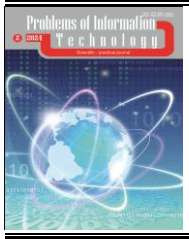

Available online at www.jpit.az

15 (2) 2024

Evasion techniques in malware detection: challenges and countermeasures

Yadigar Imamverdiyev[a], Elshan Baghirov[b]

Azerbaijan Technical University, Baku, Azerbaijan[a],

Institute of Information Technology, B. Vahabzade str., 9A, AZ1141 Baku, Azerbaijan[b]

[a]yadigar.imamverdiyev@aztu.edu.az, [b]elsenbagirov1995@gmail.com

[a] https://orcid.org/0000-0002-3710-1046; [b] https://orcid.org/0000-0002-5940-4136

ARTICLE INFO

Keywords:

Malware
Malware detection
Evasion techniques
Cybersecurity
Obfuscation

ABSTRACT

In the ever-evolving digital landscape, the escalating sophistication of malware poses a substantial threat, necessitating continual advancements in detection methods. This paper addresses the pervasive challenge of evasion techniques employed by malware to circumvent standard security measures. Focused on understanding the intricate methods employed by malware developers, our study explores the dynamic nature of this cyber threat. As malicious actors continually refine their approaches, a nuanced understanding of evasion tactics becomes paramount for developing effective countermeasures. The research emphasizes the need for robust defense mechanisms capable of adapting to the constantly changing cyber threat landscape. By unraveling the complexities of evasion techniques, this paper contributes valuable insights to the development of proactive and resilient cybersecurity measures. Through an exploration of specific evasion tactics, we aim to inform and empower cybersecurity professionals, facilitating the creation of strategies capable of effectively mitigating the risks posed by these dynamic digital threats.

1. Introduction

Malware, short for malicious software, refers to any software intentionally designed to cause damage, disrupt, or gain unauthorized access to computer systems, networks, or devices. Malware encompasses a wide range of malicious programs, including viruses, worms, Trojans, ransomware, spyware, adware, and rootkits, among others (Alakbarov, 2023; Baghirov, 2023). Common malware types and their descriptins has been shown on Table 1.

In an era of rapid technological advancement, the escalating presence of sophisticated malware poses a significant and evolving challenge to digital systems' security. The relentless innovation of malicious actors, combined with the dynamic cyber threat landscape, necessitates continual improvements in malware detection methods.

As digital systems become increasingly integral to daily life, effective cybersecurity becomes paramount, with malicious entities exploiting vulnerabilities and refining evasion tactics against traditional defense mechanisms. Understanding these intricate evasion techniques is not only critical for staying ahead of cyber threats but is also fundamental to developing proactive and resilient countermeasures (Jiaxuan et al. 2024).

In the realm of cybersecurity, end-users rely on anti-malware software as a pivotal tool to identify and address the presence of malicious entities on their machines. The conventional structure of anti-malware solutions typically encompasses two fundamental components: a

Received 7 March 2024, Received in revised form 3 May 2024, Accepted 20 May 2024
http://doi.org/10.25045/jpit.v15.i2.02

signature-based detector and a heuristics-based classifier. The signature-based method excels in identifying variations of established malware families with a commendably low error rate. However, its efficacy diminishes as the cybersecurity landscape witnesses a continual surge in the identification of an expanding array of new and previously unknown malware samples (Hojjat et al. 2020). In our previous work, we demonstrated that incorporating extracted opcode sequences as features in classification, while accounting for correlations and potential irrelevant opcode names, led to improved accuracy in malware classification when employing the Random Forest classifier, aligning with the emphasis on effective detection methods in the realm of cybersecurity (Baghirov, 2023).

Some analysis showed that integrating new intelligent technologies into existing systems is necessary to enhance cybersecurity protections (Shikhaliyev, 2024).

