CAPTCHA challenge strings: problems and improvements

Jon Bentley[a], Colin Mallows[a]
[a]Avaya Labs, 233 Mt. Airy Road, Basking Ridge, NJ 07920, USA

ABSTRACT

A CAPTCHA is a Completely Automated Public Test to tell Computers and Humans Apart. Typical CAPTCHAs present a challenge string consisting of a visually distorted sequence of letters and perhaps numbers, which in theory only a human can read. Attackers of CAPTCHAs have two primary points of leverage: Optical Character Recognition (OCR) can identify some characters, while nonuniform probabilities make other characters relatively easy to guess. This paper uses a mathematical theory of assurance to characterize the probability that a correct answer to a CAPTCHA is not just a lucky guess. We examine the three most common types of challenge strings, dictionary words, Markov text, and random strings, and find substantial weaknesses in each. We therefore propose improvements to Markov text, and new challenges based on the consonant-vowel-consonant (CVC) trigrams of psychology. Theory and experiment together quantify problems in current challenges and the improvements offered by modifications.

Keywords: CAPTCHA, challenge string, assurance, Markov text, Consonant-Vowel-Consonant (CVC) trigrams

1.INTRODUCTION

In a Human Interactive Proof, a human being reveals a secret as evidence of membership in a class. A password, for instance, is evidence that the human belongs to the singleton class that consists of just that user. In a CAPTCHA, correctly identifying the challenge string is evidence that the subject is a human, because the text is presumably degraded beyond the capacity of OCR. Intuitively, we feel that some secrets carry more assurance than others: If a CAPTCHA system chooses the challenge word "cat" from a small dictionary, and the subject correctly identifies it, then we have some assurance that the subject is a human who can read the phrase, and not a program that has made a lucky guess. If the subject correctly identifies the challenge string "2R97WZ8E" in which each of 8 characters is chosen at random, then we have substantially more assurance.

We have undertaken to develop a mathematical theory of assurance to make precise that intuition. Bentley and Mallows [2005] describe the status of the work as of early 2005. At Baird and Lopresti's [2005] Workshop on Human Interactive Proofs, we were challenged to apply our theory to CAPTCHA strings. We have done so, and a lengthy preliminary description of that work is available as Bentley and Mallows [2006]. That paper develops mathematical techniques to characterize the assurance of CAPTCHA challenge strings, and uses that theory to critique existing challenges and to guide the development of new challenges. Baird, Moll and Wang [2005] have chosen to use some of our new challenges in a major CAPTCHA experiment.

This paper summarizes the less mathematical part of that work. Section 2 of this paper briefly outlines "just enough" of the mathematical theory of assurance. Section 3 surveys existing CAPTCHA challenge strings, and shows serious flaws in each. Section 4 proposes new challenges that increase assurance. Conclusions are offered in Section 5.

2.A SKETCH OF ASSURANCE

Suppose that we generate a CAPTCHA challenge string of length 8 by tossing a fair coin 8 times. Each time the coin comes up heads, we record a 0, and each time the coin comes up tails, we record a 1. Thus the challenge string might consist of the 8 bits 01101011. We will say that a challenge string made in that fashion provides 8 *bits of assurance* against a blind attacker, that is, a guessing attacker who does not exploit any OCR. If a challenge consists of 8 uniformly selected quaternary digits (0, 1, 2 and 3), it provides 16 bits of assurance against a blind attacker.

If a certain class of attacker has probability p of successfully guessing a challenge string, then we say that that string provides $-\lg p$ bits of assurance against that attacker (we use "lg" to denote "log$_2$"). Because a blind guesser can correctly guess a string of B uniformly selected bits with probability 2^{-B}, such a string provides $-\lg 2^{-B} = B$ bits of assurance. If a challenge string consists of a single lower case letter selected uniformly, the probability of success of a guessing attacker is $1/26 \sim 0.0385$, so that single letter provides $-\lg 1/26 = \lg 26 \sim 4.70$ bits of assurance against that

guesser. Consider next a sequence of 5 such letters, each selected independently and uniformly. A similar analysis shows that the probability of success by guessing is $1/26^5 \sim 0.0000000842$. It is much easier (and more intuitive) to multiply the assurance of one uniform letter by the number of letters, showing that the five letters together give $5 \times \lg 26 \sim 23.50$ bits of assurance.

