A Simple Malware Test Environment

Sam Lundie and Daniel Rolf
School of Computing and Information Systems
University of Tasmania
Launceston, Tasmania, Australia
e-mail: {slundie, Daniel.Rolf} @utas.edu.au

Abstract— **Malware does not need to compromise the operating system kernel in order to provide an untrustworthy browsing experience for the user. This paper describes a simple, virtual machine-based, malware test environment built using freeware and open source software. The system was designed to allow the high-level behaviour of a piece of malware to be studied quickly and conveniently by monitoring network, process and file activity. The system proved effective when trialled against different samples of the well-known malware Zeus and was verified further by tests conducted with the commercially available anti-malware products PC-Tools and Trusteer. Although tests were conducted with variants of the Zeus malware, the techniques discussed in this paper are equally applicable to any other malware and can be used to quickly assess the effectiveness of potential anti-malware solutions. Also, the system is portable and simple, requiring only a general level of technical knowledge to operate, allowing it to be used as a convenient platform for a wide student and professional audience.**

Keywords – Zeus, online banking, malware

I. INTRODUCTION

Governments and corporate enterprises are increasingly dependent on computer networks for their day-to-day and critical operations. In this network-centric environment it is necessary for all participants to appreciate the importance of required network and computer security measures. It is also useful for security professionals to quickly become aware of any changed threat conditions, allowing necessary new countermeasures to be designed and implemented.

The purpose of this project was to develop a simple malware testing platform and to demonstrate its ability as an educational and research tool. The test platform was based on laptop hardware and freeware tools implemented in a virtual environment, making it affordable as well as portable.

Although, there are many different types of malware, credit card fraud is one of the most notable targets for malware developers and organised crime because of its impact on the finance industry and general public. Statistics from the Australian Payments Clearing Association show that card-not-present fraud increased by 38% for the calendar year 2010, with 35.6 cents in every $1,000 dollars falling victim to fraud [1]. However, similar figures for fraud perpetrated against Online Banking are much harder to obtain as banks are seemingly reluctant to divulge loss figures. The Symantec Corporation has claimed that cybercrime has surpassed illegal drug trafficking as a criminal money-maker [2].

It has been estimated that Zeus is guilty of approximately 44% of all banking malware infections [3]. In August 2009 Gunter Ollmann the VP for research at Damaballa [4] positioned the Zeus malware as the number one botnet threat with 3.6 million infections in the US alone (about 19% of the installed base of PCs in the US). Zeus is ever changing, and many modifications exist in the wild, each targeted towards a specific set of exploits. Currently, Gameover is one such variant [5] which, according to the United States Federal Bureau of Investigation (FBI), has "…the capability to steal usernames and passwords and circumvent most common authentication methods" [6].

Zeus was specifically designed to operate against the Windows suite of operating systems (OSs) because Windows enjoys the majority share of the OS market. The majority of people transacting online use a version of Microsoft Windows, with WinXP, Win7, WinVista and Win2003 taking up over 90% of the total desktop computer OS market share [7]. The major non-Windows OSs, MacOSX and iOS, have a combined market share of only 7%. This means that targeting machines running a windows operating system provides much better Return on Investment (ROI) for those committing fraud. Microsoft also acknowledges this threat and spends considerable time and resources on taking down Command and Control servers that target machines running its operating systems [8].

Given the above reasons, for the purpose of this project, the target systems were chosen from the range of Windows operating systems and the sample malware chosen was

Zeus. The approach taken is equally applicable to other malware affecting Windows OSs.

II. COMPROMISING WINDOWS

There are many vectors of compromise for Windows systems. Some are offensive, such as worms like Conflicker exploiting remote vulnerabilities, and others are reactive where the end user performs some sort of action that triggers the exploitation. Recent research by the SANS Institute found that the number one initial infection vector was exploitation targeting client-side software [9]. In 2010, the National Vulnerability Database [10] indicated that web browsers and document management software accounted for the majority of the applications being exploited, with multimedia players being strongly represented. Client-side applications have more integration with the Internet and offer more functionality providing a greater attack surface for developing exploits. These vulnerabilities can be exploited from many different sources: some are executed just by visiting a webpage having embedded malicious content, known as 'Drive-by' infection, whilst others are more targeted 'Phishing'-style emails containing either embedded exploited software, or lures to sites where such malicious content is hosted.

