Understanding the Threat of Banking Malware

Najla Etaher and George R S Weir

Department of Computer and Information Sciences
University of Strathclyde
Glasgow G1 1XH

{najla.etaher, george.weir}@strath.ac.uk

Abstract: Malware is a general term for all malicious and unwanted software. Such software poses a major security threat to the computer and Internet environment. As an increasing number of people use the Internet in their daily life, inevitably users become subject to malware threats. In the field of digital forensics, malware analysis has become a significant discipline. Malicious software is becoming ever more common, but also continuously more profit driven, stealthy, and targeted, often organised by illegal associations. Furthermore, malware continues to evolve in its sophistication and there are several different types of banking malware that pose a very serious threat to bank customers. This paper presents an overview of techniques, issues, and examples from the area of malware detection. In particular, we describe Zeus as a case study in banking malware. The sophistication and adaptability of such malware presents a lasting and pernicious threat to end-users and organisations. Despite this danger, we argue that an understanding of the infection mechanism coupled with circumspect behaviour on the part of the end-user can contain such malware threats.

Keywords— **malware, malicious software, Zeus, security threats, malware detection.**

1 Introduction

Malware is software that when executed deliberately performs destructive actions intended by the software author without the permission of the owner. Often such programs are designed to disable, breakdown, and damage the computer system or the network (Zolkipli *et al.,* 2010). This threat appears to be rising day by day as a result of sophisticated communication networks and computing technology (Rieck *et al*., 2008).

Malware and related network security threats have become a regular occurrence. These have been responsible for misdeeds all over the world, including Internet fraud activities, compromise of client's private information, data theft, and other types of cybercrime, which all highlight the inherent vulnerabilities and weaknesses of software platforms. Consequently, there is increasing concern for network security, with a major focus on malware detection. With malware equipped with the potential of damaging a computer system, infiltrating and breaking down the machine without user's knowledge, data and codes require protection against replacements and modifications. This also gives rise to security control applications, which become an integral part of computer systems in today's world (Zolkipli *et al*., 2011).

Recent work on malware detection suggests that malware propagation continues to increase at an alarming rate. The intensive use of the Internet and networks enhances the scope for malware to proliferate and also the effectiveness of this kind of software.

Most malware needs some form of user acceptance to propagate. As such, user awareness plays a very important part in the risk of malware infection, but Internet users are easy targets. Many users are completely unaware of the security risks and often do not know they have malware on their devices. Moreover, they do not have adequate knowledge on how to securely manage and protect their information systems. This lack of awareness and inability to take appropriate security measures increases the prevalence and the impact of malware.

2 Malware Types

There are various types of malware that have been observed in the Internet. Each class is different, and has a different effect depending on the host machine. Some types of malware can be easily traced, some are

extremely complex and difficult to disable, and others are downright destructive. The following are examples of prevalent malware types:

- *Viruses* are computer programs that are self-replicating; they modify the content of the files on the victim computer by hiding themselves within other seemingly inoffensive programs in order to perform a malicious action. Viruses copy themselves from machine to machine through media for example a USB device (Distler *et al.,* 2007; Vinod *et al*., 2009).

- *Worms* are usually small self-replicating and self-contained computer programs which perform destructive actions by invading computers on a network. Worms have the ability to propagate and self-replicate, similarly to viruses, although both are entirely different types of malware. Worms replicate and propagate directly through networks and they are designed to perform the infection action autonomously without human help.

- *Trojan Horses* are apparently harmless computer programs and may appear to be useful and entirely functional, but in fact they contain harmful components that will insert themselves into a machine and install an illicit or malicious action on it. Trojan horses require human assistance in order to spread, due to the fact they unable to self-replicate as worms and viruses. The most common one is Zeus which is a banking Trojan. The paper will present it as a case study.

- *Spyware* is any computer program that is installed on a computer in order to transmit, track and report data and information regarding the activities of an internet user without their consent. Spyware is often bundled with free software and automatically installs itself with the program that the user was intended to use.

- *Adware* is software that installs to provide advertisers with information about the browsing habits of the users, consequently allowing the advertiser to offer targeted ads.

- *RATS and Backdoors:* RATS are Remote Access Trojans, also called Backdoors. They bypass standard security controls in order to give unauthorised access to an attacker.

