How Much Assurance Does a PIN Provide?

Jon Bentley and Colin Mallows

Avaya Labs, 233 Mt. Airy Road, Basking Ridge, NJ 07920, USA
{jbentley, colinm}@avaya.com

Abstract. We would like to quantify the assurance contained in an authentication secret. For instance, how much assurance does a customer convey to a bank by revealing that his Personal Identification Number (PIN) is 1111? We review a number of previously proposed measures, such as Shannon Entropy and min-entropy. Although each is appropriate under some assumptions, none is robust regarding the attacker's knowledge about a nonuniform distribution. We therefore offer new measures that are more robust and useful. Uniform distributions are easy to analyze, but are rare in human memory; we therefore investigate ways to "groom" nonuniform distributions to be uniform. We describe experiments that apply the techniques to highly nonuniform distributions, such as English names.

1 Introduction

To gain access to a computer system, a user typically presents both a public name and a secret password. When the user tells that secret correctly, the system gains assurance that it really is that user, and not a lucky guesser. How much assurance is in a password? Very little if the password is "dog" (especially if a web search for the user reveals that he owns dogs), but much more if the password is "tbdam3CS@h". Can we quantify our intuitive notion of that assurance?

Quantification would have important implications in security engineering. Current password policies are based on rules such as insisting that a password must be eight characters long, include a mixture of upper and lower case and numerics, and contain no dictionary words. We would like to quantify our intuition that "q#acwm!" is a better password than "Supercalifragilisticexpialidocious1", even though the former violates all rules and the latter is five times longer. Ideally, a security engineer would weigh assurance against cognitive effort to choose provably good password policies.

Similar issues arise in most Human Interactive Proofs. If the correct response to a CAPTCHA is "iw5r7Acq", then we feel that we have more assurance than for the string "cat". To make a quantitative statement about that example, we would have to take into account how the CAPTCHA chooses text, how the text is transformed, how the attacker recognizes characters (and what mistakes it makes), the strategy the attacker employs, and many other issues.

Most of this paper will therefore be devoted to a problem that is relatively easy to state, yet still important: how much assurance is provided by a 4-digit Personal Identi-

H.S. Baird and D.P. Lopresti (Eds.): HIP 2005, LNCS 3517, pp.111-126, 2005.

fication Number (PIN)? Extensions to more subtle problems are examined in later sections.

2 The Technical Question

To use an Automated Teller Machine (ATM), a customer needs both a bank card and the corresponding 4-digit PIN. If the card is lost or stolen, the PIN provides some protection against its unauthorized use by an attacker. How can we measure the amount of protection the PIN affords?

We take as axiomatic that the degree of protection, which we call assurance, is measured by the probability p that an attacker can guess the PIN. We find it convenient to speak of the number of bits[1] of assurance, which is $-\log_2 p$ (all further logarithms in this paper are base 2). We write $assurance(C|A) = -\log P(C|A)$, where $P(C|A)$ is the probability that an attacker A guesses the customer C's PIN correctly. The bank may feel that by implementing a 4-digit system, they have provided each customer with $\log 10000 \sim 13.3$ bits of assurance. But can a particular customer, who may have chosen as his PIN the easy-to-remember number 1111, have this degree of assurance? And what about a sophisticated attacker, who has found a bank card, and knows a lot about how people choose PINs?[2]

We do not have access to large sets of real PINs for real ATM cards, so we did several simple experiments to study the PINs that people use in other domains. A web search for "passwords name pin cheats" gave a variety of web sites that list "cheat codes" for computer and video games, many of which require a 4-digit PIN. The web site www.gamefaqs.com/computer/doswin/code/25003.html, for instance, gives this set of 41 PINs, which we present in sorted order:

0201 0310 0322 0425 0517 0526 0530 0604 0818 1029 1111 1111
1111 1111 1111 1111 1112 1122 1221 1234 1836 2220 3141 3246
3333 3333 3333 3691 4288 4393 4440 5158 5651 6000 6660 6765
6969 7761 7777 8148 8337

The authors feel confident in asserting that a uniform random process did not generate those 41 PINs. Six of the numbers are 1111; an attacker who guesses that PIN has about a 15% chance of success, which indicates under 3 bits of assurance. Four other PINs also contain a single digit. The PINs 1112, 2220, 4440, 6000 and 6660 deviate in just a single affix digit. Of the numbers that begin with the digit pair 00 through 12, all but 1234 have second digit pairs less than 31, which make us think of dates (birthdays seem particularly likely). Excluding these PINs leaves us with less than a third of the original set:

---

[1] Shannon [1948] begins by counting the number of messages, and quickly moves to the logarithmic measure of bits. We find the logarithmic measure more appropriate for assurance for the same three reasons that Shannon found the logarithmic measure more appropriate for information: it is more useful, nearer intuition, and more mathematically suitable.