This study addresses the persistent threat of malware proliferation, acknowledging the formidable challenge it poses to the cyberspace landscape. Despite researchers offering numerous sophisticated defense mechanisms, malware writers persistently innovate to evade detection, presenting an ever-evolving and dynamic threat to computer systems. Malware evasion involves specific strategies employed by malware to circumvent security software. Current surveys on evasion techniques have been fragmented and focused, lacking systematic and comprehensive research. To address this gap, our paper proposes a strategy-driven framework from the perspective of malware writers, aiming to contribute valuable insights for the development of adaptive and robust cybersecurity measures.

2. Related works

(Jiaxuan, 2024) proposes a unified framework for Portable Executable (PE) malware evasion methods, demonstrating how malware writers can combine various strategies, including transformation, concealment, and attack-based techniques. By conducting a comprehensive survey, the study highlights that packer and code obfuscation are the primary evasion tactics, with environment analysis being a significant concealment strategy, while direct attacks on detectors are less effective. The paper emphasizes the importance of reinforcement learning in adversarial attacks and provides a future outlook on malware evasion trends, aiming to equip researchers and practitioners with a thorough understanding to develop more robust detection and prevention methods. While the proposed unified framework for PE malware evasion methods is comprehensive and insightful, it has several drawbacks. Its complexity can make it challenging to understand and implement, especially for practitioners not deeply familiar with each evasion strategy. Additionally, the framework may face scalability issues when applied to large-scale systems, requiring significant computational resources. Practical implementation in real-world environments could also be difficult, as deploying advanced detection techniques might not be feasible for all organizations due to resource constraints.

(Farnood et al. 2022) proposed a novel solution to combat evasive Android malware in smartphone security with the introduction of CamoDroid. This open-source and extendable dynamic analysis environment successfully mimics authentic device features, conceals its analytical nature, and exhibits robust resilience against both sandbox detection tools and real-world evasive Android malware, boasting a success rate surpassing 96 percent. While CamoDroid presents a promising approach to addressing the challenge of evasive Android malware, it is essential to acknowledge certain drawbacks. One potential limitation lies in the constant evolution of malware techniques, as adversaries may adapt to countermeasures over time. Additionally, the effectiveness of CamoDroid may vary based on the diversity of evasive tactics employed by malware developers, emphasizing the need for ongoing updates and enhancements to maintain its resilience in the face of evolving threats.

(Chandra et al. 2018) addressed the escalating threat of malware, particularly in the context of Internet-of-Things (IoT), where a surge in polymorphic and metamorphic malware targeting IoT devices has been observed. Focusing on the evasion techniques employed by these sophisticated malicious programs, the paper highlights the need for effective countermeasures. The contribution lies in the presentation of a comprehensive taxonomy for classifying existing evasion countermeasures, bridging a crucial gap in the current research landscape. This taxonomy aims to facilitate scalable classification, aiding

malware analysis, classification, and guiding the development of future evasion countermeasures. One drawback of this work is its acknowledgment that the landscape of sandbox evasion countermeasures, anti-debugging, and antidisassembly techniques is vast and continually evolving. While the paper provides a taxonomy based on existing strategies and techniques, it concedes that the coverage is not exhaustive, leaving room for the emergence of new evasion techniques that may not be adequately addressed. Additionally, the proposed future work outlines the need for ongoing evaluation and updating of the taxonomy, suggesting a potential challenge in maintaining the relevance and comprehensiveness of the classification system over time. Lastly, the expressed interest in exploring machine learning techniques for detecting sophisticated evasive malware lacks detailed explanation, leaving uncertainty about the specific methodologies and considerations involved in this potential future research direction.

