Application of Both Static and Dynamic Analysis Methodologies for Malware Detection

Submitted: 5th December 2024; accepted: 2nd April 2025

Andrzej Mycek, Mirosław Roszkowski

DOI: 10.14313/jamris-2026-050

Abstract:
Currently, the two most rapidly developing fields in IT are Cybersecurity and Artificial Intelligence. Both of these areas intersect significantly—advancements in AI contribute to the growth of the security sector. This is crucial because cybercriminals are also advancing, devising new methods to cause harm, such as data theft, hijacking infrastructure, turning off IT systems, or espionage. To prevent such incidents, security specialists are developing increasingly sophisticated tools that enable the detection and neutralization of attacks. Among these tools are two techniques: static analysis and dynamic analysis. In this thesis, both methods have been thoroughly examined, and the benefits of combining these techniques in the context of protection against malware have been demonstrated using specific malicious software.

Keywords: *cybersecurity, dynamic malware analysis, malware detection, ransomware, static malware analysis*

1. Introduction

The Cybersecurity market is continually expanding. According to research by [1], the Cybersecurity market has grown by over 30% in the past three years, and it is projected to increase by 58.5% over the next five years, reaching nearly $350 billion by 2026. The ongoing digital transformation of businesses drives this growth. The advancement of global economies necessitates the development and deployment of increasingly sophisticated IT infrastructures, with a rising number of new devices used daily by individuals, often referred to as the Internet of Things (IoT). Additionally, new regulations and standards concerning data protection and information security are being continuously introduced, with the European Union leading in this domain [2]. A positive development is the growing awareness among companies and ordinary users regarding cyber threats.

This awareness is critical given that many companies, including those that produce software, adopt a relatively lax approach to security. The ratio of employees in programming, administrative, and security departments is approximately 100:10:1. This approach, even among IT companies, is ruthlessly exploited by cybercriminals.

Currently, the most prevalent types of attacks include Man-in-the-Middle attacks, Brute Force attacks, Distributed Denial of Service (DDoS) attacks, Malware, Phishing, and Social Engineering [3, 4]. Man-in-the-middle attacks aim to intercept communication between two parties, allowing the attacker to capture and modify transmitted data. Such attacks frequently occur in environments with unsecured Wi-Fi networks, such as restaurants, airports, or pubs. [5] is a method that can defend against this type of attack.

Brute Force attacks involve guessing a password manually or, more commonly now, automatically testing all possible combinations until the correct one is found. The downside for attackers is the required time, making this method effective mainly for short and straightforward passwords (ideally, dictionary words) without special characters [6].

Distributed Denial of Service (DDoS) attacks are prevalent among experienced and novice cybercriminals. These attacks aim to overload a service provider's IT infrastructure by sending multiple requests from multiple sources simultaneously. Attackers may use large volumes of TCP and UDP packets or employ applications that send HTTP/HTTPS requests to web applications [7].

Among these threats, Malware is the most diverse. It encompasses any code added to or removed from software with the intent to cause harm or disrupt system functionality. Malware is a broad term that includes various forms of malicious software, such as trojans, ransomware, viruses, spyware, adware, and worms [8]. The article will explore Malware and analyse methods for detecting these threats.

The final types on the list of most common attacks are Phishing and Social Engineering. These attacks are similar as they aim to extract sensitive information such as passwords, personal data, or financial details from potential victims. They involve manipulating users to take actions desired by the fraudsters, often leveraging human nature—such as curiosity or gullibility. Many countries currently run awareness campaigns online, in print media, and on television to warn about such attacks, with a particular focus on protecting elderly individuals who are especially vulnerable [9].

2. Related Work

Research into malware in the IT world has been conducted since the emergence of the first virus. Undoubtedly, the first such threat was the Creeper

virus, created by Bob Thomas. The malicious software he developed in 1971 operated on the TENEX (TOPS-20) system within the ARPANET network. The program's purpose was to test the concept of self-replicating software. Interestingly, this virus did not cause significant harm; it merely displayed the message, "I am the creeper, catch me if you can! [10].