If a challenge is selected uniformly from a set, the number of bits of assurance that that challenge provides against a blind attacker is the (minimal) number of bits required to encode the challenge. We believe that computer programmers will find this a natural way to think about assurance. Subtlety arises when challenges are selected nonuniformly. Unfortunately, neither the Information Theory of Shannon [1948] nor the Authentication Theory of Maurer [1996] addresses the requirements of assurance. Although we have concentrated here on blind attackers, we will soon consider attackers that employ imperfect OCR. Further details regarding assurance can be found in Bentley and Mallows [2005, 2006].

3.PROBLEMS WITH CURRENT CHALLENGES

The vast majority of the textual CAPTCHAs in Baird and Lopresti [2005] were taken from three sources: dictionary words, random strings, and Markov text. We examine those sources in the next three subsections.

3.1. Dictionary words

Our first experiments were on the EZ-Gimpy dictionary of 561 English words; that dictionary is available online at openanonymity.sourceforge.net/HTML/scripts/ez-gimpy/dictionary. Each word contains 4, 5 or 6 characters. If a CAPTCHA selects a word uniformly, a blind attacker has probability of success by guessing of $1/561 \sim 0.00178$, for 9.13 bits of assurance. That is small, but sufficient for some purposes. Furthermore, familiar words such as "with", "point" and "record" are comfortable for most human users.

CAPTCHA researchers have long felt that dictionary words are vulnerable to partial OCR. We have derived a theorem that characterizes the *a priori* probability of success of an attacker armed with specified information about a word, and a program to determine that probability that is optimal in both space and time. We have used the program to build tables that show bits of assurance under varying assumptions. For instance, knowing only the number of characters (which might be provided by character segmentation without attempting recognition) reduces the number of bits of assurance from 9.13 to 7.55. Knowing only the first character of a word reduces the number of bits of assurance to 4.55. Knowing the length of the word, the first character and the last character reduces the number of bits of assurance to 0.76, so a guessing attacker with that information can succeed with probability 0.59.

We also studied a larger dictionary of 61,580 English words between 5 and 12 characters in length. The average number of characters per word is 8.37, and the number of bits of assurance against a blind attacker is 15.91. This corresponds to about 1.90 bits per character. If an attacker knows the first letter, the assurance drops to 11.21 bits. Knowing the four hints of first character, middle character, last character and number of characters drops the bits of assurance to 1.33, and a guessing attacker armed with that information has a probability of roughly 0.40 of success.

These measurements (and tables of others like them) quantify the intuition in the field that dictionaries are highly vulnerable to guessing attacks in the presence of partial OCR.

3.2. Random strings

We next consider challenge strings composed of random letters and numbers. We might start by considering the 26 upper case letters and 10 digits, for a total of 36 characters. But because the letter O and the digit 0 are easily confused, as are the letter I and the digit 1, we will omit those four characters to leave 24 letters and 8 digits, for a total of 32 characters. Here are 7-character challenges randomly chosen in this fashion:

348UVLP 6F4ZRGD BPLDPVV 2R97WZ8 SHK3AJ7 MP589ET

Dictionary words are easy for human subjects to recognize and to remember (for hunt-and-peck typists who look away from the CAPTCHA to the keyboard as they type the word), but provide little assurance. Random strings are their dual: they provide much assurance, but are difficult to recognize and to remember.

Uniformly selecting one of 32 characters provides $\lg 32 = 5$ bits of assurance against a blind attacker. In general, N such characters provide $5N$ bits of assurance.

But what if the attacker is not blind, but rather uses imperfect OCR? If an attacker has the hint of reading K out of N of the characters, then we are left with $5(N–K)$ bits of assurance. So if the OCR guesses 6 of 8 characters correctly, the $2 \times 5 = 10$ remaining bits of assurance means that an attacker can succeed with probability of $2^{10} = 1/1024$.

