An Open Source, Extensible Malware Analysis Platform

P. Michalopoulos[1], *V.* Ieronymakis[2], *M.T.* Khan[3] *and D.* Serpanos[4]

[1,2,4]ISI/ATHENA and University of Patras, Patras, Greece - [4]serpanos@isi.gr

[3]Informatics Systems Institute, Alpen-Adria University, Austria, muhammad.khan@aau.at

Abstract. A malware (such as viruses, ransomware) is the main source of bringing serious security threats to the IT systems and their users now-a-days. In order to protect the systems and their legitimate users from these threats, anti-malware applications are developed as a defense against malware. However, most of these applications detect malware based on signatures or heuristics that are still created manually and are error prune. Some recent applications employ data mining and machine learning techniques to detect malware automatically. However, such applications fail to classify them appropriately mainly because they suffer from high rate of false alarms on the one hand and being retrospective, fail to detect new unknown threats and variants of known malware on the other hand. Since anti-malware vendors receive a huge number of malware samples every day, there is an urgent need for malware analysis tools that can automatically detect malware rigorously, i.e. eliminating false alarms. To address these issues and challenges of current malware detection and analysis approaches, we propose a novel, open source and extensible platform (based on set of tools) that allows to combine various malware detection techniques to automatically detect/classify a malware more rigorously. The developed platform can be fed with malware samples from different providers and will enable the development of effective classification schemes and methods, which are not sufficiently effective without collaboration and the related sample aggregation. Furthermore, such collaborative platforms in cybersecurity enable efficient sharing of information (e.g., about new identified threats) to all collaborators and sharing of appropriate defences against them, if such defences exist.

1 Introduction

Malware is a computer program that intentionally performs undesirable and harmful tasks resulting in compromise of target IT based system resources [1]. The most popular types of such programs include worms, spyware, viruses, ransomware, to name a few. Since last couple of decades, malware has been widely used as a weapon to launch cyber-attacks to target IT based systems by cyber- criminals, e.g. Ukrainian Grid by CrashOverride [2].

---

3 is corresponding author.

These attacks may compromise IT system based internal (e.g., files, memory, communication) and external (e.g. of cyber physical systems) resources. Moreover, these attacks may also steal information, deteriorate system performance and perform legal tasks through unauthorized access, which often result in physical damage to critical information and infrastructure or financial loss [3]. In the last couple of years, malware attacks have stolen approx. 1 billion $, in addition to approx. 1 million £ of bitcoin that were stolen from the victims by a ransomware WannaCry [4]. Furthermore, approximately 2-5 million computers were affected by malware attacks during 2016 [3].

More recently, malware attacks have arisen as main security threats to application users and systems. In order to protect users and systems from such attacks, various defenses have been developed as anti-malware, for instance, Symantec and Kaspersky. However, these anti-malwares mainly employ signature based, heuristic based and hybrid methods to detect such attacks. Signature based methods are given a signature, which is a sequence of bytes characterizing a malware uniquely and identifies if a given software is a malware by looking up the signatures in the software [5]. However, this method is strongly retrospective and fail to detect novel malwares as well as variant of known malwares since malware developers keep changing the same malware code (using developed toolkits [6,7,8]) to avoid detection by employing various techniques, e.g., polymorphism, instruction virtualization, to name a few. Heuristic based method is given rules and patterns (as determined by experts) to identify malware and identifies if a given software as malware by applying rules to analyze the software [9]. These rules are often generic and cover variants of the known malware avoiding false alarms. However, this method is weakly retrospective as it only fails to appropriately detect novel malware. Signature based and heuristic based methods require manual creation of malware samples and rules/patterns, therefore, these methods are error prune on one hand and offer very limited reliability (i.e., suffers from high false alarms), scalability and efficiency on the other hand. These methods have been widely used to detect malware both statically and dynamically.

