Strengthening CAPTCHA-based Web Security

Graeme Baxter Bell
Affiliated with the Applied Artificial Intelligence Group,
Murdoch University, Australia.
E-mail: research2012@graemebell.net
Web: http://graemebell.net , http://aai.murdoch.edu.au

Abstract

Simple, universally applicable strategies can help any captcha-protected system resist automated attacks and can improve the ability of administrators to detect attacks. The strategies discussed here cause an exponential increase in the difficulty faced by automated attackers, while only increasing the inconvenience for human users in an approximately linear manner. These strategies are characterised using a new metric, the 'Captcha Improvement Ratio'. The paper concludes that presenting multiple captcha systems together in random order may provide quantitative and qualitative advantages over many typical present-day captcha systems.

Keywords: Web security, CAPTCHA, Abuse of websites and services, CAPTCHA Improvement Ratio

1 Introduction

Many of us are familiar with websites that force us to identify peculiar, warped letters in cluttered images and enter them into a text box. One of the standard chores of registering with modern Web services is the demonstration that we truly are human beings and not nefarious computer programs set on causing mischief. Yet despite the warped and cluttered characters in these images becoming harder to distinguish as each year passes, Web-forums and

blog comments still seem to be filled with unwanted spam adverts. Clearly, the system is not working as well as it should. This paper therefore addresses the topic of Web CAPTCHAs - Completely Automated Public Turing tests to tell Computers and Humans Apart - and discusses how we might make them more effective and long-lasting without greatly inconveniencing everyday users[1].

Captchas are an important and widely used modern Internet technology. They reduce the ability of automated agents to programatically exploit Web-based resources such as online e-mail accounts, online polls, Web-based comment systems, Web-based SMS portals and so on (Carnegie-Mellon University, 2009; Pope and Kaur, 2005; von Ahn et al., 2003). A captcha is a challenge that authenticates users as human; human users are generally accepted to be much less able to exploit Web-based resources than automated agents. Captchas require users to prove they are human by conveniently exhibiting human-level intelligence in some manner (Turing, 1950; von Ahn et al., 2003). Generally, a user must correctly interpret an image, sound, or text phrase that has been mathematically corrupted with noise, perturbations and transformations, though other types of captcha exist (Baird and Bentley, 2005; Baird et al., 2005; Bursztein et al., 2010; Chow et al., 2008; Coates et al., 2001; Gossweiler et al., 2009; Microsoft, 2007; Misra and Gaj, 2006; Ritendra Datta and Wang, 2006; Shirali-Shahreza and Shirali-Shahreza, 2007a; von Ahn et al., 2003). The noise and transformations are intended to make computer-based image or sound recognition effectively intractable. In principle, this means that only humans can provide a correct response and thus gain access to the Web resource protected by the captcha.

Unfortunately, in practice, captchas have not been quite so successful (Chellapilla and Simard, 2005; Golle, 2008; Huang et al., 2008; Mori and Malik, 2003; Moy et al., 2004; Yan and Ahmad, 2008a). Generally, the essential problem in captcha design is the tradeoff between the difficulty of the captcha for automated agents, and the convenience of the captcha for human agents. Figure 1 shows an example of a typical current-day captcha.

To date, the captchas that have been designed and implemented for high traffic sites such as Google, Yahoo and MSN have proven vulnerable to attack (Bursztein et al., 2011; Bursztein and Bethard, 2009; Protalinski, 2008; Vaughan-Nichols, 2008; Websense Security Labs, 2008a,b,c, 2009). Designers of malicious automated agents have shown themselves to be very capable of

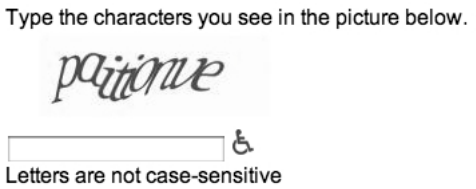

Figure 1: Example of a commercial captcha system (Google, 2009).

overcoming the obfuscation present within captchas, often combining simple heuristics with brute force attacks. In addition to these attacks by real-world malicious users, Web security researchers engage in cat-and-mouse style research in which captchas are designed and then broken, as an iterated and evolving research challenge (Baird and Bentley, 2005; Baird et al., 2005; Chellapilla and Simard, 2005; Golle, 2008; Huang et al., 2008; Microsoft, 2007; Mori and Malik, 2003; Moy et al., 2004; Yan and Ahmad, 2008a). This presents a problem for major websites relying upon captchas, as researchers who successfully break a captcha system may indirectly assist malicious users. In principle, research helps industry-based captcha system developers to produce robust and convenient captchas. In practice, the delay between captcha systems being overcome and then replaced by a newer and more successful captcha technique, allows malicious users a golden opportunity to abuse Web-based resources.

This paper aims to help break the cycle whereby captchas are proposed and rapidly broken, by encouraging a systematic approach to flexibly strengthening any type of captcha system against automated attacks, while not greatly inconveniencing human users. Put simply: any captcha system may fail swiftly and unnoticeably through the efforts of a determined attacker; therefore high value sites should consider employing several different captchas together.

Three different generic captcha augmenting strategies are assessed here by evaluating their relative impact upon human users and automated captcha-attacking software. The strategies presented are shown to be very straightforward, very effective, and retrospectively rather obvious, but they do not seem to be presently employed for real world Web-based security. Further, the strategies yield other benefits in addition to foiling attacks. This paper

concludes that the adoption of these strategies would improve the ability of real-world captcha systems to defend Web-based resources against present and future attacks.

This paper is structured as follows. Section 2 discusses classes of captcha attacks, and existing general strategies for defence. Section 3 introduces a metric for evaluating captcha improvement strategies quantitatively. Section 4 discusses several strategies that substantially strengthen the ability of any captcha system to resist automated attacks. Section 5 proposes guidelines for captcha developers. Sections 6 and 7 present suggestions for future work and conclusions, and are followed by references.

2 Background: Forms of attack

Whether a captcha is based on pictures, text, sound, or puzzle-solving, certain similarities can be seen in terms of how captchas are attacked by malicious users. Typical attack models seen to date include:

Bypass attacks Any attack that circumvents the need to solve the captcha at all. For example, network replay attacks, or cases where the captcha solution is exposed accidentally, perhaps through HTML or CGI form parameter values. Generally, any system that sends the decoded form of the captcha to a client program as part of the data stream is vulnerable to such an attack. Such attacks are not always a weakness of the captcha itself; they may instead be a weakness of the service using the captcha.

Challenge replay attacks If the captcha system can produce only a limited number of unique challenges, then the automated agent may record all or most of the possible challenges. A human associate provides a library of correct answers for the challenges. The automated agent can then replay the correct answer whenever it is faced with a particular challenge for which it knows the correct solution. Some image-based captchas are vulnerable to this weakness, particularly those based upon a finite library of photographs (e.g. the 'KittenAuth' captcha (Reimer, 2006) used a challenge library of 42 images).

