Quest Journals
Journal of Software Engineering and Simulation
Volume 4 ~ Issue 2 (2018) pp: 01-05
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Verifying the Detection of Concealed Malware

Liberios Vokorokos[1], Matúš Uchnár[1], Juraj Mihaľov[1]
[1]*Department of Computers and Informatics, FEI, Technical University of Košice, Slovakia*
Corresponding Author: Liberios Vokorokos

ABSTRACT : *This work is aimed at the testing of various malware concealment methods, namely packing and encryption. At first, the malware sample is acquired and tested. Then the sample is concealed using packers and encryption software and tested again to see changes in the detection rates. This paper also briefly describes antivirus detectors and their methods to detect different malware types and families.*
KEYWORDS – *Crypter, Detection, Encryption, Malware, Security*

Received 20 July, 2018; **A**ccepted 04 August, 2018 © The author(s) 2018. **P**ublished with open access at **www.questjournals.org**

I. INTRODUCTION

There are many methods and techniques which are used for malware concealment in the system (compression, encryption, protectors, rootkits, etc.) [1]. The aim of this work is to demonstrate commonly used malware encryption techniques and the detection difference between their original form and encrypted form via online tool VirusTotal. The malware used for this testing will be Trojan.ZeroAccess (commonly known as Max++) [2], which is fairly known malware and most antivirus software are able to detect it in non-encrypted form.

II. MALWARE ANALYSIS

Malware analysis studies individual malware components and also its behavior in the infected environment. This analysis contains lots of methods and tools. It can be divided into two categories – static methods and dynamic methods [3]. While both categories have the same goal – detection of malware – they use different approaches and require different resources. The base difference is that during static analysis, the malware is not executed, while during dynamic analysis, the malware is being executed in the safe environment.

Static analysis mainly consists of methods which are looking for a string which is a unique malware signature. While early antivirus software were just a simple scanners, there was a need for improvement because malware creators started using more sophisticated concealment methods. Modern antivirus software use the combination of many methods, because there is not just a single universal method for various malware detection. The antivirus detection mechanisms could be divided into 4 categories [3],[4]:

- First generation scanners,
- Second generation scanners,
- Algorithmic scanning methods,
- Code emulation.

The comparison of these methods using various aspects could be seen on the Table 1.

Table 1 Comparison of antivirus methods

			Promise of perfect desinfection	Scanning speed-up	Generic virus detection	Detection of new or unknown viruses	Encrypted/polymorphic viruses	Metamorphic viruses	Macro viruses	Odds of false positive	Odds for false negative
First generation scanners	Simple string scanning		Yes	No	No	No	No	No	No	Low	Low
	Optimization methods	Wildcards	Yes	No	Yes	No	No	No	No	Low	Low
		Mismatches	Yes	No	Yes	No	No	No	No	Low	Low
		Gneric detection	Yes	No	Yes	No	No	No	No	Low	Low
	Bookmarks		Yes	No	No	No	No	No	No	Very low	Low
	Speed-up methods	Hashing	Yes	Yes	No	No	No	No	No	Low	Low
		Top and tail scanning	Yes	Yes	No	No	No	No	No	Low	High
		Entry/fixed point	Yes	Yes	No	No	No	No	No	Low	Low
Second generation scanners	Smart scanning		Yes	Yes	Yes	No	No	Yes	Yes	Low	Low
	Skeleton detection		Yes	No	No	No	No	No	Yes	Low	Low
	Nearly exact identification		Yes	No	No	No	No	No	No	Very low	Very low
	Exact identification		Yes	No	Yes	No	No	No	No	Null	Null
	Heuristic analysis		No	No	Yes	Yes	No	Yes	Yes	Very high	Low
Algorithmic scanning methods	Generally		Yes	No	Yes	No	Yes	Yes	Yes	Low	Low
	Optimization methods	Filltering	Yes	Yes	No	No	No	No	No	Low	Low
		Static decryptor detection	No	No	No	No	Yes	No	No	Very high	Very high
		X-ray scanning	Yes	No	No	No	Yes	No	No	Low	Low
Code emulation	Generic detection		Yes	No	No	No	Yes	No	No	Low	Low
	Dynamic decryptor detection		No	Yes	No	No	Yes	No	No	Low	Low

In order to test the malware sample, just a single antivirus program cannot be used, because the test would not be objective. VirusTotal is an online service which contains a database of most known and used antivirus programs. The sample can be tested by up-to-date antivirus database and the history of testing for each individual file is available. Most antivirus software use static analytic methods, but some of them use also dynamic methods to ensure higher detection rate [5].

