Optimizing Windows Security Features to Block Malware and Hack Tools on USB Storage Devices

Dung Vu Pham[1], **Malka N. Halgamuge**[2], **Ali Syed**[1], **and Priyan Mendis**[2]

[1]School of Computing and Mathematics, Charles Sturt University, Victoria 3000, Australia
[2]The Department of Civil and Environmental Engineering, The University of Melbourne Victoria 3010, Australia

Abstract— Malware replicating via USB storage devices including worms, virus, Trojan horses and other malicious codes together with USB based hack tools such as Pod Slurp and USB Switchblade have accounted for many serious security issues for the last few years. The majority of these malicious codes exploit Windows Autoplay features to automatically launch attacks on host computers transparently to the users. In this paper, we will analyze the vulnerabilities in the default settings of the latest Windows operating systems which allow malware from attached USB storage devices to launch attacks on the computers. We will also propose solutions and patches for the vulnerabilities in form of ready-to-deploy scripts which can be run by any computer users. The solutions will focus on the latest Windows operating systems including Windows XP SP2, Windows Vista, Windows 7, and Windows 2008.

1. INTRODUCTION

Universal Serial Bus (USB) storage devices have been one of the most common means of malware and hack tools attacks on computers for the last few years. This has been possible due to some factors involving the lack of security mechanisms for Windows Autoplay features, and the communication channel between computer and USB drives, and the popularity of USB storage devices such as compact flash cards, flash drives, external HDDs, digital cameras, iPods, and USB phones. Malware that exploits Windows Autoplay features to replicate via USB drives and automatically activate itself is commonly known as Autorun malware. This threat vector represented the largest single category of malware for the first two quarters of 2008 which, accounted for 17.7% of the total infection detected by Microsoft [1], and from June 13 to July 13, 2009, over 27 million infections by this type of malware were detected by McAfee [2]. Moreover, there are also various USB based hack tools, especially on U3 drives, such as USB Switchblade, Pod Slurp, and USB Pocket Knife which exploit the Autoplay features to automatically launch attacks on host computers transparently. Microsoft and security firms such as McAfee, Symantec, ESET, Trend Micro Inc, and BitDefender have invested a lot of effort in re-engineering their products to reduce the impacts of Autorun malware. Recently, Microsoft had to make a decision to disable the Autoplay feature for USB drives on Windows 7, their latest operating system, in an effort to reduce the impact of Autorun malware [1]. However, such a solution only helps prevent malware from activating via Autoplay features on USB drive insertion. There have been some work on the risks of USB devices to information security in corporate environments with proposed solutions involving *data access control, USB port access control, and security policies* [3–5]. However, the most common type of risks from USB drives which was not mentioned in the previous papers is malware on USB which are supposed to be accounted for the majority of all USB based software attacks. This threat vector has not received enough attention and further work on this type of attacks is necessary. Moreover, there are important factors which were not considered in the solutions proposed in the previous papers which are personal computers and their users. Solutions proposed in previous papers are only applicable for corporate environments which require license costs, technical configurations and maintenance cost, and corporate policies [6]. In this paper, we will propose a solution package in form of ready-to-deploy scripts which helps optimize Windows security features to mitigate attacks by malware and hack tools on USB storage devices. Here, we target for a solution which does not require license costs or complex technical configuration and suits both personal and corporate computer environment.

2. MALWARE AND HACK TOOLS ON USB STORAGE DEVICES

The term *malware* in this paper refers to computer worms, viruses, Trojan horses, root kits, spyware, and adware. Malware uses two main techniques to spread via USB storage devices, executable file

infections and by exploiting Autorun.inf file and Windows Autoplay feature [6]. Malware presenting on USB drives are capable of self-replicating themselves via many means of media under the forms of executable files and scripts. Such executable files typically involve .exe, .dll, .prg, .ocx, .ovl, and .sys, and scripting files with .bat, .js, .pl, .vbs, and .wsh extension.