- *KEYLOGGERS* are a mainly nefarious type of malware, in that their goal is to secretly collect critical information such as passwords, logins or other sensitive information that are manually entered from a client device to transmit it back to the source of the infection or to a botnet controlled by that source.

- *Rootkits* are the most difficult to detect types of malware, as their goal is to entirely hide malware from the client as well as security software that can detect malware. Some rootkits are impossible to detect even with forensics. They run at the lowest levels of a specific operating system for which they are designed. Their complete function is to hide malware and only noticeable symptom of their attendance is unexplained reboots, or blue screen crashes.

- *Droppers* are a compressed package of malware. In order to decrease suspicion of surreptitious download in progress, their design dictates that they be as small as possible. Droppers are often hidden inside a document, email, or other compressed files. Their only purpose is to get access into a system. Once installed they will download other file components needed for the malware to perform. They are generally compressed with a particularly obfuscated wrapper which is designed to avoid and elude detection by anti-virus software by some methods such as encryption.

- *Exploits* are specific instances of malware designed to exploit website vulnerabilities, weaknesses and design errors in existing operating systems or software. These exploits will perform unauthorised actions on the victim's device and may exploit the poorly written code of Java applets, JavaScript or PDF files or documents (McAleavey, 2013, Huang *et al.,* 2011, Distler, *et al.,* 2007, Idika *et al.,* 2007).

3 Malware Analysis and Detection Techniques

Equipped with knowledge of malware's capabilities, malware detection is an area of major importance, not only to the research community but also to the public. Malware is continuously increasing in complexity, with most malware writers using more sophisticated hiding techniques to avoid detection tools. This makes manual detection of programmatic faults time consuming and impractical. According to Symantec, over one million new malware pieces are created every day (McAleavey, 2013). The most complex hiding techniques are polymorphic, metamorphic and packers. For these reasons, there is a strong need to develop further detection methods and solutions in order to avoid malware threats and attacks.

Malware detection identifies whether code is genuinely malicious or benign. In order to identify different malware executables, it is important to understand the behaviour of its aspects; this can be done by executing the malware binary. A crucial aspect of efficient malware detection lies in the ability to correctly handle obfuscated malware. Detection is performed through the use of a detection system that works to recognise malicious software. This takes place through analysis of signatures and use of other techniques, such as heuristics parameters (Zolkipli et al., 2011, Idika et al., 2007, Mathur et al., 2013).

This section describes existing malware detection methods, and presents a series of techniques, issues, and examples relating to malware detection. Currently malware often aims to avoid detection sophisticated methods, such as packing and obfuscation. The use of these methods creates many difficult problems in the field of malware detection, the most prominent one being a zero-day attack and false positives rates. Detection of zero-day threats is at an all-time low. A zero-day attack is a piece of malware which is new enough that it has not yet been detected by any anti-malware company and therefore they do not have a signature for it (Elhadi et al., 2012).

Traditionally, malware detection is built upon two techniques: signature-based and behaviour-based. Each of these has specific features that can be applied through static, dynamic or hybrid analysis.

A. Signature Based Techniques

Signature-based techniques use a series of commands designed specifically for the malicious software, which in turn produces the signature for the malware. Each signature can be captured by specialised researchers within laboratory conditions. The aim is that every signature produced is capable of malware recognition, with emphasis on detection of malicious behavioural patterns. Signature-based techniques are widely used by antivirus detection tools and used to carry out investigations on malware binary disassembled codes, which in turn produces the signature. There exist countless debuggers and disassemblers for this purpose. Once this has taken place, the code is analysed to extract those features necessary for populating the signature of any group of malware. The aim of this method is to replicate the malicious behaviour and use this in detection. This model frequently gets referred to as a signature.

The main advantage of a signature-based technique is its utility and efficiency. However, the biggest drawback is the difficulty presented in handling new malicious software. Also, it cannot offer the opportunity to understand and shed light on malware threats because it ignores the malware functionality and its goals. The reason for this is malware detection software relies only on producing matches and not the behaviour of malicious code or its objectives. Clearly, there is a need to understand malware behaviours and goals in order to design and apply avoidance and evasion mechanisms capable of operating successfully within data networks and computer systems (Zolkipli et al., 2010).

