Secure Passwords against Trojan Theft using CAPTCHA Keyboard

Abstract

Password is more commonly used than any other authentication methods. However, the attacks against passwords have never stopped. Protocols based on CAPTCHA (Telling Humans and Computers Apart Automatically) have been proposed to secure passwords against the widely spread dictionary attack. A typical solution is to require that a CAPTCHA test must pass before the user can input password. CAPTCHA tells machine and human apart and only human can pass the test. Therefore, it is almost infeasible for a hacker program (machine) to iteratively guess the password using dictionary words (i.e., arbitrary combinations of characters). However, current CAPTCHA based protocols are still vulnerable to password theft of Trojan horse programs. In this paper, a method is presented to secure user passwords against Trojan programs using CAPTCHA Keyboard. When password is needed, a virtual keyboard is displayed on the monitor and user clicks on the keyboard instead of typing the real keyboard. The virtual keyboard is specially designed: i) all the characters are CAPTCHAlized, i.e., they are distorted handwritten characters with noises which only human can recognize, ii) the order of characters are randomly generated, iii)when a character is clicked, the location of the character is used as the representation of the character and transmitted to server side. The benefit of this novel protocol is obvious: even the Trojan program steals what the user clicks or the transmission message is intercepted, the real password won't be revealed. The paper describes the implementation of the protocol and analyzes it's security.

1 Introduction

Passwords are so widely spread that almost everyone has to use for authentication purposes. Currently, most passwords are sequences of textual characters. The main reason behind this is their convenience and practicality in implementation. Although more secure authentication methods such as smartcards and biometrics (signature, fingerprint etc) can be used, none of them beats password in consumer markets. So far, smartcards can only be used in on-site authentication instead of web. Signature is the most popular biometrics and its major usage is to confirm payments but no signature verification is conducted in real time. Fingerprint is another popular biometrics and has begun to be used as login password in some modern computers like Toshiba Tablet laptops, but still faraway for Web account password unless most consumers' computers have built-in fingerprint modules. In the foreseeable future, password will continue to be the main authentication method, especially in the Web.

On the other side, it is well known that the user password is vulnerable to attacks. A large fraction of users choose passwords from a small domain (users name, , birthdays, initials, account name, and other relevant personal information). A small password domain enables hackers to login to accounts by automatically trying all possible passwords, until correct one is found. This attack is called dictionary attack. Almost all big websites have been attacked by dictionary attacks according to reports [9]. Empirical study also showed that 21% of 15000 passwords can be cracked cracked in one week using only dictionary attack [7].

Figure 1. An example of CAPTCHA.

Fortunately, CAPTCHA (Telling Humans and Computers Apart Automatically) is a good solution against dictionary attack [1]. A CAPTCHA is a program generating and grading tests that humans can pass but current computer programs cannot. A typical example is that humans can read distorted text as the one shown in Figure 1, but no current computer program can. The direct application of CAPTCHA is to protect websites from bots. CAPTCHA has been widely applied in major websites such as PayPal, Yahoo! and AltaVista.

The idea to use CAPTCHAs to prevent dictionary attacks in password systems is simple: prevent a bot from being able to iterate through the entire dictionary of passwords by requiring it to pass a CAPTCHA test after a certain number of unsuccessful logins [9]. Because it is easy for a program to iterate endlessly but not feasible for a human to repeat even hundreds of times. That means, if "someone" repeatedly try unsuccessful login passwords, it is potentially a "bot" instead of a human.

Unfortunately, the CAPTCHA can not prevent Phishing attack, which installs a Trojan horse program in user's computer to steal user's passwords and privacy information. A lot of Phishing attacks against banking accounts have been reported [6, 8]. The "Trojan horse" program, known as Magic Lantern, could be sent via seemingly innocent emails or be downloaded unconsciously from infected websites. After the program has installed itself, it would capture the passwords when user logins web accounts and send them to hackers. While anti-virus software may prevent Phishing attacks by detecting and removing Trojan programs to some degree, the Trojan horses are evolving just like virus. In this paper, a novel method is presented to prevent Trojan thefts.

2 Secure password against Trojan theft

Most passwords are strings of characters. When asked a password, user inputs it by typing the characters on the keyboard. The characters visually hidden on the monitor by wildcats such as '*'s or '·'s to prevent Shoulder-surfing (i.e., watching over shoulder). However, the plain text are stored in the main memory of the computer temporarily before submission to the server side. Suppose both the server side and the transmission channels are encrypted, then the client side is the most weak point raising security concerns. The plain texts are easy to expose to the Trojan programs which are actively watching in operating system. From the keyboard to the main memory, the content is always plain text, where there is no room for encryption in both hardware and software. So, how can we fix the weakness of typing plain textual passwords?

To address this concern, graphical passwords are suggested [2]. There are several schemes proposed for graphical passwords which claim to be effective alternative to improve the security of authentication systems [3, 4, 10]. However, all these schemes suffer from common weaknesses. Let's use the Déjà vu scheme as example [4], in which images are generated to substitute characters for user to choose from. The advantages are: i) harder to be stolen; ii)harder to be forgotten, since human brain memorize images longer than text. However, the disadvantages are also obvious: i)implementation cost is high and the computation of image generation is intensive, ii) not possible to hide with wildcats against Shoulder-surfing.