USB based hack tool refers to non-self-replicable malicious tools deployed on USB drives, especially U3 drives, and can be triggered from USB drives [6]. Many of these tools involve the use of scripting files and are specially designed to exploit Windows Autoplay features to trigger themselves on USB drive insertion [3–5]. USB based hack tools generally provides attackers with ultimate payloads varying from *data theft, information exploits, accounts and password exploits, data recovery, to privilege escalation* [6]. Although the number of incidents caused by USB based hack tools is limited because these tools cannot self-replicate and are not fully automated, they do cause fear in corporate environments where information security is critical.

3. WINDOWS VULNERABILITIES EXPLOITED BY MALWARE AND HACK TOOLS ON USB DEVICES

3.1. No Security Mechanism for Windows Autoplay Features

Windows Autoplay features automatically launch the content on removable media at the very moment when the media are inserted. The Autoplay process is activated with parameters specified in Autorun.inf located in the root folder of a USB drive which specifies the path to executable files to be run. Attackers exploit this benign feature to auto launch malware without any user interaction when a USB device is inserted. The computer detects newly inserted USB drive, reads the Autorun.inf file and loads the malware. Unfortunately, Windows does not provide any security mechanism, which prevents activation of malware specified in Autorun.inf files.

3.2. U3 USB Drives Remain a Major Threat Vector

U3 is an open standard developed to provide users with application mobility through an application platform available in U3 drives whereby U3 applications can be installed on and run from U3 drives *independently from host computers*. In a U3 drive, a small partition located at the beginning of the drive is marked as a CDFS (CD file system) partition so that Windows recognizes it as a CD rather than a USB drives. U3 applications are self-contained applications run from the CDFS partition. While the Autoplay features for USB drives are disabled on Windows 7, they are still enabled for the CDFS partition. U3 technology is supported on all Windows x86 and x64 platform from Windows 2000 SP. Attackers can customize their own ISO images with necessary hack tools and malware to install in the CDFS partitions to exploit the Autoplay feature available for CDFS partitions or directly run the hack tools from the U3 Launchpad [6]. The CDFS partition contains irremovable Autorun.inf file and malware which cannot be removed by Antivirus software even when they are detected.

3.3. Driver Signing Is Not Enforced on 32 Bit Platform

Digital signatures allow users to know whether a legitimate publisher has provided the software package. The x64 families of Windows Vista and Windows Server 2008 require Kernel Mode Code Signing (KMCS) in order to load kernel-mode software. Kernel Mode Code Signing (KMCS) policy requires kernel-mode code to be signed with a valid Authenticode certificate rooted by a famous code signing authorities, such as VeriSign. However, The 32 bit family does not provide any protection to private read/write system memory used by components running in kernel mode and, therefore, once in kernel mode the software has complete access to all operating system data [7]. This was the reason why attackers could craft their own driver for their USB drives to cause buffer overflow attacks on Windows 32 bit operating systems [8].

3.4. Vulnerabilities in User Account Control (UAC)

Windows operating systems including Vista and the later are still vulnerable to unsigned code execution. Although UAC does warn users when there is an unsigned application activated and try to make changes to system settings or system files, these warning messages are not always effective as users can simply allow the execution of the unsigned code. This is due to the lack of standardization in the software industry where a lot of software is released without digital certificates which make it difficult to make the decision of allowing only signed code execution on Windows computers. Figure 1 illustrates an alert by UAC created when an unidentified (unsigned) application which is trying to make changes the computer. In this situation, if the user chooses "Allow", the unsigned code gets the full permissions to make changes to the operating system files

and settings. Unfortunately, in many cases, users feel UAC alerts annoying and immediately allow the unsigned application to execute.

Moreover, information gathering tools which collect user's information left in temporary files and Windows registry are generally not blocked because UAC only alert users when there is a process trying to *make changes* to the computers. Therefore, in many situations, malware and hack tools on USB drives can still successfully execute their payload under the monitor of UAC. Figure 2 shows the UAC setting options under Windows 7 where it shows that only application which try to modify or make changes to the current settings and system files will generate an alert. This explains why some spyware and information gathering tools can still successfully exploit user's private information such as online accounts, credit card information from windows temporary files under the UAC monitor.