B. Behaviour-Based Techniques

Behaviour based techniques' main purpose is to analyse the behaviour of recognised malicious programs. In addition, the basis and target addresses of the malware are included in these behaviours (Mathur et al., 2013; Elhadi et al., 2012; Idika et al., 2007).

Both malware detection techniques have benefits and drawbacks. Signature-based techniques take less time for scanning and give few false positives. On the other hand, such approaches cannot deal with unknown

malware and cannot cope with easy obfuscation in static analysis. One advantage of behaviour-based detection is that it gives best results in detecting polymorphic malware (Elhadi *et al*., 2012; Mathur *et al.,* 2013).

There is reason to suppose that combining malware detection approaches can give improved detection. For example, dynamic techniques may be used to analyse the file if static techniques fail to generate significant insight. Moreover, infected files can be analysed using both signature and behaviour based techniques to obtain better results (Elhadi *et al*., 2012; Mathur *et al.,* 2013; Vinod *et al.,* 2009).

Some researchers on malware seek to address the limitations of signature-based approaches and improve detection rates using techniques such as call graphs, control flow graphs, machine learning and data mining techniques and Objective-Oriented Association (OOA). Other data mining and analysis techniques such as finite automaton, and neural networks have also been employed in order to improve behaviour-based detection techniques. Fig. 1 illustrates the organization of malware detection techniques (Elhadi *et al*., 2012).

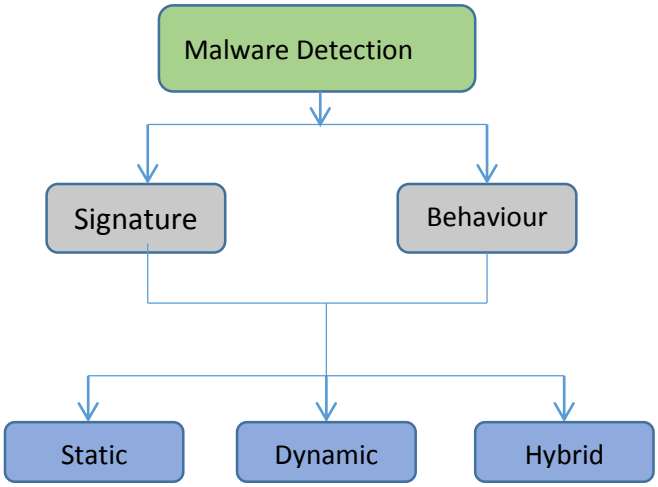

Fig.1: The Organization of Malware Detection Techniques

Static analysis occurs when the infected file undergoes analysis without any execution. It was the first attempt to detect malicious software. This analysis can extract information in a low-level for example, system call analysis, (CFGs) control flow graphs, and (DFGs) data-flow graphs. By using various tools, this information can be collected by decompiling or disassembling the infected files. In order to evade auto malware execution it is sometimes better to analyse infected files in a dissimilar environment. Applying static analysis allows for the retrieval of safe, fast and low false positive rates. All these advantages help to get fundamental information for the purpose of analysis. In contrast, unknown malicious programs that use obfuscation systems are difficult to analyse statically, proving that this methodology is not sufficient for every scenario (Elhadi *et al.,* 2012).

In order to accomplish a detailed analysis of the malware code and to provide an internal opinion of the malware functionality which is referred to as static analysis, software disassemblers and debuggers (such as IDA Pro and OllyDBg) can be used. In contrast, dynamic analysis runs the malware and detects the interaction of the running malware with the computer from a behavioural perspective (Verma, *et al.,* 2013).

Dynamic analysis relates to analysing infected files during its execution within a simulated environment. In order to analyse the file's malicious functions, this environment can be a debugger, an emulator or a virtual machine. With malware developers using tools like anti-virtual machines to hide their malware functions, the static analysis environment becomes unseen to them. In addition, dynamic analysis will fail to detect malicious activities if the malware changes its behaviour. This is due to the fact that a particular execution path can be examined in every attempt made.

Both static and dynamic have advantages and disadvantages. Static analysis is fast and safe, whereas dynamic analysis is neither fast nor safe. Static analysis is good for analyzing multipath malware while this can be difficult using dynamic analysis. Static analysis faces difficulties to analyse unknown malware, whereas dynamic analysis

is good in detecting unknown malware. In addition, static analysis gives a low level of false positives (Elhadi *et al*., 2012; Mathur *et al*., 2013).