Rootkits are designed to hide a program from the guest OS to avoid detection and sustain their life span on the infected machine. They have the ability to hide files, processes, registry keys, open network ports and other system objects. By subverting the OS they cloak their operations from anti-virus/malware products that search for them. Modern Trojans, such as Zeus, leverage the power of rootkits, to keep their activities hidden from the OS and any anti-virus software that may be running on the compromised PC. Trojans want to run but also want to remain hidden and, as such, face a similar paradoxical situation to rootkits [11]. Another problem faced by Trojans is how to install themselves on a system without alerting the user to their malicious intent. Some of the more prevalent online banking Trojans, such as Zeus, or its predecessor Torpig, only operate within the context of the user who executed them and this restricts the depth to which they can embed themselves within the OS [12]. They are unable to employ kernel level rootkit techniques because they require access to the kernel to execute and thus require a higher level of privilege. Such Trojans are restricted to using a shallower user level rootkit.

One reason that these Trojans may choose to avoid using a higher level of privilege is to avoid triggering Windows User Access Control (UAC). As part of Microsoft's security push to increase the security of their OSs, UAC was introduced into Windows Vista ™ [13]. The goal of UAC was to lock the screen and give the user a visual prompt requesting explicit consent to allow a process to execute actions requiring a higher level of permission than that of its current context.

The Trojan appears to weigh up the advantages of running at a higher privilege level against the risk of alerting the user to its malicious intentions (because of UAC). The fact that Trojans such as Zeus and Torpig can implement User Mode Rootkits, inject code into other processes and establish outbound connections via injected processes such as Internet Explorer, indicates that the permission level of the system account used as the context for their execution is too high. However Windows is at pains to avoid over-prompting the user for requests to elevate privileges. Microsoft overhauled the UAC functionality after feedback from users of Vista™ indicating that it was too onerous and that too much of their time was spent accepting the UAC prompts from the OS [14]. As such, Microsoft is in the tricky position of having to provide better security for its users whilst still allowing the flexibility of using the OS unhindered.

III. FUNCTIONALITY OF THE ZEUS TROJAN

The Zeus Trojan is a bot that installs itself on the victim machine and provides a framework for remote control of that machine. Once infected, the machine will report back to a Command and Control (C&C) server periodically to update it with the status of the local machine. Zeus is designed to be deployable to end systems that sit behind sophisticated network infrastructure. It is specifically designed for the Windows OS and will install on a machine with only Guest privileges. It has the ability to communicate with its C&C over TCP port-80 (http) and be operated using the SOCKs protocol. It will spy on the user and has the ability to capture information from within the browser which, thanks to its API hooking, includes sessions secured by Secure Socket Layer (SSL).

Zeus is a modular Trojan designed for easy customisation to the needs of those deploying it. Once a bot is loaded on a system it calls to the C&C address (which is hard-coded in the binary) for a newer version of a configuration file. This allows the controllers of the bot net or 'Bot-herders' to update their Botnet by simply uploading a new configuration file containing a list of web-injects - the sites from which the Trojan is targeting to steal information. This communication is via a thread injected into the Explorer Process by the Trojan. Apart from the initial call to the C&C, all communications are encrypted with the RC4 algorithm using a pre-shared key that is obfuscated within the Trojan executable. Zeus and its operation have been studied at depth by Binsalleeh and his group [3].

One of the most powerful methods that Zeus uses to subvert customer Online Banking sessions is its ability to perform Man-in-the-Browser (MitB) style attacks. In an MitB attack the web-browser is subverted to display the content that the Trojan wants the user to see. By using API hooking

techniques the Trojan can intercept the content before it is rendered by the browser and insert its own content instead. Zeus uses web injects to target specific financial institutions by including custom scripting tailored to emulate the layout of the targeted bank, thereby enticing the user to think the content they are viewing is that of the bank, thus negating any of the end to end security of SSL encryption between the source website and the victim.
According to Gühring [15], by injecting content directly into the web browser it is possible to circumvent many of the security mechanisms currently used by financial institutions.

