Making CAPTCHAs Clickable

Richard Chow
Palo Alto Research Center
rchow@parc.com

Philippe Golle
Palo Alto Research Center
pgolle@parc.com

Markus Jakobsson
Palo Alto Research Center
mjakobss@parc.com

Lusha Wang
Indiana University at
Bloomington
luswang@indiana.edu

Xiaofeng Wang
Indiana University at
Bloomington
xw7@indiana.edu

ABSTRACT

We show how to convert regular keyboard-entry CAPTCHAs into *clickable* CAPTCHAs. The goal of this conversion is to simplify and speed-up the entry of the CAPTCHA solution, to minimize user frustration and permit the use of CAPTCHAs on devices where they would otherwise be unsuitable. We propose a technique for producing secure clickable CAPTCHAs that are well suited for use on cell phones and other mobile devices. We support the practical viability of our approach by results from a user study, and an analysis of its security guarantees.

Categories and Subject Descriptors

K.6.5 [**Computing Milieux**]: Management of Computing and Information Systems—*Security and Protection*

General Terms

Security, Human Factors

Keywords

CAPTCHA, reverse Turing test, usability, mobile devices.

1. INTRODUCTION

We have seen a remarkable increase of automation of behavior that is criminal, in violation with terms of service, or plainly unwanted by service providers. Only ten years ago it was hard to imagine that computers today would be engaged in automated web navigation and requests for contents and resources, yet it is a fact that now permeates the Internet. One of the most common reminders of this type of abuse is the so-called *CAPTCHA*—short for *Completely Automated Public Turing test to tell Computers and Humans Apart.*

CAPTCHAs typically consist of sequences of contorted characters that are hard for computers (but possible for humans) to recognize. Early CAPTCHAs were rather straightforward for people to solve, but that is no longer the case. Advances in automated techniques for solving CAPTCHAs have necessitated the development of ever harder CAPTCHAs to elude attackers. CAPTCHAs have become sufficiently hard for humans to solve that many service providers now balk at deploying them for fear of deterring potential clients.

For users of handheld computers, the problem is compounded by the fact that entering the CAPTCHA solution on a mobile device is more cumbersome and time-consuming than on a traditional keyboard. Given that close to 40% of all adults in Japan browse the Internet from a mobile device [9], this is a significant problem — at least in Japan, and increasingly elsewhere too. A test to distinguish humans from computers that can be solved only with clicks of a mouse or stylus would be far less distracting and more acceptable to most users, particularly mobile users—provided the number of clicks is relatively small, and the cognitive task within reason.

We introduce the notion of clickable textual CAPTCHAs, and the related idea of combining several textual CAPTCHAs into a grid of clickable CAPTCHAs. A traditional textual CAPTCHA consists of a sequence of distorted characters (the solution is the string of characters). Our clickable CAPTCHA consists of a grid (or matrix) of textual CAPTCHAs (e.g. a 3-by-4 grid). The solution to a clickable CAPTCHA is the determination (e.g. by clicking) of the grid elements which satisfy some given requirement. For example, the user may be asked to identify in the grid the subset of CAPTCHAs which embed English words (assuming some, but not all, do). While traditional textual CAPTCHAs require the entry of the obfuscated word, ours only requires the selection of some elements in a grid. This can be done with a mouse, a stylus, or even a cell phone keyboard, as our user study demonstrates.

The security of clickable CAPTCHAs is based on an assumption similar (though not identical) to traditional textual CAPTCHAs: the difficulty of recognizing distorted characters. This gives us confidence that clickable CAPTCHAs based on grids of strong textual CAPTCHAs will be hard to solve automatically (a detailed security analysis is given in section 5). At the same time, clickable CAPTCHAs may be easier for humans to solve than regular textual CAPTCHAs (which is a desirable property). The cognitive task of deciding whether a given sequence of characters satisfies a certain condition (e.g. being an English word) often does not require (and is thus easier than) deciphering the complete sequence. This claim is supported by research such as [13, 5] for example, which shows that people can identify correct words even if the order of the letters is permuted.