Although the virus was not harmful, it changed people's approach in the IT industry. IT personnel realized that there are no completely secure systems. Moreover, the appearance of the first virus opened a new chapter in the IT industry. This chapter undoubtedly discusses the development of antivirus systems. Creeper contributed to creating the first antivirus program in history, Reaper [11].

This software laid the foundation for developing highly advanced systems capable of detecting and removing malicious software. However, it also encouraged others to create malware, and the battle between cyber criminals and IT security specialists continues to this day and will persist indefinitely. One of the first researchers to take on the fight against malware was Fred Cohen. His academic research provided the groundwork for future methods of malware detection [12].

One of the first antivirus software producers was the American company McAfee Corp., founded by John McAfee in 1987. This company quickly became a leader in antivirus solutions, and its software used signature-based technology for many years, as discussed later in this work. McAfee's success was mirrored by companies like Symantec with its Norton AntiVirus product, F-Secure, and Kaspersky Lab—all tools above utilized signature-based technology during that period.

Just a few years ago, cybersecurity researchers primarily used two methods for detecting and analyzing malware: Signature-based Detection and Heuristic-based Detection. The former was the most commonly used. The signature-based detection method involves using pre-established signature databases and specific fragments of malicious code. The advantage of this approach is its high effectiveness in detecting known threats; however, its drawback is its inefficiency in dealing with new malware variants [13].

The second prevalent method used in malware detection was the Heuristic-based Detection method. This method involves analyzing the code of malicious software for suspicious behavior that may indicate the presence of malware. Unlike Signature-based Detection, this technique allows for the detection of new and unknown viruses and so-called zero-day attacks. Despite its advantages, this method has drawbacks, the most significant being the risk of generating false alerts (false positives) [14].

Over time, new concepts for detecting malicious software began to emerge. The late 1990s and early 2000s marked a period when behavioral analysis methods gained significant popularity. The rapid expansion of the Internet, along with the widespread sharing and distribution of malicious software, drastically increased the rate of malware propagation. Despite displaying very similar behaviors, these types of software often have varying structures due to different code modification techniques, such as polymorphism and metamorphism, which pose a substantial challenge for signature-based detection. It is also important to note that malware frequently relies on various API calls provided by operating systems, which makes behavioral analysis particularly effective in identifying such threats [15].

In addition to behavioral analysis, the beginning of the 20th century saw the development of several other techniques. Among the most important of these, which should be highlighted, are undoubtedly sandboxing, memory analysis, and reputation-based analysis.

Sandboxing is an isolated environment in which malicious software is executed and thoroughly analyzed. This allows for the verification of software without impacting critical infrastructure. Sandboxing has now become an integral part of corporate infrastructure, enabling the testing of malicious code without risk [16]. Unfortunately, even sandboxing is currently being 'bypassed' by cyber criminals. There are now malware strains capable of detecting whether they operate within a sandbox environment. They only activate once they find themselves outside the sandbox's perimeter. Attackers assess whether their application runs in a sandbox primarily by analyzing input data. In a typical environment, users often click the mouse or type on the keyboard. However, when malware operates within a sandbox, the attacker receives no input data. Fortunately, there are methods to 'hide' the sandbox environment from the malware. In article [17], the authors demonstrate how this can be achieved using Delphix. Delphix is an AI tool designed to mask data, making it more difficult for malware to analyze. Delphix replaces any sensitive or confidential data with realistic-looking alternatives that do not contain critical information. Cybercriminals in the current century are increasingly focusing on concealing their malicious software. Malicious software has been developed that leaves no traces on the disk, operating solely in the system's RAM.