Since last decade, hybrid methods have been used to detect malware, which are a mix of signature based and heuristics-based methods on one hand and mix of static and dynamic analysis on the other hand [33]. The hybrid systems provide better accuracy however, they fail to detect novel malware, in particular zero-day malware. Recently, heuristics-based method has started employing data mining and machine learning based techniques to automatically detect malware. However, such methods are not robust and suffer from high false alarms. In case, such methods detect an unknown malware, they fail to explain why the detection is malware.

Existing malware detection techniques fail to detect malware appropriately mainly because these techniques aim to detect features (that characterizes a potential malware) that are identified by unreliable malware analysis techniques. Most popular approaches for identifying malware are static and dynamic analysis, which provide impractical abstract features that often result in expensive and false detections. Such techniques are the main reason for inaccurate malware detection. Recently, hybrid analysis techniques have been developed to analyze malware more appropriately and accurately [33].

Since anti-malware vendors are confronted with a huge number of malware samples every day (i.e., 42 million malwares collected by Kingsoft, few years ago [11]), therefore, there is an urgent need for malware analysis tools that can automatically detect/classify malware rigorously. With the fast-growing rate of malware and unwanted code may exceed benign software applications, as reported by Symantec [10]. To address these challenges, we propose a novel, open source and extensible platform (based on set of tools) that allows to combine various malware detection techniques to automatically detect/classify a malware more rigorously. The developed platform can be fed with malware samples from different providers and will enable the development of effective classification schemes and

methods, which are not sufficiently effective without collaboration and the related sample aggregation. Furthermore, such collaborative platforms in cybersecurity enable efficient sharing of information (e.g., about new identified threats) to all collaborators and sharing of appropriate defences against them, if such defences exist.

The rest of the paper is organized as follows: Section 2 sketches various kind of malwares, while Section 3 discusses main malware detection techniques. In Section 4, we discuss popular techniques for malware analysis (static and dynamic analysis). Section 5 highlights the challenges to current malware analysis and detection methods, while Section 6 introduces our proposed malware analysis platform.

2 Malware

In this section, we discuss the popular types of malware.

2.1 Ransomware

Since last few years, ransomware is the most popular malware, which evasively installs on the target machine and executes a crypto-virology attack to adverse the software parts of the machine [12]. Typically, in case of such an attack, the victim has to pay a ransom to recover from the attack through decryption.

2.2 Trojans

Trojan is a computer program that appears to be useful but in fact does malicious job at the backend [13]. For instance, a recent trojan Zeus has stolen banking related information by keystroke logging and form grabbing [14]. The aforementioned trojan compromised approx. 0.075 million FTP accounts including well- known companies, e.g., Amazon, Oracle and NASA, to name a few.

2.3 Viruses

Virus is a code that adds itself to other (legitimate) program and infects the affected parts when the other program executes [15]. Viruses propagate to other computers and can also infect any shared resources, e.g. files.

2.4 Worms

In contrast to a virus, worm executes independently, i.e. without appending itself to some other program [15]. Furthermore, worm can propagate a fully functional copy of itself to other machines. Some popular worms (e.g., MyDoom, Code Red [16]) have reduced various Internet websites access by 10%-50% [17]. Furthermore, the cost of removing one of the earlier worms (i.e., Morris) from infected machines was approx. 1 million $ [18].

2.5 Bot

One of recent malware is bot, which is a computer program which allows its master to remotely control the infected machine/parts [19]. Such malware usually exploits software vulnerabilities or employ social engineering to spread themselves. Some bots form a botnet to infect other machines over the network. More recently Mirai has infected various

machines in Rutgers University, which required university to increase budget to 1 million $ to handle the malware, in addition to increase in tuition fee for that academic year [20].

2.6 Spyware

Like its name, a spyware a computer program that spies user activities without alarming user or prior consent of users and transfers the information to the attacker [21]. The information of interest for such malware could be user accounts, emails, documents, history contents etc. Typically, attacker uses spyware as part of a larger attack.