Signal processing attacks The noise and perturbations that are commonly used to obfuscate captcha images or sounds are intended to be one-way; a computer should be able to add them, but not reverse them easily. In principle, only a human's flexible image and sound recognition capabilities should be capable of conveniently reversing the transformations and recovering the original message. In practice, captcha researchers and malicious attackers have both proven highly capable of reversing captcha transformations approximately via mathematical heuristics and machine learning approaches (Chellapilla and Simard, 2005; Hocevar, 2004; Huang et al., 2008; Mori and Malik, 2003; Yan and Ahmad, 2007, 2008a). For text-based captchas, this is achieved by removing image noise and clutter items, and isolating individual characters within the captcha in order to allow optical character recognition (OCR) technologies a maximised opportunity for success. Attacking heuristics often have a parameterised design, so that their behaviour may be adjusted to attack several different but related forms of captcha.

Mechanical Turk attacks Here, the problem of solving the captcha is automatically 'outsourced' to a paid human agent. They immediately solve the challenge and quickly return the answer to the automated agent in real time. The automated agent then presents the human-provided answer, and is able to programatically exploit the online resource (Barr and Cabrera, 2006; Bursztein et al., 2010; The Economist Newspaper Ltd., 2008; Websense Security Labs, 2008b). A human 'Turk' agent working full time to support such attacks can solve thousands of captchas per hour, depending on the type of captcha. There is little that can be done to defend against such attacks, other than to perhaps raise the inconvenience of the captcha for all users in order to reduce the economic viability of this attack. Generally, an increase in inconvenience can be achieved either by increasing the difficulty of the captcha for humans, or by requiring users to regularly re-authenticate themselves as human.

Trivial guessing attacks If there is an unlimited range of challenges, but a very limited range of possible answers (e.g. 'which of these 10 choices is correct?'), a high success rate may be achieved by an attacking program by merely guessing randomly from the available answers. Particularly, any graphical captcha that requires the user to select a correct position within

an image - but which has a wide error tolerance for user inaccuracy - may be vulnerable to a trivial guessing attack.

Brute force attacks If there is a somewhat limited range of possible answers - e.g. a numerical 4-digit captcha would have 10,000 possible answers - then it is possible for a distributed group of automated agents to attack the captcha by exhaustively trying answers at random or according to a selected sequence. This differs from the 'trivial guessing attack', in that it relies upon having access to a large number of attacking agents - i.e. a 'botnet' (Websense Security Labs, 2008b, 2009) - rather than relying upon having access to a poorly designed captcha.

Hybrid attacks It is possible to combine these attacks. For example, if a signal processing attack can estimate 5 of 6 captcha characters with a high degree of confidence, a guess may be made on the remaining character, yielding a success rate of between 1.5% (mixed case alphanumerical characters) and 10% (numerical digits). For example, the 'Question-Based captcha' (Shirali-Shahreza and Shirali-Shahreza, 2007b) presents a mathematical problem, which can be broken by an attacker who uses OCR to recognise the numerical digits mentioned in the puzzle, combined with a random guess of one of the few possible ways in which the numbers may be combined arithmetically.

2.1 Success rates for automated attacks.

Automated attacks against captchas may have success rates ranging from a positive infinitesimal up to 100%, depending on the type of captcha and the form of attack. For signal-processing attacks, a low degree of success is typical for initial efforts against a new captcha (e.g. 0-5%). This increases over time as newer and more successful heuristics are introduced but rarely rises beyond 70-80% accuracy against well designed captchas (Bursztein et al., 2011; Bursztein and Bethard, 2009; Chellapilla and Simard, 2005; Huang et al., 2008; Moy et al., 2004; Yan and Ahmad, 2008a). For challenge replay attacks, success rates are proportional to the amount of the challenge library that has been previously seen and solved. Mechanical Turk attacks are as

successful as regular human attempts. Guessing attacks and brute force attacks are successful in inverse proportion to the range of possible answers to the captcha, and in proportion to the attacking agent resources available for the attack.

The issue of attack success rate is of critical importance to captcha designers. Automated agents are generally free to attack many times, often from different source networks, and with arbitrary delays between attacks to avoid detection. An attack which succeeds a mere 5% of the time, requires only 20 attacks on average in order to achieve one successful result in gaining access to the protected resource. This is well within the capabilities of attacking agents, and is economically viable for attacks. Consequently it is well accepted that there is a need for techniques that limit successful attacks upon captchas to at most 1 case in 10,000 (0.01%) (Chellapilla and Simard, 2005). In practice, however, this degree of effectiveness against attack has never been maintained for any substantial length of time. Consequently, it may be helpful for designers to target extremely low expected rates of success for attacks - i.e. perhaps 0.00001% or less. Furthermore, it seems there is also a need for techniques that can help to limit the speed with which attackers improve their success rate against captchas over time.

2.2 Issues addressed in this paper.

This paper seeks to address two matters. Firstly, do general strategies exist which increase the difficulty faced by automated agents very substantially, while having a limited impact upon human user convenience? Secondly, how might an objective a-priori estimate be made of the degree of improvement offered by a proposed strategic modification to an existing captcha system, given the qualitative nature of the terms 'difficulty' and 'convenience'?

There are very few systematic techniques that can be applied to any form of captcha in order to reliably improve their utility. The most commonly used approaches are:

- *Adding random noise/clutter* - first formally discussed in (Mori and Malik, 2003). In the case of a graphical captcha, some obfuscating characters, dots, or backgrounds are chosen to make the underlying challenge hard for algorithms to process. In the case of an audio

captcha, static and crackling noises may be added to simulate line noise.

- *Distorting globally* (image-level transformations) or *distorting locally* (character / sub-image transformations) - introduced as PIX in (von Ahn et al., 2003) and analysed in (Chellapilla et al., 2005). Here, parts of characters and groups of characters are made mis-shapen through mathematical formulae, so that they no longer resemble the original font. This can cause character detection, segmentation and recognition techniques to fail.

- *Removing obvious avenues of attack* - this includes *limiting the number of answer attempts* and *distrusting the client*. These are essential general strategies that have been re-used throughout many distinct captcha systems. Von Ahn et al. gave other useful design principles in their 2003 captcha paper, e.g. *avoid reliance upon secrecy* (von Ahn et al., 2003).

However, the section of Von Ahn et al.'s paper that briefly discusses the idea of *"gap amplification by sequential repetition"* is still (in this author's opinion) the clearest attempt to characterise a flexible and systematic strategy that can be shown through analysis to augment the challenge of an arbitrary captcha system of any type. It will be discussed further below.

Besides these techniques, the idea of developing and measuring captcha-augmenting strategies that could be applied to any type of captcha system to make it better does not seem to have been broadly discussed in existing literature. Instead, research has focused on a search for new types of captcha challenge that are more effective from the start.

