A Security Concern in MS-Windows: Stealing User Information From Internet Browsers Using Faked Windows

Lior Shamir
Department of Computer Science,
Michigan Tech
1400 Townsend Drive, Houghton, MI 49931,
{lshamir@mtu.edu}

May 9, 2006

Abstract

A simple method that might be used by malicious attackers for stealing usernames, passwords, credit card numbers or other valuable small pieces of information is described. Hidden processes running on Windows-based machines might create faked controls and place them on the browser exactly on top of the real controls of web pages such as on-line banking account login pages. Users might then type in their passwords or credit card numbers into the faked controls, allowing the malicious process to capture the data. Since spyware is a large and growing threat to Internet users, it is not unlikely that such a technique will be used for stealing valuable information. An example based on *hotmail* web-based email service is demonstrated, but this technique might be used for a variety of password-protected web services. The paper also discusses different approaches of protection against the described attack.

1 Introduction

Spyware is a large and growing threat to Internet users. In fact, in 2003 the National Cyber Security Alliance reported that 90% of all broadband users have spyware installed on their computers, and one year later the congress started an investigation regarding this issue. While some spyware simply record web addresses visited by the user, others are more dangerous, and try to collect more valuable pieces of information such as usernames and passwords of on-line services, as well as credit card numbers that are used for on-line shopping. In the case of regular home users, this can lead to privacy problems such as unauthorized access to email accounts, or to severe financial damage such as abuse of

on-line banking accounts or captured credit card numbers. One common technique used by spyware for capturing small pieces of information is by using key loggers. Software-based key loggers are stealth surveillance applications that keep records of the user keyboard activities. By logging the user keystrokes, key loggers can capture personal information such as bank account numbers, credit card numbers, usernames, passwords, home addresses, etc, that can be used for non-legitimate purposes such as unauthorized activities in bank accounts, and might provide an infrastructure for identity theft. In the popular MS-Windows operating system they commonly do not show up in the "Task Manager", and their executable files use random names and are hidden among the system files so that even expert users find it difficult to know whether a key logger is active in their system.

Some key loggers are marketed as legitimate tools for tracking employees or family members [Blazing, Spector, Invisible, Keysnatch]. However, despite the legitimacy of some keystroke loggers, malicious keyboard monitors in the form of worms, spyware or Trojan horses are considered a concern in the field of information security [Stafford 2004]. Recorded past incidents that involve key loggers include the identity theft from Kinkos customers [Network Security 2003], the password stealing from Microsoft's Redmond, WA office [Farrow 2003] and more [Baldwin & Klingdon 2003, Levine 2004].

Due to the increasing popularity of key loggers, some anti-key logger approaches have been proposed. One approach is by using anti-hook techniques and detecting processes trying to intercept messages on their way to their destination window [**?** , 1]. Other approaches are based on scanning the system for known key loggers, and are similar to anti-virus programs.

However, stealing a password does not necessarily require key logging. The powerful and flexible window hierarchy in MS-Windows operating system allows a malicious process running on a Windows-based computer to replace the password controls for known pre-selected web pages. This feature can easily be used by even moderately sophisticated computer programmers for stealing passwords and transmitting them to a public domain. With the assertive workforce of hackers and malicious software developers, one can reasonably assume that it is only a matter of time before this threat becomes a practical concern in computer security of on-line banking and electronic commerce. In this paper we describe a potential attack that is based on using faked windows in MS-Windows operating system. In Section 2 we describe the tools and methods that can be used for such an attack, and in Section 3 we propose methods of protection against such an attack.

2 Capturing Usernames and Passwords by Using Faked Windows

The flexible window positioning and hierarchy management in MS-Windows operating system provides powerful features for application developers. However,

due to the presence of hidden malicious processes running on the system, the ability to access and modify windows of other processes also introduces some software security concerns.

A potential faked window attack can use known applications that require their users to type in passwords such as popular Internet browsers. The specific type of Internet browser being used on a given system can be easily deduced by accessing the system registry. After the Internet browser type is known, the malicious process can find the URL of the current web site by using the browser application interface. By comparing the URL to a list of pre-selected addresses of specific known services such as web-based email, the malicious process can determine whether the user has opened a password web page and is now about the enter her password and username in order to use a password-protected service. For instance, if the malicious process notices that the URL is www.hotmail.com, it can infer that the user is probably about to type-in the username and password into the textboxes at the top of the page in order to check or send email messages using the *hotmail* service.

After it has been determined that the user is about to enter her password and access some known pre-selected victim web-service, the malicious process simply creates textboxes and buttons at the exact size of the controls of the web page, and places them on top of the Internet browser so that the new controls cover entirely the real controls of the web page. This can be implemented by first finding the handle of the browser window by using the FindWindow API function. If the popular Microsoft Internet Explorer 6.0 is used, it can be simply done by calling the function FindWindow("IEFrame",NULL).

If more than one window is used, the title of the web page can also be used in order to find the specific instance of the Internet browser. In the case of *hotmail*, this can be done by the FindWindow API function by using also the window name, e.g. FindWindow("IEFrame", "Sign In - Microsoft Internet Explorer").