[2] A bank may allow a customer several attempts at entering a PIN, to allow for mistyping and the like. If the attacker fails on his first attempt, he should simply ignore that PIN and proceed with the next-best possibility. For simplicity, we ignore this complication, and assume that only one attempt can be made.

1836 3141 3246 3691 4288 4393 5158 5651 6765 6969 7761 8148 8337

Even this subset does not appear to have been chosen at random; only three of the thirteen PINs have four different digits, though more than half of the 10000 possible PINs have that property.

This simple experiment is not atypical. At another site, 14 of 34 players used the PIN 1111. One of the authors is willing to admit that at one time, one of the three ATM cards in his wallet had the (default) PIN 1111 (that card is now discarded). Such experiments and decades of bitter experience (see Morris and Thompson [1979]) lead to our basic assumption:

Humans tend to choose secrets in nonrandom and repeated patterns.

3 Assurance Is Not Entropy

We might assume that a 4-decimal-digit keyspace of PINs means that an individual PIN yields about 13.3 bits of assurance. But if many users choose the PIN 1111, then that particular PIN should carry fewer bits of assurance. How can we quantify that intuition? Many information scientists (including the authors and several of their colleagues) jump to the answer, "Entropy, of course!" That answer is wrong.

One approach to authentication is to ask people questions that are easy to remember yet are hard for attackers to guess (see O'Gorman, Bagga and Bentley [2004]). Such a question with four answers can be viewed as a 2-bit PIN. Our internal corporate web site offers a daily straw poll with questions such as "What is your favorite way to celebrate a holiday?" The answers and their (rough) percentages are

Gather with family and friends	71%
Catch up on chores	17%
Attend a wild party	11%
Volunteer at a shelter	1%

The maximum possible entropy for four answers is 2 bits (when each occurs uniformly, 25% of the time). Interpreting the percentages as probabilities gives an entropy for these nonuniform answers of 1.2 bits. On the other hand, an attacker who guesses "family and friends" has a 71% chance of being right on this question, and slightly more than a 50% chance of being right on two such questions in a row. Our intuition says that such an answer should therefore be worth at most half a bit of assurance. Evidently, assurance is not entropy.

Shannon [1949] first examined related issues in terms of "secrecy systems". A great deal of work has been done since then in "authentication theory". Cachin [1997, Section 3.1] surveys information measures that have been used in cryptography to characterize probability vector $p = (p_1, p_2, ..., p_N)$, where the probabilities have been ordered so that $p_1 \geq p_2 \geq ... \geq p_N$. He starts by reviewing the classical Shannon entropy of $-\Sigma_i p_i \log p_i$. He also describes the min-entropy of $-\log p_1$, which characterizes the probability of guessing the most likely element, and therefore the largest security hole (such as "family and friends" above). The guessing entropy $\Sigma_i i p_i$ gives the expected cost of a sequential search (Knuth [1973, Section 6.1]) for a secret, starting at the most likely guess and progressing until the answer is found. This measure might describe the time required to guess a hashed password, given a dictionary of all

passwords ordered by frequency. Other measures that Cachin describes include *relative entropy*, *Renyi entropy of order* α, *collision probability*, and *variational distance*. None of these measures, though, directly addresses the issue of concern to us: how much assurance is in a particular secret? (Though we will see in the next section that many of these measures are very relevant in particular contexts.)

In his "Unified and generalized treatment of authentication theory", Mauer [1996] cautions us that "Compared to the theory of secrecy, authentication theory is more subtle and involved." Ellison, Hall, Milbert and Schneier [1999] point out that "Research needs to be done on the actual entropy (from the attacker's point of view) of a given class of answers…." With that warning and challenge, we now address those issues.

4 Three Views of Nonuniform Probabilities

Three players are in the game. The first is the Bank, B, which chooses N, the size of the key-space (N=10000 for a 4-digit PIN). The second is the customer, C, who chooses a particular PIN $x(C)$, and who is interested in the probability that this particular PIN can be guessed. The third is the attacker, A, who (we assume) may know everything about the key system, except the value of the particular PIN that goes with the bank card he has found.[3] In particular, we assume that the attacker may have access to statistical information regarding the frequencies of the various PINs.

Each of these players has different concerns. The customer is interested in the degree of protection afforded by his own PIN. The bank is interested in one or more summary statistics regarding its customers, for example the average degree of protection they have, or the protection given to the least-protected customer. The attacker is interested in the probability that he will be able to guess the PIN associated with the particular bank card he has found, which we assume is randomly chosen from all the bank customers.