Assuming that an attacker can read precisely 6 out of 8 characters is simple to analyze but not particularly realistic. We will therefore assume the simple model that imperfect OCR recognizes each character with a fixed probability p. Chellapilla, Larson, Simard and Czerwinski [2005, Section 3.1] use a similar model to characterize their (very effective) attacks on commercial CAPTCHAs. We will assume that when the OCR identifies a character correctly, it knows that it has done so. When it cannot identify the character, it fails with no partial knowledge about its value.

Let's suppose that we wish to achieve 20 bits of assurance against an attacker with OCR capable of recognizing half the letters, on the average. Since 4 letters suffice to achieve 20 bits of assurance against a blind attacker, we might (erroneously) assume that 8 characters would suffice against an attacker who can read each letter with probability 1/2. But since each character can be read with probability 1/2, each character provides only about 1 bit of assurance (because an attacker can guess unread letters, each character in fact provides $-\lg 33/64 \sim 0.96$ bits of assurance). We therefore require 21 such characters to achieve 20 bits of assurance!

If OCR can read a character with probability p, that character provides at most $-\lg p$ bits of assurance. If OCR succeeds with probability 1/4, for instance, then a character provides at most two bits of assurance. So even though short random strings provide much assurance against blind attackers, one needs long strings to defend against attackers with weak OCR. This observation justifies our examination of long challenges.

3.3. Markov text

Chew and Baird [2003] suggest using Markov text as CAPTCHA challenges. Markov text is described by Shannon [1948, Sections 2-4]. Our experiments use the King James Bible as training text. We chose that text because it is widely available, has a vocabulary and style familiar to many readers of English, yet has not (as far as we know) been used to generate challenges for any commercial CAPTCHAs; we do not wish to assist attackers. We stripped book, chapter and verse markings from the text, removed all non-alphabetic characters (including spaces), and converted upper case letters to lower case.

The algorithm starts with an arbitrary letter pair, or digram. At each state, it chooses the next character according to the probabilities derived from the input text. That new character is printed, and becomes the second letter in the new digram. Every specified number of letters, we print a newline to form challenge words of a fixed length. Here are some sample 7-letter challenge strings produced by that process:

famingi tseywif erandav erecent opletih atobeir makefor ibeasst ohoiare

These are not English words, and therefore do not succumb to a dictionary attack. Yet they are vaguely familiar and often even pronounceable.

We have estimated how vulnerable Markov text is to a global blind attacker who guesses a string using various strategies. Markov strings provide 4.70 bits of assurance per character against an attacker who uniformly guesses one of 26 letters. We believe that many CAPTCHA researchers have therefore implicitly assumed that Markov strings of length 7 give about 33 bits of assurance, which might be sufficient for many applications. However, Markov text provides only 3.86 bits against an attacker who guesses letters with the appropriate frequencies, and 2.97 bits against an attacker who guesses common letters. An engineering approximation and experiment together indicate that the average character gives about 2.07 bits of assurance against an attacker who guesses a string produced by the same Markov model.

Some Markov strings occur much more frequently than others, and are therefore vulnerable to more clever blind attackers. The most common string among 100,000 random 5-character challenges was "andth" (due in large part to the common word pair "and the"); it occurred 341 times. That observation reduces the number of bits of assurance per character from 2.07 to just 1.64. A seven-character string therefore provides only about 11.48 bits of assurance against such an attacker. An alternative attack starts with the most common digram ("th"), and at each stage always takes the most common next step in the Markov chain; this attack yields similar assurance.

We considered in Section 3.1 a "contextual attack" in which partial OCR reveals to the attacker certain characters of a dictionary word. If an attacker can read the first two letters of a Markov trigram, then he is able to guess the next letter with average probability about 0.38, which corresponds to just 1.41 bits of assurance.

4.IMPROVEMENTS TO CHALLENGES

Dictionary words are vulnerable to attackers with partial OCR; they should not be used in contexts that require much assurance. Short random strings provide much assurance against blind attackers, but strings must be kept long to defend against lucky OCR, and long random strings confuse users. Each character of Markov text provides only about 1.64 bits of assurance against clever attackers. What should a CAPTCHA designer do?