> *"The WYSIWYG concept of the browser is successfully broken. No advanced authentication method (PIN, TAN, iTAN, Client certificates, Secure-ID, SmartCards, Class3 Readers, OTP, ...) can defend against these attacks, because the attacks are working on the transaction level, not on the authentication level. PKI and other security measures are simply bypassed, and are therefore rendered obsolete."* [15]

Any form of second-factor authentication that is not bound to the transaction the customer wished to perform can be subverted for the purposes of the Trojan. For example, during a customer payment transaction, once the Trojan detects the customer has submitted the payment, Zeus can intercept the POST request and populate it with a malicious payment. When the user is presented with an authentication screen requesting confirmation of payment Zeus will manipulate the output seen by the user so they see, not the malicious payment, but the payment they were attempting to make. When the user supplies an authentication credential Zeus will use it to authenticate the malicious payment. The balances and payment details of the customer can then be altered by the Trojan to display as if the original payment was performed and not the malicious payment. This on-the-fly transaction manipulation presents a complete compromise of the online banking system.

IV. The Test Environment

A. Physical Set Up

Static analysis methods can be time-consuming and while the effort expended could be warranted for a new strain of malware with new behaviours the focus on an iteration of a known variant can be excessive. The ability to quickly execute a piece of malware to observe its high level functionality can be useful in determining if the particular sample exhibits any new traits or behaviour not already seen.

Using virtual machines (VMs) has the distinct advantage of providing a platform that is both easily reset to a known clean state, and easily controlled. Note that some versions of the Zeus is VM aware and as such may not function correctly if it can see particular process IDs running. Multiple machines can also been run on the same host to test the functionality of the malware against different OS types and versions. C&C infrastructure can be simulated within the confines of a single computer, hosting multiple VM images. Fig.1 shows the basic logical network configuration used. Note that all computers shown are
virtual devices.

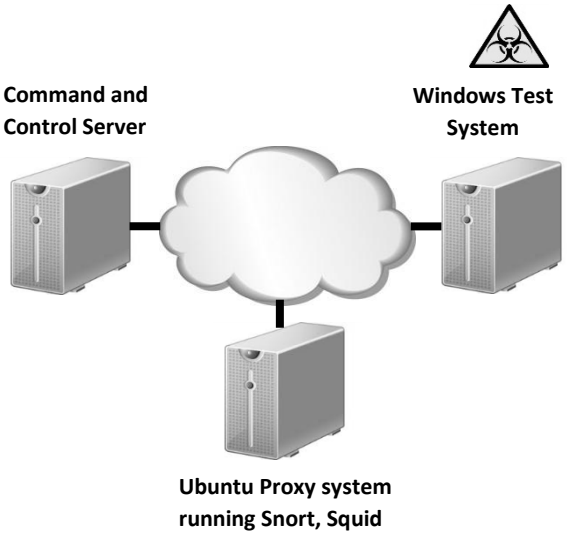

Figure 1: Logical Network Configuration

For our experimentation a test system consisting of 4 virtual machines, all running on a Virtual-Box platform was configured on a Windows-7 64 bit system. The underlying hardware comprised an Intel i5 processor running at 2.66GHz X4 with 4GB RAM. The virtual machines comprised the C&C server, an Intrusion Detection and Prevention System (IDS/IPS), and two target machines. All virtual machines could be run simultaneously, with some noticeable degradation to the performance of the host machine when significant program loads were placed on both host and guest machines.

The IPS virtual machine was setup to run the Snort engine, to obtain detailed information about generated traffic. This was built from Insta-Snorby [16], an Ubuntu build that already has Snort configured and only requires a limited amount of setup. Alerts are displayed in a very clean web interface which runs on the Ruby on Rails platform.

Windows XP and Windows 7 were chosen as the target machines' Operating Systems as, currently, they represent the two most dominant Windows OS versions in use. The RAM allocation to each can be easily adjusted within the Virtual-Box software and was chosen to be low enough to allow all virtual machines to run simultaneously with sufficient functionality from the one host machine (Table 1).

The network was controlled by a simple ADSL Router/Modem running the 802.11g wireless protocol and hosting the DNS Server. Statically mapped IP addresses were used for the hosts to enable the bots to call back to the C&C Server. The C&C Server has its network interface configured to bridged mode, using a device driver on the host system to filter data from the physical network adapter. This driver is therefore called a "net filter" driver. It allows Virtual-Box to intercept data from, and inject data into, the physical network, effectively creating a new network interface in software. When a guest is using such a new software interface, it looks to the host system as though the guest were physically connected to the interface using a network cable: the host can send data to the guest through that interface and receive data from it. This means that you can set up routing or bridging between the guest and the rest of your network [17].