III. ENCRYPTION

There have been at least 360 000 new malicious files detected every day in 2017 [1]. While this number seems very high, most of them are just known malware families which are using various concealment methods in order to avoid detection [6]. This way the known malware may seem like a new one and may be able to avoid detection.

Packers are very popular amongst malware creators. It is a method which can conceal malware from detecting by both static and dynamic detection methods [7]. Packer is an executable program which compresses a file and then packs it in a new executable file. If we take a known malware, antivirus software should be able to detect it in its original form. But if the malware is packed, its signature is different and the antivirus file may not be able to detect it until the file is extracted. The package consists of two base components. The first are many data blocks, which are basically a compressed original executable file. The second part is the stub, which can dynamically recover the original executable file. The execution of the original file is usually unchanged and begins from the original entry point (OEP) without performance sanctions on execution (after unpacking) [8]. Many packers use various techniques in order to avoid reverse engineering. For example they can use multi-level packing or sophisticated self-debugging techniques to prevent other debugging tool to access parent process, because of the simple fact that in one user environment, only one process can be debugged at a time [9]. The whole packer process can be seen on the Figure 1.

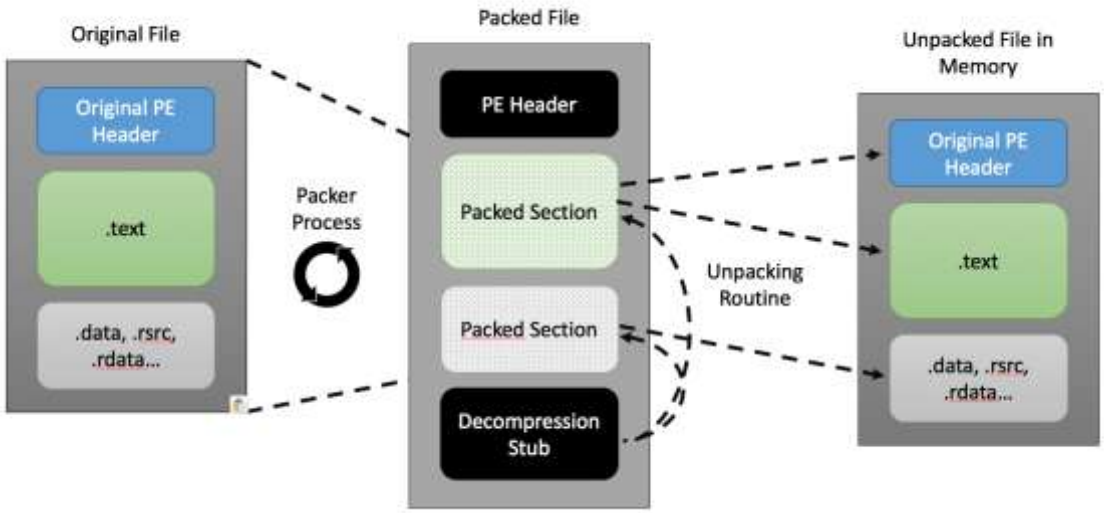

Figure 1 Packer process [10]

The idea behind crypting is simple, just like with packers. If we change the malware code in order to eliminate the known malware signature string, it can avoid detection by antivirus systems. The software which use this method is called crypter. The crypter takes the binary executable file on the input and encrypts it using various algorithms (for example DES, 3DES, AES, or it could use simple cyphers like XOR) [11]. The behavior of the file remains unchanged except for the need to be decrypted. After file encryption by chosen algorithm and randomly generated key, the program creates a stub which is an entry program which contains everything it needs for decryption and execution of the original file. The decryption process can be also bound to various conditions in order to avoid unpacking while being tested by antivirus software [12]. This approach allows to bypass antivirus detection mechanisms because the encrypted file has unknown signature and thus is not detected by static and heuristic methods. It can only be detected by dynamic methods and also some heuristic methods aimed at detecting the decryption algorithm commonly used in malware encryption [13]. The difference between scantime and runtime crypters can be seen on the Figure 2.