The improved usability of clickable CAPTCHAs potentially opens up many new application domains. One example is defense against click-fraud [4]. In the cost-per-click advertising model, clicks from machines are never desired and potentially fraudulent. While human clicks may also be fraudulent or of low quality, distinguishing humans from machines would be a large step forward in the fight against ad fraud. Traditional CAPTCHAs, however, may be too awkward to use for this purpose, especially on UI-limited mobile devices.

Organization. We discuss related work in section 2. We present our construction for clickable CAPTCHAs in section 3, the results of our user study in section 4 and our security analysis in section 5. We conclude in section 6.

2. RELATED WORK

The notion of a CAPTCHA was first proposed by researchers at Carnegie Mellon University [14]. Early CAPTCHA proposals were text-based, displaying obfuscated text for human to decipher. One example is EZ-Gimpy [12], used in the past on Yahoo!. Many of these early approaches have been shown to be vulnerable to character recognition techniques developed specifically for CAPTCHAs [3]. Modern CAPTCHAs use harder character recognition problems such as strings with overlapping characters to complicate segmentation, or other techniques such as image and voice recognition. A survey of recent progress on these techniques can be found at [12].

Some CAPTCHAs with a user-friendly design are briefly discussed in [10]. An example is to let the user select the right orientation of a page through a click. In a similar vein, Microsoft's ASIRRA project [6] asks users to identify the cats among images of cats and dogs, and BotBarrier [1] asks users to click on a specified location in an image. The usability of our clickable CAPTCHAs is similar to the ASIRRA and BotBarrier work. However, we believe our clickable CAPTCHAs enjoy two critical technical advantages over existing image-based approaches. First, the security of our clickable CAPTCHAs can be reduced to the security of well-studied textual CAPTCHAs. In contrast, image-based CAPTCHAs such as ASIRRA or BotBarrier are based upon hardness assumptions that have not been as carefully vetted. In fact, recent work [7] shows that the problem of telling cats from dogs automatically is significantly easier than hypothesized by the ASIRRA designers [6]. The second advantage of our clickable CAPTCHAs is that they can be generated entirely algorithmically, unlike image-based approaches which rely on a static database of images. Reliance on a database of images introduces a security vulnerability (if the database is too small, or compromised) and complicates deployment.

3. CLICKABLE CAPTCHAS

We describe an embodiment of clickable CAPTCHAs based on Google textual CAPTCHAs [8]. We create 12 Google CAPTCHAs and tile them in a 3-by-4 grid. Of the 12 CAPTCHAs, 3 represent real English words (chosen at random from a dictionary of English words) whereas the remaining 9 do not correspond to any English word. The 3 CAPTCHAs containing English words are placed in random locations in the 3-by-4 grid (see Figure 1).

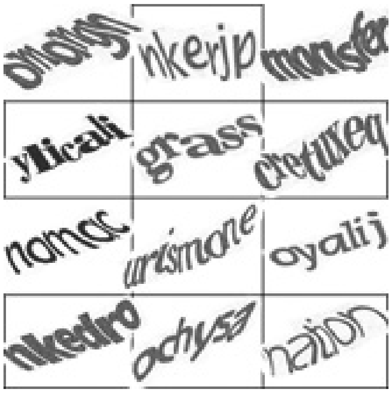

Figure 1: Example of a clickable CAPTCHA. The user's task is to identify the three valid English words (in this example: *monster*, *grass* and *nation*). In our experiment, the data entry was done using the cell phone keypad, and each of the grid element is mapped to one of the keys. Data entry may be somewhat faster for devices with touch screens.

To solve this clickable CAPTCHA, the user must click on the 3 cells which contain English words, and only those. The correct solution requires exactly 3 clicks. Any click on another CAPTCHA cell invalidates the solution. Note that an English-speaking human user can trivially solve this clickable CAPTCHA: the user scans the 12 cells, recognizes the 3 cells which contain English words and clicks on them. A computer, on the other hand, cannot decode the individual CAPTCHA cells and thus cannot solve the clickable CAPTCHA either (see section 5 for a more detailed security analysis).