Table 1. Common malware types and descriptions

Malware types	Description
Viruses	Self-replicating programs that attach to legitimate files and spread when the infected files are executed.
Worms	Standalone, self-replicating programs that spread across networks, exploiting vulnerabilities to infect other devices.
Trojans	Disguised as legitimate software, Trojans deceive users into installing them, enabling unauthorized access or causing harm.
Spyware	Monitors and gathers sensitive information, such as login credentials and personal data, without the user's knowledge
Adware	Displays unwanted advertisements, often bundled with free software, generating revenue for the attacker.
Ransomware	Encrypts files or systems, demanding a ransom for their release. It can severely impact individuals and organizations.
Botnets	Networks of compromised computers controlled by a single entity, often used for coordinated attacks or distributing malware.
Rootkits	Conceals malicious software by modifying the operating system, enabling unauthorized access and persistent control.
Keyloggers	Records keystrokes to capture sensitive information, such as passwords and credit card details.

3. Evasion techniques in malware detection

In this study, we use the term "evasion" to describe techniques, including obfuscation, that lead to a benign classification even when malicious actions are executed, as opposed to anti-analysis methods that prevent malicious actions during the analysis process (Jiaxuan et al. 2024; Haikoi et al, 2023; Borja et al. 2023). Obfuscation involves making modifications to an executable while preserving its functionality (Melissa and Vivek, 2018). The primary goal is to hinder both automated and manual scrutiny of the code, making it more challenging for analysis. After reviewing several papers in the field, it is evident that evasion techniques are systematically categorized into three distinct categories, as summarized in Table 2.

Evasion techniques for disassembling. In this category, evasion techniques are tailored to hinder the disassembly and reverse engineering processes. Attackers consistently modify their malicious software to increase intricacy and minimize the likelihood of detection. They create novel malware, often labeled advanced malware, owing to its capacity to alter its structure and camouflage, thereby confusing or eluding malware analysts. This adaptive malware is referred to as polymorphic malware. Metamorphic viruses, particularly insidious, dispense with the need for a decryption routine, directly modifying the code body by embedding a morpher (code for morphing transformations) within itself to generate diverse malware

variants. Common tactics employed by malware creators for morphing include the insertion of dead code, renaming variables, and reordering statements. The majority of metamorphic or polymorphic malware showcase intricate and mutable traits, rendering them challenging to identify (Abijah and Geetha, 2021).

According to VirusTotal reports, an average of over 680,000 new samples are analyzed daily. Some of these samples are identified as re-packed versions of previously encountered samples, exhibiting identical behavior (Hojjat et al. 2020). Packing is a technique used by malware authors to alter the signature of malicious code, evading detection by security software, particularly signature-based antivirus programs, and requires subsequent unpacking during execution, prompting ongoing efforts by security researchers and antivirus vendors to develop effective detection methods. It serves as a hindrance to extracting informative features from executable files (Hojjat et al. 2020)

Evasion techniques in sandbox. Evasion techniques within sandbox environments are designed to recognize if the malware is running in a controlled, isolated environment used for analysis, commonly known as a sandbox. The malware checks for signs of sandboxing, virtualization, or emulation and adjusts its behavior accordingly. Techniques may include environmental checks, time delays, and mimicking user interactions. The objective is to avoid detection and analysis by eluding execution in an artificial environment, forcing analysts to resort to more sophisticated and dynamic analysis techniques.

Evasion techniques for debugging. Evasion techniques for debugging focus on detecting whether the code is being actively debugged or analyzed. If a debugger is detected, the malware alters its behavior to impede the analysis process. Anti-debugging tricks, intentional code errors triggered by debugging events, and checks for debugging tools in memory fall into this category. The intention is to thwart debugging attempts and maintain the stealth and integrity of the malware during analysis.

Table 2. Classification of evasion techniques

Category	Examples for evasion techniques
Evasion techniques for disassembling	Code obfuscation, opaque predicate, encryption and decryption, self-modification, anti-debugging tricks, conditional code execution, polymorphism and metamorphism, dynamic code loading, string manipulation, anti-analysis checks, indirect jumping, binary packing, try-catch, overloading etc.
Evasion techniques in sandbox	Environment checks, time delay execution, sensor mimicry, (for example, mimicking mouse movements, keyboard input), anti-vm checks, API call randomization, memory inspection evasion, anti-snapshotting techniques, evasion through uncommon file formats, anti-emulation techniques, fake resource utilization etc.
Evasion techniques for debugging	Check for debugger presence, intentional code errors, code triggers based on debugging events, anti-stealth techniques, register and memory state manipulation, timing-based detection of debuggers, hardware breakpoint avoidance, interrupt descriptor table attacks, debugging API obfuscation, check for debugging tools in memory etc.