4.1 The Attacker

It is convenient to start with the attacker. A naïve attacker will guess all possible values of $x(C)$ equally often, with the chance of success of precisely $1/N$. The bank therefore achieves $\log N$ bits of assurance against that attacker, which we will refer to as the *max-entropy* (by analogy with min-entropy). At the other extreme, a well-informed attacker might know the probability vector $p = (p_1, p_2, ..., p_N)$, which gives the probabilities with which each of the N possible PINs is used. We suppose the possible PINs have been ordered so that $p_1 \geq p_2 \geq ... \geq p_N$. Now suppose that $p_1 = p_2 = ... = p_k > p_{k+1}$. Then the attacker's optimal strategy is to guess one of $x_1, x_2, ..., x_k$, and (whether or not he randomizes among these possibilities) $P(C|A)$ is just p_1. Any

[3] The Attacker (A), Bank (B) and Customer (C) of authentication collectively apologize to Alice (A), Bob (B), and Carol (C) of cryptography for unintentional identity theft.

other strategy will decrease $P(C|A)$. In this case, the min-entropy accurately characterizes the weakest link in the chain.

In the first case, the attacker has information about no PINs, while in the second case, he knows the distribution of all PINs. To define a state of knowledge intermediate between these two, we consider an attacker who has access to a single random PIN, say y, chosen from all the PINs of the bank's customers. The optimal strategy for this "single-peek" attacker is to guess $x(C) = y$, and his chance of success on this occasion is p_y. The overall chance of success for an attacker of this type is $P_A = \sum_y (p_y)^2$ (because it is the probability that the customer's choice of PIN matches the one the attacker has seen). We have $1/N \le P_A \le p_1$, so the assurance is between the max-entropy and the min-entropy. Furthermore, the weighted average assurance of a customer with respect to this single-peek attack is the Shannon entropy $-\sum_i p_i \log p_i$.

One can consider the more general case of an attacker who has a sample of size m of the PINs; the previous results are special cases for $m = 0$, 1, and ∞. Bentley and Mallows [2005] prove that P_A is in fact a non-decreasing function of the size of the sample.

4.2 The Customer

Now consider a customer C with a particular PIN, $x(C)$. This customer wants to know the probability that an attacker will guess exactly this PIN. If the corresponding p_x is not equal to p_1, we could argue that this customer is perfectly protected, since an optimal attacker with complete knowledge will never guess this PIN. But what about an attacker who chooses a different strategy? To measure the degree of assurance the customer has, we need to be specific as to what attacks are considered. Clearly, there is no protection against an attacker who has inside information as to this particular PIN. We assume that the most an attacker can know is statistical information as to the distribution of PINs over the bank's customers, that is, the complete vector p. We cannot allow the possibility that an attacker may have any possible (mistaken) value of p, since this would allow the possibility that he believes that the particular PIN $x(C)$ is very likely. We need to specify the information the attacker may have, ranging from none to complete (accurate) knowledge of p.

A naive attacker will guess $x(C)$ correctly with probability $1/N$, no matter what p_x is. An attacker who has seen a single PIN, say y, and who guesses that value for $x(C)$ will succeed with probability p_y. An attacker who knows p completely, and who therefore guesses some y with $p_y = p_1$, can succeed (in guessing $x(C)$) only when $p_x = p_1$ and otherwise is sure to fail. We suggest that an appropriate measure of the assurance that a customer C has is the minimum of these three values, so that

$$assurance(\mathrm{x}) = -\log \max(1/N,\ p_x) \qquad (\mathbf{1})$$

Note that this conservative formula is correct both when $p_x = p_1$ and when $p_x \neq p_1$.

The formula (1) is not completely satisfactory. Consider a key-space with N=5, and two alternative p-vectors: p^a = (.4, .3, .1, .1, .1) and p^b = (.3, .2, .2, .2, .1). According to (1), the PINs with p = .3 are equally secure in these two cases, but this seems wrong since an attacker who knows p (or even an approximate value of p) will

guess this PIN correctly in case (b) but will never do so in case (a). More appropriate measures of assurance might take rank into account.

4.3 The Bank

While a naive view is that a 4-digit PIN affords the max-entropy of log 10000 ~ 13.3 bits of assurance, the bank cannot claim that each individual customer has this degree of assurance. More relevant measures include the mean assurance (the Shannon entropy, for a single-peek attacker) and the assurance given the most vulnerable customer (the min-entropy). The bank can improve these measures in several ways. One way, which is done by few banks and resented by many customers, is to issue random PINs to customers instead of allowing customers to choose their own PINs. But even this approach raises difficult questions: What if current customers have typical PINs, and a random process issues a new customer the common PIN 1111?

Alternatively, the bank can urge its customers to avoid "common" PINs, but how can this be done without giving away information as to which PINs are common? One possibility is for the bank to urge its customers to choose PINs that contain four different digits. This reduces the key-space to N = 5040, and so surrenders about one bit of assurance relative to the full N=10000. Yan, Blackwell, Anderson and Grant [2000] apply such an approach to computer passwords.

4.4 Who Knows What When?

We have seen that assurance depends strongly on what the attacker knows about the distribution of the PINs. If the attacker assumes that each element is equally likely, then the assurance is the max-entropy of log N bits; if the attacker knows the most likely key, then the assurance is the min-entropy of $-\log p_1$ bits; if the attacker samples a single key, then the weighted average assurance is the Shannon entropy.