3 Measuring the effectiveness of strategies

Recall that the goal of captcha research is generally considered in a qualitative and subjective manner: to minimise the inconvenience for human users, while maximising the difficulty for automated attacking programs. Unfortunately, it is very difficult to quantify the convenience of individual captcha systems in a numerical form; and impossible to estimate the robustness of a system

to unknown present and future automated attack techniques. Further, there is the problem of combining these two aspects of captcha design into a single metric that might allow direct comparisons to be made between alternative captcha systems.

Here, a metric is adopted based upon the estimation of *relative improvement* due to a strategy applied to a captcha system, rather than the absolute outcome, in terms of 'convenience for humans' and 'difficulty for computers'. This sidesteps the challenge of establishing numerical values for absolute difficulty and absolute convenience, in a world where no two captcha systems, users, or attacks are identical. "*Relatively speaking*, for each type of user, how much work must now be done after application of the strategy?"

The idea of *work* is awkward. Clearly, there are many factors and complexities involved in both the human and computer efforts in overcoming a captcha challenge. The nature of these factors for humans (often psychological, perceptual, or time-based), and for computers (often computational, memory, time, or network-based), as well as their interactions, makes precise absolute estimation of work almost impossible. Nonetheless, certain key principles can be hypothesised.

For example, we can suppose that a captcha formed of two equivalent parts might be approximately half as convenient as solving only a single part. There are few indications that the majority of people have rebelled at the gradually increasing complexity and workload of captchas over the years, as services guarded by captchas continue to be used in greater and greater numbers. Consequently, providing that a captcha does not stray 'too far' from existing levels of work or complexity, users can be expected to perceive it as being of proportional convenience.

However, this assumption cannot be taken for granted! Certain non-linearities will come into play as these principles are stretched further from known cases. For example, a human required to solve 50 text captcha challenges laid out in sequence without any error might refuse outright to even try, which would raise the inconvenience of the captcha by an infinite degree. A computer heuristic, on the other hand, would readily begin to process the data and might experience only a finite increase in difficulty, particularly if it has an extremely high rate of success at captcha character recognition. A degree of care should be taken when evaluating captcha strategies, so as not to stray very far from what is already established or

reasonably plausible.

I therefore suggest the following principles:

- Firstly, all other things being equal, solving m distinct captcha challenges of the same type probably requires approximately m times as much effort as a single challenge, whether it is a human or computer that is solving the challenge - for small m.
- Secondly, a captcha that requires m times more effort from a human user in order to achieve a single success, is consequently m times less convenient - for small m.
- Thirdly, a captcha that requires m times more effort from an automated attacker in order to achieve a single success, is consequently m times more robust against attack.

From this, the ideas of human effort and computer-heuristic effort can be merged into a combined metric that expresses the overall effect of a strategy, in terms of convenience for humans and robustness against automated attacks. This paper proposes that the Captcha Improvement Ratio (CIR) of a given captcha-strengthening strategy is said to be a ratio equal to the *approximate relative increase in average-case work performed by a computer* in order to pass the modified captcha, divided by the *approximate relative increase in average-case work performed by a human* in order to pass the modified captcha. Here, 'average' refers to the mean.

$$\text{CIR} = \frac{\text{approximate average-case increase in work for computers}}{\text{approximate average-case increase in work for humans}} \tag{1}$$

The higher the CIR, the greater the improvement of the captcha, and the worse the situation becomes for automated attackers compared to humans

- for a given improvement strategy, and relative to the original unmodified captcha challenge.

A key problem with this approach is that the competence of present and future attack algorithms is unknown and varies over time; and the work required by a computer to pass the captcha is largely contingent on an attack's degree of competence. Consequently, numerical values for the CIR can only ever be estimated, perhaps by using worst-case assumptions informed by known trajectories for attacker success over time. Nevertheless, algebraic comparisons can be usefully made between the CIRs of strategies that rely upon the same factors, without the need to assign numerical values to those factors.

4 Proposed captcha improvement strategies

A series of strategies are presented in this section that are intended to strengthen captcha systems. Here, it will be assumed that a human passes a captcha on average $h\%$ of the time, and that a computer agent passes a captcha on average $c\%$ of the time, by guessing, signal processing, or some other form of attack. The phrase 'user' will refer to an unknown agent who may be a human or a computer, 'human' will refer to a known human, and 'attacker' will refer to a known automated software agent. 'Administrator' will refer to the person managing the Web service and captcha system. It will be assumed that for the base case, a well-designed unmodified real-world captcha has been chosen.

4.1 The base case: Lifecycle of a captcha

What happens during the lifetime of a normal captcha? Here, we consider an unmodified captcha which faces increasingly more effective attacks over time.

Throughout the lifespan of the captcha, the human rate of success (h) with a well-designed captcha will be quite close to 100%. Historically, Chellapilla suggested that human success rates should be around 90% (Chellapilla et al., 2005), and the most commonly used current-day captcha project

(reCAPTCHA (von Ahn et al., 2008)), which is in use at over 40,000 websites, records human rates of success that are consistently in the range 93% to 97% (Yan and Ahmad, 2008b)[2].

As far as the attacker is concerned, success rates for computer agents against well-designed captchas are known to vary from a positive infinitesimal at time of captcha design, to as high as 50-80% over time, depending on the attack method in use and the form of the captcha.

On average, the user needs a total number of attempts equal to $1/h$ if they are human, and $1/c$ if they are an attacker. The base case is that the attacker takes h/c times more attempts to solve a captcha once than a human user would need.

4.1.1 Stage A: Uninformed guessing attack

If we assume $h = 90\%$ and $c = 0.0001\%$ for a newly developed and well-designed real world captcha, it would take 900000 times more attempts for an attacker to pass the captcha once, than would be required by a human. This is somewhat impractical and unrewarding for an attacker, and so the captcha will be effective in deterring and preventing automated attacks. Furthermore, it would be trivial for an administrator to determine if an attack was taking place, given the high number of failed attempts, by looking at website access logs.

4.1.2 Stage B: Weak heuristic attack

Over time, a revised attack may be developed with a success rate of 0.1% or more. At this point, the captcha is passed at least 1 time in every 1000 attempts by the attacker, as opposed to 1 time in every 1.11 attempts by the human. Although the attacker is failing up to 99.9% of the time, their occasional successes will allow them to abuse the service to a limited degree. The administrator might be aware that an attack of some form is taking place, because of the large proportion of failed attempts at solving the captcha; but they may be unaware that the attacker is achieving slight success unless they can separately observe the abuse of the system. At this point, the captcha's main value lies in it's ability to slow the rate of abuse and potentially allow

the automated detection of attacks from particular IP addresses.