2.7 Scareware

Scareware is a computer program that tricks user to download it or buy it for useful activity, e.g., as an anti-virus, but in fact, it infects the machine, once downloaded and executed. Such malware often results either in financial loss or threatens user policy by spying personal information.

2.8 Rootkits

Recently introduced, rootkit, is a computer program that hides certain process and information from the user and provides continuously privileged access to its author. They usually instrument API calls in a user mode or tamper with operating system structures to perform malicious activity.

Initial malware authors helped system developers to identify various vulnerabilities in their system by launching malware but now their intent is malicious that either gives (financial, business) loss to the victim or earns some (financial, business) benefit from the victim. Therefore, modern day malware is typically hybrid, i.e. the authors mix several of the abovementioned malwares to write a more powerful and novel malware, which helps attackers to make desired loss to the target victim. In order to achieve such malicious goals in a more robust way, the authors have started using concealment techniques that either evades the detection as malware or makes the malware resilient, i.e. continue malicious operation despite detection. For instance, “obfuscation” allows to hide harmful goal of the malware [22]. Another technique, “polymorphism” enables a malware look different each time it replicates itself [23]. Recently developed, “metamorphism” allows malicious code to change itself, so that the new one has no resemblance with the others but same “goal” malwares [23]. In order to accomplish that, various toolkits have been introduced, which help to automatically create a new or customize a malware, e.g. Zues [24], SpyEye [25]. Furthermore, the malware authors encrypt the malware to even harden malware detection.

3 Malware Detection

In this section, we discuss the major approaches to detect malware.

3.1 Signature-based Detection

In order to defend against the malware, various companies (e.g., MacAfee, Symantec, Kaspersky and Kingsoft) provide anti-malware software. The anti-malware software widely uses the signature-based method to identify the known threats/malware. A signature is a short sequence of bytes that uniquely characterizes any malware [5]. The signatures are created manually by an analyst by analyzing the malware. The signatures are later used to

decide if a new software is a certain type of malware. The process of generating signature is very labor and time consuming on one hand and is error prune on the other hand. Furthermore, this method is retrospective and thus fail to detect any novel malware or a variant of a known malware.

3.2 Heuristics-based Detection

As discussed in Section 2, a malware author can easily evade detection by signature-based method by applying various techniques (e.g. obfuscation, polymorphism, encryption). Therefore, to protect legitimate users and system operations from new malware, heuristic-based method can be used, which identified malware based on patterns and rules that are determined by experts and analysts manually [9]. The rules are typically generic and cover variants of known malwares. However, this method also fails to detect novel malware.

Modern day malware authors have created toolkits that can create/mutate/customize thousands of malwares every day, which cannot be detected from signature-based and heuristics-based methods [10]. Therefore, manual analysis is the main bottleneck in the workflow of malware analysis. Hence, more intelligent and automatic techniques are required to identify novel malware.

3.3 Data Analysis-based Detection

Recently, various data analysis-based methods have been used to automatically detect novel malware. These methods typically employ data mining and machine learning techniques to identify if a software is a novel malware. These methods suffer from high false alarms, which hinders system performance as it prevents legitimate users and system to perform legitimate operations. Furthermore, these methods require more labor and time, because in order to enable legitimate users and operations, a manual analysis is required on top of these methods.

More recently, anti-malware industry has started developed hybrid detection methods that are mix of abovementioned methods, e.g. Comodo's Anti-Virus products [5, 26, 27] and Symantec Anti-Malware products [28]. These methods have better results but still fail to avoid false alarms on one hand and fail to detect novel malware on the other hand.

The accuracy of the above methods lies in the underlying techniques that analyst use to understand the malware's malicious code by observing various features/characteristics of the malicious code. Understanding and analyzing a malware is the real challenge in malware detection. Typically, a malware is analyzed either statically or dynamically, which we discuss in the following section.