The growth of the Internet, in addition to techniques like sandboxing, has significantly contributed to the emergence of reputation-based analysis. This method involves evaluating the trustworthiness of websites, domains, or files. It enables quick blocking of access to dangerous sources based on globally gathered data. The most famous example of such a service is VirusTotal, which we also utilized in our work [18]. The recent history undoubtedly belongs to machine learning and artificial intelligence. Thanks to the remarkable advancement in computational power and machine learning techniques, significant progress has been made in malware detection driven by these technologies. The analysis of vast datasets and the extraction of patterns based on specific behaviors enable the detection of increasingly sophisticated cyberattacks.

Furthermore, anomalies in abnormal behavior in systems, users, or networks have become crucial to modern security systems [19].

3. Malware Characteristics

Cybercriminals create malicious software to cause harm, steal data, or gain access to IT systems. The "malware family" encompasses various malicious software types, with the most common examples being viruses, trojans, worms, spyware, adware, ransomware, and rootkits. Each type behaves and operates differently, carrying out unique tasks. Viruses aim to replicate and modify other computer programs by inserting their infected code. They are most commonly spread through email attachments and files shared on hosting services. Trojans, in contrast, disguise themselves as legitimate software and rely on deceiving users to disrupt systems, perform harmful actions like data theft, or create backdoors.

Worms are similar to viruses primarily in their ability to replicate, but unlike traditional computer viruses, they do not attach themselves to existing programs. Spyware, as the name suggests, monitors user activity, gaining access to emails and even capturing keystrokes. Adware behaves similarly regarding user analysis, but its primary goal is to display unwanted ads to users, generating revenue for cybercriminals. Adware is often bundled with spyware.

In the 1990s, rootkits began gaining popularity. Rootkits aim to access the root (administrator) account while remaining undetected. They are capable of modifying software while staying hidden. Nowadays, rootkits are primarily used by cybercriminal groups and in Advanced Persistent Threat (APT) attacks, where the key objective is to conceal presence and maintain long-term control over the victim's system [20].

Another highly prevalent type of malware is ransomware. The goal of ransomware is to block access to a computer system by encrypting its files. In theory, the victim may receive a decryption key upon ransom. The most common currency in these cases is cryptocurrency, as transactions using cryptocurrencies are difficult to trace, making it challenging to identify the attacker [21].

We have examined ransomware and trojans in more depth later in this work, as these two types of malware were the primary focus of our research. We aimed to demonstrate how applying static and dynamic analysis methods enables the identification of malware and how these methods can contribute to the development of Intrusion Detection Systems [22].

4. Problems and Research Methods Used in the Project

The ever-increasing volume of threats in cyberspace has led to a growing demand for cybersecurity professionals year on year. Threat detection and response have become critical components of every organisation's operational framework. Due to the evolving characteristics of modern malware variants, traditional signature-based methods are proving to be progressively less effective with each passing month.

In light of these challenges, static and dynamic analysis techniques are gaining prominence and recognition, offering more accurate—and often profoundly comprehensive—insights into the behavior of executable files, thereby enabling the effective identification of malicious activities.

Static analysis allows us to examine malicious code without executing it, which facilitates the rapid and safe acquisition of valuable information, such as utilised libraries and the structural layout of the file. Conversely, dynamic analysis involves executing the code within a fully controlled environment, providing detailed visibility into the malware's runtime behavior—such as registry modifications, file operations, and attempts to establish connections with command-and-control (C2) servers.

This paper presents a comprehensive analysis of both methodologies, illustrated by an investigation of a well-known piece of malware. With the rapid evolution of cyber threats, increasingly sophisticated attack methods are emerging. The malware examined in this work, specifically ransomware and Remcos RAT, represent significant risks to individual users, critical infrastructure, and financial institutions, often leading to substantial financial losses, operational downtime, and privacy breaches.