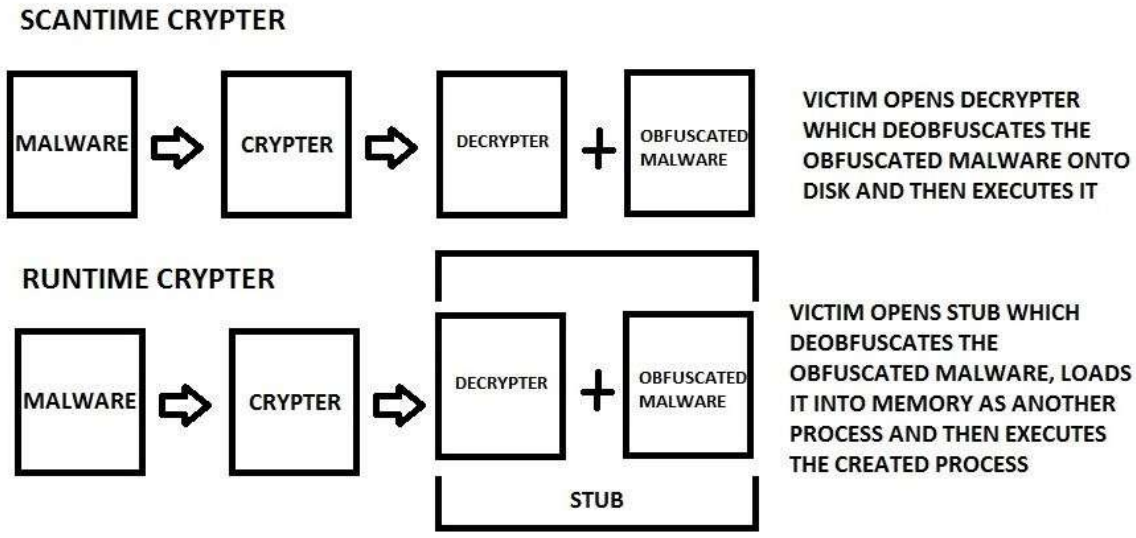

Figure 2 Scantime and Runtime crypters [14]

While both methods are very good at malware concealment against static detection methods, they are vulnerable against dynamic detection methods. This can be solved by binding the execution of decompression/decryption algorithms to some conditions which are specifically crafted in order to not be fulfilled in the testing environment, usually exploiting antivirus limitations during dynamic analysis [15].

IV. ENCRYPTED MALWARE ANALYSIS

At first, we have tested the original malware file via VirusTotal tool. It was detected by 53 out of total 61 antivirus programs. The malware md5 hash is "d8f6566c5f9caa795204a40b3aaaafa2" which matches the VirusTotal malware description.

For packing, we have used UPX packed, which is one of the most used packers. The compressed UPX file is just slightly smaller than original file on the hard drive, but is slightly bigger in the memory where it decompresses. The md5 hash ("2fe1315a51d4bd5c0b948337d8b40827") is different than the original file. Then we tested the compressed file containing malware via VirusTotal and it was detected by 28 out of 67 antivirus programs. As we can see, there have been big decrease in the detection rate and also some scanners only detected the stub as malicious and not the payload.

For the second chosen malware concealment method, encryption, we have chosen one of the free crypter tools – File Crypter. We have used various algorithms for the encryption and also variable stub settings. We have discovered that the chosen encryption algorithm had very little impact on the detection rate, because the scanners were mainly focused on decryption subroutine detection, or they ran dynamic analysis, so we set the algorithm to AES. At first, we have tested just file encryption without using any stub. The detection rate was 48 out of 66, which was only minor change against original, unencrypted form. Then we encrypted the file using standard stub implemented in the program. The detection rate for this test was 34 out of 66, which is slightly better, but not as good as packing method. At least, we have used external stub. For this test, we have chosen some non-malicious program as a base for this stub. Then we injected the stub into this program and tested the whole file. This approach has proven to be the best, as the detection rate was 2 out of 67. We have also tested all the files via dynamic analysis to ensure that the behavior of the malware remains unchanged. The comparison of the individual concealment methods could be seen on the

Table 2.

