced to either admit that a user does not exist or provide some set of ”fake” seeds. Fortunately, there is a simple solution: each month (or suitable password change period), the server generates a new secret and retains the previous month's secret. When a login request for a non-existent user is received, the username is hashed with one of the two seeds, based on a secret function of the username. This provides a consistent seed for each non-user, with random updates to the seed with the same regularity as a real user.

4.2 User Computable Hash Function

In the inkblot authentication scheme a user-computable hash function is needed to reduce a user's full association with an image to a small sequence of characters. The simple example we have used so far is to take the first character of the description of the user's association. Besides being easy for a user to compute, the hash function should be able to take an arbitrary association and map it in a deterministic way to a constant number of characters. (While it might be possible to use a non-constant length hash function, the user would need an additional entry to indicate that the next inkblot should be displayed. For this reason, we will only consider constant-length functions.)

A simple hash function of output length ℓ consists of the first ℓ characters of the association. Unfortunately, some associations can be as short as two characters (e.g. ”ox”). Also, this would produce english n-grams, which do not have very much entropy. In fact, any contiguous subset of characters from the association would suffer from the same problem.

Another possible hash function would take the first letter of each of the first ℓ words of the association. However, this function could not deal with associations that have fewer than ℓ words in them, or would require a special rule for such cases. We decided to avoid such special-case rules.

As the length of the hash function's output increases, so too should the amount of randomness present. However, simply harvesting more and more characters from each association produces diminishing returns. For example, predicting the first letter of a random English word is much more difficult than predicting the next letter of the English word “walkin.” Therefore, for a constant-length password, the tradeoff between number of blots and characters per blot must favor more blots in order to ensure that the generated passwords have sufficient entropy.

The hash function used by the final scheme takes the first and last letter of the association. We assumed that no blot would be associated with a single letter. This choice of hash function was made based on the data from prior inkblot experiments. While this choice produced good distributions of associations, as seen in section 5, it also had drawbacks. For example, quite a few of our test users saw many of a single object in blots, and their associations thus ended with the English plural morpheme “s' When recalling their associations, some users were unable to remember whether they had used the plural or singular form. Therefore, it might make sense to instruct users to always choose singular forms, even when they see multiple objects.

It is also possible for each user to choose a different hash function, so long as they all have the same output length. This would make an attacker's job more difficult, as the distribution of likely hash outputs would be more complex. However, our experiments showed that users who (against our instructions) chose to change their hash function had more difficulty correctly entering their passwords. This is discussed further in section 6.

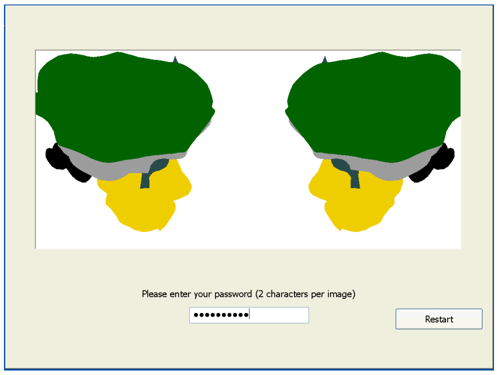

Figure 3: A screenshot of the program used in the inkblot authentication experiments

4.3 Experimental Methodology

In order to confirm our design decisions and validate the performance of the system, we conducted two user trials. In each trial, a group of self-selected users drawn from researchers, programmers, testers, administrative assistants, and secretaries were asked to use the system on three separate days. On the first day, each user was shown the inkblots and asked to choose a password based on them. (The actual instructions given to the users are included as appendix A.) The next day, the users returned, were shown the same inkblots, and were given up to three tries to reenter their chosen password. One week later, the users were again shown the blots, and again given three chances to authenticate.

When choosing their passwords, the users were first presented with the images in a randomly-chosen order. Then the order was randomly shuffled and the users were asked to reenter the password. This was done so that users would be forced to actually form associations with the images rather than simply ignoring them in choosing their password. The shuffled ordering was then used on all subsequent authentication attempts. A screenshot of the program used in the experiment is shown in figure 3.