Section 4.1 shows how Markov text from a corpus can be improved, and Section 4.2 considers Markov text generated from other training sources. Section 4.3 then examines the CVC trigrams of experimental psychology.

4.1. Grooming Markov text

Bentley and Mallows [2005] introduce a technique called "grooming" to induce uniformity into nonuniform phenomena. We can use that idea to remove the biggest targets from the sights of attackers. We will call this "common-string grooming". A sample of 100,000 random 5-character Markov strings yielded 49,947 distinct words. Among those, 32,910 occur just a single time. We could therefore ignore all duplicated strings, and present as challenges such as

yheri warem usshy tieda snate paria ommai gerth eciph ceint

This scheme might combine the best of both worlds: text that is vaguely recognizable yet still hard to guess. While five uniform lower-case letters yield $5 \times \lg 26 \sim 23.5$ bits of assurance, this scheme would yield $\lg 32{,}910 \sim 15.0$ bits of assurance (roughly 3 bits per character) but with increased readability. We also studied one million Markov strings of length 7. Grooming that set leaves 624,284 unique strings. Since $\lg 624{,}284 \sim 19.25$, that yields roughly 2.75 bits of assurance per character.

Our next approach is "limited successor grooming". This method yields 3 bits of assurance per character against an attacker with OCR that has identified the two preceding characters. Among the letters following each digram, consider only the 8 most common, and choose one of those uniformly. That uniform choice from 8 characters gives $\lg 8 = 3$ bits of assurance. Similarly, choosing one of the 4 most common characters gives 2 bits of assurance per character. One might fear that while regular Markov text tends to look like English, groomed Markov text might not. For instance, grooming to 4 characters might look unnatural by cutting off later characters. Then again, grooming to 8 characters might look unnatural by giving equal weight to later characters. The following 7-letter challenges were generated either as "pure" Markov text, or with grooming to 4 or 8 characters; can you identify which is which?

hecketu heretto hantara lthimst ediwoes areopro
andundd thoushe thatedt tliedan onganci fivilie
ontohil rtohins omentot butpunc phribut vadotti

The first and fourth columns are ungroomed, the second and fifth columns are groomed to 4 characters, and the third and sixth columns are groomed to 8 characters. We do not note any differences in "feel" among the three methods. In generating 10,000 characters groomed to 4, we always generated 2 bits of assurance per character. If a digram does not have 8 following characters, we chose from among fewer successors. That process generated an average of 2.93 bits of assurance per character.

4.2. Markov text from dictionaries

Our previous experiments on Markov text follow Chew and Baird [2003] in using a body of "real" English text as a training sample. Chew and Baird trained on the million-word Brown Corpus, while we used the (roughly) 800,000-word King James Bible. The text generated from the resulting models therefore has many properties of written English, including its letter frequencies and digram and trigram frequencies. Unfortunately, we saw that such text tends to contain common words and even word pairs (such as "andthe"), and therefore offers enticing targets for clever guessers.

This section describes an alternate version of Markov text, in which we use a dictionary as the training text. Our hope was that a dictionary would yield nonsense words that "looked like" English, while being more statistically diverse. In particular, our training text was the larger of the two dictionaries mentioned in Section 3.1, which contains 61,580 English words. In the rest of this section, we will describe how the resulting "dictionary Markov text" compares to the "Biblical Markov text".

Experiment and approximation together reveal that Biblical Markov text gives 2.07 bits of assurance per character against a guesser generating such text. Similar analyses of dictionary Markov text give 2.53 bits of assurance per character. This fact was encouraging: each character of dictionary Markov text potentially offers about 22% more assurance than Biblical Markov text!