4.1.3 Stage C: Competent heuristic attack

Subsequently, an attack is produced with a success rate of perhaps 10%. At this point, the captcha is overcome 1 time in every 10 attempts by the attacker. By this stage, the captcha is almost worthless as a defence against automated attackers; it is reduced to a trivial inconvenience. Attackers may freely abuse the protected Web service. However, it is still possible for administrators to detect that attackers exist by examining the website logs for the captcha (because of the relatively greater number of failed attempts).

4.1.4 Stage D: Expert heuristic attack

Finally, an attack developer produces an attack with a success rate close to or equal to that of a human, perhaps 50% or more, e.g. (Chellapilla and Simard, 2005; Huang et al., 2008; Moy et al., 2004; Yan and Ahmad, 2008a). The defending administrator now loses the ability to discern through the captcha system that the site is being attacked, as there will be very few failed attempts per success. If an attacker can achieve an initial attack with this kind of success rate, then the defender may never realise that the captcha system has been compromised. They can only detect the attack if they observe the resulting abuse of the protected resource - by which time it may be too late. This is a qualitatively different situation to Stage C.

4.2 Strategy 1: Simultaneous instances (m-captcha).

A basic situation has been considered involving the presentation of a typical current-day captcha.

We now consider the case where several (m) distinct, unique instances of a captcha are presented together or in sequence, and the user required to solve all of the instances correctly in order to pass the challenge as a whole.

This idea was first put forward in (von Ahn et al., 2003) in the section "Gap Amplification". The authors write:

> "We stress that any positive gap between the success of humans and current computer programs against a captcha can be amplified to a gap arbitrarily close to 1 by serial repetition... amplifying a gap can roughly be thought of as increasing the security parameter of a captcha: if the best computer program now has success 0.10 against a given captcha (for example), then we can ask the prover to pass the captcha twice (in series) to reduce the best computer programs success probability to 0.01." (von Ahn et al., 2003)

They conclude: "Since captchas involve human use, it is desirable to find the smallest m possible" (von Ahn et al., 2003)

This approach seems trivial to implement, but the vast majority of mainstream captcha services have not currently adopted it[3]. Even those few services that have something of this nature have considered only the case of two captcha instances side-by-side. Yet the m-captcha strategy can be easily applied to any form of captcha, in the same way that noise and clutter can be added to almost any captcha presented as an image, and it has a very useful and significant effect upon the ability of a captcha to resist automated attacks, as described in the quote above. However, minimising m in the interests of human convenience as they suggest may not be the best approach, given that:

- The rate of success that can be achieved by an attacker is always unknown and varies over time. This makes it hard to optimise m.
- Human convenience is likely to be subjective and hard to estimate; is it better to have a greater number of simple captcha instances which must be passed together or a single hard-to-pass captcha instance?
- Modern attacks improve very quickly, on a scale of weeks.
- It may be useful to be able to monitor attacks even when the captcha challenge is failing to provide a barrier; the selection of a desired 'gap' may be influenced by the desire to monitor attacks as well as control access.

The key idea that this paper now presents in regard to m-captchas, is that increasing m has a disproportionately damaging effect on automated agents, compared to humans. For this reason, we should be willing to consider raising m beyond the current value of 1 or 2, perhaps to 4 or 5. The reasoning is as follows.

Consider a human user who must solve m instances of the captcha correctly, together and without any mistakes or retries at any individual instance. The chance of the human achieving success is equal to h^m, where h is the chance for success on a single instance of the captcha. The average number of attempts taken to pass the m-captcha once is therefore $1/h^m$. However, the human is now undertaking m times more effort in total in each attempt to solve the combined m-captcha challenge. The total effort required therefore increases as $m(\frac{1}{h^m})$. We know that the human's success rate is high - e.g. $h = 0.9$. Consequently, the growth in the value of $m(\frac{1}{h^m})$ is dominated by the linear aspect rather than the exponential term, for small values of m ($m \leq 5$). This is because the exponent of a fraction very close to 1.0 remains close to 1.0, for small m.

In short: human effort scales near-linearly with the number of instances to be passed together - providing there are not too many and providing they are not exceptionally challenging.

A computer attacker will also experience a growth in effort required per success, as $m(\frac{1}{c^m})$. Since c is known to be far from 1.0, typically in the range 0.001 to 0.5, the growth in the value of $m(\frac{1}{c^m})$ is dominated by the exponential aspect for $m \geq 2$.

In short: attacker effort scales exponentially with the number of instances to be passed together. Consequently, for each extra instance - m=3,4,5,... - the m-captcha strategy vastly amplifies the difficulty experienced by a computer attacker, while inconveniencing a human to a disproportionately smaller extent.

The CIR for m-captchas can be derived for the general case by considering the relative improvement in the ratio between human and computer average-case efforts. Previously, the attacker effort was $1/c$. Now it is $m((1/c)^m)$. The relative increase is therefore $m((1/c)^{(m-1)})$ times greater. Previously, the human effort was $1/h$. Now it is $m((1/h)^m)$. Simplifying the ratio yields $(h/c)^{(m-1)}$. It is therefore established that by presenting m distinct instances

of a captcha and requiring all to be correctly solved together, the relative effectiveness of the captcha is improved by a factor:

$$CIR = (h/c)^{(m-1)} \tag{2}$$

How significant is this improvement? As noted above, this is a real-world captcha, so it is known that h is very close to 1.0. Therefore, h^m is close to 1.0 for small m. Consequently, the dominant factors are c (the skill of the computer agent) and m (the number of instances of the original captcha to be answered simultaneously). Table 1 displays CIR values for Strategy 1, varying according to c and m, and assuming h is equal to 0.9. Values are given for 1-captchas - the base case - through to 5-captchas. These figures describe the relative effect for automated agents as opposed to humans.

Attacker success %	0.001%	0.1%	1%	10%	50%
CIR, m=1	1	1	1	1	1
CIR, m=2	90000	900	90	9	1.8
CIR, m=3	$(8.1).10^9$	810000	8100	81	3.2
CIR, m=4	$(7.29).10^{14}$	$(7.29).10^8$	729000	729	5.8
CIR, m=5	$(6.6).10^{19}$	$(6.6).10^{11}$	$(6.6).10^7$	6561	10.5

Table 1: Captcha Improvement Ratio for an m-captcha.

The success rates of the attacker at all of the levels described earlier are vastly diminished by the m-captcha strategy. This means that in turn, a substantially increased challenge is presented for the programmer of the attack software, who must now fully replicate something much closer to human-level abilities, rather than merely construct a partially successful initial attack that enables a subsequent brute-force attack. Consequently, the timeframe over which any form of captcha is useful can be greatly extended. It may also be the case that by raising the threshold for initial attacker success so substantially, some attackers will be discouraged from attempting to break the captcha with weak heuristic attacks and will not go on to develop stronger attacks. Finally, any subsequent improvement to a captcha design that increases the difficulty for attackers will have its effect greatly amplified by the m-captcha approach, making it very worthwhile to develop and implement minor improvements.