4.3.1 The First Experiment: Original Rorschach Blots

In the first experiment, we used the original Rorschach inkblots instead of the computer-generated blots in order to compare the results obtained in our small trial to the larger data set of Rorschach responses from the psychology literature. We also broke our user population into three groups. Each group saw a different number of blots—either 6, 8, or all 10—so that we could estimate the maximum number of associations that a given user can retain.

4.3.2 The Second Experiment: Computer Generated Blots

In the second experiment, the computer-generated blots were used. However, instead of generating new blots for each user, we presented all users with the same set of blots, so that we could measure the entropy of responses to a single blot. Also, in this experiment, all users were shown a complete set of 10 blots, since the first experiment showed that this was no more difficult than remembering associations with 6 or 8. The ten inkblots, generated at random, used in the experiment are shown in figure 4.

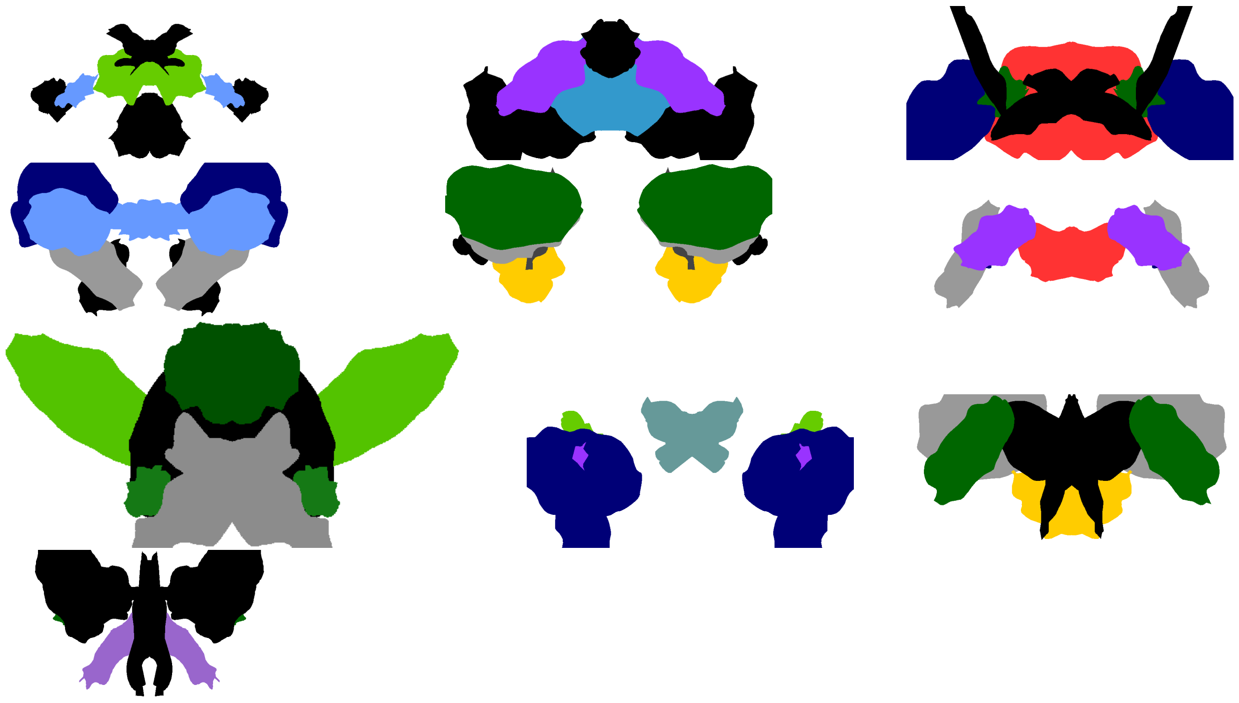

Figure 4: The ten randomly generated inkblots used in the user study

5 Password Strength

The most important criterion by which to evaluate inkblot authentication is the security of the passwords it produces. To quantify the security of our schemes we have measured the entropy in a set observed passwords. This measure corresponds to the difficulty facing an attacker trying to guess passwords given the best possible information about a group of users (specifically, the exact distribution of their responses to each blot). In principle, the attacker could approach this level of information by paying a large group of subjects to form associations with the target blots, and using the resultant distribution to attack the password. However, since in a real system each user has different blots, this technique would be impractical for a non-targeted attack or a casual adversary.