The attacker's strategy can also change as a function of what he knows (or assumes) about the assurance that the bank assigns to each guess. If an attacker knows the complete probability distribution, and also knows that the bank assigns the same assurance to every correct answer, then his optimal strategy is to choose the most likely PIN. But if the bank knows that that attacker will behave in exactly that way, then the bank should assign small assurance to the most likely answer. In fact, if the bank assumes that the knowledgeable, rational attacker makes that choice with probability 1, then it must associate $-\log 1 = 0$ bits of assurance with that answer, and infinite assurance with every other answer.

But if the attacker in turn knows that the bank employs that modified policy, then the attacker's revised optimal strategy is to choose the second most frequent PIN. And so it goes, depending on who knows what when. After wandering through a game-theoretic analysis reminiscent of "Rock-Paper-Scissors", we soon arrive at "The Paradox of the Surprise Examination" (see Wischik [1996]).[4]

[4] In that paradox, the teacher announces to a class that there will be an exam one day next week (Monday through Friday) on a day when the students do not expect it. But the exam cannot

Many analyses show such a lack of stability. The bank posits a set of probabilities and assurances, and analyzes the attacker's strategy, which results in a new and distinct set of assurances. A stable strategy always exists in a two-person, zero-sum game. Unfortunately, we do not see how to formulate the present problem as such a game. We will study an alternate approach to stability in the next section.

5 Inducing Uniformity

Analysis of uniform probabilities is straightforward. Unfortunately, few events in human memory are truly uniform; humans tend to know obscure but nonuniform facts. In this section we will study ways in which we can induce uniformity.

5.1 Accumulating Assurance

So far we have considered the assurance of a single transaction: how much does one answer yield? In many contexts, though, we are interested in accumulating the assurance of a sequence of questions. To gain access to personal financial information, for instance, one has to give correct answers to a series of questions such as "what is your birth date?" and "what are the last four digits of your Social Security number?" We now turn our attention to systems that collect a large number of questions and answers from a user at registration, and at login ask a subset of the questions to authenticate the user in the presence of potential attackers.

The probability that an attacker correctly guesses one of 16 items chosen uniformly is 1/16, which corresponds to 4 bits of assurance. If the bank presents a second independent question of 16 choices, then the probability and bits are the same. The probability of correctly guessing both answers multiplies to 1/256, and the bits correctly sum to 8. Straightforward accumulation of assurance is indeed straightforward.

Accumulation becomes subtle when the attacker knows more about the bank's mechanism. If the attacker has guessed enough correct answers to need just one more bit of assurance to break in to an account, for instance, then he might take a very different approach than when he still needs to accumulate many bits, and this can substantially change the attacker's optimal strategy. The change in strategy can change probabilities, which also changes bits of assurance. Henceforth we will ignore this complication, and assume that the attacker's goal is to act for the long run.

5.2 Grooming a Single Question

Perfectly uniform distributions are hard to attack, easy to analyze, and, unfortunately, relatively uncommon in human memories. Even gender is not determined by a fair

---

take place on Friday, because after Thursday had passed with no exam, the students would expect it on Friday. For the same reason, the exam could not take place on Thursday, and so on.

coin toss -- the *CIA World Factbook* (at www.cia.gov/cia/publications/factbook/) reports that at birth the male/female ratio in the USA is 1.05. But just as dieters hope that "inside every fat person is a thin person trying to get out", so we observe that "inside every skewed distribution is a uniform distribution trying to get out".

To induce uniformity into a nonuniform binary question, for instance, we can randomly exclude some members of the larger set. For example, assume that 1000 registrants report that their gender is female, and 1050 report male. An attacker might gain a slight advantage by guessing male more often than female. We can remove that advantage by randomly selecting 50 of the males to exclude from that question; we instead use other questions to verify their identity. The result is a perfectly balanced question, which provides precisely one bit of assurance.

For a multiple-choice question, assume that 1000 responses to a four-answer question occur with these nonuniform frequencies:

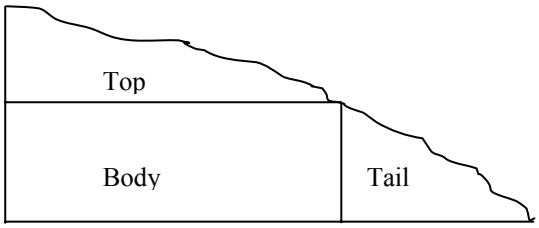
A	B	C	D
350	325	300	25

We can *groom* this into a perfectly uniform question for 900 of the respondents by excluding all 25 Ds and randomly excluding 50 As and 25Bs:

A	B	C
300	300	300

We have chosen answer C as the *grooming point*. We will no longer ask this question of the 100 excluded respondents (50 As, 25 Bs and 25 Ds), and instead ask other authentication questions of them. (In this single-question grooming, it is important to exclude the respondents before any login attempts; if we randomize at each login, an attacker might observe that a certain question is asked rarely, and thereby deduce that the particular respondent gave a common answer.) We have induced perfect uniformity by trimming ten percent of the responses, at the cost of reducing the number of bits of assurance per response from the maximum possible of 2 to just $\log 3 \sim 1.585$. If a knowledgeable attacker does not know that the distribution has been groomed, he might still tend to answer A; his probability of success is exactly 1/3, which is accurately reflected in the bits of assurance.