Hybrid analysis is a combination of static and dynamic analysis. The signature specifications of malware codes are analysed initially then, for the improvement of complete analysis, they are combined with other behavioural parameters. As a result, the limitations of both dynamic and static analysis can be alleviated by use of hybrid analysis (Mathur *et al*., 2013). There have been several techniques that used hybrid detection for analysing and controlling malware execution in a newer, well-organized way, such as the HERO technique (Guo *et al*., 2010). Despite being an effective technique, further developments are needed in future for better detection correctness, wider detection range, and lower false alarm rates. Nevertheless, the use of such a hybrid technique can obtain better results than applying either static or dynamic analysis separately.

4 Zeus

4.1 Zeus Overview

In the crime-ware world, a global threat for banking organizations is represented by financial botnets that purposely intend to perform financial fraud and other critical information from a client's computer without the owner's consent (Riccari et al., 2010). Generally, there are several types of banking malware; however, all types have the same malicious intent. A common example of banking malware is Zeus. A significant feature of Zeus is that it is available as source code and not just as executables. In 2011, the source code of Zeus was released to the public. Predictably, this resulted in an explosion of new Zeus variants.

The Zeus Trojan, also called Zbot, WSNPOEM, NTOS, or PRG is the king of malicious software in the financial industry, in both its effectiveness and infection rate. Symantec calls this malware "Zeus, King of the Underground Crime-ware Toolkits". In recent reports, Zeus botnets have been found responsible for 44% of online malware infections during financial transactions and for approximately 90% of global banking fraud (Alazab et al., 2012). Zeus is mainly a crime-ware tool that is aimed at stealing users' online banking credentials. Furthermore, the Zeus Trojan is extremely dynamic and applies obfuscation techniques such as polymorphic, metamorphic encryption and packers in a network of bots. In order to defeat signature-based detection techniques, Zeus re-encrypts itself automatically in each infection thereby creating a new signature. Thereby, Zeus poses a serious risk since it is able to hide malicious intent and can effectively avoid malware signature detection engines.

Zeus is a Trojan horse which penetrates large numbers of computers to steal data by logging keystrokes and spreads copies of itself to other computers via instant and email messages. Once successfully installed, hackers can control and monitor infected devices to obtain access to unauthorised data such as online accounts and credentials.

As previously mentioned, Zeus is the most significant banking malware currently in existence. It is a toolkit that is used to make a particular strain of Trojans designed to damage and steal information. Stealing details of online banking and other login credentials is the major focus of Zeus.

The Zeus kit can be obtained from underground forums with older versions, available for free, and the newest versions costing many thousands of dollars (Wyke, 2011). The impact of Zeus infection can be very costly to an organization and differs to that of individuals. For example, stealing online banking details, or theft of personal login details can feel terrible to an individual, whereas the impact of infection for an organization can be devastating and felt on a much larger scale.

4.2 Brief History of Zeus

Since Zeus first emerged in 2007, it has continued the same in its goal for information theft, however, there have been several obvious changes in how it addresses this aim. Zeus is simple to use and only requires minimal technical knowledge (Wyke, 2011). From 2007, several Zeus Trojan variants have been documented. For example, there were about 3.6 million infected computers of Zeus in the USA alone during the period of 2009 and 2010 and this era is considered the most productive period for Zeus (Binsalleeh et al., 2010).

The Zeus crime-ware toolkit has grown to be one of the preferred tools for attackers due to its competitive price and because it has a user-friendly interface. The Zeus toolkit control panel manages and monitors the infected systems; it also controls the gathered stolen data and information. In addition, attackers use this crime-ware tool to steal important information such as users' credentials (Binsalleeh et al., 2010).

According to several research labs (Alazab et al., 2012), the Zeus botnet is still developing with new plugin releases which can infect even the newest operating systems. For example, in order to take the threat of Zeus to a different level, and to generate more sophisticated bots, Zeus has recently combined with the 'Spy-Eye Trojan' released in 2010. This combined version has two versions of a control panel that are utilised for managing compromised systems and committing fraud.

