CAPTCHA as a Web Security Tool

UDIT AGARWAL
Astt. Prof (CS&IT), RBMI Group of Institutes, Bareilly (U.P.)

ABSTRACT

CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) are widespread security measures on the World Wide Web that prevent automated programs from abusing online services. They do so by asking humans to perform a task that computers cannot yet perform, such as deciphering distorted characters. CAPTCHAs provide a method for automatically distinguishing a human from a computer program, and therefore can protect Web services from abuse by so-called “bots.” Most CAPTCHA consists of distorted images, usually text, for which a user must provide some description. Web sites are increasingly under attack from automated scripts. In many cases, sites could deter automated attacks if they could distinguish human users from machine users. Artificial Intelligence experts identify a category of security controls called “CAPTCHAs” which can help Web sites distinguish between human and machine users by posing a problem that is easy for humans to solve, but difficult for machines to solve. This paper discusses CAPTCHA’s variety of modes, implementations, methods of deployment, relative advantages and disadvantages, effectiveness against various attacks, role in security strategies, accessibility considerations, known weaknesses, pitfalls to avoid, and practicality in Web applications.

Keywords: CAPTCHA, Web Security, spam, distortions.

1. INTRODUCTION

A captcha (an acronym for Completely Automated Public Turing test to tell Computers and Humans Apart, trademarked by Carnegie Mellon University) is a challenge-response test frequently used by internet services in order to verify that the user is actually a human rather than a computer program. The term was coined in 2000 by Luis von Ahn, Manuel Blum, Nicholas J. Hopper of Carnegie Mellon University, and John Langford of IBM. The P for Public means that the code and the data used by a CAPTCHA should be publicly available. This is not an open source requirement, but a security guarantee: it should be difficult for someone to write a computer program that can pass the tests generated by a CAPTCHA even if they know exactly how the CAPTCHA works (the only hidden information is a small amount of randomness utilized to generate the tests). The T for “Turing Test to Tell” is because CAPTCHAs are like Turing Tests. In the original Turing Test, a human judge was allowed to ask a series of questions to two players, one of which was a computer and the other a human. Both players pretended to be the human, and the judge had to distinguish between them. CAPTCHAs are similar to the Turing Test in that they distinguish humans from computers, but they differ in that the judge is now a computer. A CAPTCHA is an Automated Turing Test. We deliberately avoid using the term Reverse Turing Test (or even worse, RTT) because it can be misleading- Reverse Turing Test has been used to refer to a form of the Turing Test in which both players pretend to be a computer.

2. Why use CAPTCHA?

The main goal of CAPTCHA is preventing automated software or bots from performing certain actions on a site. Sure, the automated code may access the site, but you don’t want it posting comments (spam), creating user accounts, or placing orders. There are a variety of situations targeted by CAPTCHA, which include the following:

• Site registration: A site can limit access to its registration system or page by using CAPTCHA as a gate to accessing it.
• Comments: Sites that allow comments often have generated content that is obviously not entered by a user. For instance, sites such as Blogger use CAPTCHA to control access to posting comments.
Online Polls: CAPTCHA can help maintain the integrity of a poll by letting only humans participate.

Passwords: A common way to attack a site is via a dictionary attack where brute force is applied to a password field in an attempt to guess it. CAPTCHA can be used to control access to the password field, thus leaving bots in the cold.

While providing an audio CAPTCHA allows blind users to read the text, it still excludes those who are both visually and hearing impaired.

The use of CAPTCHA thus excludes a large number of individuals from using significant subsets of such common Web-based services as PayPal, GMail, Orkut, Yahoo!, many forum and weblog systems, etc. Even for perfectly sighted individuals, new generations of CAPTCHAs, designed to overcome sophisticated recognition software, can be very hard or impossible to read. Even some of the demo CAPTCHAs at the software sites listed below are indecipherable to many if not all humans.

3. Varieties of CAPTCHAs available
CAPTCHAs further differ from the original Turing Test in that they can be based on a variety of sensory abilities. The original Turing Test was conversational the judge was only allowed to ask questions over a text terminal. In the case of a CAPTCHA, the computer judge can ask any question that can be transmitted over a computer network. Basically we can specify three types of CAPTCHAs

1. Text based
   - Gimpy, ez-gimpy
   - Gimpy-r, Google CAPTCHA
   - Simard’s HIP (MSN)

2. Graphic based
   - Bongo
   - Pix

3. Audio based

GIMPY is one of the many CAPTCHAs based on the difficulty of reading distorted text.
GIMPY works by selecting seven words out of a dictionary and rendering a distorted image containing the words (as shown in Figure).

GIMPY then presents a test to its user, which consists of the distorted image and the directions: "type three words appearing in the image." Given the types of distortions that GIMPY uses, most humans can read three words from the distorted image, but current computer programs can’t. The majority of CAPTCHAs used on the Web today are similar to GIMPY in that they rely on the difficulty of optical character recognition (the difficulty of reading distorted text).

