A Survey on Malware Detection and Analysis

Joshua Smallman

Senior Manager, IT & Security Operations, Modsquad, California, USA

DOI: 10.55662/JST.2024.5401

Abstract

Malware, or malicious software, poses a significant threat to the security and functionality of computer systems globally. This survey provides a comprehensive analysis of current malware detection and analysis methods, focusing on data mining methodologies. The study categorizes malware detection techniques into signature-based and behaviour-based approaches, highlighting their respective strengths and weaknesses. It explores heuristic techniques enhanced by artificial intelligence, including neural networks and genetic algorithms, to improve detection accuracy. The literature review examines host-based and network-based intrusion detection systems, hybrid systems, and virtual machine introspection. The paper also discusses static and dynamic analysis methods, emphasizing the importance of analysing malware in controlled environments. Through detailed examination, this survey aims to present a thorough understanding of contemporary malware detection strategies and their applications, offering insights for future advancements in the field.

Keywords

Malware, Malware Detection, Data Mining, Signature-Based Detection, Behaviour-Based Detection, Artificial Intelligence, Intrusion Detection Systems, Static Analysis, Dynamic Analysis, Virtual Machine Introspection

This work is licensed under CC BY-NC-SA 4.0. View complete license here

Introduction

Malware refers to any piece of malicious software that is designed to disrupt the normal functioning of a machine, steal confidential information, or gain access to private computer systems. Software that produces unexpected damage owing to a defect is not considered harmful software since it does not have the destructive intent that defines malware. In various contexts, the phrase badware is used to refer to both malicious software and accidentally destructive programs. Everyday life has been impacted by the proliferation of malware, which has reached into spheres as diverse as e-governance and social networks, digital automation, and mobile networks. There is a vast variety of malicious software, including viruses, worms, trojan horses, rootkits, backdoors, botnets, spyware, adware, and so on. These categories of malware are not exclusive of one another, which means that a single piece of malware may exhibit traits from many different categories at once. Malware developers apply polymorphism to the harmful components to avoid detection. Harmful files from the same malware "family," exhibiting the same types of malicious behavior, are regularly updated and/or obfuscated using different techniques, making them seem to be a wide variety of distinct files. Malware represents a significant and perhaps catastrophic danger to Internet security. Symantec ran a survey in February 2019 to find out how well-prepared businesses were to assess the threats and motives behind a malicious code sample. The data gathered in this way may be utilized to counteract emerging malware patterns or prepare for future dangers. Malware analysis may provide useful features for classifying new malware samples into recognized families (Ye et al, 2009).

This study provides a comprehensive literature analysis of the most up-to-date malware detection strategies that make use of data mining methodologies, with the goal of addressing some of the limitations of previous methods. This article divides the methods for detecting malware into two categories: signature-based and behavior-based. As for the paper's contributions, they are as follows:

- This paper will Summarize the present difficulties of malware detection methods in data mining.
- Providing a comprehensive, annotated summary of existing methods
- By Investigating a framework of the primary techniques that matter in the context of malware detection approach

This work is licensed under CC BY-NC-SA 4.0. View complete license here

- Classification malware techniques in data mining: a discussion of key considerations for future improvement.

Detection techniques

Signatures and anomalies-based Techniques respectively

To identify malicious software, almost every malware scanner uses some combination of signature-based and anomaly-based methods. However, there are ways that make advantage of these techniques, including static methods that are performed by extracting characteristics from static malware that is stored on a disk, and hybrid methods that combine the dynamic and static approaches (Blount et al, 2011).

In order to determine whether or not a file is harmful, signature-based procedures use scanning software to compare the file's contents to a database containing thousands of viral signatures. The main benefit of these methods is their efficiency. However, the fundamental drawback of signature-based approaches is that they are not able to protect against undiscovered infections.

Malware with a fingerprint in their databases may be detected by signature-based systems, whereas any misuse of a computer system can be identified by anomaly-based systems. To identify malicious software, anomaly-based detection techniques compare system activity with predefined norms and flag any deviations as suspicious. Signature-based malware detection relies on patterns, while anomaly-based detection relies on categorization.

Heuristic based Techniques

Signature and anomaly-based approaches were made more effective with the use of artificial intelligence (AI). Because of their flexibility in response to shifting conditions and their capacity for accurate prediction, neural networks (NNs) have found widespread use. Fuzzy logic is a kind of AI that borrows from the fuzzy theory of approximate reasoning rather than strict classical logic. In order to derive classification rules and choose acceptable features or ideal parameters for an optimal solution, genetic algorithms, another machine learning-based approach, are utilized in the malware detection process. Inheritance, mutation, selection, and hybridization, all important concepts in evolutionary biology, are put to use. The fundamental benefit of this method is that it may derive answers from many angles without requiring any previous knowledge of the system's behavior (Zhou, 2009).