To further prevent computers from automatically telling apart English words from non-English words, the sequences of letters which are not English words are generated in our experiments according to a Markov process which models the distribution of bigrams and trigrams in English. Alternately, non-English words could be generated with simple transformations of English words.

Many alternate embodiments of clickable CAPTCHAs are possible. For example, different grid sizes are possible. Any textual CAPTCHA may be substituted for the underlying Google CAPTCHA. Any binary textual classification problem can be substituted for the recognition of English words. For example, the user may be asked if the middle character of a string is alphabetic or numeric (this variation does not assume familiarity with English). Alternately, variations based on the semantics of the words displayed are also possible (e.g. click on words that correspond to edible items).

4. USER STUDIES

We ran a user study to compare the time required to solve a traditional Google CAPTCHA and our clickable CAPTCHA. In our study, a clickable CAPTCHA consisted of identify-

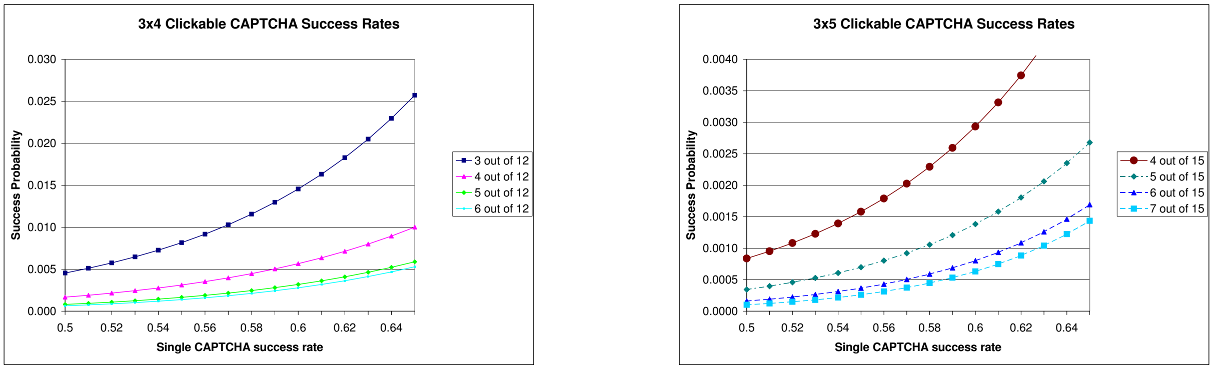

Figure 2: Probability of correctly solving a Clickable CAPTCHA automatically for 3x4 and 3x5 grids and for various values of the number of correct cells. The X-axis is the probability of correctly classifying one of the underlying individual CAPTCHAs. The Y-axis is the probability of success for the Clickable CAPTCHA.

ing 3 English words in a 3 × 4 grid of Google CAPTCHAs. Timing measurements were taken first for CAPTCHAs displayed on a regular computer screen and solved with the computer's keyboard, then for CAPTCHAs displayed on a low-cost cellular phone screen (a NOKIA 5200 with a screen resolution of 128 × 160 pixels) and solved with the phone's keypad. A total of 48 subjects participated. All subjects were undergraduate or graduate students, ages 17-38, with a variety of majors and nationalities, and equal proportions of men and women. All had used the Internet for at least six years.

Apart from start-to-finish timings, we also recorded the subjects' familiarity with English and their familiarity with computers, using two classes for each. Subjects were considered familiar with English if they had lived in an English-speaking environment for more than 5 years; they were considered familiar with computers if they were or had been enrolled in a computer-related program. We found that familiarity with English had a notable impact on the timings, while familiarity with computers had only a marginal impact. We therefore describe a version of the analysis in which we ignore the subjects' familiarity with computers and only consider the two language-based classes of subjects.