More specifically, we have calculated,

$$-\sum_{x \in R} \frac{x}{n} \lg \frac{x}{n}$$

where $x \in R$ represents the number of times a specific response was given and n is the total number of responses, for each character in an inkblot password. We are assuming here, for simplicity, that each character is chosen independently. If, for example, the hash used picked consecutive letters from the association, the second letter could be very dependent on the first (e.g. "u" following "q"). This assumption also means that we believe an attacker who is given some of a user's other associations will be no more likely to guess a new association than an attacker who has no information on the user.

Before conducting our own experiments, we computed entropy information from a large scale Rorschach data set [8]. This data set was gathered by projecting each inkblot onto a large screen at the front of an auditorium while study participants were asked to write down their associations, a situation surprisingly similar to the authentication scenario. Unfortunately, as with a traditional Rorschach test, the subjects were

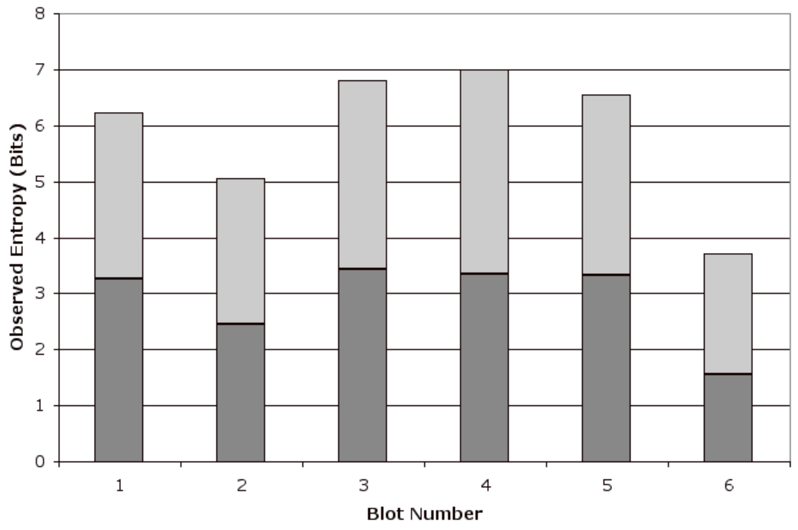

Figure 5: Observed entropies for six of the original Rorschach inkblots. Each bar shows the contribution for the first (lower) and second (upper) character.

encouraged to provide multiple associations for each blot.

While the data set is extensive (632 college age subjects, 116 adults, as well as prison inmates and psychopaths), it is far from raw. The researchers have grouped like responses together (e.g. disregarding pluralization) and only present frequency data for categories of objects (e.g "Anatomy (animal)"). However, by assuming the worst where information is missing, we can use the data to develop a lower bound on the amount of entropy to expect from each blot.

The data on the ten original inkblots shows that we can expect to see at least 2 bits of entry per character in the very worst case (remembering that most of the entropy in the data set was either removed during the grouping of the responses or by making worst-case assumptions when data was missing). Though this may not seem like much, even mildly strong, highly memorable passwords are useful for some situations, such as web service authentication. Though not indicative of what we would expect to see in user trials, we used this estimate to suggest a range of password lengths to test.

5.1 The First Experiment

Since different users were shown different numbers of blots in order to determine an optimal length for memorability, we will only compute the entropy for the six blots seen by all users. This data is shown in figure 5. The mean entropy per character was 2.94, with the total size of the measured entropy space for these 12 character passwords being 35.38. The blot that performed worst, number 6, was predicted by the large scale experiment data and by Rorschach himself, who declared it to be the easiest to associate [1]. The most common association on this blot was "bat," followed by "bird."