Table 1: Details of the Virtual Machine Images used in the test system

Specifications	**C&C Server**	**Target 1**	**Target 2**	**IPS**
Operating System	Ubuntu 11.04	Windows XP-32bit	Windows7-32bit	Ubuntu 10.04 (Snorby)
Base Memory	1267MB	314MB	1287MB	822MB
Disk Space	10GB	10GB	10GB	6GB
Network Configuration	Bridged	NAT	NAT	Bridged
Virtual-Box Image Size	1.8GB	3.6GB	3.06GB	1.02GB

Network Address Translation (NAT) mode was used for the two Windows target machines. NAT mode allows a machine to access the internet via Virtual-Box's NAT filtering but the machine can't be used as a server as it is not internet-addressable in this mode.

B. Software Tools used for Testing:

In order to understand the malware's operation we used tools to capture information about network, process, and file activity. Network activity was monitored using the packet sniffing tool WireShark, but running local copies of Fiddler (a plugin for Internet Explorer) was used to look at all HTTP/HTTPS communications because of its simplicity. Fiddler also "allows one to fiddle with incoming and outgoing data" if desired [20].

Based on the Windows Mini-Spy filter-driver, CaptureBat has the ability to monitor processes at a kernel and user level. Results should be reliable with malware such as Zeus that only work within the malware executes kernel level hooking. CaptureBat saves any files that are deleted, a useful feature as the tested Zeus samples were observed to delete certain files associated with their unpacking process. Graphical monitoring of processes and their associated threads and files was used to show the lineage of a process and determine parent and child relationships between processes. The utility "SysInternals Process Monitor", a freely available advanced monitoring tool for Windows, was used as it allows real-time activity on the file system and registry as well as process and thread activity to be viewed. GMER was chosen to look for evidence of hooking on Windows API calls by comparing its output from before and after installation of the malware. GMER finds hooks in the System Service Dispatch Table (SSDT) - a kernel structure that lists native system service' addresses. GMER also encrypts traffic from the keyboard and decrypts it again in the browser, to stop keyboard logging. HandleDiff.exe was also useful for finding evidence of hooking as it operates by comparing the difference between two snapshots of open file handles. The interval between the snapshots is a configurable parameter.

V. THE TEST PROCEDURE

As the Zeus Trojan uses the Explorer process on the machine it proved easier to use Fiddler on the local machine to capture the traffic between the bot and the C&C rather than rely on the network packet sniffer. "Fiddler" has the added advantage of being able to see the other http requests without having to reconstruct them from observed network traffic. As Zeus communicates at the application layer such a level of detail would add unnecessary complexity.

The actual network design included an IDS/IPS to capture and monitor traffic on the network (see Fig. 2). Any communication between the infected PCs and the internet or C&C server could be monitored in this way. VRT labs have a rule set for detecting the initial call from the infected machine back to the C&C. The rules capture the initial POST and GET request traffic between the infected machine and the C&C, and this is the only traffic that is not encrypted [18]. The remaining traffic is encrypted using RC4 and a shared secret key.

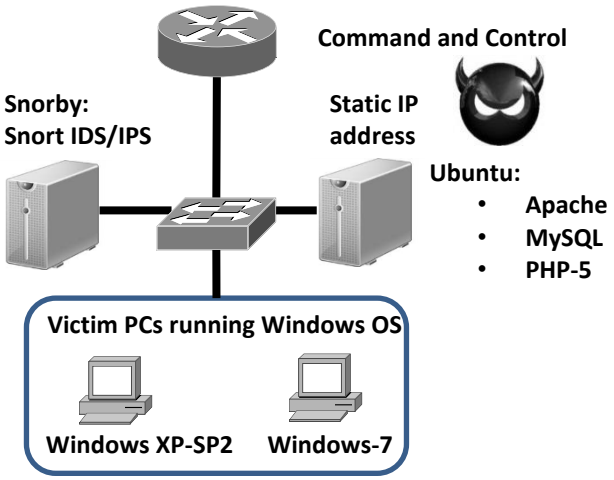

Figure 2: Detailed Logical Configuration used for Malware Dynamic Analysis

A. Capturing Network Traffic

A snapshot was taken of the victim machine before it was infected to allow a quick reset to a known clean state. Then, for each test, a clean VM Snapshot was loaded into VirtualBox, with all the monitoring tools already running. The programs running on the virtual machine, prior to the malware being executed, were:

- System Internals TCP View, to show current open TCP/IP and UDP Ports
- Fiddler IE Plugin, to monitor http/s traffic
- CaptureBat
- WireShark
- GMER (run to ensure that the machine is in a clean state)
- HandleDiff.exe

The malware was loaded and executed and a second snapshot was taken by HandleDiff after 60secs (using the command HandleDiff.exe -d -s 60 -f zeus.txt). GMER was run a second time to check the system for evidence of Root-Kits, and WireShark was stopped so the network traffic could be checked. CaptureBAT was checked for artefacts and the output from Fiddler was examined for calls made to the C&C (requesting an updated configuration file). The malware sample was then checked in 'Virus Total' ™ to determine how many AV products were able to detect it.