Example: Previously, it was noted that if a traditional 1-captcha is presented, the attacking program is slightly effective when $c \approx 0.1\%$; moderately effective when $c \approx 1\%$; and highly effective as c reaches 10% or thereabouts. The attacker becomes essentially invisible if it can succeed around 50% of the time or better. In contrast, assume that the corresponding 4-captcha is presented. If the attacking program can achieve a success rate of 0.1% on the original 1-captcha, then it is ineffective against the 4-captcha. Even at a success rate of 10% against the original 1-captcha, the attacker remains quite ineffective against the 4-captcha, succeeding 729 times less often than it would against the original 1-captcha. It is not until the attacker's success rate on the original captcha nears 50%, that it becomes sufficiently effective to be able to overcome the 4-captcha and exploit the system at all. Even then, the attacking program does not become invisible to detection, as it would with a traditional unmodified captcha.

Strategy 1 (m-captcha) essentially forces all captcha attack developers to produce attacking heuristics whose initial success rate are close to 50%, which is at the very top end of existing research capabilities. It also forces attack developers to produce techniques whose initial success rate is indistinguishable from a human, if they wish to avoid detection. The m-captcha strategy provides good protection against the 'trivial guessing attack' approach, whose success rates on traditional 1-captchas are seldom higher than 10%. It protects effectively against all weak signal-processing attacks with success rates under 10%. It proportionally raises the economic cost of 'Turk' attacks according to the value of m. Lastly, this technique can be conveniently and rapidly scaled up to cope with an increased level of success by automated attackers; the value m can simply be raised higher.

4.3 Strategy 2: Simultaneous distinct types of instances: (t-captcha).

It is clear though that as the automated agent begins to approximate the skill of a human on a particular captcha, the utility of the m-captcha reduces. Table 1 highlights this phenomenon clearly. In the scenario given earlier, the 5-captcha is initially almost 10^{20} times more challenging than the original captcha. However, if the attacker is eventually able to achieve a 50% success rate, then the 5-captcha is merely 13 times more difficult than a regular

captcha. As the attacker's abilities increasingly approximate human ability, the gap trends towards zero, and 'amplification' becomes meaningless.

Instead, when an attacker's success rate improves dramatically against a challenge, the attacking agent should still be left unable to compete against a human's natural abilities. At the very least, a website administrator should retain the ability to detect that attacks are taking place as the captcha fails. What is needed is a mechanism that amplifies the gap when it exists, yet acts as a fallback as the gap diminishes and disappears.

Strategy 2 (t-captcha) augments the initial captcha with other types of captcha challenge that must all be solved together within a single attempt. A human is generally able to achieve very high success rates on *any* kind of captcha; whereas an automated agent is generally only able to attack one form of captcha[4]. With the t-captcha strategy, this inflexibility on the part of the attacker will be exploited.

A t-captcha will be any captcha instance that contains individual sub-instances of t unique, distinct captcha systems, all of which must be solved in sequence as a single challenge response. If the base case captcha is modified by adding $(t-1)$ further distinct, simultaneous captcha challenges, the CIR is the product of the expected success rates of the attacker against each new part of the captcha $(c_2, c_3, c_4 ... c_t)$. There are a number of benefits to this approach relative to m-captchas.

$$CIR = \frac{h^{(t-1)}}{c_2.c_3.c_4...c_t} \tag{3}$$

4.3.1 Breaking part of the captcha has a minimal impact upon the effectiveness of the captcha as a whole

Firstly, any attack system that is particularly effective against one of the captcha sub-instance types gains little against the t-captcha as a whole. Imagine that an automated attack is designed, that achieves a remarkable 100% success rate against system C_2. Even this remarkable attack gains the attacker no useful ground against the system as a whole. The combined effort required to pass the whole t-captcha once is now defined by the competence

of the attacker against the remaining parts of the t-captcha, that is, $C_3...C_t$. The typical success rate per attempt would still be $(0.01^{(t-1)})$ and the resource would remain protected.

Example: For a t-captcha based upon 4 sub-challenges (a $4t$-captcha) under the assumptions above, it would still require the equivalent of 4.10^6 times as much effort for a 'remarkable attacker' to pass the $4t$-captcha once, compared to the single traditional captcha. A human, in contrast, might require merely $4/(0.9^3) = 4.49$ times as much effort per success compared to the traditional captcha. Consequently, even this theoretical 'remarkable attacker' faces a typical relative workload that is still around 729000 times greater than that faced by a human.

Generally, the CIR will be an extremely high value as it was with Strategy 1. It could be said that with the t-captcha, we are not only exploiting the flexibility of human perception within each captcha instance; additionally, we are exploiting the flexibility of human perception in aggregate across different forms of captcha, in order to exponentially amplify the challenge of the captcha for attackers but not for humans. Consequently, the overall effort for a human will be essentially no different for an t-captcha than it would for an m-captcha; after all, the human has an approximately equal level of skill against any reasonably designed captcha instance. The effort required from the human will therefore still scale approximately linearly for small values of t as before. However, for an attacker to achieve a level of competence that is equivalent to a human overall, the attack developer would need to produce multiple distinct attack heuristics targeted against each unique part of the t-captcha, with each capable of nearly human-level success rates. This is a substantially greater challenge for attackers.

4.3.2 Attackers must become multi-skilled

A second and indirect benefit of t-captchas exists in terms of the challenge presented to the developers of attacks against captchas. Previously, the developers implementing the attack system must have expert-level ability in one field (such as image-processing or sound-processing) in order to overcome a single captcha system. Now, the developers must possess expert-level skills across a far wider range of computing disciplines. A text-captcha attack

developer may find themselves stumped by a simultaneous 'photo captcha', 'logic captcha' and 'audio captcha'. By forcing the attack developer to learn multiple new skills (or hire extra developers), the economic viability of captcha attacks is sharply reduced, and the longevity of the overall system increased[5]. This represents a significant challenge for attack developers in future, providing that industry-based captcha developers take advantage of the full range of captcha systems that have been invented, besides text-based captchas.

4.3.3 New captcha systems can be prototyped within live systems with less risk

Many captcha challenge systems have been proposed in research papers, but few have been adopted in practice. t-captchas make it possible to prototype academic systems for effectiveness without taking a risk by abandoning established approaches to captchas such as text-recognition captchas.

Any agent that can immediately pass every part of a newly deployed t-captcha system is essentially guaranteed to be a real human user. By tracking IP addresses and attempts, it would be possible to gather statistics about humans who accidentally fail against one or more parts of the t-captcha on their first effort but succeed following a subsequent attempt. It is therefore possible to conveniently and automatically track the success rates that are being initially achieved by real humans within each part of the t-captcha. Using this information, captcha developers can put forward new types of captcha challenge within a t-captcha, and rapidly assess how convenient they are for humans, by tracking failure rates from the moment of first publication. A captcha-instance which is not effective within the t-captcha can be easily (and perhaps automatically) replaced by a more convenient alternative. Crucially, this can be done without endangering the overall integrity of the t-captcha against attackers. This strongly contrasts with the risk taken by any Web service provider who employs a newly designed captcha test in isolation, i.e. the traditional and current-day practice. It is possible to imagine a framework for t-captchas that evolves in parallel with captcha-breaking toolkits, with deployed systems using plug-ins to provide a suite of the most effective known types of captcha at any time.