The grooming process divides a distribution into three parts:



The grooming point is the right boundary of the Body, and the height of that particular value defines the top of the Body. The elements in the Top are discarded at random; the elements in the Tail are all discarded, and the result is the uniform Body. Those remaining elements are truly uniform. If the grooming point is at the G^{th} element, then $\log G$ bits of assurance are assigned, and the probability of an attacker guessing a randomly chosen element on the first try is $1/G$, and by the K^{th} try is K/G; the expected number of guesses before success is $(G+1)/2$, and the worst-case number

is G. It is not coincidental that after grooming, we have the condition $p_1 = p_2 = ... = p_G > p_{G+1} \geq ... \geq p_N$ that we analyzed in Section 4.1.

We have extended the basic idea of grooming in several ways. We can admit the elements in the Tail (the 25 people who answered D above, for instance) without compromise by still assigning just log G bits for each answer. This conservative policy provides a lower bound on the number of assurance bits and therefore an upper bound on the probability of an attacker successfully guessing.

In the example above, answer C was an obvious grooming point. Grooming to answer D would have reduced all counts to 25, and therefore excluded too many answers. Grooming only to answer B would have raised one count from 300 to 325, but at the cost of reducing the bits of assurance from log 3 to log 2. A linear-time program can scan a sorted sequence of frequencies (or probabilities). If registration time is critical, then one might choose the grooming point as the item that maximizes the number of bits per response times the number of non-excluded responses, which is the bits per original response. If login time is critical, then one might choose a larger grooming point to collect more bits at each authentication attempt.

The table below shows the effect of grooming point. The first four columns give the percentages of a population (in nonincreasing order), and the last three columns present the bits per original response for the three grooming points. The first three rows give uniform distributions (over 4, 3 and 2 answers), the fourth row gives the example we used earlier in this section, and the next seven rows give real distributions from the straw poll referred to in Section 3. Because the first row describes a uniform distribution, the right entries give the max-entropy. The 8th line in the table shows that bits per original response is not unimodal in the grooming point. Section 6.1 describes grooming larger data sets.

Percentages				*Grooming Point*		
				2	3	4
25	25	25	25	1	1.58	2
33.3	33.3	33.3		1	1.58	
50	50			1		
35	32.5	30	2.5	0.98	1.47	0.20
31.3	30.9	20.6	17.2	1.00	1.25	1.37
34.5	34.4	16.0	15.1	1.00	1.00	1.21
38.6	32.7	20.5	8.3	0.94	1.10	0.66
40.7	35.4	12.7	11.2	0.95	0.78	0.89
63.2	26.1	6.6	4.2	0.63	0.38	0.33
63.6	18.3	10.8	7.3	0.55	0.63	0.58
65.7	17.0	9.8	7.5	0.51	0.59	0.60

Some pairs of questions that are singly well suited for authentication have the disadvantage of being statistically correlated. At an extreme, the questions of state of birth (with the answers Hawaii and Minnesota) and favorite childhood sport (with the answers surfing and ice hockey) likely show a strong correlation. Once an attacker guesses one answer to a question, he should be constrained about further guesses.

We can incorporate this fact into the analyses of Section 4 by using conditional probabilities.

Alternatively, we can remove the need for conditional probabilities by pairwise grooming to induce independence. Suppose that two potential authentication questions ask of voters in the 2004 USA presidential election "With what party are your registered?" and "For whom did you vote for president?" Further assume that in one (illustrative, not typical) community, 1000 respondents had this matrix of answers:

	Republican	Democrat
Bush	300	200
Kerry	200	300

That is, 300 Republicans voted for Bush and 200 Republicans voted for Kerry, while the Democrats voted in exactly the opposite numbers. We can replace that matrix with the uniform 2×2 matrix that consists of four entries of 200 by randomly grooming out 100 of the 300 Republican Bush voters and 100 of the 300 Democratic Kerry voters.

5.3 Grooming a Series of Questions

Suppose that the bank has recorded the answers that each customer has given to a large set of questions. We assume that an attacker knows the frequency with which each answer is given in the population of customers, but does not know anything about the particular customer he is attacking. To assess an attack, the bank needs to choose some set of questions to ask, such that the probability that the attacker succeeds in answering all these questions correctly is less than some preassigned value.