Table 2 Comparison of the detection rates of various concealment methods

File	Stub	Algorithm	Detection rate
Original Max++	-	-	53/61
Packed Max++	-	-	28/67
Encrypted Max++	-	AES	48/66
Encrypted Max++	Stub.exe	AES	34/66
Encrypted Max++	External	AES	2/67

V. CONCLUSION

The aim of this work was to demonstrate how it is possible to conceal known malware. At first, we have tested the original malware to ensure that it is known and detected by majority of the antivirus software available. Then we have tried to conceal the malware sample using packing and encryption techniques. Packing has proven to be semi-useful as the detection rate lowered by a significant amount, but still the major antivirus scanners were able to detect the file as malicious. The results of the encryption techniques varied and they were mainly dependent on the stub. It has proven to be the best to use external program as a stub to ensure lowest detection rate. It is important to note that while these methods work, if the concealed malware gets widespread, it can get manually patched and new detection string could be released for this version, so it can be detected via standard static analysis.

In the future, we will try to implement some subroutines in the stub to ensure even lower detection rates. They will be mainly focused on avoiding program being decrypted in an antivirus testing environment, which is slightly different and limited as opposed to the real OS.

ACKNOWLEDGEMENTS

This work was supported by KEGA Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under Grant No. 077TUKE-4/2015 "Promoting the interconnection of Computer and Software Engineering using the KPIkit" and Grant No. 003TUKE-4/2017 "Implementation of Modern Methods and Education Forms in the Area of Security of Information and Communication Technologies towards Requirements of Labour Market" and by the Slovak Research and Development Agency under the contract No. APVV-0008-10.

REFERENCES

[1]. Benzmüller, R. Malware trends 2017. Available online: https://www.gdatasoftware.com/blog/2017/04/29666-malware-trends-2017.

[2]. Symantec. Trojan.Zeroaccess. Available online: https://www.symantec.com/security_response/writeup.jsp?docid=2011-071314-0410-99&tabid=2.
[3]. Rad, B. B. et al. Evolution of Computer Virus Concealment and Antivirus Techniques: A Short Survey. International Journal of Computer Science Issues, 2011. Vol. 8, Issue 1, January 2011. ISSN 1694-0814. pp. 113-121.
[4]. Hurtuk, J., Madoš, B. Simple obfuscation tool for software protection. Journal of Software Engineering and Simulation. Vol. 3, no. 7 (2017), p. 14-21. - ISSN 2321-3809
[5]. Chovancová, E., et al. Securing distributed computer systems using an advanced sophisticated hybrid honeypot technology. Computing and Informatics. Vol. 36, Issue 1 (2017), p. 113-139. - ISSN 1335-9150
[6]. Aycock, J. Computer Viruses and Malware. Springer, 2006 ISBN 978-0-387-30236-2. p27.
[7]. Flake, H. Structural comparison of executable objects. Proceedings of the IEEE Conference on Detection of Intrusions, Malware and Vulnerability Assessment (DIMVA), 2004, pp.161-173.
[8]. Brosch, T. and Maik M. Runtime packers: The hidden problem. Black Hat USA (2006).
[9]. Chiueh, T. and Fanglu G. Automated unpacking of executables packed by multiple layers of arbitrary packers. U.S. Patent No. 7,996,904. 9 Aug. 2011.
[10]. Roccia, T. Malware Packers Use Tricks to Avoid Analysis, Detection. McAfee. Available online: https://securingtomorrow.mcafee.com/technical-how-to/malware-packers-use-tricks-avoid-analysis-detection/
[11]. Rad, B. B., Maslin M. and Suhaimi I. Camouflage in malware: from encryption to metamorphism. International Journal of Computer Science and Network Security 12.8 (2012): p. 74-83.
[12]. Sharif, M. I., et al. Impeding Malware Analysis Using Conditional Code Obfuscation. NDSS. 2008.
[13]. You, I. and Yim, K. Malware obfuscation techniques: A brief survey. Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference. pp. 297-300. IEEE.
[14]. Security-Oriented C Tutorial 0xFB - A Simple Crypter. Wonder How To. Available online: https://null-byte.wonderhowto.com/how-to/security-oriented-c-tutorial-0xfb-simple-crypter-0168089/
[15]. Nasi, E. Bypass Antivirus Dynamic Analysis: Limitations of the AV model and how to exploit them. Sevagas, 2014. p21

Liberios Vokorokos"Verifying the Detection of Concealed Malware.” Quest Journals Journal Of Software Engineering And Simulation, Vol. 04, No. 02, 2018, Pp. 01–05.