4. Malware evasion countermeasures techniques

In response to the increasing prevalence of evasive malware, researchers have put forth measures to counteract the evasion employed by such malware. While specific mitigation techniques target certain types of evasion once identified, others aim for a broader enhancement of transparency without the necessity of detecting evasion attempts. Assessing the effectiveness of these mitigation techniques poses challenges, requiring researchers to establish their own metrics, often involving comparisons of malware activity across different executions. Additionally, researchers must mitigate or eliminate variations between analysis sessions that could result in divergent behavior unrelated to evasion, such as differences in execution time or environmental

discrepancies (Alexei et al. 2017).

The comprehensive countermeasures presented in Table 3 aim to fortify cybersecurity defenses, offering practical solutions to mitigate the impact of sophisticated evasion tactics.

The countermeasures outlined encompass a diverse set of approaches, ranging from dynamic analysis and controlled execution environments to advanced static analysis tools, behavioral analysis, and machine learning applications. Recognizing the dynamic nature of cyber threats, these countermeasures provide a multifaceted defense against the arsenal of techniques deployed by malware authors to obfuscate, evade detection, and impede analysis.

Table 3. Malware evasion countermeasures techniques

Countermeasures for	Name of countermeasures
Evasion techniques in disassembling	• Advanced static analysis tools • Behavioral analysis and dynamic unpacking
Evasion techniques in sandbox	• Diverse sandbox configurations • Environment randomization
Evasion techniques for debugging	• Controlled debugging environments • Code emulation for debugging

5. Comparative analysis

The purpose of this comparative analysis is to systematically review and evaluate the effectiveness of various published machine learning-based malware detectors. By examining a range of state-of-the-art detectors, we aim to highlight their respective strengths and weaknesses, particularly in terms of detection accuracy, robustness against evasion techniques.

To conduct this comparative analysis, we systematically reviewed the most recent advancements in machine learning-based malware detection across various operating systems published over the last five years. We conducted an extensive literature search using multiple academic databases, including Elsevier, IEEE Xplore, SpringerLink, ACM Digital Library, and Google Scholar. The selection criteria focused on peer-reviewed articles, conference papers, and significant technical reports that proposed novel detection methodologies or demonstrated significant improvements over existing techniques. We used keywords such as "malware detection," "evasion," and "obfuscation" during our search. While we analyzed over 30 papers in our review, we selected five unique papers that represent the broader trends and findings within this body of research.

This analysis aims to provide insights that could guide future research and development in the field of obfuscated malware detection. Table 3 shows a comparative analysis of reviewed works, where methods, advantages, and limitations have been highlighted.

Table 3. Comparative analysis of suggested detection mechanisms

Author(s)	Method	Advantages	Limitations
Hemant, R. et al. (2022)	• Evaluated robustness of malware detection models • Used MalDQN neural network agent with deep reinforcement learning • Generated adversarial malware to evade detection • Reduced accuracy of twenty detection models • Improved detection and resistance to adversarial samples	The MalDQN attack reduced the average accuracy from 86.18% to 55.85% in the above twenty malware detection models.	The study focuses only on Android malware, which might limit its applicability to other operating systems. It uses static features (permissions and intents), which may not cover all aspects of malware behavior.