Two variants of Zeus were used, as shown in Table 2 and Table 3. The first sample was retrieved from the Trusteer website (http://www.trusteer.com/). Before executing the malware, GMER was run to look for anything hooking the system and, as expected initially, it located no rootkits.

Table 2: Wild Zeus Variant

Specs	**Sample 1**
Description	Zeus 2.x Binary
MD5	c0b87175875743a7c560e915b711b50e
Source	Rapport
Timestamp	Sat, 2011-07-30 20:04
Size	194048 bytes
Filename	0.886293863363183.exe

Table 3: Zeus Bot from Source Code

Specs	**Sample 2**
Description	Zeus 2.x Binary
MD5	2d5ec50a525269dc9e05c04bd57a116d
Source	Zeus Source Code
Timestamp	Sat, 2011-09-22 22:1
Size	163840 bytes
Filename	Bot.exe

Execution of the Wild Sample:
MD5(c0b87175875743a7c560e915b711b50e)

As soon as the malware was executed on the XP Virtual Machine, the dropper file created two executable files and a batch file. The malware launched the executable files and used a command window to launch the batch file. The three artefacts detected by captureBAT (blank lines have been inserted for readability) were:

device\harddiskvolume1\documents and settings\guest\ application data\uhkete\ychy.exe

device\harddiskvolume1\documents and settings\guest\ application data\owveny\ucrie.are

device\harddiskvolume1\documents and settings\guest\local settings\temp\tmp92a62147.bat

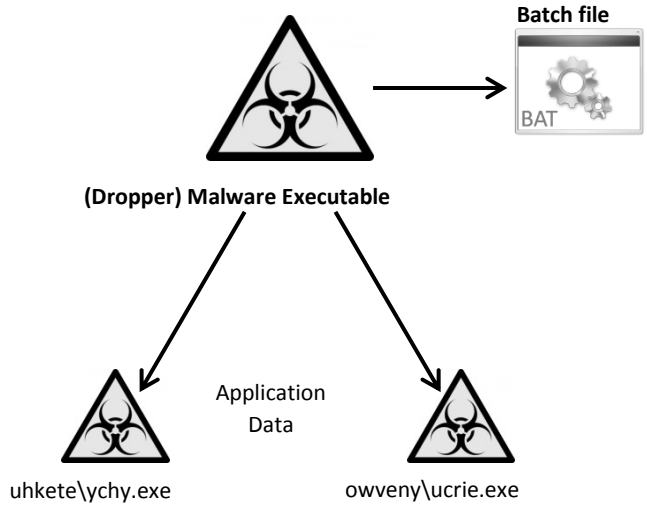

Figure 3: Files created on Zeus Execution

The batch file appeared to be responsible for deleting both itself and the original dropper.

"SysInternals Process Monitor" was used to verify each of the events and show that no child processes were created for the second executable. GMER was then run a second time whereupon it detected the rootkit.

B. Establishing persistence

The output from HandleDiff showed the malware had registered one of the newly created executable files with the Windows registry, ensuring that it would execute every time Windows was started.

Registry Keys modified by Zeus to reload on reboot

HKCU\Software\Microsoft\Windows\CurrentVersion\Run

Key (Not Verified) = {01C7C6E1-3DCC-BCB5-3E13-6DCD65A2CF91}
File it wants to run = c:\users\admin\appdata\roaming\isbaga\lier.exe

Shortly after execution of the malware the default Windows firewall detected an operation trying to make a modification. A "Windows Security Alert" was displayed advising that a program had been blocked, however, the malware's communication had not been effected as it had disabled the firewall. The malware issued a request for an updated version of the configuration file using an http GET request and Fiddler captured this as:

GET http://commanderseeckings.com/config.bin HTTP/1.1

As a verification step these details were given to ZeuS Tracker, a web based facility which tracks Zeus hosts around the world. Zeus Tracker was searched for the configuration file based on the outbound call to commanderseeckings.com and showed that this ZeuS C&C was listed previously. However the file was removed on 2011-08-04 at 04:51:28 (UTC) due to the C&C no longer being operational as the hosting domain had been blacklisted. The output from ZeuS Tracker is shown in Table 4.