The length of generated words varied widely, so we chose to focus on words of length 8. When our program generated 100,000 such words, we found that each character had an average 2.49 bits of assurance against a guesser generating similar random text (or just slightly less than the 2.53 bits of the original text, which included the extremely long words). Here is a random sample of those 8-character words:

palimard reembous liancobs diroting explipar defatine rathwass

Among the 100,000 words, the most frequent was "comation", which appeared 9 times; 95,917 of the words were unique. This suggests that these strings are not highly susceptible to blind global attacks. The fact that almost 96% of the generated words were unique indicates that 8-letter dictionary Markov challenges are particularly appropriate for the

"common-string grooming" described in Section 4.1. Because so few words are repeated, an efficient implementation of grooming might keep a dictionary of repeated words, which are then discarded if they happen to be randomly generated. One can also apply the "limited successor grooming" of that section to dictionary Markov text; here are challenges of length 8 generated by grooming to 8 successors:

pansonet andobhon celfdoze centhous sumencry subrouse purocele

Instead of a dictionary of words, one might train on a file of names. Here are 8-character names produced by training on three files of common names from the U.S. Census Bureau (at www.census.gov/genealogy/names/). Can you guess whether a particular name was produced by training on the female first names, male first names, or last names?

Cargilyn Ryjoline Vicorlee Chelinne Chiredry Micharia Ferrilla
Jakelmed Lemannie Jefferne Judichan Javerick Broylory Samellas
Letherpe Bondreno Debraund Siesimes Wajlalin Polninel Fingengs

The first row represents female first names, the second row represents male first names, and the third row represents last names. Experiments indicate that the (ungroomed) 8-character last names provide about 2.67 bits of assurance per character. Grooming further increases that assurance.

4.3. Consonant-vowel-consonant (CVC) trigrams

Dictionary words are easy to read and to remember but are also easy to guess; random strings are hard to guess but are also hard to read and to remember. Markov text provides one tradeoff between those two extremes. In this section, we will investigate another intermediate point on that spectrum: "consonant-vowel-consonant" or "CVC" syllables. Because they contain three letters, we will sometimes refer to them as "CVC trigrams".

Ebbinghaus [1885] introduced CVC trigrams in his pioneering study of human memory that commenced in 1879; he referred to them as "nonsense syllables". In their simplest English form, we classify the 26 letters of the Roman alphabet as the five vowels a, e, i, o and u, while the remaining 21 letters are consonants. We then generate a trigram as a consonant-vowel-consonant or CVC trigram, such as het, som, zih, qox and cat. We assume that a CAPTCHA challenge would consist of either two or three CVC triples, such as hetsom or zihqoxcat.

One can be quite elaborate in forming CVC trigrams. Ebbinghaus [1885, Chapter III, Section 11, Footnote 1] writes:

> The vowel sounds employed were a, e, i, o, u, ä, ö, ü, au, ei, eu. For the beginning of the syllables the following consonants were employed: b, d, f g, h, j, k, l, m, n, p, r, s, (= sz), t, w and in addition ch, sch, soft s, and the French j (19 altogether); for the end of the syllables f, k, l, m, n, p, r, s, (= sz) t, ch, sch (11 altogether). For the final sound fewer consonants were employed than for the initial sound, because a German tongue even after several years practise in foreign languages does not quite accustom itself to the correct pronunciation of the mediae [b, d, g] at the end. For the same reason I refrained from the use of other foreign sounds although I tried at first to use them for the sake of enriching the material.

He discarded three-letter German words to avoid mnemonic associations that do not clutter true nonsense syllables; he was left with about 2300 syllables.

As a brief step in the direction of well-considered CVC trigrams, we evaluated CVC trigrams assembled from the following components:

Initial Consonant: b c d f g h j k l m n p r s t v w y z
Vowel: a e i o u
Final Consonant: b c d f g j k l m n p r s t v x z

We employed the five standard vowels, but excluded some consonants using the "looks funny" test. As initial consonants, we excluded q and x; q looks strange without a following u and vowel, and x looks strange at the start of a syllable (we apologize for our xenophobia). We similarly ignored h, q, w and y as final consonants; q still looks strange without the following u, and ih, iw and iy are all rare digrams. We believe that this simple policy results in plausible CVC syllables. It yields pairs such as

sixbeg pembus letsuv degyov dedvox nefbak kojnud zunzib tafbis

and triples such as

mabjertot jajlabcav lidfaggas mazhilcoz nubnabcux pifgoxkam lirbagjus

Each of those sequences was produced by a program, and selected without human editing.