5.2 The Second Experiment

In the second experiment, all users were given the same number of blots (10) and so more complete data is available. The individual blot entropies are shown in figure 6. The mean entropy for each character of

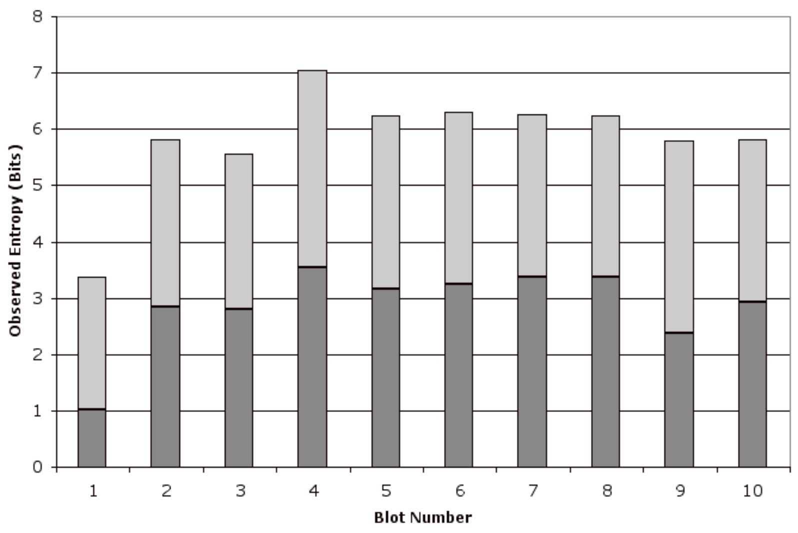

Figure 6: Observed entropies for the ten computer generated inkblots. Each bar shows the contribution for the first (lower) and second (upper) character.

these blots was 2.92, approximately the same as for the original blots. Thus the size of the full 20-character password space is 58.45 bits. Interestingly, the blot that fared worst in this set also caused users to either see a "bat" or "batman." It it also worth pointing out that besides the "bat" outlier, the other generated blots performed similarly. This is evidence that the blot generator can be expected to output blots with passwords of approximately the same strength, so long as a reasonable number of blots are used.

Because "bt" appears as an extremely common response in both experiments, the system could disallow the use of "bt" as a response. This could be accomplished by simply replacing each blot that a user selects "bt" for with a new blot during password generation.

5.3 Analysis

These entropy estimations provide excellent expectations for the strength of inkblot passwords against a very powerful adversary. It is interesting to also consider attacks that could be launched without the use of per-blot distribution information—that is, how an attacker might launch a very general, low-cost attack on the system. A natural hypothesis might be that the distribution from which associations are drawn is similar from blot to blot. By computing frequency statistics, we see that this is indeed the case. Figure 7 shows the distribution of characters in both the original blots and in the computer generated blots. It is clear that the distributions both resemble each other and the distribution found in English, though the "b" spike in both distributions is unexpected. The distribution also shows that some users chose characters outside of the expected lowercase letters. One user chose a proper noun as an association and decided to capitalize it. Even more inventively, one user typed his associations in Dvorak on the QWERTY keyboard, something he claims to do with all his passwords.

Though these frequencies could be used to launch an attack better than brute force, the amount of entropy present in the combined data is increased. For both the original and computer generated inkblots, the entropy across the blots was 3.97 and 4.09 bits per character, respectively. Thus, for 20 character passwords, the entropy in the password space would be approximately 80 bits. While this measured entropy is smaller than

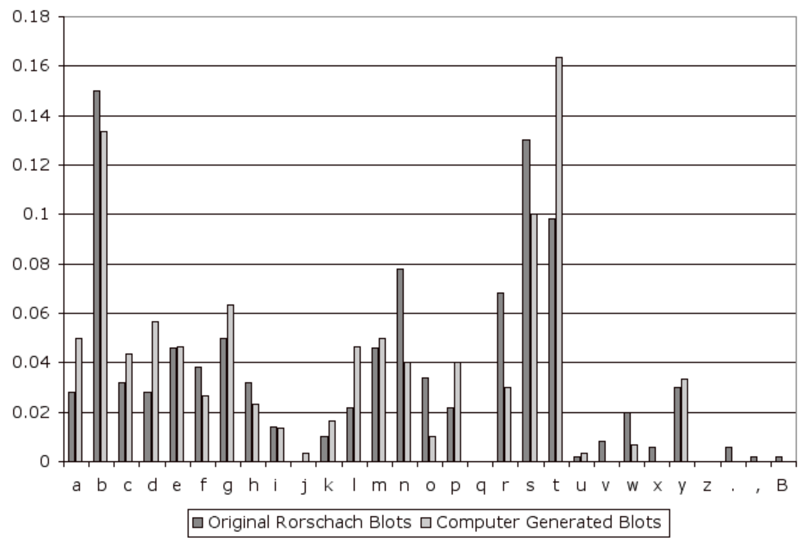

Figure 7: The distribution of each character across the two sets of blots.

the theoretical maximum (94 bits, assuming only lowercase characters; 160 bit assuming an extended ASCII character set), it is more than sufficient for preventing a search of the space.