Traditional CAPTCHAs. The average time for solving a regular Google CAPTCHA (we chose 8 letter words) with a *computer screen and keyboard* was 4.3 seconds for subjects familiar with English, and 4.4 for subjects not familiar with English. The estimated 95% confidence intervals were [3.8,4.9] and [3.8,5.0] respectively. In comparison, the average time for solving a regular Google CAPTCHA with a *cell phone screen and keypad* was 15.9 seconds for subjects familiar with English, and 17.7 for subjects not familiar with English. The corresponding 95% confidence intervals were [14.5,17.4] and [14.9,20.4]. A final experiment showed no difference between the time required to solve a CAPTCHA displayed on a computer screen and on a cell phone screen, when in both cases the solution was input with a computer keyboard. We conclude that the sharp increase in the time required to solve a Google CAPTCHA on a cell phone is due to the more cumbersome data entry on a cell phone keypad.

Clickable CAPTCHAs. The average time for solving a clickable CAPTCHA with a *computer screen and mouse* was 7.6 seconds for subjects familiar with English, and 11.5 for subjects not familiar with English. The estimated 95% confidence intervals were [6.7,8.5] and [10.1,12.9] respectively. The average time for solving a clickable CAPTCHA with a *cell phone screen and keypad* was 11.1 seconds for subjects familiar with English, and 18.2 for subjects not familiar with English[1]. The estimated 95% confidence intervals were [10.0,12.1] and [16.3,20.1].

We observe that our clickable CAPTCHAs can be solved 30% faster with a cell phone screen and keypad than regular Google CAPTCHAs. Solving clickable CAPTCHAs is only 52.2% slower on a cell phone screen than on a computer screen, while solving a regular CAPTCHAs is 282% slower.

On Error Rates. In our experiments, we also measured the error rates, i.e., the portion of responses that were incorrect. We found that subjects familiar with both English and computers had a zero error rate, while other subjects exhibited error rates in the interval [0.1,0.3] on a PC and [0.1,0.2] on a cell phone (with 95% confidence).

5. SECURITY ANALYSIS

We base the security of our clickable CAPTCHA on the security of the underlying CAPTCHAs that make up the clickable grid. For that, we must define a binary classification problem for our clickable CAPTCHA whose hardness is as closely reducible as possible to the hardness of solving the underlying CAPTCHAs. For example, if the classification problem is to distinguish English words from non-

[1] It is not surprising that subjects with limited familiarity of English took more time to solve the CAPTCHAs, given that the clickable CAPTCHA test we used was based on identifying valid English words.

English words, we generate non-English words which are locally similar to English words, using a Markov model of English bigrams and trigrams. According to [2], the difficulty in solving textual CAPTCHAs lies in segmenting the characters. If segmentation is less than completely successful, local similarity ensures that pieces of non-English words are hard to distinguish from pieces of real English words.

For concreteness, we analyze the problem of recognizing 3 CAPTCHAs from a 3x4 clickable CAPTCHA grid (e.g. picking the 3 English words out of a grid of 12 CAPTCHAs). The analysis can be easily adapted to other choices of parameters. Let p be the probability of success of a machine adversary for an individual binary classification problem (we assume that this probability is uniform over all CAPTCHAs). The best strategy for the adversary is to submit 3 CAPTCHAs at random out of all the CAPTCHAs that it has classified as English words. If the adversary has classified less than 3 CAPTCHAs as English words, it makes up the difference with ones selected randomly from those classified as non-English words. Note that our clickable CAPTCHAs, unlike [6], reveal to the adversary the number of cells to select in the grid. We believe this makes our CAPTCHAs more usable, but at a cost of slightly increasing the adversary's chance of success. Since the probability of success for each of the 12 CAPTCHAs is independent, we have $\Pr(\text{success}) = \Pr_1 + \Pr_2$, where

- $\Pr_1$ is the probability that the 3 solution words are classified correctly and these 3 words are chosen among all those classified as words.
- $\Pr_2$ is the probability that 2 or fewer of the solution words are classified correctly, and the 9 non-words are classified correctly and the remaining solution word(s) are chosen from those classified as non-words.