We assume that CVCs used as CAPTCHA challenges would be employed as pairs or triples. How much assurance is in such a challenge against a blind attacker? A uniform selection from 19 initial consonants provides 4.25 bits of assurance, 5 vowels provide 2.32 bits, and 17 final consonants provide 4.09 bits, for a total of 10.66 bits per CVC trigram, and an average of 3.55 bits per character. A CVC pair thus provides 21.31 bits, and a CVC triple provides

31.97 bits of assurance. Thus a CVC triple of 9 letters provides slightly more assurance than 6 characters of the random strings chosen from a 32-symbol alphabet that we saw in Section 3.2:

3fremr 7j6hwu ldsu5b dwazdr jx62z7 7n62lg 9fys7y dgtycs gjwqbq

We conjecture that even though the CVC triples are 50% longer, the average user will find the CVC triples easier to read, easier to remember (for non-touch typists), and more "friendly".

How do CVC challenges fare against imperfect OCR? If an attacker knows that a string is generated in this fashion, and recognizes some of the characters, we can still add the assurance left in the unrecognized characters, which is 3.55 bits on the average. Since the CVC letters were generated uniformly and independently, knowledge of the surrounding characters does not aid an attacker.

Some people find CVC doubles and triples to be awkward impersonators of English words. A friend (who prefers to preserve his anonymity) once generated random pronounceable passwords by combining 4 CV pairs based on Japanese syllables, like "yokohama". With the same (non-Japanese) initial consonants and vowels as described earlier, his scheme generates words like:

pahimotu cusefutu vicebagi hevurero huzedara duviyihe vufelato tosiruka

As before, a selection from 19 consonants provides 4.25 bits of assurance against a blind attacker, and a selection of 5 vowels provides 2.32 bits, for a total of 6.57 bits for the digram, and an average of 3.28 bits per character, a slight decrease from the 3.55 bits per CVC character.

5.CONCLUSIONS

This paper makes two primary contributions to CAPTCHA engineering. The first is applying the framework of assurance to the area, and thereby quantifying an important property of any CAPTCHA: how much assurance does the challenge string provide against various attackers? Baird and Bentley [2005] apply a preliminary version of this theory to "Implicit CAPTCHAs" in which the user clicks on a navigation aid, potentially without even realizing that he has thereby passed a test. They make statements of the form that a simple test "provides 5 bits of assurance against a blind attacker, or 3 bits of assurance against an attacker that can group words". Rui, Lie, Kallin, Janke and Paya [2005, p. 54] report that a facial recognition CAPTCHA based on many mouse clicks "is robust at a rate of 2 out of a million". We would hope that they might now rephrase that as "it provides 19 bits of assurance" against a specified class of attacker. We hope that this theory of assurance can broadly applied to CAPTCHAs and in the larger field of Human Interactive Proofs.

The second contribution of this paper concerns particular classes of challenge strings. We believe that this paper represents the first time that the following difficulties have been quantified:

Dictionary Words. Knowing a few hints reduces the assurance to at most a few bits.

Random Strings. While short random strings provide much assurance (5 bits per character) against blind attackers, they provide little assurance against even weak OCR (1 bit per character if the success rate is 50%). Long random strings provide adequate assurance, but are awkward for humans.

Markov Text. Typical Markov text starts by providing only 2.07 bits per character against an attacker who generates text from the same model, and that is reduced to 1.64 bits per character against more clever attackers.

Fortunately, we have also suggested ways in which challenge strings can be improved:

Markov Text. Training on a dictionary rather than on a text corpus has several benefits: the initial assessment of assurance increases by 22%, and the challenge strings feel more wordlike. The assurance of any Markov text can be increased by two kinds of grooming: removing common strings and uniformly choosing a successor from a limited set.

CVC Trigrams. This old psychological tool yields strings that are pronounceable, memorable and have an average of 3.55 bits of assurance per character, or about 70% that of random strings.

Baird, Moll and Wang [2005] have chosen to include both groomed dictionary Markov challenges and CVC doubles in their substantial CAPTCHA experiment.