We fix the weakness of the textual passwords in more easy and reliable way, namely "CAPTCHA Keyboard": the regular textual password remain the same but the way user inputs password is secured. Instead of typing keyboard, a virtual keyboard is displayed on the screen and user clicks keys using mouse. The location sequence of the clicks is transmitted to server side for verification. The virtual keyboard is made different from regular keyboard: i)every character is distorted with noises, generated using CAPTCHA;ii) the order of keys on the

Figure 2. Example of Virtual Keyboard that can be generated by java script.

keyboard is random.

The characters on the CAPTCHA keyboard are distorted but still recognizable by human. The reason is same as why we use CAPTCHA: no bot can recognize them. In addition, the order of the characters on keyboard must be randomly disordered, otherwise, the bot can easily determine the characters from the user's click locations given the fixed keyboard.

Using traditional textual password, user does not have to change their habits in choosing passwords (changing habit itself is resistant for a majority of people). Given the existing popularity of virtual keyboard and CAPTCHA, our password input method is acceptable in practice in contrast to the graphic password which has inherent weakness in acceptance.

The CAPTCHA Keyboard will not cost much more than regular CAPTCHA in implementation. The CAPTCHA randomly generates around 6 distorted characters in the server side. Without incurring significant extra computation cost, the CAPTCHA program can be adjusted to generate 62 alphanumeric characters (26 letters uppercase and lowercase, 10 digits) and display them on the virtual CAPTCHA Keyboard. Both CAPTCHA generator and soft keyboard generator are available in public websites [1, 5]. Figure 2 is an example of virtual keyboard which can be easily implemented using java scripts. It costs little in programming to combine the two generators. The server's database of user profiles does not need any change, since the password format remain the same.

3 Security analysis

In the common client-server architecture, the most vulnerable point is the client side, the next is the transmission channel (the Internet), then is the server side. We assume server side is safe, otherwise, no password is secure.

Using the CAPTCHA Keyboard, we can secure password against Trojan programs which steal what user types when inputting passwords, which have happened and are still happening [8]. Suppose some Trojan programs can even capture what user clicks on the CAPTCHA Keyboard, i.e, it can steal the sequence of locations which will be transmitted to sever for verification. However, due to the guard of CAPTCHA, the hiding Trojan can't read and recognize the keyboard. Even the locations are known to Trojan, it can't determine the what are the real characters that user clicks on the keyboard because the locations are random. That is, only the server who randomly generated the keyboard knows what each location stands for. The transmission is also secured by the CAPTCHA Keyboard, since even the sequences of locations are intercepted, they are useless for next time login. Every time user logins, the order of the keys on the virtual CAPTCHA Keyboard is different since it is randomly generated.

Of course, although CAPTCHA Keyboard as a solution enhances password security significantly, we have to acknowledge that it won't stay non-vulnerable forever, since Trojans are evolving with more intelligence. If a Trojan can capture the locations of mouse click and at the same time make a screen-shot and then send both to hacker, then the real password will be stolen. However, this can be prevented by either lock the screen-shot or block unauthorized programs from getting connected to Internet. The former can be configured in the Operating System and the latter can be set using common Firewall software.

4 Conclusions

The CAPTCHA Keyboard significantly enhances security of passwords against Trojan thefts. Its implementation cost is almost same as regular CAPTCHA. The distorted characters on the virtual keyboard prevents Trojan programs from recognizing them. The random order of characters hides the real passwords, which makes the theft useless. The only way that a hacker can break the CAPTCHA Keyboard system is: i) obtain the sequence of locations that user clicks, ii) take the screen-shot of the CAPTCHA Keyboard, and iii) send all these information over internet to hacker. No Trojan program so far can do the three things simultaneously and this attack tunnel can be blocked by setting the Firewall or disable the screen-shot.

References

[1] The CAPTCHA Project, http://www.captcha.net.

[2] G. Blonder. Graphical passwords. *United States Patent 5559961 (1996)*.

[3] D. Davis, M. Monrose, and M. Reiter. On user choice in graphical password schemes. *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[4] R. Dhamija and A. Perrig. Déjà vu: A user study using images for authentication. *Proceedings of the 9th USENIX Security Symposium*, August 2000.

[5] B. GreyWyvern. Javascript Graphical/Virtual Keyboard Interface. *http://www.greywyvern.com/code/js/keyboard.html*.

[6] D. Ilett. Trojan horse spies on Web banking. *Special to CNET News.com*, November 2004.

[7] D. Klein. Foiling the cracker; a survey of, and improvements to unix password security. *Proceedings of the Second USENIX Security Workshop*, pages 5–14, 1990.

[8] W. Knight. FBI's 'Trojan horse' program to grab passwords. *New Scientist Print Edition*, November 2001.

[9] B. Pinkas and T. Sander. Securing passwords against dictionary attacks. *Proceedings of the ACM Computer and Security Conference*, pages 161–170, November 2002.

[10] J. Waters. Authentication using graphical passwords: Basic results. *The Graphical Passwords Project, http://clam.rutgers.edu/ birget/grPssw/*.