4 Malware Analysis

In this section, we discuss various techniques to comprehend and analyze the malware.

4.1 Static analysis

In static analysis, the desired features of malware are identified without executing it. Static analysis can be applied on different representations of the program, e.g. source code, binaries, intermediate code. The goal of the analysis is to infer an abstract model of the source code mainly by analyzing function calls, parameters, information and control flow.

The analysis based on source code is more useful as it helps to identify various vulnerabilities, detect memory flaws and to proves the correctness of the models of program [29, 30, 31].

Statically analyzing binaries is of little help because binaries often do not contain valuable information, for instance, data structure sizes and variable, which is critical and desirable for understanding a malware.

Recently, static analysis has been used to collect useful information about the program, e.g. based on call graphs, to better analyze the malware. The graphs help to understand possible values of various parameters, which can help to develop defense mechanism against specific malware attacks [32]. However, some parts of the program cannot be analyzed statically, e.g. current system date and indirect jump instructions.

Typically, source code of the malware is not readily available because it is either encrypted and obfuscated or is executable, which hinders the applicability of static analysis to analyze a malware. In order to analyze executables of malware, dynamic analysis is helpful, which we discuss in the following subsection.

4.2 Dynamic analysis

In dynamic analysis, a malware is analyzed by executing it and observing the desired features. Dynamic analysis is applied to monitor various characteristics of the program execution, which helps to understand various features of the malware.

Function call monitoring [9] is widely used technique for dynamic analysis of malware to understand control flow of the program. Since each program has functions, where each function performs a specific task. If one knows the function calls in a program, one can understand the potential intent of the program by looking up the details of the called functions in corresponding APIs or System Calls. Such calls can be monitored by instrumenting/hooking the executable, which helps not only to understand the flow of calls but also to understand function parameter values based on function parameter analysis. Usually, malware contains user-defined functions which are part of executable and cannot be comprehend by function call monitoring.

Orthogonal to function calls, monitoring information and data flow technique can help to understand how a program processes data. This technique allows to understand propagation of interested data by marking (i.e. tainted) such data with label. The analysis helps to understand data and control flow dependencies of a program. Furthermore, the analysis reveals the flow of implicit information. This technique is not scalable because abstracting a control or data flow graph of a given complex program is usually not possible.

Recently, hybrid analysis has been used to analyze a malware, which is a mix of static and dynamic analysis. This analysis helps to comprehend a malware with higher accuracy, however, this method is inherently limited. The accuracy of malware detection depends on the accuracy of underlying static or dynamic analysis techniques to comprehend the malware. We discuss the challenges in malware analysis and detection in the following.

5 Challenges

The main challenges that hinder the accuracy of different malware analysis and detection techniques are discussed in the following.

- Identifying “right features” of an unknown malware or variant of known malware that can accurately and uniquely describe a malware is the major challenge that hinders the accuracy of malware analysis techniques. Current approaches either identify such features manually, which is time consuming and error prune or identify

them automatically using static and dynamic analysis techniques that fails to identify features with practical and adequate abstraction due to underlying limitations of these techniques, which further results in inaccurate malware analysis.

- Encoding the identified features "adequately" in detection techniques such that they can be extracted from a given malware without any error is also a challenging task. Current approaches encode identified features as rules and patterns, which either are not exhaustive or are overlapping and thus extract inaccurate features resulting in high rate of false positives and negatives.

Beside the aforementioned challenges, lack of collaboration among anti-malware community and other security and policy communities (e.g., state security departments) makes it challenging to share, among them, the critical information, results and knowledge about potential malware and threats in general and security in particular. This isolation results in inadequate and outdated knowledge about potential threats and malwares among all the communities, which strongly hinders the development of robust and rigorous malware detection techniques.

6 Proposed Malware Analysis Platform

To address the identified challenges (in Section 5), we propose a novel, open source and extensible platform (based on set of tools) that allows to combine various malware analysis and detection techniques to automatically detect/classify a malware more rigorously. In the following, we discuss the workflow of the platform.