Many problems remain open. Perhaps the first problem on that list is measuring psychological issues of the challenge strings. We conjecture that rigorous studies will support our intuition that the string "diroting" is superior to the string "dgtycsoe" in dimensions such as readability, memorability and user-friendliness. CAPTCHA engineers should aspire to an "applied psychology" along the lines described by Card, Moran and Newell [1983, Section 1.3].

Three-letter CVC syllables are among the simplest models of random English text. One could use a more complex model involving consonant sequences and vowel pairs to generate random syllables such as "splaint". Starting with such a base, one could use affix analysis to generate random challenge words such as "desplaintify". Dan Bentley of Google has suggested increasing the effective size of a dictionary by intentionally introducing misspellings, so the word

"suggest" might render challenge strings such as "soggest", "sugest", "sojast", and many others. Much of the research on spelling correction might be inverted to transform a "spell checker" into a "mispeling genarater". Simple CVC syllables only scratch the surface of word models beyond Markov text.

With the exception of uniform alphanumeric challenges and the German CVC syllables of Ebbinghaus, most of the work that we have described applies predominantly to English. How are these results related to the haphazard nature of English spelling? French spelling is much more regular; it is often said that if a native speaker of French can pronounce a word, then he can spell it. What are the implications of that for Markov text or nonsense syllables in French? How robust is a French dictionary to an attack with partial OCR? Apart from issues of transforming other alphabets, how should CAPTCHA challenges for various languages be implemented in the Latin alphabet? Random *Romaji* syllables for Japanese users should consist of consonant-vowel pairs from a well-known table of about 40 syllables or *kana* (different from the CV pairs at the end of Section 4.3), perhaps with a final n. Random words for Arabic and Hebrew speakers might exploit the triconsonantal structure of those Semitic languages. Analyzing the assurance of internationalized CAPTCHA challenges is an interesting open problem.

6.ACKNOWLEDGMENTS

The authors are grateful for the helpful feedback of many of the participants at the HIP 2005 Workshop, especially Henry Baird, Andrei Broder, Kumar Chellapilla, Dan Lopresti, and Patrice Simard. For helpful comments on this paper, we are grateful to Henry Baird, Dan Bentley and Brian Kernighan.

REFERENCES

Baird, H. S., M. A. Moll and S.-Y. Wang [2005]. "ScatterType: a legible but hard-to-segment CAPTCHA," *Proceedings IAPR International Conference on Document Analysis and Recognition*, Seoul, Korea, August 29 – September 1, 2005.

Baird, H. S. and J. L. Bentley [2005]. "Implicit CAPTHCAs," *Proceedings IS&T/SPI Document Recognition and Retrieval XII Conference (DR&R 2005)*, Santa Clara, CA, January 2005.

Baird, H. S. and D. P. Lopresti [2005]. *Proceedings of Second International Workshop on Human Interactive Proofs*, May 2005.

Bentley, J. L. and C. L. Mallows [2005]. "How much assurance does a PIN provide?," *Proceedings of Second International Workshop on Human Interactive Proofs*, May 2005, pp. 111–126.

Bentley, J. L. and C. L. Mallows [2006]. "Analyzing and Improving Challenge Strings for CAPTCHAs," to appear as an Avaya Labs Research Technical Report.

Card, S. K., T. P. Moran and A. Newell [1983]. *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, NJ.

Chellapilla, K., K. Larson, P. Y. Simard, M. Czerwinski [2005]. "Building Segmentation Based Human-Friendly Human Interaction Proofs (HIPs)," *Proceedings of Second International Workshop on Human Interactive Proofs*, May 2005, pp. 1–26.

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

Ebbinghaus, H. [1885]. *Memory: A Contribution to Experimental Psychology.* English translation in 1913 published by Teachers' College, Columbia University, NY, NY, http://psychclassics.yorku.ca/Ebbinghaus/.

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.

Rui, Y., Z. Liu, S. Kallin, G. Janke and C. Paya [2005]. "Characters or faces: A user study on ease of use for HIPs," *Proceedings of Second International Workshop on Human Interactive Proofs*, May 2005, pp. 53–65.

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.