4.3 Functionality

Zeus Trojan's key purpose is to steal online credentials as specified by the attacker. Some of the many actions it performs are information system gathering, online credential information stealing, command and control (C&C) server contacting and protected storage information stealing (Falliere *et al.,* 2009). Although technically Zeus is a crime-ware kit designed to steal money, from other perspectives, it is a new online illegal business enterprise. Within this enterprise different organizations can cooperate with each other in order to commit entire online fraud and theft. This becomes a component part of a fully organized cybercriminal organization. In fact, Eastern European Organized Crime is the cybercriminal underground that is behind Zeus. Generally, the top bulletproof-hosted Zeus domains exist in Ukraine and Russia (Micro, 2010).

4.4 Zeus Crime-ware Tool Components

In general, to gain monetary profits, the Zeus toolkit takes control of devices causing them to perform as spying agents. There are five components that make up the general structure of this toolkit:

1. Control panel: this manages, controls and gathers the stolen data and information, it also consists of PHP scripts which observe the botnet and display the information to the botmaster.

2. A builder: two files are generated here; the 'bot.exe' which is the malware binary and the 'config.bin' which is the encrypted configuration file.

3. Configuration files: they involve two files; the 'config.txt' which is the configuration file. The crucial information is listed in this file; and the 'webinjects.txt' which is the web injects file. This file is also responsible for the recognition of targeted websites and defines the content injection rules. Moreover, the configuration files modify the botnet parameters.

4. Generated encrypted configurations files 'config.bin'. An encrypted version of the botnet configuration parameters is held in these files.

5. Generated malware binary files 'bot.exe', these files infect the victims' devices as the binary of the bot (Binsalleeh, *et al*., 2010).

As the Zeus Trojan is designed to steal sensitive information stored on devices or transferred through web browsers and protected storage, it carries a very light foot print. Once the victim's computer has been infected, the stolen data gets immediately sent to a bot 'C&C' (a command and control server) through an encrypted 'HTTP POST' request, whereby the stolen data is saved. Furthermore, Zeus allows cybercriminals and hackers to inject content into the web page of a bank as it is shown in the infected computer browser. Hackers can control the infected systems remotely, as the stolen data is sent to a drop server controlled by a cybercriminal known as the botmaster.

5. Conclusion

Writers of malware are becoming more and more profit driven and are incorporating methods in order to make their code as stealthy and undetectable as possible. Malware is being written by professional programmers who

have a very good understanding of digital forensic techniques and endeavour to create forensic analysis as difficult and complicated as possible. The more vulnerable the technology, the more likely it is to be exploitable through malware.

This paper has looked at the varieties of malware and the ways in which they pose major security threats to users. To help understand the risks, we have detailed how malware operates and propagates, and how malware may be addressed through the latest malware detection techniques. The study of malware and techniques for its detection is ongoing as a specialisation for forensic analysts. Generally, in malware detection there is no single technique best suited to detect all types of malicious software. Moreover, countermeasures cannot detect unknown malware or unknown signatures that are unique to a specific malware. For example, signature-based detection is disadvantageous as it cannot be used to detect novel attacks. As malicious software poses a major security threat to Internet users, there is a clear need for end-users to understand the nature of malware and the incipient danger of malware threats.

Using obfuscation techniques, developers of malicious software are able to create serious threats that render traditional anti-virus techniques obsolete and outdated. Detection techniques that rely on software signatures are a weak defence against such threats and also prove ineffectual against unknown threats. Anomaly and similar behavioural based detection techniques are likely to be more effective against such adversaries.

Since Zeus has become the most common banking crime-ware kit in the criminal underground world for wholesale financial theft, all Internet and mobile device users should be fully aware of the threat. Importantly, the 'common user' should be educated to increase the likelihood that precautions will be taken against infection.

A Zeus infection is most likely through receipt of spam email claiming to come from a major organisation, such as an insurance agency, internal revenue service, Microsoft or Facebook (Symantec, 2014). Clicking on a link within such an email results in a 'drive by' compromise (if the user's computer is not already protected).