6.1 Malware analysis

The proposed platform provides rich integration that enables communities to combine compatible of the existing analysis tools and techniques to analyze a given malware. A user has to choose number of tools and then has to specify a "analysis configuration" that will be used to integrate the selected tools. The configuration is a logical specification that includes the description of features of interest and relationships among them (if any), the criteria to evaluate and combine the extracted features. The platform employs formal verification to attest the reliability of the extracted features as a potential malware.

6.2 Malware detection

Based on the verified extracted features in the previous step, the platform allows user to select number of tools and techniques and then to specify a "detection configuration" that will be used to detect the extract features from a given software with the help of selected tools and techniques. Like analysis configuration, detection configuration is also a logical specification that includes the constraints on the features, the criteria to evaluate and combine the detected features. The attestation engine of the platform will employ formal verification to attest that detected malware is indeed a malware, i.e. malware rigorously characterized based on detected features.

6.3 Results dissemination

Besides malware analysis and detection, the platform can be fed with malware samples from different providers/communities and will enable the development of effective classification schemes and methods, which are not sufficiently effective without

collaboration and the related sample aggregation. Furthermore, such collaborative platforms in cybersecurity will enable efficient sharing of information (e.g., about new identified threats) to all collaborators and sharing of appropriate defences against them, if such defences exist.

In addition to the collaborative research described, the platform will enable

1. the sharing of software/malware platforms among all project partners. Common databases will be developed exploiting appropriate tools, such as MISP (Open source platform for threat sharing). New classification schemes will be developed and
2. the development of new tools, which will be available to the research community through open software distribution platforms (e.g. github).

7 Conclusion

Current approaches for malware analysis and detection fail to effectively classify/detect new malware or variants of known malware. We have discussed the popular techniques for malware analysis and detection. We have identified the main challenges that hinder the accuracy of current malware analysis and detection approaches. To address the challenges, we have proposed a collaborative platform (i.e., set of tools) that can help to combine several tools and approaches to detect and analyse a given software as a malware more effectively. The platform also enables collaboration among various security providers to effectively handle malware threats to legitimate users.

Part of this work that was performed at ISI/ATHENA was supported by the project “I3T - Innovative Application of Industrial Internet of Things (IIoT) in Smart Environments” (MIS 5002434) which is implemented under the “Action for the Strategic Development on the Research and Technological Sector”, funded by the Operational Programme "Competitiveness, Entrepreneurship and Innovation" (NSRF 2014-2020) and co-financed by Greece and the European Union (European Regional Development Fund).

References

1. A. Moser, C. Kruegel, E. Kirda, In *Proceedings of the IEEE Symposium on Security and Privacy* (2007)
2. A. Bindra, In *IEEE Pow. Electr. Mag.*, **4**(3), pp. 20-27 (2017)
3. Kingsoft, Internet Security Research Report in China. Retrieved from http://cn.cmcm.com/ news/media/2016-01-14/60.html (2016).
4. E. Kalita, Independently published, IEEE (2017).
5. Y. Ye, T. Li, S. Zhu, W. Zhuang, E. Tas, U. Gupta, M. Abdulhayoglu, In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining* (2011)
6. D. Song, D. Brumley, H. Yin, J. Caballero, I. Jager, M. G. Kang, Z. Liang, J. Newsome, P. Poosankam, P. Saxena, In *Proceedings of the 4th International Conference on Information Systems Security* (2008)
7. P. Beaucamps, E. Filiol, In *Jour. in Comp. Vir.* **3**(1), 3–21 (2007)
8. E. Filiol, G. Jacob, M. L. Liard, In *Jour. in Comp. Vir.* **3**(1), 23–37 (2007)

