UserTrust

Allen Madsen

Computer Science

B. Thomas Golisano College of Computing and Information Sciences

August 27, 2010

Abstract

TODO: Abstract

I. Introduction

A Completely Automated Public Turing test to tell Computers and Humans Apart or CAPTCHA is a test which uses a difficult artificial intelligence problem to distinguish between a human and a computer. The most common form for a CAPTCHA is an image with obscured or transformed text that the user must decipher. This type of CAPTCHA is effective because it is difficult for computers to recognize letters that are distorted, whereas humans find it much easier. As a result, this sort of test is useful for defending various types of systems from automated usage by computers [2].

For example consider the website Reddit, which allows its users to submit links to content they like and vote for their favorite links. Links that have a high number of votes are promoted to the front page and receive a lot of views. A malicious user would see this behavior and attempt to take advantage of it by promoting her links to the front page. Since Reddit has a limit of one vote per person the malicious user needs access to a lot of accounts to effectively boost her link to the front page. Creating the necessary amount of accounts and voting manually would be unreasonable for the malicious user to do. As a result, the malicious user would like to automate account registration and voting. However, by placing a CAPTCHA on the account registration form, Reddit prevents the malicious user from automating the creation of accounts.

For other sites, restricting account registration with a CAPTCHA is not enough to prevent abuse. As an example consider MySpace. Once a malicious user manually creates an account on MySpace she can automate the sending of friend requests. With a large amount of friend requests the malicious user is bound to generate a large set of friends even if most of the friend requests are rejected. Now the malicious user can automate the posting of comments on peoples pages. Since a page is visible by all of the page owners friends the comment gains large distribution even if few people accept friend requests. In order to prevent this type of automation MySpace could use a CAPTCHA on both the friend request and commenting form.

In the cases of Reddit and MySpace CAPTCHAs are useful. However, the actions that need to be protected from automation on MySpace are much more extensive than those needed on

Reddit. This is because on Reddit links are not treated equally, whereas on MySpace each user is. Placing a CAPTCHA on these two actions would help MySpace solve the automation that is possible on their site. Using a CAPTCHA is problematic though, because it is cumbersome for a user to solve [1], [3][2, 3]. Combining the extra burden with the fact that these two actions occur very frequently and it becomes desirable to relieve the users of this burden. This creates a conflict between the security and the user experience on the site.

Adopting either of these views represents two extremes on a scale. They can be stated as always showing and never showing a CAPTCHA on a form. It would be nice if a middle ground could be found where a CAPTCHA is only shown sometimes, because this would allow a balance between security and the user experience. A nave approach would be to show a CAPTCHA randomly. However, this method is insufficient because a computer could be designed to refresh a page with a form until there is no CAPTCHA. This method is also problematic because it still treats the normal user the same as the automated user. The ideal approach would be to always show a CAPTCHA to automated users and never show one to normal users. In order to do this, more information is needed to differentiate between the two types of users.

The type of information that would be useful is information that can be used to predict the future performance of a user on a CAPTCHA. If it is known that a user will likely pass a CAPTCHA if shown one, then it is a reasonable conclusion to not show one. A users history of previous performance on CAPTCHAs is a good indicator of how they will perform in the future. From this history can be defined as:

$H = (T_1, T_2, \ldots, T_n)$, where $n \geq 1$

$$T_k = \begin{cases} 1, & \text{if } CAPTCHA_k \text{ passed} \\ 0, & \text{if } CAPTCHA_k \text{ failed} \end{cases}$$

, where $1 \leq k \leq n$

There is one problem inherent with a CAPTCHA that needs to be addressed, however. Passing a CAPTCHA provides reasonable certainty that the user is not automated, however, failing a CAPTCHA does not necessarily mean a user is automated. It is not uncommon for a normal user to incorrectly fill out a CAPTCHA. In this case a site would typically retest the user with another CAPTCHA. This lends itself to the idea of a session where a user can attempt to pass a CAPTCHA multiple times until they succeed or give up. Thus a session is defined as:

$S = (T_1, T_2, \ldots, T_n)$, where $n \geq 1$

$T_k = 0$, where $n > 1$ and $1 \leq k < n$

$$T_n = \begin{cases} 1, & \text{if } CAPTCHA_k \text{ passed} \\ 0, & \text{if } CAPTCHA_k \text{ failed} \end{cases}$$

Before session is incorporated into history it is important to decide how a sessions value is determined. A very simplistic approach would be to use the value of T_n for each session. This approach neglects to account for the number of tries a user performs before they succeed. This information is important because a normal user should complete a CAPTCHA correctly in relatively few tries, whereas an automated user may be able to complete a CAPTCHA correctly after many tries. Another approach would be to use an average; however, this may not punish the automated user enough. Instead, a weighted average can be used, where p^{k-1} is used as the weight and $0 < p \leq 1$. Thus we get the equation:

$$SV = \sum_{k=1}^{n} T_k * \frac{p^{k-1}}{\sum_{k=1}^{n} p^{k-1}}$$

Since all $T_k = 0$ where $n > 1$ and $1 \leq k < n$, the above equation can be simplified to:

$$SV = T_n * \frac{p^{n-1}}{\sum_{k=1}^{n} p^{k-1}}$$

II. Conclusion

TODO: Conclusion

REFERENCES

[1] N. Ben-Asher, J. Meyer, S. Moller, and R. Englert. An experimental system for studying the tradeoff between usability and security. In *Availability, Reliability and Security, 2009. ARES '09. International Conference on*, pages 882 –887, 2009.

[2] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford. CAPTCHA: using hard AI problems for security. In *Advances in cryptology—EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Comput. Sci.*, pages 294–311. Springer, Berlin, 2003.

[3] J. Yan and A. S. El Ahmad. Usability of captchas or usability issues in captcha design. In *Proceedings of the 4th symposium on Usable privacy and security*, SOUPS '08, pages 44–52, New York, NY, USA, 2008. ACM.