4.3.4 Costs and risks

This strategy may also present some new challenges for administrators and developers. Users may simply be unwilling to accept the extra burden of completing different types of captcha in order to authenticate themselves during registration or use of resources. Some thought may be required to address the issue of impaired users who rely on alternative captcha challenges such as audio; perhaps it will be necessary to produce new types of audio captcha challenge. Finally, progress in real-world captcha systems has been slow to date; there may be even less enthusiasm for the parallel development of multiple types of challenge.

4.4 Strategy 3: Randomly-ordered t-captchas.

Strategy 3 adds an extra degree of difficulty for computer attackers, while adding little or no extra difficulty for humans. Under Strategy 3, the t distinct types of captcha within the t-captcha are presented in a random, changing order. They should be presented in such a way that they cannot be trivially distinguished from each other e.g. by interface, or representation within the page HTML. In the best case for attacking, the attacker will be able to correctly identify the nature of each part of the combined t-captcha with 100% accuracy, and then use an appropriate attacking heuristic for each corresponding part. On the other hand, if the attacker has no way to determine the order of the types of captcha they are attempting, they must correctly guess the order of each part of the t-captcha at random. This implies choosing from t-factorial possible orderings in the hope of applying the correct attack to each part of the t-captcha and achieving overall success.

In contrast, a human who is interacting with the captcha on multiple occasions will experience little increase in effort, as they will naturally respond in an appropriate way to each part of the t-captcha. There is no chance of a human applying the wrong type of 'solving heuristic' to some part of the t-captcha, as human captcha-solving abilities derive from human general intelligence and audiovisual pattern-matching capabilities rather than explicit algorithms. Further, a human encountering the captcha only once, or for the first time, has no reason to expect any one ordering more than another. Consequently, the CIR under Strategy 3 when compared to Strategy 2 is '1'

in the best case for an extremely effective attacker (i.e. no improvement), and 't-factorial' in the worst case, i.e. a newly implemented or naive attack program. The CIR relative to a single base captcha challenge will be:

$$CIR = t! \frac{h^{(t-1)}}{c_2.c_3.c_4...c_t} \qquad \text{(initial/worst case, attacker)} \qquad (4)$$

$$CIR = \frac{h^{(t-1)}}{c_2.c_3.c_4...c_t} \qquad \text{(best case, attacker)} \qquad (5)$$

For example, in the case of a 4-part t-captcha, the CIR is raised by a further factor of 12 (4x3x2x1) relative to Strategy 2, until the attacker can reliably identify the ordering of the parts of the captcha. Strategy 3 should never produce a weaker captcha system than Strategy 2.

4.5 Standardisation and Implicit Captchas

I have suggested that t-captchas should ideally be presented in such a way that each type of sub-captcha cannot be trivially distinguished. How might this be achieved in practice?

Standards for captchas: It would be easier to replace failing captchas, and implement t-captchas in the real world, if there were agreed standards for certain types of captcha. For example: graphical captchas could perhaps be constrained to bitmaps of a particular width, height, expected screen resolution, colour palette and user input method (e.g. touchscreen/cursor position, gestures, typed characters). Any necessary instructions could be communicated as part of the captcha image itself, to prevent attackers from trivially identifying the type of captcha from surrounding HTML-based instructions.

However, invariants might still be present in the image which could allow attackers to easily categorise the captcha. For example, if the instructions

were always in a particular corner of the image, or if they always used the same text and font within the image, the lack of variation would give away the captcha's type to a determined attacker. This leads to a second suggestion.

The Implicit Captcha: The captcha could 'imply' how it should be approached and deliberately exclude explicit instructions. Humans have a remarkable ability to impose order when no direct instructions are given. The idea of 'forcing understanding by failure to communicate' is excellently described with examples in "The Strategy of Conflict" (Schelling, 1980). Schelling describes graphical, textual and mathematical challenges which seem to yield inherently coordinated responses when given to many human subjects. His examples include:

> "Write some positive number. If you all write the same number, you win."[6] (Schelling, 1980)
>
> "Name an amount of money. If you all name the same amount, you can have as much as you named."[7] (Schelling, 1980)

A trivial example of an Implicit Captcha might present an image on the screen containing a classical distorted text captcha alongside an on-screen keyboard. An 'obvious' action to be taken might be to click on the keyboard part of the image, according to the distorted letters present. A more complex example might remove the explicit instructions from a system such as 'Captcha Zoo' (Lin et al., 2011), as shown in Figure 2.

In Captcha Zoo, the user is expected to identify randomly coloured and oriented overlapping 3D animals in a visual scene. The original CAPTCHA Zoo challenge also included text instructions. However, humans may instinctively realise that when 3 horses and 12 dogs are present in a challenge image, they should click on each of the horses, because it is the simplest intelligent pattern of action that could be taken.

At the extreme end, highly-challenging visual Implicit Captchas might draw inspiration from graphical puzzle adventures such as 'Myst' (Wikipedia, 2011b) and 'Braid' (Wikipedia, 2011a). In these games, the user must first interactively explore the puzzle and its interface, and develop an understanding of its nature and dynamics, without instructions to guide them. When

Figure 2: Example of Captcha Zoo without instructions, from (Lin et al., 2011). Red circles have been added to the CAPTCHA image to highlight the areas of the screen containing the solution.

the user finally understands the nature of the puzzle, they can begin to form their solution.

It is also possible to imagine examples of audio-based Implicit Captchas. In the context of a puzzle, if you were to hear 4 Fs, followed by 3 Es, then 2 Ds, then a gap, how might you naturally respond? I believe many people would (after a moment's thought) try to imitate the sound of a single C - regardless of whether they had heard the letters 'F,E,D' being spoken or the notes 'F,E,D' being played. An extremely challenging audio Implicit Captcha might involve listening and responding 'naturally' to the words of a randomly generated artificial language based around real human language structures.

The advantage of Implicit Captchas is that they make the problem of 'captcha-type characterisation' for attackers as difficult as the problem of solving a captcha itself. Consequently, such an approach should allow the quantitative and qualitative benefits of the t-captcha strategy to be realised. However, further study with real-world users would be necessary to identify

captcha-like situations where humans are able to quickly, naturally, and consistently discover the path to a solution.