This objective can be achieved in the following way, at least when all questions have the same number of answers. Suppose the responses to each question have been ordered from most popular to least popular. The bank chooses a sequence of M desired responses as a sequence of independent random variables having some common distribution F, which the bank chooses; the attacker may know F. The bank then asks questions that this particular customer has answered in the desired way. From the attacker's point of view, he is trying to guess a sequence of independent random variables, distributed according to F, so his optimal strategy is to give the most likely response (according to F) every time. The fact that he can observe which questions are being asked does not help him. (This remark is not entirely trivial.) The probability that the attack succeeds is just f_1^M where f_1 is the largest probability in F.

This scheme cannot be applied to protect a customer who has not given a sufficiently diverse set of responses.

While this scheme does provide the customer with the desired measure of assurance, a customer who is successfully attacked may feel cheated because although he answered many questions with responses other than the most popular one, on this occasion the (incompetent) bank asked only questions that were easy to guess. To avoid this outcome, let us change the bank's strategy. The bank can work out how many realizations of each possible response can be expected in M questions; it constrains the sequence of desired responses so that it is a random permutation of these numbers of each response. The bank announces this information, so an attacker who has been paying attention will now choose his guesses in the same way: as a random

permutation of these numbers of each type of response. The probability that the attacker succeeds is now the reciprocal of a multinomial coefficient. Any other attack has a smaller chance of succeeding. The bank can therefore design the scheme to provide any desired degree of assurance.

6 Applications

So far we have described our techniques in straightforward contexts. In this section, we will see how the methods can be applied in more substantial domains.

6.1 Names

Many humans are able to remember the names of childhood friends, and that information is often difficult for attackers to learn (at least it was before membership in youth clubs was posted on the Web). How much assurance is contained in a name such as Mary or Ardelia or Smith or Aalderink? We will study data from the U.S. Census Bureau's web site at www.census.gov/genealogy/www/freqnames.html. A file of 88,799 last (family) names accounts for 90% of the sample population and begins with these three lines:

```
SMITH          1.006  1.006      1
JOHNSON        0.810  1.816      2
WILLIAMS       0.699  2.515      3
```

The names appear in decreasing order by frequency. The second line says that the name Johnson accounts for 0.810 percent of the sample population, that the names so far in the file account for 1.816 percent, and that Johnson is the second most frequent name in the sample. The min-entropy is –log 0.01006 = 6.635 bits for Smith. The Shannon entropy of that file is 9.969 bits, which is the weighted average assurance (over all customers) against a single-peek attacker.

We wrote a linear-time program to groom the set of names, and found that the grooming point that maximizes total assurance bits was at Adkins, the 394th name in the file, which accounts for 0.029 percent of the names. All names are therefore assigned log 394 = 8.622 bits of assurance. The 394 names account for 32.478% of all names, but we must exclude some fraction of those (that occur above the threshold of 0.029%). For instance, we exclude the unlucky 0.670% of the population that had the last name of Williams and was chosen to be excluded by grooming; we also exclude half of the people named Dunn (the 160th most common name, with 0.058% of the sample). Altogether, grooming excludes 21.448% of the population, but assigns 8.622 bits of assurance to the remaining 78.552% of the population. That gives an average of 6.807 bits per (original) name.

The probability of success of any attacker against a randomly chosen name is at most 1/394. Still, the few Smiths lucky enough to survive grooming might feel that they are particularly vulnerable against an attacker who is knowledgeable about the distribution of last names yet ignorant of the fact of grooming. The Bank could soothe those fears by announcing to all potential Attackers that the data had been groomed.

We applied similar analyses to files of male and female first names from the same web site. (Since the popularity of first names changes quickly over the years, a clever attacker should take birth date into account, and use names popular at the time.) The 1219 male names accounted for 90% of the population, with James the most popular at 3.318%. The optimal grooming point was at the 77th name (Aaron, which accounted for 0.240% of all names). This assigned log 77 = 6.267 bits of assurance to the non-excluded 63.77% of the population, for an average of 3.997 bits per (original) name. The 4275 female names accounted for 90% of the population, with Mary the most popular at 2.629%. The optimal grooming point was at the 112th name (Rosa, which accounted for 0.194% of all names). This assigned log 112 = 6.807 bits of assurance to the non-excluded 76.24% of the population, for an average of 5.190 bits per (original) name. Our experiments are summarized in this table.

	Total Names	*Shannon Entropy*	*Min-Entropy*	*Grooming Point*	*Bits Per Name*	*Names Excluded*	*Bits Per Original Name*
Last	88,799	9.969	6.635	394	8.622	21.45%	6.807
Female	4275	8.591	5.249	112	6.807	23.76%	5.190
Male	1219	7.386	4.914	77	6.267	36.23%	3.997

6.2 Grooming PINs

Grooming is straightforward to apply to multiple-choice questions in which there are a handful of options. Grooming is also useful for four-digit PINs. Suppose, for instance, that of the 10,000 possible four-digit PINs, 8000 were rarely chosen, while 2000 were chosen frequently. We would choose a grooming point of 2000 in the descending-frequency list of PINs, and therefore conservatively assign just log 2000 (or about 11) bits of assurance to each PIN. We would still ask about the 8000 PINs in the Tail, but we would not use them to compute assurance.