Table 4: ZeuS Tracker Output for commanderseeckings.com

ZeuS C&C:	**commanderseeckings.com**
Date added:	2011-08-01 20:03:20 (UTC)
Last updated:	2011-08-03 05:39:09 (UTC)
Uptime (hh:mm:ss)	33:35:49
Removal date:	2011-08-04 04:51:28 (UTC)
Removal reason:	Domain suspended

C. Testing of security products against Zeus

Once the test system had been used to observe and verify the operation of the two Zeus samples, two commercial security products, Trusteer Rapport (for PC and Mac security) and the PC-Tools suite were evaluated.

1) Trusteer

Trusteer uses a similar method to CaptureBAT, to monitor changes in the operating system, monitors Windows systems at a Kernel level by implementing its own driver. The Trusteer Rapport product website says about their product:

> *"Protects end-user endpoints against financial malware and phishing attacks. By preventing attacks such as Man-in-the-Browser and Man-in-the-Middle, it secures credentials and personal information and stops financial fraud and account takeover. And, it keeps endpoints malware-free by blocking malware installation and removing existing infections."* [19]

The Trusteer product is capable of both detection and mitigation of the Zeus malware, as well as implementation of some other anti-phishing techniques. Trusteer proved capable of detecting both samples of malware tested. After installation on a Windows-7 machine with sample-1 installed, Trusteer lay dormant until the Australia and New Zealand (ANZ) bank website was visited (http://anz.com.au/), specifically the sign-in page for their internet banking. Sample-1 had the URLs for ANZ internet banking within its configuration file, and would have been capturing the form fields relating to customer number and password from the ANZ site, most likely triggering the detection.

2) PC-Tools

"PC-Tools" is an anti-virus (AV) suite that was recently purchased by Norton. When the malware was run against it, PC-Tools detected events on various levels. Malware binary sample-1 was not detected as malicious by PC-Tools, however once executed it triggered several alerts. When the malware attempted to inject itself into the Windows Explorer Process PC-Tools alerted to the fact that another process was attempting to write to the memory space of Explorer. PC-Tools also alerted when the malware attempted to deactivate the windows firewall.

Sample-1 was packaged with a 'cryptor', meaning that detection based on the signature of the binary alone would have been hampered, as the binary would not look like other bots produced with that version of Zeus. However when the malware attempted to place hooks into the address space of Explorer, the anti-virus triggered based on heuristic detection mechanisms.

Sample-2 was built from the Zeus source code and had no such packaging or obfuscation. It was detected solely based on the binary and raised Generic Zbot/Zeus detection by PC-Tools. As the MD-5 hash for this binary had never been submitted to any AV engines, some form of analysis of the compiled code may have triggered the detection.

VI. CONCLUSION

Using open source and freeware tools allowed a simple virtual environment to be built for observing the behaviour of malware. These tools require only general technical knowledge for their set up and understanding making this system useful for a wide student and professional audience.

The well-known, well-researched, malware Zeus was used for system testing and evaluation of the performance of anti-

malware tools. Tests were conducted with PC-Tools and the Trusteer suite of products, and both demonstrated the ability to detect the Zeus variants being utilised. It was found that the use of signature-based detections was not as effective as the heuristic rules used in both the PC-Tools and Trusteer products. Once the malware started to perform the actions it needed to complete its subversion of a browser session, its hand was also revealed to the AV Products, showing that malware such as Zeus, used to subvert the browser can be detected using the techniques discussed in this paper.

The popularity of the Windows platform for its modular and flexible nature appears to be at odds with its security as a platform to perform online banking. The core Kernel of the OS can be compromised because of the lack of separation between trusted OS components and untrusted user programs.

Malware such as Zeus does not need to compromise the Kernel to provide an untrustworthy browsing experience for the user. Although Microsoft has taken measures such as Kernel Patch Protection and User Access Control (UAC), there is still plenty of leeway for malware to subvert the OS and perform internet banking fraud. Without a path of trust from the kernel level up, the trustworthiness of the information being displayed by the OS will always be questionable. If banks want their customers to perform their banking online with confidence, solutions that mitigate the risk posed by the current Windows OS's need to be considered.