5.4 Guessing Individual Passwords

While the bounds on the entropy of the password space provide a measure of the security for a *set* of passwords, they do not precisely characterize the difficulty in guessing an individual password from the set. However, the blot-specific frequency information suggests a natural strategy for searching the space: on each iteration the attacker tries the most probable (according to the frequencies) password that has not yet been guessed. Using this strategy, the weakest password in our computer generated inkblot data set would be guessed only after more than 5 million tries. While not resistant to an offline attack, even this weakest password would be secure in an online setting where entry delays and an eventual account lock could be introduced. This compares favorably to passwords, which can often be guessed using dictionaries with under 3 million entries (e.g. 25% of 15,000 passwords were guessed using a dictionary with approximately 2.8 million entries by Klein [11]).

In the more realistic threat model, however, the attacker only has access to the cross-blot frequency data. Using this data, the weakest password will only be guessed after more than 564 million attempts. If the attacker wants to guess a specific password, his job becomes even tougher. For example, the mean number of tries needed to successfully guess a password in the data set was more than 10^{17} (equivalent to searching a 56-bit space). Thus, many of the passwords are even resistant to offline attack. In this scenario, however, the attack often has an additional piece of information: the identity of a particular user. We have found no evidence to support the conclusion that knowing a person assists an attacker in guessing associations. Besides having to guess the semantic association, the attacker must also guess the textual association that the user has decided to enter. Even the most ardent Rorschach followers have not claimed that this is possible.[2]

[2]One could imagine the following experiment being used to reject this hypothesis. First, gather a group of subjects containing

6 Password Memorability

Though the memorability of Rorschach associations is not a part of the original personality test, research on the variability of responses in repeated tests has been done [1]. Even when the tests are conducted months apart and subjects are only asked to form free associations (i.e. not to attempt to remember their previous responses), “normal” subjects repeat many of their associations, sometimes with new associations added [14]. Unfortunately, this result is not directly relevant to the authentication scheme, as users are forced to choose a single association per blot. Also, it is unclear from the recorded data whether the subjects simply reassociated the same semantic idea, or whether they also used the same verbal association to describe it—that is, whether the subjects always used, say, the word ”cat” to describe a blot, or whether the examiner simply lumped the responses ”feline” and ”kitty” together with ”cat” when recording responses. For these reasons, the memorability data from the two user studies is particularly important.

6.1 The First Experiment

The memorability results from the original Rorschach inkblots are shown in figure 8. With one exception, all users were able to remember all but one of the blot associations correctly after one day, and again after one week. The one outlier is actually surprisingly easy to explain. One user, in an effort to see if he could “beat the system,” decided to hold down a single key for all of the blots. Thus, his password was “gggggggggggg.” However, upon returning the next day he was unable to correctly remember what key he had held down and his three guesses were all incorrect. Fortunately, he was able to guess correctly a week later. This example shows that it was actually *easier* for the users who followed the system and constructed strong passwords to remember them than for the one user who attempted to choose a very poor password. Also, this situation could easily have been avoided, in retrospect, had the restriction been imposed that no more than two of the hashes in a given password be the same.

6.2 The Second Experiment

The memorability results for the computer-generated inkblots are shown in figure 9. In this experiment, all users were able to remember at least nine out of their ten associations, after both a day and a week. Additionally, after the experiment was complete, users who missed an association were able to identify the association they missed. Besides the expected problems with choosing the wrong association, there were two additional classes of problem. First, two users admitted to using the hash function incorrectly. When setting their passwords, for some of the associations they used the “first-last” hash, as instructed; however, each used, in one case, the first letter of each word of a two-word association. When they were re-entering the password later, they could not remember on which response they had used the alternative hash function. A second problem occurred when some users got confused between the singular and plural forms of their association. For example, when choosing their password they chose to hash “cats,” whereas when re-entering the association they dropped the plural and hashed “cat.”