Let us further suppose that of the 2000 frequent PINs, most were chosen relatively uniformly, while a popular few were chosen often (such as 1111, 1234, 9876 and the like). We might coerce the users to change them those particularly common PINs, or insist on some additional information at authentication. Alternatively, we could assign less assurance, using formula (1) from Section 4.2. The small assurance might be enough to inquire about account balance, for example, but not enough to drain money from an account.

6.3 How Much Assurance Does a CAPTCHA Provide?

In a Completely Automated Public Test to tell Computers and Humans Apart (CAPTCHA), the Bank wishes to distinguish a human Customer from a robotic At-

tacker. Rather than providing each human a distinct secret password, the bank gives a test that is easy for humans yet hard for computers. A typical test is to read a sequence of distorted letters. The test should provide little inconvenience to a human, but enough difficulty so that an attacker usually fails the test. Assurance is an appropriate measure of that difficulty.

How well does a blind Attacker do against a CAPTCHA of visual letters? If the challenge text is chosen uniformly from a dictionary of 1024 common words, it carries 10 bits of assurance (the randomly guessing Attacker has probability 1/1024 of success). If 8 characters are chosen uniformly from a set of 32 characters (start with 26 upper-case letters and 10 digits, and throw out near misses like "I" and "1" and "2" and "Z"), then the text will have 8´5 = 40 bits of assurance. When Chew and Baird [2003] generate a pronounceable 8-character challenge from an order-3 Markov chain, how many bits of assurance does it carry? While the exact probability of the process selecting the given challenge is straightforward to compute, we conjecture that the inherent nonuniformity will make the probability of a clever attacker succeeding in guessing very hard to determine. We also conjecture that a "groomed" Markov chain would be easy to analyze yet still yield challenges that "look like" English text.

Smart attackers of CAPTCHAs use Optical Character Recognition systems that are far from blind. How do we analyze their probability of success? We conjecture that a misrecognition matrix showing how often each character is mistaken for another ("C" might be mistaken for "O" more frequently than it is for "X") will be key for such analysis.

6.4 Lotteries

Consider a lottery where patrons choose a key (from some known finite set) and winners are decided by a random draw (perhaps on TV using a physical randomizing mechanism). This is an instance of a special case of our problem where the Attackers are the patrons, and the secret password is the randomly selected winning key. It differs from our general case in that there is no analog of the particular Customer, who owns an individual password and has an interest in its security. For such a uniform lottery, both the bank and the patrons can agree that the security of the key is properly measured by the size of the key-space (or the logarithm of this).

Now suppose that the randomizing mechanism does not produce keys uniformly. (This happened in the early days of the US draft lottery.) We assume that successive draws are independent, but not uniformly distributed. If the true distribution is not known by the patrons, they will still measure security by log N, and will have no reason to prefer one key over another. If they do know the distribution, however, or if they get an estimate of it by accumulating data over time, then they will need to reassess. The system becomes all the more interesting if the winnings are divided equally among fellow successful attackers, so attackers benefit from selecting an answer that is guessed by few other attackers. Becker, Chambers and Wilks [1988, Section 1.2] describe how this happened early in the New Jersey Pick-It Lottery.

7 Conclusions

Dictionaries define assurance as a statement that inspires confidence. The goal of this paper is to quantify the amount of assurance contained in a particular message. In Human Interactive Proofs, that message might be the answer to a multiple-choice question, a PIN, the text in a visual CAPTCHA, or a password.

Section 2 phrases the problem in terms of PINs, and gives anecdotal evidence to show that human secrets tend to be nonrandom. Section 3 shows that neither the classical Shannon entropy nor other more recent entropic proposals completely captures assurance.

Section 4 surveys ways of quantifying assurance from various viewpoints. Many of the modifications of entropy are in fact relevant in various contexts. For nonuniform distributions, though, the assurance varies greatly with assumptions regarding the knowledge of the attacker and regarding "who knows what when". Section 5 therefore proposes methods to "groom" nonuniform distributions to induce uniformity. By guaranteeing uniformity we ensure that we can employ the straightforward measure of max-entropy. Section 6 shows how the methods can be applied to problems in and beyond human authentication.

This paper has taken steps towards a theory of assurance. Our long-term goal, though, is assurance engineering, which would allow authentication engineers to quantify the assurance of various schemes. In designing a CAPTCHA, for instance, we might want to know how much assurance is contained in 10-characters of order-4 Markov text versus 6 characters of order-1 Markov text. We could use such numbers together with analyses of readability, pronouncability, familiarity and so forth to achieve a provably good design. We consider the following problems particularly ripe for further work.