Malware detection makes use of statistical and mathematical methods by using statistical and mathematical models to data about system activities including network connections, bandwidth, memory utilization, system call utilized by objects, etc.

Literature Review

According to Jiang et al. (2007), hosts-based intrusion detection systems track the real-time actions and states of individual computers to identify malicious attempts to subvert security measures. "To improve upon the capabilities of current dynamic behavior-based detectors, they provided a framework. When applied to realistic and varied cloud-based contexts, the proposed technique opens the way for sophisticated behavior-based analysis of harmful applications. Through this new paradigm, potential victim computers and the security lab would both run sample code. Study results indicated that analyzing several installation records of the same malware sample in the environments of different end users may improve analysis conclusions with little extra effort. On the other hand, the proposed framework is vulnerable to a wide variety of detection and evasion assaults, which presents serious privacy and security concerns. Improving the framework's efficiency is as simple as fixing its security flaws and making it resistant to evasion assaults.

Xiao et al. (2017) conducted research on the cloud-based malware detection game, which includes mobile devices transmitting their application traces to security servers in dynamic networks through base stations or access points. They developed a method to make use of a Dyna architecture that improves performance and a post-decision state learning-based mechanism that speeds up reinforcement learning. Based on their studies, the authors assert that their suggested methods outperform the state of the art in the dynamic malware detection game in terms of accuracy rate, detection latency, and mobile device utility. Several factors may reduce performance during detection due to the number of parties involved and their need to communicate with one another; they include network transmission delay, mobile

This work is licensed under CC BY-NC-SA 4.0. View complete license here

device detection delay, cloud processing time, and local detection delay. The performance may be enhanced by minimizing these lags.

According to Garfinkel and Rosenblum (2003), all of the packets on network nodes may be analyzed by using a network-based intrusion detection system (IDS). The traffic on a given network segment is monitored by a single sniffer module in this configuration. A distributed network-based intrusion detection system, in contrast, monitors traffic by using numerous modules located at each node. Malware detection systems that run over a network are sometimes referred to by their colloquial moniker, "out-of-the-box," to emphasize that they are located away from the host they are keeping tabs on. In this research, we offer a design for a host-agnostic intrusion detection system (IDS) that maintains the visibility of a host-based IDS while moving it outside of the host to improve its resilience to outside attacks. Specifically, we employ a virtual machine monitor to do this. Using this method, we can separate the IDS from the host being monitored without sacrificing any of our insight into the health of the host. Because of the VMM, researchers also have the rare opportunity to function as a go-between for communications between the host operating system and the hardware components. As part of our architectural research, we introduce Livewire, a working prototype. To show how Livewire works, we've put into practice a set of low-effort intrusion detection protocols and used them to foil actual assaults.

Basicevic et al. (2005) state that there exist hybrid detection systems that combine host-based and network-based features. An IDS of this kind is made up of several monitoring and data-gathering subsystems, each of which is located on a different node in the network. These subsystems gather data, which is then sent to the main system. There is a trade-off between host-based and network-based detection systems in terms of efficacy; although the former efficiently protects the internal system, it is vulnerable to external assault, the latter can prevent external attack but cannot defend the interior of the host.

Virtual machines (VMs), as described by Yin and Dawn (2013), are "an exact copy of a physical computer that runs in its own isolated environment and retains all of the original machine's functionality while emulating its entire set of capabilities" (p. Malware detection solutions that run in a virtual machine are built using this idea. Among the three types of virtual machines (VMs) used for malware detection, the first is the sandbox, in which the system gets information about a suspicious executable program from a user, evaluates its behavior by

This work is licensed under CC BY-NC-SA 4.0. View complete license here

running it in a controlled environment (sandbox), and then gives its findings back to the user. Second, efficiency is the defining feature of emulators, and emulation achieves this by simulating an entire computer system in order to run the guest operating system and the VMM, creating an execution environment for programs that is identical to the original machine with the exception variances caused by the availability of system resources or by timing dependencies. Emulators are computer programs that replicate the operation of hardware. Instead of running code itself, an emulator intercepts instructions and translates them into a sequence of instructions native to the target platform. Due to the fact that malware cannot be detected in system emulators, they are often used for this purpose. Thirdly, a virtual machine monitor (VMM) is a tiny piece of privileged code that grants the VM elevated privileges on the host computer in native system virtual machines. Because of this quality, it is a native VM that performs well but is vulnerable to mistakes and manipulation.