Once the window handle is determined, the malicious process can create new text and button controls using the CreateWindow function, and place them exactly on top of the username, password and the submit button using the SetParent and SetWindowPos API functions. Using SetParent in the form of SetParent(FakedControlWindowHandle,BrowserWindowHandle), the new faked controls become child windows of the internet browser, just like the real username and password controls. The SetWindowPos can be used in order to place the faked controls exactly on top of the real controls and make sure they get the focus when the user clicks them or tries to type in text. After the faked controls are in place, the user who wishes to check her hotmail account enters her password and username normally to the text controls, thinking that the data are about to be transmitted to *hotmail*, without knowing that the information actually goes to the malicious process that records the data and transmits them to its evil master.

After recording the data, the malicious process can copy the data to the real browser controls using functions such as SetWindowText or by sending WM_KEYDOWN messages. It can also simulate a mouse click on the submit button by sending (using the SendMessage API function) a WM_LBUTTONDOWN

message. With the correct data entered to the username and password controls, the user logs into the account normally, and therefore cannot know that the password has been stolen by another process.

An example application that implements faked controls to MSN hotmail using MS Internet Explorer 6.0 is available for download at http://www.phy.mtu.edu/ lshamir/downloads/faked_windows. The application creates faked window controls of the username and password, and places them on top of the real controls whenever the user opens the hotmail login page (at http://www.hotmail.com) using Microsoft Internet Explorer 6.0. When the user clicks the "Sign In" button, a message pops up with the username and password entered by the user. Obviously, the username and password are not recorded or transmitted, but users are encouraged to test this sample application using random faked usernames and passwords rather than their real password. This example application demonstrates *hotmail* password stealing, but unfortunately, the exact same approach can be applied to many password-protected web sites such as popular electronic commerce web sites and on-line banking services. One can reasonably assume that unlike the sample application, spyware using such an approach will also be invisible to the Taskbar and to windows "Task Manager".

3 Protection Against Faked Window Attacks

While the potential damage introduced by faked windows can be severe, the protection against this kind of attack is fairly simple, and can be applied in several ways. The safest way is probably by checking that no text window covers the username and password textboxes. This can be done by searching all windows in the system and examining their positioning and Z-order in order to check if they cover username or password text controls. For instance, if a textbox is found to have the same parent window and approximately the same size and position of another textbox control, it can be suspected for being a faked control. However, this approach uses Windows APIs, and therefore can be supported only by the browser or by a third-party process running in the system (such as anti-virus). Web sites, however, cannot defend using this technique, and are dependent on the protection level of the user's system.

Since Internet browsers do not yet provide this kind of protection, web masters might search for an alternative that is not dependent on the web browser or software installed on the user's system. In order to confuse potential malicious processes from creating and using faked windows, the web page can be changed dynamically every time it is accessed by a remote user so that the relevant textboxes and buttons randomly change their sizes and positions. When different users have different sizes and positions of textboxes, faked controls of fixed sizes and positions will not be placed exactly on top of the real controls. The web page can also include a random number of unused textbox controls and buttons in order to confuse the malicious process.

4 Conclusion

The flexible and powerful window handling in MS-Windows operating system allows processes running on the system to create faked windows that cover specific windows of other applications. This feature can be used by malicious hidden processes in order to steal confidential information such as usernames, passwords and credit card numbers. By creating faked textbox windows that cover the real windows, users simply type in the information into the textboxes that they see on the screen, but these are actually textboxes controlled by a malicious process.

The protection against this attack can be provided by the browser or by a third party application such as an anti-virus, by checking that all relevant textboxes, buttons, etc' are not covered by other windows of approximately the same size. Web masters can randomly add textboxes and buttons, or change the size and position of the textboxes in the relevant web pages in order to confuse potential malicious processes.

References

[Baldwin & Klingdon 2003] BALDWIN, R.W. AND KLINGDON, K.W. 2003. Survey of Spyware and Countermeasures. Plus Five Consulting, Inc. http://www.plusfive.com/reports.html

[Blazing] Blazing Tools Software, Perfect Keylogger, http://www.blazingtools.com/bpk.html.

[Farrow 2003] FARROW, R. 2003. Is your Desktop Being Wiretapped? Network Magazine 18, 8, 52–53.

[Gunetti & Picardi 2005] GUNETTI, D. AND PICARDI, C. 2005. Keystroke Analysis of Free Text. ACM Trans. Info. Syst. Security 8, 3, 312–347.

[Invisible] Invisible Keylogger, http://www.amecisco.com/iks2000.htm.

[Keysnatch] Keysnatch keylogger, http://www.fileheaven.com/Keysnatch/download/2975.htm.

[Levine 2004] LEVINE, J.R. 2004. Written Comments of Dr. John R. Levine. U.S. Senate Committee on Commerce, Science and Transportation, http://commerce.senate.gov/pdf/levine032304.pdf

[Network Security 2003] Network Security. 2003. Man steals passwords with keystroke logger. Network Security 8, 20–21.

[Spector] Spector keylogger, http://www.spector.com.

[Stafford 2004] STAFFORD, T.F. 2004. Spyware: The Ghost in the Machine. Communications of the Association of Information Systems 14, 291–306.

[WinMacro] WinMacro, http://www.phy.mtu.edu/~lshamir/downloads/WinMacro/WinMacro.html.

[1] XU, M., SALAMI, B., AND OBIMBO, C. 2005. How to Protect Personal Information against Keyloggers. Proceedings of Internet and Multimedia Systems and Applications (IMSA05), 275–280.