Improvements to Grooming. Section 5 described straightforward grooming algorithms, and Section 6 showed that they are fairly efficient in some applications. We conjecture that more advanced grooming algorithms might be even more efficient.

Authentication Contexts. We blithely assumed that an attacker is allowed a single guess to produce an absolutely correct answer. Many systems give a user a few tries to allow for mistyping or misremembering before taking draconian measures (such as seizing an ATM card). We conjecture that in many contexts, allowing success on the K^{th} guess increases the probability of success by a factor of at most K, so $\log K$ bits should be subtracted from assurance thus attained. We suspect that similarly simple expressions could be found to quantify the assurance of schemes that allow "near" answers, such as one wrong multiple-choice question or missing a character or two in a visual CAPTCHA.

Real Distributions. Section 6.1 applied grooming to English names. People also tend to remember a variety of other "personal facts" that could be used for authentication, such as telephone numbers, street names and street numbers, postal codes and the like. It would be useful to collect and to analyze such distributions. Dates are particularly interesting. Barring insider information, it seems that the response to the question "my sister's birthday" might have about $\log 365.25$ bits of assurance, even if the attacker could guess the year. The response to "James and Mary's wedding anniversary" might carry far fewer bits, because of cultural propensities towards "June

brides" and weekend weddings (assuming that an attacker could guess the year in question).

Structure of Passwords. Humans tend to compose passwords in predictable fashions. A user might combine a name, a punctuation mark, and a month into the password "ophelia-april". A dog fancier might memorize the personally significant phrase "the best dogs are my 3 Cocker Spaniels at home" and from it derive the password "tbdam3CS@h". It would be interesting to characterize the assurance in such well-defined password schemes.

Acknowledgments

The authors are grateful for the helpful comments of Henry Baird, Dan Bentley, Joe Hall, David Jefferson and several anonymous referees.

References

Becker, R. A., J. M. Chambers, A. R. Wilks [1988]. *The New S Language: A Programming Environment for Data Analysis and Graphics*, Wadsworth & Brooks/Cole, Pacific Grove, CA.

Bentley, J. L. and C. L. Mallows [2005]. Problem submitted to *American Mathematical Monthly*.

Bishop, M. and D. V. Klein [1995]. "Improving system security via proactive password checking," *Computers and Security 14*, 3, 1995, pp. 233-249.

Cachin, C. [1997]. "Entropy measures and unconditional security in cryptography," Ph.D. Thesis, ETH Zurich.

Chew, M. and H. S. Baird [2003]. "BaffleText: a Human Interactive Proof," *Proceedings IS&T/SPI Document Recognition and Retrieval X Conference (ER&R 2003)*, Santa Clara, CA, January 2003.

Ellison, C., C. Hall, R. Milbert and B. Schneier [2000]. "Protecting secret keys with personal entropy," *Future Generation Computer Systems 16*, 4, February 2000, pp. 311-318.

Feldmeier, D. C. and P.R. Karn [1990]. "UNIX password security – ten years later," *Advances in Cryptology – CRYPTO '89 Proceedings*, Springer-Verlag, 1990, pp. 44-63.

Knuth, D. E. [1973]. *The Art of Computer Programming, Volume 3: Sorting and Searching*, Addison-Wesley, Reading, MA.

Mauer, U. M. [1996]. "A unified and generalized treatment of authentication theory," *Proc. 13th Symposium on Theoretical Aspects of Computer Science (STACS '96),* Springer-Verlag LCNS 1046, pp. 387-398.

Morris, R. and K. Thompson [1979]. "Password security: A case history," *Comm. ACM 22*,.11, Nov. 1979, pp. 594-597.

O'Gorman, L., A. Bagga, and J. Bentley [2004]. "Call center customer verification by query-directed passwords," *8th Int. Financial Cryptography Conference*, Florida, 9-12 Feb. 2004.

Shannon, C. E. [1948]. "A mathematical theory of communication," *Bell System Tech. J. 27*, July, October, 1948, pp.379-423,623-656, http://cm.bell-labs.com/cm/ms/what/shannonday/paper.html.

Shannon, C. E. [1949]. "Communication theory of secrecy systems," *Bell System Tech. J. 28*, October 1949, pp. 656-715.

Smith, R. E. [2002]. *Authentication – From Passwords to Public Keys*, Addison-Wesley, Boston, 2002, pp. 87-99.

Wischik, L [1996]. "The Paradox of the Surprise Examination", http://www.wischik.com/lu/philosophy/surprise-exam.html.

Yan, J., A. Blackwell, R. Anderson, and A. Grant [2000]. "The memorability and security of passwords – some empirical results," TR 500, University of Cambridge, Computer Laboratory, September 2000, http://www.cl.cam.ac.uk/TechReports/UCAM-CL-TR-500.pdf