According to Ye's (2009) research study, agent technology's autonomy, decentralization, platform independence, scalability, and mobility are essential to the success of agent-based intrusion and malware detection systems. It's taking use of the idea that having no nerve center also means having no single point of failure. Agent-based systems were developed to address the shortcomings of both host-based IDS and distributed IDS architectures. While host-based IDS is only good at preventing attacks from inside the network, distributed IDS excels at preventing assaults from outside.

An APIDS, as defined by Goldman (2003), is an intrusion detection system that monitors and analyzes traffic based on a certain application protocol. The system constantly keeps tabs on the application protocol's changing behavior and condition. The system is made up of a service or agent that resides between a set of servers and keeps tabs on the application protocol passing across them. An APIDS is often placed between a web server and a database management system to keep an eye on the middleware/business logic's native SQL protocol during database interactions. Email spam may be stopped with the use of anti-spam technologies. In contrast to the inherent approaches employed automatically by email server systems, spam treatment involves both end users and administrators. Four distinct types of anti-spam methods exist, depending on whether they need user participation, are administered by system administrators, are automated by senders, or are used by investigators and law enforcement.

This work is licensed under CC BY-NC-SA 4.0. View complete license here

A methodology for the identification of behavioural malware on Android devices was proposed by Shabtai et al. (2012). The proposed framework makes use of a host-based malware analysis system to keep an eye on the mobile device's characteristics and events in real time and then uses ML anomaly detectors to determine whether or not the data was received from a malicious source. To determine the optimal setup for detecting new Android malware, they tested a wide variety of anomaly detection techniques, feature selection methods, and the number of top features. The authors assert that their suggested framework works well on Android and other mobile platforms. However, in order to conduct these tests, researchers have used simulated malware.

A novel technique based on sequence grouping and alignment was suggested by Borojerdi and Abadi (2013), and it's called MalHunter. For polymorphic malware, it may automatically build signatures depending on the virus's behaviour. This new approach involves the following steps: At first, sequences of behaviors are created from various malware samples. Later, the information is organized into subsets based on shared behavioural patterns. To identify a malware sample, we collect behavior sequences and compare them to those that have been previously created and saved in a database. The sample is classified as malware or clean depending on the results of the comparison It is the authors' contention that their proposed schema may be utilized to recognize any polymorphic malware, regardless of how the latter may have been disguised. Further, the authors assert that their approach is better to the industry standard for creating signatures.

Zolkipli and Jantan (2010) provided a novel framework for the identification of malware that included the combination of s-based detection with an evolutionary algorithm (GA), as well as a signature generator. The authors claim that their technology can recognize previously undisclosed kinds of malware, although the report is lacking in essential data concerning the suggested architecture. The number of malwares tested, the outcomes of those tests, and a comparison of the suggested technique to existing literature are all included. Authors created an accurate exploit-based signature generator for polymorphic worms as a bioinformatics approach. Multiple sequence alignment encourages sequential substring extractions, noise treatment gets rid of noise effects, and signature modification ensures that the simplified regular expression signature is compatible with up-to-date IDSs. The authors assert that their proposed schema outperforms prior exploit-based signature generation schemas in terms of accuracy and precision while being robust to noise. This is because it is better able to extract

This work is licensed under CC BY-NC-SA 4.0. View complete license here

polymorphic worm characteristics, such as invariants of a single byte and distance limits between invariant bytes. But the suggested paradigm can only be used to polymorphic worms and no other forms of malware.

Table 1: Malware detection analysis

Detection system	**Techniques and Technology**	**Characteristic**	**Strength**	**Weakness**	**Release year**	**Author name**
Effective mobile devices malware detection using signatures	Signature based Techniques for malware detection	Signature based Heuristic techniques	Malware detection effectiveness	Unable to detect new malware	2008	(Deepak, 2008)
Network intrusion detection based on anomalies: Methods, infrastructure, and problems	Anomaly based	Techniques such as AI analysis, statistical analysis, and behavior analysis are included here.	Detecting unknow new malware	Higher rate of false/negative and false positive	2007	(Jiang et al, 2007)
For mobile devices, an intrusion detection framework	Specification based	Interrupts generated by the use of a keypad or touch	Detecting unknow new malware	Higher rate of false/negative and false positive	2011	(Razeghi and Abadi, 2013)