3.5. In Effective Malware Detection Based on Behavior Patterns by Microsoft Antimalware

The recent test by AV-Test.org, independent anti-virus testing lab, showed that Microsoft Security Essentials (MSE) achieved 98.44 per cent detection rate using signature based detection but it does not have any effective dynamic detection features which can analyze malware based on malicious behaviors. Detection and cleaning of infected computers worked but in many cases MSE does not correct the problem left behind after malware attacks [9]. Therefore, MSE is not effective to malware whose signatures are yet not there in malware definition database and chances for stopping malware on activation is lower than a thorough scan over all the files before the malware gets triggered.

In Figure 3, the drivers for USB devices provided by hardware vendors are located in User Mode layer where access to system resources is limited to user right and privileges only. This model is generally applied to Windows Vista and the later. However, in previous Windows version such as Windows XP and 2003, USB driver was located in Kernel Mode layer where it has unlimited access to system resources. Thus, attackers who successfully commit USB drivers can have system rights and privileges. Moreover, attackers were able to craft drivers for USB drives and successfully injected the drivers into Windows possibly due to the lack of driver signing enforcement in Windows XP and other 32 bit editions. The enforcement of signed drivers will prevent unsigned drivers from being injected to Windows kernel and thus help mitigate this threat vector effectively.

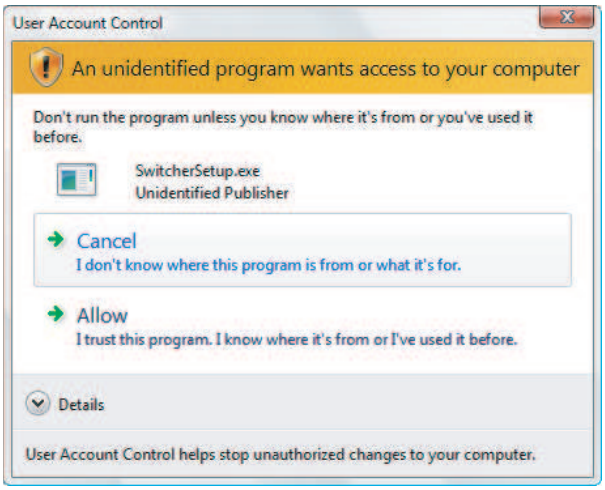

Figure 1: UAC alert on unsigned code execution.

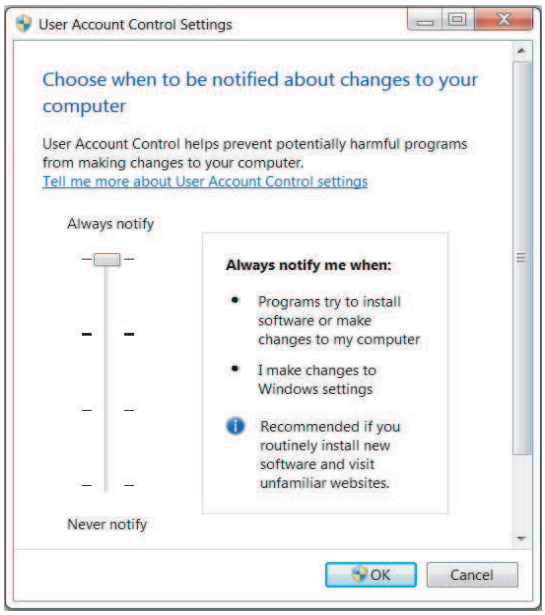

Figure 2: User account control settings.