5 Author recommendations

- If you have an important Web-based resource to protect, yet only one captcha system in place, then consider employing Strategy 1 (m-captcha) immediately to achieve a substantial improvement in the effectiveness of your captcha system. This ensures that any attack type with a success rate of less than 10% becomes significantly less effective; and all attacks below human-equivalent level are noticeable.

- If you are able to present a combination of several randomly arranged captcha instances of different types (Strategy 3, the t-captcha), then consider doing so; it will make your combined captcha more robust even against determined attackers.

- Several simple captchas posed simultaneously may produce a better user experience than a single captcha that is inconvenient for humans to solve, while still representing a significant challenge for attackers.

- If you are using an s-captcha, track the success rate of each part of the captcha for known human users independently, and respond to it.

- Consider scaling the number of captcha instances (m or t) that are presented simultaneously, to suit the value of the resource being protected. Users may be sympathetic to an adaptive strategy that inconveniences them only as much as is absolutely necessary given the circumstances.

6 Future work

From the perspective of defenders, future work in this area might involve verifying the psychological response of users to an increased number and variety of captchas, particularly in the absence of instructions i.e. Implicit Captchas. It would be worthwhile to determine if typical Internet users prefer

the present trend of increasingly obfuscated and hard-to-interpret captchas, or if multiple simple captchas would be more welcome.

From the perspective of attackers, future work might involve building systems that can identify captcha features so as to allow automatic classification of the types of captcha in a randomly ordered t-captcha. A computer could then automatically select attacks from an appropriate 'toolbox' against a presented t-captcha.

7 Conclusions

This paper has outlined methods and analysis showing that any captcha system may be improved by presenting multiple instances of different types of captcha challenges.

The analysis of this paper suggests that successful attacks on captchas might be postponed by combining several distinct types of captcha challenges together, in a random order (referred to here as a 'randomly-ordered t-captcha'). Increasing the number of captcha challenges causes the difference in success rates between humans and computers to become exaggerated. Generally, humans will experience an approximately linear increase in effort required as the number of captcha instances increases, whereas automated attackers will experience an approximately exponential increase in effort per success until they achieve near-human pass rates. This benefits both prevention and detection of automated attacks, and assists in improving the lifespan of captcha systems.

Attackers can only silently defeat a t-captcha system by reliably achieving human-level success against all of the types of captcha instance presented in the t-captcha. The challenge posed by a randomly-ordered t-captcha containing multiple Implicit Captchas would be well beyond the capabilities of any research-based attack system published to date in the field of captcha research.

The author hopes that the approach of this paper will provoke thought towards the development of generic techniques that enhance existing real-world captchas rather than the continued proposal of new captcha systems that never reach commercial deployment.

Notes

[1]To improve the presentation of the text, CAPTCHA has been written 'captcha' throughout the text of this paper.

[2]Interestingly, new research in (Bursztein et al., 2010) suggests that while many captchas have success rates around 90-95%, some recent captcha designs in use at commercial sites now have human success rates below 40%, in particular, audio captchas.

[3]To the author's knowledge, there are almost no global commercial Web services undertaking this strategy at present. The closest examples are the Facebook.com captcha system (which presents two dictionary text phrases side by side as a single captcha) and reCAPTCHA (which presents a text captcha instance alongside an OCR work unit) (von Ahn et al., 2008).

[4]However, 'toolboxes' exist that can attack several distinct captcha systems (Hocevar, 2004).

[5]Captcha longevity is becoming an increasingly important issue; even the best modern commercial captchas are now being broken within mere weeks or months by determined malicious users (Websense Security Labs, 2009).

[6]Forty percent of people asked chose '1', in the absence of any other information to guide their choice. It is remarkable that from an infinitely large selection of possible numbers, so many people naturally come to the same choice.

[7]Twelve of 41 people asked instinctively chose $1,000,000; only three in 41 people surveyed chose a number which was not a power of 10.

References

Baird, H. and Bentley, J. (2005). Implicit CAPTCHAs. In *Proceedings of IS& T/SPIE Document Recognition & Retrieval XII Conf., San Jose, CA*, pages 191–196. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.87.1071&rep=rep1&type=pdf.

Baird, H. S., Moll, M. A., and Wang, S.-Y. (2005). A highly legible CAPTCHA that resists segmentation attacks. *Human Interactive Proofs (LNCS)*, 3517/2005:27–41. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.87.8850.

Barr, J. and Cabrera, L. F. (2006). AI Gets a Brain. *ACM Queue*, 4(4):24–29. Available from: http://queue.acm.org/detail.cfm?id=1142067.

Bursztein, E., Bauxis, R., Paskov, H., Perito, D., Fabry, C., and Mitchell, J. C. (2011). The failure of noise-based non-continuous audio CAPTCHAs. In *Security and Privacy (SP)*. Available from: http://129.81.170.14/~rbeauxis/audio_captcha.pdf.

Bursztein, E. and Bethard, S. (2009). Decaptcha: Breaking 75% of ebay audio CAPTCHAs. In *Proceedings of WOOT '09, Montreal, Canada.* Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.160.2013.

Bursztein, E., Bethard, S., Fabry, C., Mitchell, J. C., and Jurafsky, D. (2010). How good are humans at solving CAPTCHAs? a large scale evaluation. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, SP '10, pages 399–413, Washington, DC, USA. IEEE Computer Society. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.164.7848.

Carnegie-Mellon University. The CAPTCHA project 'Completely automatic public turing test to tell computers and humans apart' [online]. (2000-2009) [cited Saturday, 4th June 2011]. Available from: http://www.captcha.net.

Chellapilla, K., Larson, K., Simard, P. Y., and Czerwinski, M. (2005). Building segmentation based human-friendly human interactive proofs (HIPs). In *In Proceedings of the Second International Workshop on Human Interactive Proofs*, pages 1–26. Springer-Verlag. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.90.3911.

Chellapilla, K. and Simard, P. Y. (2005). Using machine learning to break visual human interaction proofs (HIPs). *Advances in Neural Information Processing Systems*, 17:265–272. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.125.2105.

Chow, R., Golle, P., Jakobsson, M., Wang, L., and Wang, X. (2008). Making CAPTCHAs clickable. In *HotMobile '08: Proceedings of the 9th workshop on Mobile computing systems and applications*, pages 91–94, New York, NY, USA. ACM. Available from: http://crypto.stanford.edu/~pgolle/papers/click.pdf.

Coates, A. L., Baird, H. S., and Fateman, R. J. (2001). Pessimal print: A reverse turing test. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR*, pages 1154–1158. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.7923&rep=rep1&type=pdf.

Golle, P. (2008). Machine learning attacks against the asirra CAPTCHA. In *CCS '08: Proceedings of the 15th ACM conference on Computer and communications security*, pages 535–542, New York, NY, USA. ACM. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.140.2197.

Gossweiler, R., Kamvar, M., and Baluja, S. (2009). What's up CAPTCHA? a CAPTCHA based on image orientation. In *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain.*, pages 841–850. Available from: http://www.richgossweiler.com/projects/rotcaptcha/rotcaptcha.pdf.