This work is licensed under CC BY-NC-SA 4.0. View complete license here

based on specifications		screen may help distinguish between malicious software and human action.				
Experimental Validation of a Learning Classifier System for Adaptive Malware Detection	Rule based	A rule-based system an evolutionary algorithm based on reinforcement learning	Detecting unknow new malware	Higher rate of false/negative and false positive	2005	(Basicevic et al, 2005)
An Approach Based on behavioural malware	Malware behavioural approach	ML anomaly detectors	Detecting new malware	Not all legitimate free application	2012	(Shabtai et al.) 2012
Detecting malicious software using a combination of file content and file relations in the cloud	Cloud based application	File content information is represented by a parametric component, whereas file relationship information is represented	Protect effectively eternal system	Susceptible to external attack	2011	(Xiao et al, 2017)

This work is licensed under CC BY-NC-SA 4.0. View complete license here

		by a non-parametric one.				
A Method for the Detection and Prevention of Intrusions That Is Based on Networks	Network based	One packet-sniffing device per network segment.	Prevent external attack	Can't avoid inside attack	2010	(Zolkipli and Jantan, 2010)
An architecture for the detection of intrusions that is based on virtual machine introspection	Virtual machine	a more manageable portion of the confidential code	Performant	Liable to errors and tamper resistant	2003	(Garfinkel and Rosenblm, 2003)

Methodology

Analysis of malware cannot be undertaken in a typical setting or on a machine being used for commercial output at the time. Malware will be analyzed inside of a computer forensics lab utilizing a PC simulation. The malware analysis VM was built for that same reason. VMware makes it easy to set up an analysis lab. To create a virtual machine, you need a real host computer with sufficient memory and storage space, as well as VMware Workstation or Server and the operating system installation disks. Since VMware imitates computer hardware, the examiner must install the operating system on each virtual host individually employing VMware's new Virtual Machine Installer. After the operating system has been installed, the VMware Tools package should be put in place to make the computer run most

This work is licensed under CC BY-NC-SA 4.0. View complete license here

efficiently within VMware. Then, download some virus detection software. Having a variety of virtual computers running various Operating system in the lab is advised, since this will allow you to simulate the environments that malware is most likely to affect. This paves the way for the study of harmful software in its natural habitat. To examine malware at a certain patch level, it is recommended to use VMware Workstation and take snapshots at various times throughout the installation of security updates. Malware uses "vectors," sometimes known as pathways or methods to infiltrate computer systems. There are a few possible routes that might transport the payload and tie it back to the targeted machine. The term "attack vector" refers to any potential entry point by which an attacker might compromise a computer or network server and then release a malicious payload. Insecurity in any system, even that introduced by humans, may be exploited by hackers via the use of attack vectors. Viruses, malicious links in emails, websites, pop-up windows, IMs, chat rooms, and outright fraud are all potential entry points for attackers. Firewalls and anti-virus programs may help reduce the effectiveness of certain attacks. However, there is no kind of defense that can completely prevent an assault. Hackers are always improving their methods and looking for new ways in to computers and servers, so even if a protection strategy is successful today, it may not be tomorrow. Viruses , Trojan horses, worms, and spyware are some of the most prevalent harmful payloads (Jiang et al, 2007).

Malware detection Analysis techniques and tools

It is necessary to conduct an analysis into newly discovered malware in order to have an understanding of the dangers and objectives connected with it before one can create signatures for it. You may learn about the malicious application and its capabilities by either analyzing its source code or running it in a secure setting.

Static analysis

Static analysis is the process of examining harmful code without actually running it. Static analysis makes use of many detection patterns, such as text signatures, byte-sequence n-grams, syntactic library calls, control flow graphs, opcode (operational code) frequency distributions, etc. If static analysis is to be performed, the executable must first be unzipped and encrypted. Windows executables may be decompiled with the help of a

This work is licensed under CC BY-NC-SA 4.0. View complete license here

disassembler/debugger and a memory dumper. The malware's code is shown as Intel X86 assembly instructions in disassemblers and debuggers like IDA Pro and OllyDbg; these tools reveal a great deal about the malware's behavior and reveal patterns that may be used to track down its authors. Software known as "memory dumpers," such LordPE and OllyDump, may be used to retrieve and save encrypted code from the computer's RAM. This method is helpful for analyzing compressed executables that are otherwise challenging to decompile. Malware binaries may be obfuscated via binary obfuscation methods, which create self-compressed and uniquely organized binary files to thwart reverse engineering. This makes static analysis prohibitively costly and prone to error. In addition, static analysis performed on binary executables loses information like the size of data structures or variables, which further complicates the analysis of malware code [11].