4. SOLUTIONS

4.1. Block "Untrusted" Executable Files on USB Drive from Being Executed

A *trusted executable files* must be a *valid executable file signed with a non-expired digital signature by a trusted publisher or a reputable certificate authority* such as VeriSign. Viruses and worms

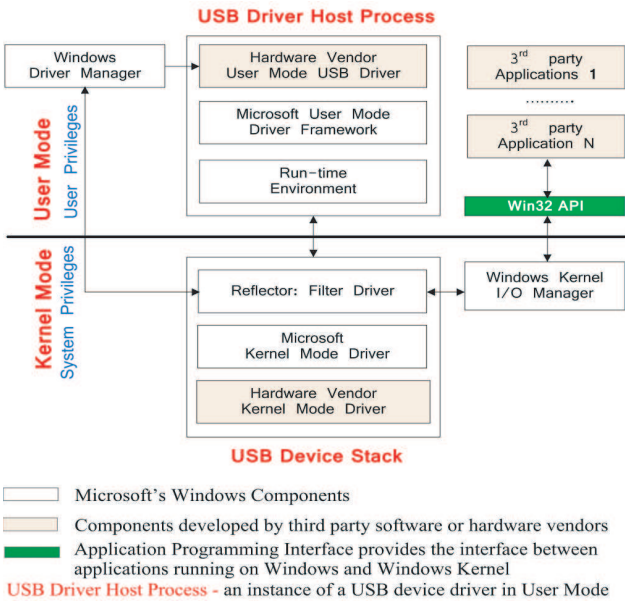

Figure 3: Windows USB driver architecture [6].

are often deployed with social engineering technique to trick users into triggering them. Therefore, allowing only trusted executable files on USB drive to be executed will help blocking any potentially dangerous code from executing even by users. This can be done via either software restriction policies or AppLocker feature. While software restriction policies feature is first introduced in Windows XP and 2003, AppLocker is a new advanced feature first introduced in Windows 7 Ultimate/Enterprise, and Windows 2008 R2.

The implementation of *software restriction policies* for executable files on USB drives are handled by certificates rules *specifying a code-signing, software publisher certificate and path rules specifying a fully qualified path to the USB drives* using wildcards to address all executable and script files. *AppLocker* is the more flexible option where we can specify only files belonging to trusted publishers can be executed from USB drives on specific user or users groups. Our ready-to-deploy script package allows users to choose to implement either software restriction policies or AppLocker based on their needs and their own operating systems.

4.2. Enforcement of Driver Signing

Under Windows XP, the driver signing option must be set to "block — never install unsigned driver software" under system property console. This will prevents crafted USB drivers to be installed on computers. While under the later 64 bit versions of Vista, Windows 2008, and Windows 7, driver signing enforcement is enabled by default, their 32 bit versions still allows users to force unsigned driver installation [7]. Therefore, manually enforcement of driver signing option must be done through *Local policy/security options* with *Devices: Unsigned driver installation behavior* option set to "*do not allow installation*". We also provides a script which automatically checks on the integrity check option for driver to make sure the driver signing option is enabled.

4.3. Prevent Autorun File from Being Created on Drive Root Folders

The most common feature of Autorun viruses and worms is Autorun.inf file creation on any drives to which they replicate as this is a part of the strategy on which they replicate and launch attacks on the next computer [10]. If the malware cannot create Autorun.inf on the next drives, it cannot automatically trigger itself and the chances for replication are much lower. The script that we provides here will automatically create Read-only Autorun.inf folder on USB drives and any connected drives on the host computer and thus preventing Autorun malware from overwrite Autorun.inf file to complete an attack cycle.

4.4. Enforcement of UAC and updated Microsoft Antimalware

The most common infection methods by malware involve through bundling (packaged with legitimate software) and social engineering (tricking users into activate the malware). UAC, Windows Defender and MSE offer significant protection against both bundling and social engineering. UAC

will always alert users on any process which try to make changes to system files and settings which helps users aware of what is going on and take necessary actions. Windows Defender and MSE provide real-time protection and scans executable files before they are activated. However, the detection mechanism of MSE relies solely on malware signatures [8] and therefore, enforcing malware definition update will provide the best protection results. Our script package is provides self check for UAC and automatically update the malware definition database for Windows Defender as well as MSE on daily basis.