---

pairs of mutual acquaintances such as spouses, family members, or close friends. Ask one member of each pair to provide their full (non-hashed) association to a number of blots. The other member of the pair is then asked to choose their partner’s response out of the set of responses for each blot. If a close friend is unable to even differentiate their partner’s responses from other users, it is unlikely that an attacker could *create* correct responses by knowing a particular user’s identity.

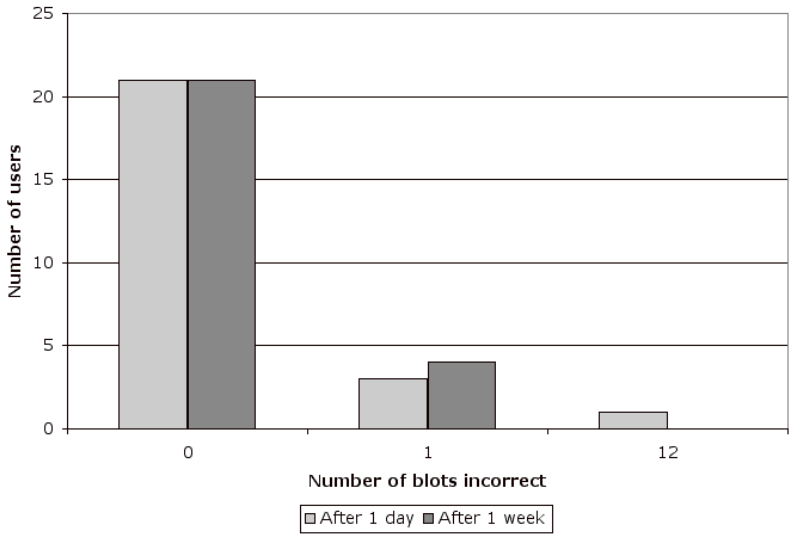

Figure 8: Memorability of original Rorscach inkblots. The outlier is explained in the text.

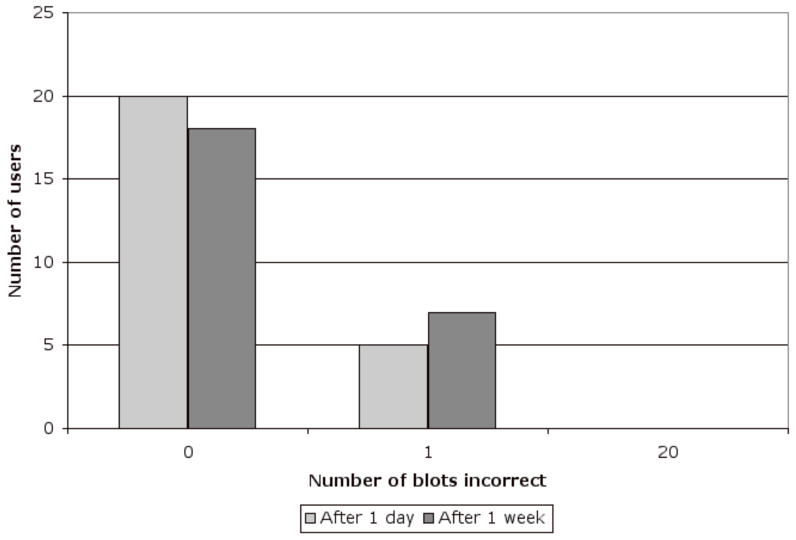

Figure 9: Memorability of computer generated inkblots. No user was able to remember less than 9 of their 10 associations.

6.3 Analysis

In both experiments, users missed at most one association, even after having not used the system for one week. Thus it may be advisable to modify the system to allow for successful authentications when *k* out of a possible *n* associations are correct. Assuming that all blots produce an equal distribution on responses, this reduces the security of passwords to the level of the original system with only *k* blots. Therefore, it might be advantageous for users to have to enter associations for more blots. A disadvantage of this approach, however, is that authentication would take longer.

7 Conclusion

Our preliminary data suggest that inkblot authentication offers a potentially significant improvement over existing widely-deployed user authentication mechanisms. In addition to gathering our quantitative results, we also asked users who had taken part in our experiments for their comments on the system. In almost all cases we received the same response: the users were happily shocked that they could remember such a "huge password." In fact, many users asked if there were any plans to allow the use of the system in their production environment. This kind of positive user experience is arguably as important to the eventual adoption, acceptance and scrupulous use of an alternative password system as any measure of security.