Dynamic Analysis

Dynamic analysis refers to the process of seeing and analyzing the actions of malicious code as it interacts with a system in a simulated or controlled environment (such as a virtual machine, simulator, emulator, sandbox, etc.). The proper monitoring tools, such as Process Monitor and Capture BAT, Process Explorer and Process Hacker replace (for process monitoring), Wireshark (for network monitoring), and Regshot (for system change detection), are installed and activated before the malware sample is executed. Dynamic analysis may be carried out using a variety of methods, such as function call monitoring, Things like AutoStart extensibility points, analysis of function parameters, monitoring of information flows, and instruction traces. Dynamic analysis is superior than static analysis since it does not need disassembling the executable. It reveals the malware's true, static-analysis-resistant nature. There are scalability concerns since it is laborious and requires a lot of resources. Malware may operate in unexpected ways when run in a simulation rather than the actual world. Another issue is that malware activity is often conditional, making it impossible to identify in a simulated setting. Norman Sandbox, CWSandbox, Anubis and TTAnalyzer, Ether and ThreatExpert are only some of the internet automated tools available for dynamic analysis of malware. Insight into virus activity and the activities they do may be gained using these technologies, thanks to the analysis reports they provide. Malware must be represented accurately in the analysis system before it can be classified using a similarity measure or feature vectors. The vast volume of daily malware samples sent to anti-virus providers, however, necessitates an automated technique to reduce the number of samples that need to

This work is licensed under CC BY-NC-SA 4.0. View complete license here

be examined in detail by humans. In the past, automated malware analysis and categorization has been accomplished with the help of a variety of Artificial Intelligence (AI) approaches, most notably those based on machine learning.

Conclusion

The purpose of this paper was to provide a comprehensive analysis of the current status of malware, malware detection methods, and malware detection technology. In specifically, it describes a variety of malware detection technologies and includes a recent comparison research covering the vast majority of malware types. Even though malware and malware detection procedures are always evolving, this research may serve as a valuable resource for experts in the area. The literature review provides a comprehensive look at how to detect and rank malware using a firm grasp of domain-specific analytics.

References

Blount, J.J., D.R. Tauritz, and S.A. Mulder. (2011) Adaptive Rule-Based Malware Detection Employing Learning Classifier Systems: A Proof of Concept. in Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual. 2011.

Basicevic, F., M. Popovic, and V. Kovacevic. (2005) The use of distributed network-based IDS systems in detection of evasion attacks. in Telecommunications, 2005. advanced industrial conference on telecommunications/service assurance with partial and intermittent resources conference/e-learning on telecommunications workshop. aict/sapir/elete 2005. proceedings. 2005.

Deepak Venugopal, G.H., (2008) Efficient signature based malware detection on mobile devices. Mob. Inf. Syst., 2008. 4(1): p. 33-49.

This work is licensed under CC BY-NC-SA 4.0. View complete license here

Garfinkel, T. and M. Rosenblum, (2003) A virtual machine introspection based architecture for intrusion detection. 2003: p. 191--206.

Goldman, E., (2003) Dissecting Spam's Purported Harms.

H. Razeghi Borojerdi and M. Abadi. (2013) ''MalHunter: Automatic generation of multiple behavioral signatures for polymorphic malware detection,'' in Proc. ICCKE. Mashhad, Iran: Ferdowsi Univ. Mashhad, vol. 1.

Jiang, X., X. Wang, and D. Xu, (2007) Stealthy malware detection through vmm-based "out-of-the-box" semantic view reconstruction, in Proceedings of the 14th ACM conference on Computer and communications security. 2007, ACM: Alexandria, Virginia, USA. p. 128-138.

L. Xiao, Y. Li, X. Huang, and X. Du, (2017) ''Cloud–based malware detection game for mobile devices with offloading,'' IEEE Trans. Mobile Comput., vol. 16, no. 10, pp. 2742–2750.

M. F. Zolkipli and A. Jantan, (2010) ''A framework for malware detection using combination technique and signature generation,'' in Proc. 2nd Int. Conf. Comput. Res. Develop.

Ye, D., An Agent-Based Framework for Distributed Intrusion Detections. 2009.

Yin, Heng & Song, Dawn. (2013). Dynamic Binary Analysis Platform. 10.1007/978-1-4614-5523-3_2.

Ye, Y., et al., (2009) Intelligent file scoring system for malware detection from the gray list, in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM: Paris, France. p. 1385-1394.

A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, (2012) ''Andromaly: A behavioral malware detection framework for Android devices,'' J. Intell. Inf. Syst., vol. 38, no. 1, pp. 161–190.

This work is licensed under CC BY-NC-SA 4.0. View complete license here