9. M. Egele, T. Scholte, E. Kirda, C. Kruegel, In *ACM Comp. Surveys (CSUR)* **44**(2), 6 (2012)
10. Symantec, Symantec global internet security threat report. Retrieved from http://eval.symantec.com/mktginfo/enterprise/white_papers/b-whitepaper_internet_security_threat_report_xiii_04-2008.en- us.pdf (2008)
11. Kingsoft, Internet Security Report in China. Retrieved from http://www.ijinshan. com/news/2014011401.shtml (2014.
12. R. Brewer, In *Network Security* 2016, **9**, 5-9, ACM (2016.
13. M. G. Schultz, E. Eskin, F. Zadok, S. J. Stolfo, In *Proceedings of the IEEE Symposium on Security and Privacy* (2001)
14. K. Hannah, S. Gianvecchio, In *Jour. of Comp. Sci. Colls.,* **30**(3), 109-116, ACM (2015).
15. E. H. Spafford, In *Proceedings of the 2nd Euro. Softw. Engg. Conference*, (1989)
16. D. Moore, C. Shannon, In *Proceedings of the Internet Measurement Workshop* (2002)
17. Bizjournals, McAfee: Trends in a decade of cybercrime. Retrieved from http://www.bizjournals.com/sanjose/news/2011/01/25/mcafee-trends-in-a-decade-of-cybercrime.html?page=all (2011)
18. T. Eisenberg, D. Gries, J. Hartmanis, D. Holcomb, M. S. Lynn, T. Santoro, In *Comm. ACM* **32**(6), 706-709 (1989)
19. E. Stinson, J. C. Mitchell, In *LNCS: Detection of Intrusions and Malware, and Vulnerability Assessment* 4579, 89–108 (2007).
20. Rutgers Attack, https://eu.northjersey.com/story/news/2017/12/13/union-county-man-pleads-guilty-rutgers-cyber-attacks/949591001/ (2017)
21. K. Borders, A. Prakash, In *Proceedings of the 11th ACM Conference on Computer and Communications Security* (2004)
22. A. H. Sung, J. Xu, P. Chavez, S. Mukkamala, In *Proceedings of the 20th Annual Computer Security Applications Conference* (2004)
23. Z. Bazrafshan, H. Hashemi, S. M. H. Fard, A. Hamzeh, In *Proceedings of the 5th Conference on Information and Knowledge Technology (IKT)* (2013)
24. Trend Threat Research Team TrendMicro, Zeus: A Persistent Criminal Enterprise. Retrieved from http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/whitepapers/wp_zeuspersistent -criminal-enterprise.pdf, (2010)
25. P. Coogan, SpyEye Bot Versus Zeus Bot. Retrieved from http://www.symantec.com/connect/ blogs/spyeye-bot-versus-zeus-bot (2010).
26. L. Chen, W. Hardy, Y. Ye, T. Li, In *Proceedings of the International Conference on Web Information Systems Engineering (WISE)* (2015)
27. W. Hardy, L. Chen, S. Hou, Y. Ye, X. Li., In *Proceedings of the International Conference on Data Mining* (2016)
28. D. H. Chau, C. Nachenberg, J. Wilhelm, A. Wright, C. Faloutsos, In *Proceedings of the SIAM International Conference on Data Mining (SDM)*. (2011)
29. H. Chen, D. Wagner, In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*. 235–244 (2002)
30. H. Chen, D. Dean, D. Wagner, In *Proceedings of the 11th Annual Network and Distributed Systems Security Symposium (NDSS'04)* (2004)
31. H. H. Feng, J. T. Giffin, Y. Huang, S. Jha, W. Lee, B. P. Miller, In *Proceedings of the IEEE Symposium on Security and Privacy*. 194 – 208 (2004)
32. M. Egele, M. Szydlowsk, E. Kirda, C. Krugel, In *Proceedings of the 3rd International Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*. 17–36. (2006)
33. Y. Ye, T. Li, D. Adjeroh, S. S. Iyengar, In *ACM Comput. Surv.* **50**(3) (2017)