Hemant, R. et al. (2023)	• Investigated robustness of 36 malware detection models using permissions and intents • Designed two reinforcement learning attacks: TRPO-MalEAttack and PPO-MalEAttack • Identified ten most vulnerable Android permissions and intents • Proposed MalVPatch defense, improving detection accuracy and robustness	TRPO-MalEAttack: 95.75% fooling rate, reduced accuracy from 86.01% to 49.11%; PPO-MalEAttack: 96.87% fooling rate, reduced accuracy from 86.01% to 48.65%	The evaluation is based on specific evasion attacks which may not represent all possible attack vectors.
Hayyan et al. (2023)	• Proposed Maaker framework for Android malware detection • Utilizes both static and dynamic analyses through hybrid execution • Incorporates human-in-the-loop approach using Model Driven Engineering (MDE) • Compared with Ares, IntelliDroid, and Defuzer tools • Criteria: number of detected evasions, reached targets, required executions, and time to reach targets • Evaluation results: Maaker outperforms rivals in effectiveness, efficiency, and scalability	Maaker outperformed the three rival tools regarding effectiveness, efficiency, and scalability.	The framework could still face issues with certain malware samples that cause crashes during execution; Reliance on human expertise could introduce variability and limit scalability.
Kowshik et al. (2023)	• Introduced MalHyStack, a hybrid model using stacked ensemble learning. First layer: ExtraTrees, XgBoost, Random Forest and Second layer: Deep learning. • Feature selection with Pearson correlation improves accuracy and reduces time complexity. • Evaluated using CIC-MalMem2022 dataset.	MalHyStack shows superior performance in accuracy and efficiency compared to existing models.	Current reliance on trial and error for hyperparameter optimization, which can be inefficient compared to automated tuning algorithms; Potential issues with scalability and computational overhead due to the use of multiple classifiers and deep learning layers.
Lichen et al. (2023)	• Proposes ERMDS, a new dataset designed to evaluate the robustness of learning-based malware detection systems (LB-MDS); • Incorporates multiple obfuscation methods across three levels: binary, source code, and packing; • Conducts a detailed analysis of factors contributing to accuracy decline in LB-MDS and antivirus software	Generates an instance called ERMDS-X, consisting of 86,685 malware samples and 30,455 benign samples.	The extensive use of diverse obfuscation techniques may introduce complexity and scalability issues, making implementation resource-intensive; The reliance on specific tools and methods for generating obfuscation techniques could introduce biases.

6. Results and conclusions

The research endeavors to confront the escalating sophistication of malware and the challenges posed by its evasion techniques. Through a detailed exploration of specific evasion tactics, the study has unveiled practical countermeasures aimed at fortifying cybersecurity defenses. The findings emphasize the critical need for ongoing research and collaboration in the realm of cybersecurity. The paper underscores the

imperative for developing robust defense mechanisms capable of adapting to the ever-changing cyber threat landscape. by unraveling the complexities of evasion techniques, this study contributes valuable insights to the creation of proactive and resilient cybersecurity measures.

As malicious actors refine their approaches, a nuanced understanding of evasion tactics becomes paramount. The research not only informs but empowers cybersecurity professionals, facilitating the creation of strategies capable of effectively mitigating the risks posed by these dynamic digital threats. To keep our digital world safe and strong, the study suggests we need to stay alert, conduct ongoing research, and work together to stay ahead in the constant fight against changing cyber threats.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used ChatGPT in order to language editing and refinement. After using this tool/service, the author reviewed and edited the content as needed and takes full responsibility for the content of the publication.

References

Abijah, R., & Geetha, S. (2021). A comprehensive survey of tools and techniques mitigating computer and mobile malware attacks. Computers & Electrical Engineering, 92, 1-18. https://doi.org/10.1016/j.compeleceng.2021.107143

Alakbarov, R. (2023). Security issues and solution mechanisms in cloud computing systems: a review. *Problems of Information Technology*, 14(2), 12-22. http://doi.org/10.25045/jpit.v14.i2.02

Alexei, B., & Bulent, Y. (2017). A Survey On Automated Dynamic Malware Analysis Evasion and Counter-Evasion. ROOTS: Proceedings of the 1st Reversing and Offensive-oriented Trends Symposium, 1-21. https://doi.org/10.1145/3150376.3150378