Another example of a CAPTCHA is the program we call BONGO. BONGO is named after M.M. Bongard, who published a book of pattern recognition problems in the 1970s. BONGO asks the user to solve a visual pattern recognition problem. It displays two series of blocks, the left and the right. The blocks in the left series differ from those in the right, and the user must find the characteristic that sets them apart. A possible left and right series is shown in Figure.

After seeing the two series of blocks, the user is presented with a single block and is asked to determine whether this block belongs to the left series or to the right. The user passes the test if he or she correctly determines the side to which the block belongs.

Audio Based CAPTCHAs
1. Pick a word or a sequence of numbers at random
2. Render them into an audio clip using a TTS software
3. Distort the audio clip
4. Ask the user to identify and type the word or numbers.
4. CAPTCHA as a Web Security

Artificial Intelligence (AI) experts propose CAPTCHA as a category of security controls to help Web sites distinguish between human and machine users. Turing (1950) proposes a test for AI in which a computer must fool a panel of humans into believing the machine is human. Blum, Ahn, and Langford (2000) propose a class of ATTs called a Human Interactive Proof (HIP), which Hopper (2001) describes as a protocol "that allows a human to prove something to a computer". Hopper and Blum (2001) propose a HIP called a Secure Human Identification Protocol, or HUMAN OID, in which a computer must verify a human's membership in a group without requiring a password, biometric data, electronic key, or any other physical evidence. A HUMANOID test must also remain effective even when others know and witness the authentication process. Blum, Ahn, and Langford (2000) further propose a more specific form of HIP called a Completely Automated Public Turing Test to Tell Humans and Computers Apart (CAPTCHA) in which the computer must be able to "generate and grade" a test "that most humans can pass", but "current computer programs cannot pass". Blum, Ahn, Hopper, and Langford implemented a CAPTCHA called EZ-Gimpy, which Manber deployed to deter spam attacks launched from Yahoo e-mail addresses registered in bulk (Spice, 2001).

Web sites can deploy CAPTCHAs either by installing turnkey CAPTCHA software, by programming custom CAPTCHA software, or by subscribing to a remote CAPTCHA service. Installing existing CAPTCHA software comes with the same considerations as installing any packaged security control, like a firewall or antivirus. To be effective, the software makers have to update the software frequently to patch vulnerabilities in previous versions and to combat cracks. There are dozens of strong turnkey options among thousands of weak options. The two primary disadvantages of subscribing to a CAPTCHA service are that another potential avenue of attack results from passwords or keys exchanged over the network, and that all remote CAPTCHA services require one organization to trust another with the power to abuse the site. Assuming other security measures are in place and the CAPTCHA is only being used to keep scripts out of an area already available to the human public, the potential damage from a rogue employee would not seem to be greater than the potential damage from any other member of the public.

5. The Rise and Fall of CAPTCHA

CAPTCHA was created in 2000 by researchers at Carnegie Mellon University, and by 2007, the technology was being used almost everywhere on the Web. Unfortunately, beginning in early 2008, crackers started getting the better of the CAPTCHA systems. In short order, Yahoo Mail's, Gmail's and Hotmail's CAPTCHA defenses were cracked.

Then, adding insult to injury, the crackers started releasing their work in the form of do-it-yourself CAPTCHA cracking software that anyone could use. For example, a program called CL Auto Posting Tool attempts to post bogus ads to Craigslist while automatically overcoming Craigslist's antispam protections. These programs work by using OCR (optical character recognition) software to try to make sense of CAPTCHA's disguised text. If they fail, they try again. They take advantage of the fact that some CAPTCHA systems don't automatically give users a new CAPTCHA image to puzzle out. Instead, they'll let you, or a cracker program, keep working at the hidden text until it's solved.

Because they are clearly insecure, CAPTCHA systems that allow unlimited or multiple attempts are becoming uncommon. Still, today's automated bots are capable of breaking even those systems that make users respond to a new CAPTCHA image after the first or second unsuccessful attempt. (On average, of course, the bots' efforts are less likely to work at one-try CAPTCHA systems.)

Another way to crack a badly designed CAPTCHA program is to reuse the session identification URL of a solved CAPTCHA image. In this case, either the cracker, or more likely a cracking program, first gets the right answer to a CAPTCHA. It then reconnects to the Web site with a URL containing the solved session identification information with a new username.

6. Conclusion

In conclusion, care must be taken when installing a CAPTCHA to avoid causing new
problems, particularly with regard to accessibility. Federal regulations and professional ethics require Web sites to remain accessible to users with disabilities, but many CAPTCHA implementations result in denial of service to people with visual impairments, auditory impairments, or both. In addition, some CAPTCHA software opens new security holes. Web professionals should carefully test a CAPTCHA’s strengths and weaknesses before integrating it into their sites, but if properly implemented and deployed, many CAPTCHAs can be effective controls against the most common forms of automated attacks.

7. References

10. H. Baird, M. Luk. Protecting Websites with Reading-Based CAPTCHAs.