REFERENCES

[1] Australian Payments Clearing Association, "Payments fraud in Australia , media release," 22 June 2011. [Online]. Available: http://www.apca.com.au/Public/apca01_live.nsf/ResourceLookup/Media_Release_Payments_Fraud_Statistics_June_2011.pdf/$File/Media_Release_Payments_Fraud_Statistics_June_2011.pdf. [Accessed 24 September 2011].

[2] Symantec, "Cybercrime has surpassed illegal drug trafficking as a criminal moneymaker," 2009. [Online]. Available: www.symantec.com. [Accessed 24 September 2011].

[3] H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Yousse, M. D. and L. Wang, "On the analysis of the zeus botnet crimeware," National Cyber Forensics and Training Alliance Canada, 2009.

[4] G. Ollmann, "Sizing a botnet – "You're doing it wrong!" 25 August 2009. [Online]. Available: http://blog.damballa.com/?p=326. [Accessed 1 May 2012].

[5] R. Naraine, "Zeus returns: FBI warns of 'Gameover' ID-theft malware," 9 January 2012. [Online]. Available: http://www.zdnet.com/blog/security/zeus-returns-fbi-warns-of-gameover-id-theft-malware/10002. [Accessed 30 April 2012].

[6] FBI, "Federal Bureau of Investigation," 6 January 2012. [Online]. Available: http://www.fbi.gov/news/stories/2012/january/malware_010612 [Accessed 30 April 2012].

[7] StatCounter, "Top 5 operating systems from April 2011 to April 2012," 02 May 2012. [Online]. Available: http://gs.statcounter.com/#os-ww-monthly-201104-201204-bar. [Accessed 02 May 2012].

[8] Microsoft, "Microsoft joins financial services industry to disrupt massive Zeus cybercrime operation that fuels worldwide fraud and identity theft," 25 March 2012. [Online]. Available: http://www.microsoft.com/en-us/news/press/2012/mar12/03-25CybercrimePR.aspx. [Accessed 26 April 2012].

[9] SANS, "Top cyber security risks," September 2009. [Online]. Available: http://www.sans.org/top-cyber-security-risks/summary.php. [Accessed 19 September 2011].

[10] DHS National Cyber Security Division/US-CERT, "National vulnerability database version 2.2," 19 September 2011. [Online]. Available: http://nvd.nist.gov/home.cfm. [Accessed 19 September 2011].

[11] K. Jesse, "Exploiting the rootkit paradox with Windows memory analysis," International Journal of Digital Evidence, vol. 5, no. 1, 2006.

[12] B. Andreas, "Torpig – back to the future or how the most sophisticated trojan in 2008 reinvents itself," 16 June 2011. [Online]. Available: http://www.tidos-group.com/blog/?p=362. [Accessed 1 October 2011].

[13] P. Cooke, "Windows 7 security enhancements," 2009. [Online]. Available: http://technet.microsoft.com/en-us/library/dd560691.aspx. [Accessed 01 October 2011].

[14] O. Garnham, "Microsoft to overhaul UAC in Windows 7," Computerworld, 10 October 2008. [Online]. Available: http://news.idg.no/cw/art.cfm?id=E761F194-17A4-0F78-31064BD6CDF046A7. [Accessed 2 August 2012].

[15] P. Gühring, "Concepts against man-in-the-browser attacks," 24 January 2007. [Online]. Available: http://www.cacert.at/svn/sourcerer/CAcert/SecureClient.pdf. [Accessed 2 October 2011].

[16] Terracatta, "TurnKey Linux," 24 January 2011. [Online]. Available: http://www.turnkeylinux.org/forum/general/20101206/insta-snorby-official-snort-snorby-turn-key-solution. [Accessed 29 May 2012].

[17] VirtualBox, "VitualBox manual," 2011. [Online]. Available: http://www.virtualbox.org/manual/ch06.html. [Accessed 01 October 2011].

[18] VRT Labs, "VRT labs - Zeus trojan analysis," [Online]. Available: http://labs.snort.org/papers/zeus.html. [Accessed 01 August 2010].

[19] Trusteer, "Trusteer Solutions," 2011. [Online]. Available: http://www.trusteer.com/solutions. [Accessed 10 October 2011].

[20] Fiddler, "Introducing Fiddler," 02 May 2012. [Online]. Available: http://www.fiddler2.com/fiddler2/. [Accessed 02 May 2012].