Baghirov, E. (2023). Malware detection based on opcode frequency. *Problems of Information Technology*, 14(1), 3-7. https://doi.org/10.25045/jpit.v14.i1.01

Borja, M., Antonio, R., Alessio, M., et al. (2023). Light up that Droid! On the Effectiveness of Static Analysis Features against App Obfuscation for Android Malware Detection. *arXiv:2310.15645*. https://doi.org/10.48550/arXiv.2310.15645

Chandra, S. V., Peter, L. K., Zhaohui, T., & Forest, T. (2018). Taxonomy on Malware Evasion Countermeasures Techniques. *IEEE 4th World Forum* on Internet of Things (WF-IoT), 1-6. https://doi.org/10.1109/WF-IoT.2018.8355202

Farnood, F., Mohammad, Z., & Steven, D. (2022). CamoDroid: An Android application analysis environment resilient against sandbox evasion. *Journal of Systems Architecture*, 125, 1-10. https://doi.org/10.1016/j.sysarc.2022.102452

Haikuo, Y., & Brandon, L. (2023). A Method for Summarizing and Classifying Evasive Malware. Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses, 455-470. New York. https://doi.org/10.1145/3607199.3607207

Hayyan, H., Behrouz, T., & Bahman, Z. (2023). Maaker: A framework for detecting and defeating evasion techniques in Android malware. Journal of Information Security and Applications, 78, 14-26. https://doi.org/10.1016/j.jisa.2023.103617

Hemant, R., Adarsh, N., Sanjay, K. S., & Mohit, S. (2023). Adversarial superiority in android malware detection: Lessons from reinforcement learning based evasion attacks and defenses. Forensic Science International: Digital Investigation, 44, 1-10. https://doi.org/10.1016/j.fsidi.2023.301511

Hemant, R., Animesh, S., & Sanjay, K. S. (2022). Defending malware detection models against evasion based adversarial attacks. Pattern Recognition Letters, 164, 119-125. https://doi.org/10.1016/j.patrec.2022.10.010

Hojjat, A., Fabio, G., & Francesco, M. et al. (2020). When Malware is Packin' Heat; Limits of Machine Learning Classifiers Based on Static Analysis Features. Network and Distributed Systems Security (NDSS) Symposium, 23-26 February, San Diego, CA, USA, 1-20. https://dx.doi.org/10.14722/ndss.2020.24310

Imamverdiyev, Y. (2021). Analysis Of Cybersecurity Problems In Process Control Systems. Problems of Information Technology, 2, 16–29. https://doi.org/10.25045/jpit.v12.i2.02

Jiaxuan, G., Junfeng, W., & Zhiyang, F., et al. (2024). A survey of strategy-driven evasion methods for PE malware: Transformation, concealment, and attack. Computers & Security, 137, 1-10. https://doi.org/10.1016/j.cose.2023.103595

Kowshik, S.R., Tanim, A., & Pritom, B.U., et al. (2023). MalHyStack: A hybrid stacked ensemble learning framework with feature engineering schemes for obfuscated malware analysis. Intelligent Systems with Applications, 20, 1-17. https://doi.org/10.1016/j.iswa.2023.200283

Lichen, J., Yang, Y., Bowen, T., & Zihan, J. (2023). ERMDS: A obfuscation dataset for evaluating robustness of learning-based malware detection system. BenchCouncil Transactions on Benchmarks, Standards and Evaluations, 3, 1-13. https://doi.org/10.1016/j.tbench.2023.100106

Melissa, C., & Vivek, B. (2018). Effectiveness of Android Obfuscation on Evading Anti-malware. CODASPY '18: Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, 143–145. https://doi.org/10.1145/3176258.3176942

Shikhaliyev, R. (2024). Cybersecurity risks management of industrial control systems: A review. Problems of Information Technology, 15(1), 37-43. https://doi.org/10.25045/jpit.v15.i1.05