Hocevar, S. PWNtcha - CAPTCHA decoder. [online]. (2004) [cited Saturday, 4th June 2011]. Available from: http://caca.zoy.org/wiki/PWNtcha.

Huang, S.-Y., Lee, Y.-K., Bell, G., and Ou, Z.-H. (2008). A projection-based segmentation algorithm for breaking MSN and YAHOO CAPTCHAs. In *Proceedings of the 2008 International Conference of Signal and Image Engineering (ICSIE'08), London, UK.* Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.148.8730.

Lin, R., Huang, S.-Y., Bell, G., and Lee, Y.-K. (2011). A new captcha interface design for mobile devices. In *Australasian User Interface Conference, Australasian Computer Science Week (ACSW2011), Perth, Australia, and to appear in the ACM digital library.* Available from: http://graemebell.net.

Microsoft (2007). Asirra: a CAPTCHA that exploits interest-aligned manual image categorization. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 366 – 374. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.187.8153.

Misra, D. and Gaj, K. (2006). Face recognition CAPTCHAs. In *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, AICT/ICIW*. Available from: http://portal.acm.org/citation.cfm?id=1116289.

Mori, G. and Malik, J. (2003). Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:134. Available from: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.12.9732.

Moy, G., Jones, N., Harkless, C., and Potter, R. (2004). Distortion estimation techniques in solving visual CAPTCHAs. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:23–28. Available from: www.cse.psu.edu/~datta/imagination/moy.pdf.

Pope, C. and Kaur, K. (2005). Is it human or computer? Defending e-commerce with CAPTCHAs. *IT Professional*, 7(2):43–49. Available from: http://www.wiphala.net/research/proposal/papers/is_it_human_or_computer._defending_e-commerce_with_captchas.pdf.

Protalinski, E. Gone in 60 seconds: Spambot cracks live hotmail CAPTCHA [online]. (2008) [cited Saturday, 4th June 2011]. Available from: http://arstechnica.com/security/news/2008/04/gone-in-60-seconds-spambot-cracks-livehotmail-captcha.ars.

Reimer, J. The most adorable spambot killer ever [online]. (2006) [cited Saturday, 4th June 2011]. Available from: http://arstechnica.com/old/content/2006/04/6554.ars.

Ritendra Datta, J. L. and Wang, J. Z. (2006). IMAGINATION: A robust image-based CAPTCHA generation system. In *Proceedings of the ACM Multimedia Conference*, pages 331–334. Available from: http://academic.research.microsoft.com/Publication/1764981/.

Schelling, T. (1980). *The Strategy of Conflict*, pages 53–80. Harvard University Press. Available from: http://en.wikipedia.org/wiki/Thomas_Schelling#The_Strategy_of_Conflict.

Shirali-Shahreza, M. and Shirali-Shahreza, S. (2007a). Collage CAPTCHA. In *Proceedings of the 9th International Symposium on*

Signal Processing and Its Applications, 2007. ISSPA 2007. Available from: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=4555329&isnumber=4555273.

Shirali-Shahreza, M. and Shirali-Shahreza, S. (2007b). Question-based CAPTCHA. In *ICCIMA '07: Proceedings of the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, pages 54–58, Washington, DC, USA. IEEE Computer Society. Available from: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=4426449.

The Economist Newspaper Ltd. (2008). Playing for profit. *The Economist*, August 26th. Available from: http://www.economist.com/businessfinance/displayStory.cfm?story_id=11997115 [cited Saturday, 4th June 2011].

Turing, A. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460. Available from: http://www.jstor.org/pss/2251299.

Vaughan-Nichols, S. J. CAPTCHA security - more worthless by the day [online]. (2008) [cited Saturday, 4th June 2011]. Available from: http://www.techworld.com/security/features/index.cfm?featureid=4168.

von Ahn, L., Blum, M., and Langford, J. (2003). CAPTCHA: Using hard AI problems for security. In *Proceedings of Eurocrypt*, pages 294–311. Springer-Verlag. Available from: www.captcha.net/captcha_crypt.pdf.

von Ahn, L., Maurer, B., Mcmillen, C., Abraham, D., and Blum, M. (2008). reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321(5895):1465–8. Available from: http://www.cs.cmu.edu/~biglou/reCAPTCHA_Science.pdf.

Websense Security Labs. Google's 'blogger' under attack by streamlined Anti-CAPTCHA operations for spam [online]. (2008) [cited Saturday, 4th June 2011]. Available from: http://securitylabs.websense.com/content/Blogs/3073.aspx.

Websense Security Labs. Google's CAPTCHA busted in recent spammer tactics [online]. (2008) [cited Saturday, 4th June 2011]. Available from: http://securitylabs.websense.com/content/Blogs/2919.aspx.

Websense Security Labs. Microsoft live hotmail under attack by streamlined Anti-CAPTCHA and mass-mailing operations [online]. (2008) [cited Saturday, 4th June 2011]. Available from: http://securitylabs.websense.com/content/Blogs/3063.aspx.

Websense Security Labs. Microsoft's CAPTCHA revolutions busted by spammers - again and again [online]. (2009) [cited Saturday, 4th June 2011]. Available from: http://securitylabs.websense.com/content/Blogs/3306.aspx.

Wikipedia. Braid (video game) [online]. (2011) [cited 9th June 2011]. Available from: http://en.wikipedia.org/wiki/Braid_(video_game).

Wikipedia. Myst (video game) [online]. (2011) [cited 9th June 2011]. Available from: http://en.wikipedia.org/wiki/Myst.

Yan, J. and Ahmad, A. S. E. (2007). Breaking visual CAPTCHAs with naive pattern recognition algorithms. In *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*, pages 279–291. Available from: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.97.2434.

Yan, J. and Ahmad, A. S. E. (2008a). A low-cost attack on a Microsoft CAPTCHA. In *Proceedings of the 15th ACM conference on Computer and communications security*. Available from: http://homepages.cs.ncl.ac.uk/jeff.yan/msn_draft.pdf.

Yan, J. and Ahmad, A. S. E. (2008b). Usability of CAPTCHAs or usability isssues in CAPTCHA design. In *Proceedings of the 4th symposium on Usable privacy and security*, volume 337 of *ACM International Conference Proceeding Series*. Available from: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.140.9396.

Thanks

The author is grateful to C Bell and S Walker for proofreading this paper and offering feedback.

About the author

Graeme Bell is affiliated with the Applied Artificial Intelligence research group at Murdoch University, Australia. He holds a Ph.D. in computer science from the University of St. Andrews, U.K. He was the top science graduate from the University of St. Andrews in 2001 and also the winner of the 2001 Scottish Young Software Engineer of the Year award. His research interests include artificial intelligence, bioinformatics, robotics, image processing, steganography, and the Internet.