5. RESULTS

Table 1 summarizes the changes made to Windows operating system after implementing the solutions. This generally shows how the security features in Windows operating systems are utilized to protect computers from malware on external USB drives.

Table 1: Changes applied to Windows after implementing the solution.

Setting & Features	**Default Settings**	**Changes in Settings**
Execution files running from USB drive	Yes	Trusted signed applications only
Autoplay feature on USB devices	Yes: Before Windows 7; No: Windows 7	Yes
Unsigned USB driver installation	Yes: 32 bit; No: 64 bit	No: Signed driver only
Executable files copied to System locations	Yes	No
Group Policy: Software Restriction Policy	Not implemented	Yes, block untrusted codes on USB drives
User Account Control	Yes by default, can be turned off	Yes, checked and turned on at system booting
AppLocker	Not implemented	Yes, block untrusted codes on USB drives
Windows Defender	Available on Windows Vista, 7, 2008	Installed on all versions from Windows XP
Microsoft Security Essentials	Not implemented	Installed on all versions from Windows XP
Windows Update	Not enforced	Enforced, malware definition update
Windows security features utilized	UAC, Windows Defender (Vista & Win 7 only), Windows Firewall	AppLocker, MSE, Windows Defender, UAC, Windows Firewall, Windows Update, GPO

6. DISCUSSION

The solution package helps to optimize and maximize the effectiveness of all currently available security features on Windows operating systems to mitigate the threats from malware and hack tools on USB drives. The solution does not require complex configuration or additional software license costs and the most complex processes are automated via script packages which make the solution suitable for all computer users in both office and home environment. Moreover, the enforcement of signed applications and trust is just another step towards a secure computing environment which is currently the common trend in the computing industry. However, this solution does not directly address and remove un-activated or "sleeping" malware and hack tools inside USB storage devices. Therefore, we suggest a more comprehensive solution which involves a new Windows

security service architecture with new Windows security services and features which coordinate the available Windows security services and features to provide a scheme based real time scanning on USB drives for malware and hack tools right at the time of USB drive insertion in our next paper.

REFERENCES

1. Cohen, A., "Improvement to autoPlay," Engineering Windows 7, Microsoft Developer Network Blog, Microsoft Corporation, 2009 (accessed November 2009), http://blogs.msdn.com/e7/archive/2009/04/27/improvements-to-autoplay.aspx.
2. McAfee Avert Labs., "McAfee threats report: Second quarter 2009," *McAfee, Inc.*, 2009.
3. Alzarouni, M., "The reality of risks from consented use of USB devices," *Proceedings of the 4th Australian Information Security Conference*, 2006.
4. Fabian, M., "Endpoint security: Managing USB-based removable devices with the advent of portable applications," *Information Security Curriculum Development Conference*, 2007.
5. Wong, S., "Risk associated with USB memory sticks and high capacity storage devices," *Siemens Insight Consulting*, 2007 (access January 2010), http://www.insight.co.uk/files/whitepapers/Risks%20Associated%20with%20USB%20 Memory%20Stick%20(White%20 paper).pdf.
6. Pham, V. D., A. Syed, and M. N. Halgamuge, "Universal Serial Bus based software attacks and protection solutions," Unpublished Result.
7. Russinovich, M. E. and D. A. Solomon, *Windows Internals*, 5th Edition, Microsoft Press, 2009.
8. Roberts, P. F., "USB devices can crack windows," *Eweek*, 2005 (accessed February 12, 2010), http://www.eweek.com/c/a/Security/USB-Devices-Can-Crack-Windows/.
9. Leyden, J., "One thumb up for MS security essentials in early tests," *The Register*, 2009 (accessed January 2010), http://www.theregister.co.uk/2009/10/01/ms_security_essentials_review/.
10. Thomas, V., P. Ramagopal, and R. Mohandas, "The rise of autorun- based malware," *McAfee Avert Labs., McAfee Inc.*, 2009.