We have:

$$\Pr_1 = p^3\Big(p^9 + \frac{\binom{9}{1}p^8\,(1-p)^1}{\binom{4}{3}} + \frac{\binom{9}{2}p^7\,(1-p)^2}{\binom{5}{3}} + \cdots + \frac{\binom{9}{8}p^1\,(1-p)^8}{\binom{11}{3}} + \frac{(1-p)^9}{\binom{12}{3}}\Big)$$

$$\Pr_2 = \frac{\binom{3}{1}p^2\,(1-p)^1\,p^9}{\binom{10}{1}} + \frac{\binom{3}{2}p^1\,(1-p)^2\,p^9}{\binom{11}{2}} + \frac{\binom{3}{3}p^0\,(1-p)^3\,p^9}{\binom{12}{3}}$$

This formula shows, e.g., that the probability of solving a clickable CAPTCHA is approximately 0.015 for a machine with a classification accuracy of 0.6. The left side of Fig. 2 shows the probability of success as a function of classification accuracy (for a 3x4 grid).

Concretely, experts estimate [11] that the Google CAPTCHA may have an adversarial success rate on the order of 10%. Assuming no information is obtained in the remaining cases, a 10% success rate corresponds to a 55% classification success rate (10% will be classified correctly and half the remainder will be correct by random guessing). As a comparison, our 3×4 grid-based version of this CAPTCHA has an adversarial success rate on the order of 1%. Some applications [3] require a chance of success on the order of 1 in $10,000$. The right half of Fig. 2 shows that these success rates can be achieved with a 3x5 grid configuration.

6. CONCLUSION

We present a generic technique for converting regular textual CAPTCHAs into clickable CAPTCHAs. Our user study shows that our clickable CAPTCHAs can be solved faster than textual CAPTCHAs on mobile devices. Unlike image-based clickable CAPTCHAs, our clickable CAPTCHAs do not rely on a static database of images. The security of our clickable CAPTCHAs is reducible to the security of textual CAPTCHAs, which have withstood the test of time.

7. REFERENCES

[1] BotBarrier.com. http://www.botbarrier.com/

[2] K. Chellapilla, P. Simard. Using Machine Learning to Break Visual Human Interaction Proofs (HIPs). In *Proc. of NIPS 2004*.

[3] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski. Designing Human Friendly Human Interaction Proofs (HIPs). In *Proc. of 2005 SIGCHI conference on Human factors in computing systems*, pp. 711–720.

[4] Click Fraud, the dark side of online advertising. BusinessWeek, October 2006. http://www.businessweek.com/magazine/content/06_40/b4003001.htm

[5] M. Davis. "Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy". http://www.mrc-cbu.cam.ac.uk/~mattd/Cmabrigde/

[6] J. Elson, J. Douceur, J. Howell and J. Saul. Asirra: a CAPTCHA that exploits interest-aligned manual image categorization. In *Proc. of ACM CCS 2007*, pp. 366-374. http://research.microsoft.com/asirra/

[7] P. Golle. Machine Learning Attacks Against the ASIRRA CAPTCHA. IACR Cryptology ePrint Archive, report 2008/126.

[8] Google CAPTCHA. https://www.google.com/accounts/DisplayUnlockCaptcha

[9] Ipsos Research. Face of the Web 2006. http://www.ipsos-na.com/news/pressrelease.cfm?id=3049

[10] D. Lopresti. Leveraging the CAPTCHA problem. In *Proc. of the Second International Workshop on Human Interactive Proofs*, pp. 97–110. Springer Verlag, 2005.

[11] G. Mori. Personal communication.

[12] The Official CAPTCHA Site. http://www.captcha.net.

[13] G. E. Rawlinson, "The significance of letter position in word recognition," PhD Thesis, Psychology Department, University of Nottingham, Nottingham UK, 1976.

[14] L. von Ahn, M. Blum, N.J. Hopper and J. Langford. CAPTCHA: Using hard AI problems for security. In *Proc. of Eurocrypt '03*, pp. 294–311.

[15] O. Warner. Kittenauth. http://www.thepcspy.com/kittenauth