The ready availability of Zeus in source code or kit-form ensures that the malware will continue to be used by cybercriminals to steal personal information and intercept online financial dealings. Such malware continues to evolve and alter the means by which it infects computers. Principally, it relies upon unpatched or zero-day exploits to gain another bot in its net. Such exploits depend for their effect upon users lacking fully patched software (vulnerability through legacy exploits), upon users misguidedly activating email attachments (through Trojan attachments), or upon users clicking on Web links that lead to unexpected sites (through drive-by zero-day exploits). In each of these cases, security awareness on the part of the end-user can successfully avoid such infections and protect against data theft. Despite the sophistication of Zeus and other banking Trojans, a little security knowledge and circumspect behaviour can successfully contain such malware.

REFERENCES

Alazab, M., Venkatraman, S., Watters, P., Alazab, M., & Alazab, A. (2012). Cybercrime: the case of obfuscated malware. In *Global Security, Safety and Sustainability & e-Democracy* (pp. 204-211). Springer Berlin Heidelberg.

Binsalleeh, H., Ormerod, T., Boukhtouta, A., Sinha, P., Youssef, A., Debbabi, M., & Wang, L. (2010, August). On the analysis of the zeus botnet crimeware toolkit. In *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference on* (pp. 31-38). IEEE.

Distler, D., & Hornat, C. (2007). Malware Analysis: An Introduction. *Retrieved (2009, Novembe*, *8*.

Elhadi, A. A., Maarof, M. A., & Osman, A. H. (2012). Malware Detection Based on Hybrid Signature Behaviour Application Programming Interface Call Graph.*American Journal of Applied Sciences*, *9*(3), 283.

Guo, H., Pang, J., Zhang, Y., Yue, F., & Zhao, R. (2010, October). Hero: A novel malware detection framework based on binary translation. In *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on* (Vol. 1, pp. 411-415). IEEE.

Huang, H. D., Lee, C. S., Kao, H. Y., Tsai, Y. L., & Chang, J. G. (2011, April). Malware behavioral analysis system: TWMAN. In *Intelligent Agent (IA), 2011 IEEE Symposium on* (pp. 1-8). IEEE.

Idika, N., & Mathur, A. P. (2007). A survey of malware detection techniques.*Purdue University*, 48.

Falliere, N., & Chien, E. (2009). Zeus: King of the Bots. *Symantec Security Respons e (http://bit. ly/3VyFV1)*.

Micro, I. T. (2010). ZeuS: A persistent criminal enterprise. *Retrieved February*,*6*, 2011.

Mathur, K., & Hiranwal, S. (2013). A Survey on Techniques in Detection and Analyzing Malware Executables. *International Journal*, *3*(4).

McAleavey, K., 2013. Malware analysis – Detecting and Defeating Unknown Malware (2013). *eForensics Magazine*, Vol.2, No.1.

Riccardi, M., Oro, D., Luna, J., Cremonini, M., & Vilanova, M. (2010, October). A framework for financial botnet analysis. In *eCrime Researchers Summit (eCrime), 2010* (pp. 1-7). IEEE.

Rieck, K., Holz, T., Willems, C., Düssel, P., & Laskov, P. (2008). Learning and classification of malware behavior. In *Detection of Intrusions and Malware, and Vulnerability Assessment* (pp. 108-125). Springer Berlin Heidelberg.

Symantec. (2014). Trojan.Zbot. http://www.symantec.com/security_response/writeup.jsp?docid=2010-011016-3514-99

Vinod, P., Jaipur, R., Laxmi, V., & Gaur, M. (2009). Survey on malware detection methods. In *Proceedings of the 3rd Hackers' Workshop on Computer and Internet Security (IITKHACK'09)* (pp. 74-79).

Verma, A., Jeberson, W. & Singh, V. (2013) A LITERATURE REVIEW ON MALWARE AND ITS ANALYSIS. *IJCRR*, 5 (16), 71-82.

Wyke, J. (2011). What is Zeus? *A Sophos Labs technical paper.*

Zolkipli, M. F., & Jantan, A. (2010, September). Malware behavior analysis: Learning and understanding current malware threats. In *Network Applications Protocols and Services (NETAPPS), 2010 Second International Conference on*(pp. 218-221). IEEE.

Zolkipli, M. F., & Jantan, A. (2011, March). An approach for malware behavior identification and classification. In *Computer Research and Development (ICCRD), 2011 3rd International Conference on* (Vol. 1, pp. 191-194). IEEE.