More experiments would help confirm or discount our security and memorability results, and could answer such questions as: How many inkblots (that is, how much entropy) can be used before the resulting passwords are no longer memorable? What is the best way to help users retain their inkblot associations? What inkblot-to-character hash function generates the most entropy without sacrificing ease of use? And what inkblot generation algorithms create inkblots with the highest-entropy (or the fewest low-entropy) association spaces?

While inkblot authentication should be quite easy to deploy in a wide variety of settings, there exist some environments (such as devices with tiny screens) where it is unworkable, and alternatives are needed. Adapting the inkblot password scheme to other password-using contexts, such as those in which the user interface is under the control of a (possibly uncooperative or legacy) application, may also require some innovative thinking.

References

[1] BECK, S. J., BECK, A. G., LEVITT, E. E., AND MOLISH, H. B. *Rorschach's Test*, vol. 1. Grune and Stratton, New York, New York, 1961.

[2] BISHOP, M. Improving system security via proactive password checking. *Computers and Security 14*, 3 (Apr 1995), 233–249.

[3] BLONDER, G. Graphic passwords. United States Patent 5559961, 1996.

[4] BROSTOFF, S., AND SASSE, A. Are passfaces more usable than passwords? a field trial investigation. In *HCI 2000* (Sep 2000).

[5] CURTIS, C., ANDERSON, S., SEIMS, J., FLEISCHER, K., AND SALESIN, D. Computer-generated watercolor. In *SIGGRAPH '97* (Los Angeles, CA, Aug 1997).

[6] DHAMIJA, R., AND PERRIG, A. Deja vu: A user study using images for authentication. In *9th USENIX Security Symposium* (Aug 2000).

[7] FELDMEIER, D., AND KARN, P. Unix password security — ten years later. In *Crypto '89* (Aug 1989).

[8] HARROWER-ERICKSON, M. R., AND STEINER, M. E. *Large Scale Rorschach Techniques*. Charles C. Thomas, Springfield, Illinois, 1945.

[9] HOLTZMAN, W. H., THORPE, J. S., SWARTZ, J. D., AND HERRON, E. W. *Inkblot Perception and Personality*. University of Texas Press, Austin, Texas, 1961.

[10] JERMYN, I., MAYER, A., MONROSE, F., REITER, M. K., AND RUBIN, A. D. The design and analysis of graphical passwords. In *8th USENIX Security Symposium* (Aug 1999).

[11] KLEIN, D. Foiling the cracker: A survey of, and improvements to, password security. In *Proceedings of the Second USENIX Security Workshop* (Aug 1990).

[12] MORRIS, R., AND THOMPSON, K. Password security: A case history. *Communications of the ACM 22*, 11 (Nov 1979), 594–597.

[13] PERLIN, K. An image synthesizer. *Computer Graphics 19*, 3 (1985).

[14] RICKERS-OVSIANKINA, M. *Rorschach Pyschology*. John Wiley and Sons, New York, New York, 1960.

[15] RORSCHACH, H. *Psychodiagnostik*. Bircher, Bern, 1921.

[16] WOOD, J. M., AND LILIENFELD, S. The rorschach inkblot test: A case of overstatement? *Assessment*, 6 (1999), 341–349.

[17] YAN, J., BLACKWELL, A., ANDERSON, R., AND GRANT, A. The memorability and security of passwords – some empirical results. Tech. Rep. 500, Computer Laboratory, University of Cambridge, 2000.

A Text of instructions given at beginning of experiment

In this experiment, you will be testing a new system designed to help users remember passwords. You're going to be shown a series of random inkblots. Your password will be what you see in each blot. However, instead of typing in the entirety of what you see, you'll only type in the first and last letter. For example, if I saw the "jolly green giant" in on of the blots, I would enter "jt."

In the experiment today, you will be shown the blots and asked to enter your password once. Then the blots will be shuffled and you will be asked to confirm your password. The blots will be in a different order, so you must match your two letter responses to the blots.

In the two subsequent experiments, you will be shown the same blots and asked to reenter the same password that you select today. Please take your time on each blot and wait until you see something you think you will remember.