This study thoroughly analyzes the behavior of these two threats using two essential techniques: static analysis and dynamic analysis. Static analysis enabled an assessment of the malicious software's code, facilitating the identification of specific attack patterns and vectors without executing the malware. On the other hand, dynamic analysis allows for observing malware behavior in a controlled environment.

Thanks to this, the characteristic patterns of WannaCry and Remcos RAT were observed in a practical example, and the effectiveness of static and dynamic analysis was assessed in the context of their potential use in IDS-type systems.

We began the study of the selected malware by setting up a test environment wholly isolated from the systems we use in our lab daily. Two virtual machines were connected to a virtual network: one running Security Onion, a comprehensive Linux distribution designed for network monitoring, traffic analysis, threat detection, and incident response, and the other with a Windows 10 Pro client system. Security Onion has many pre-installed tools for detecting and analyzing network traffic, such as Suricata, Zeek, Wireshark, Network Miner, Sguil and, among others, ELK Stack (Elasticsearch, Logstash, and Kibana) for managing and analyzing logs [23]. The Ghidra tool, created by the National Security Agency (NSA), was installed on a Windows 10 machine and used in reverse engineering as a tool that allows for binary code analysis and source code recovery from binary files.

Figure 1. Security Onion Desktop

This study examined two types of malware: the WannaCry ransomware and Remcos RAT. WannaCry, a form of ransomware that emerged in 2017, spread rapidly by exploiting a vulnerability in the Microsoft Windows operating system known as EternalBlue. The NSA initially identified this vulnerability and intended to use it by U.S. intelligence agencies as a cyber intelligence tool. However, a hacker group called "The Shadow Brokers" publicly released a collection of NSA tools, including EternalBlue. As a result, WannaCry quickly spread worldwide, infecting and encrypting data on over 200,000 computers across more than 150 countries [24].

The second type analyzed in the work was Remcos RAT. This Trojan allows unauthorized remote access to the victim's computer in order to spy and steal data. The most common method of spreading is via infected email attachments [25].

5. Laboratory Environment and Malware Analysis

The malware research was preceded by preparing a comprehensive lab environment in controlled conditions that allowed for simulated attacks on an endpoint running Microsoft Windows 10 (malware infection) and subsequent analysis of the malicious software.

Two virtual machines were connected to the simulated internal network: one running Security Onion for threat detection and analysis and another with Windows 10 functioning as the victim machine, equipped with a static analysis tool. After installing Security Onion, the essential monitoring components and a suite of analytical tools were configured.

On the Windows 10 machine, the Ghidra tool was installed and configured for static code analysis. Once configuration work was complete, selected malware research samples were downloaded from the MalwareBazaar website for use in the lab environment. This is a public database of malware samples. It offers access to even the latest malware, making this site a beneficial tool for researchers and security analysts [26, 27].

5.1. WannaCry Static Analysis

The first step in conducting static analysis is always to check the file's properties. Our executable file with the `.exe` extension was imported into the Ghidra tool. The import report revealed that the analyzed WannaCry is built for the x86 architecture, with 45 data types, 46 functions, and 96 symbols. The file was then passed on for detailed analysis.

Ghidra translated the binary code into assembly language. This step is one of the most important in conducting this type of analysis because it allows for understanding the operation of the malware, especially when the source code of the malicious software is not available.

Ghidra can quickly identify functions in the analyzed file, allowing us to examine its interactions with other operating system components. Inspecting the function labelled "entry" enabled us to study the entry code generated for executable files in Windows. Thanks to Ghidra's powerful capabilities, we were able to convert assembly code into C language, significantly simplifying and accelerating the malware analysis process.

`WinMain()` function was identified in the decompiled code, containing critical operations that initiate the WannaCry ransomware. This included functions responsible for establishing network connections to download resources. In WannaCry's case, a variable named `strange_url` was declared within the function assigned to a specific URL. This URL served as a "kill switch." The malware attempted to connect to this address—if the server responded, the malware would terminate its operation; if not, WannaCry would continue spreading the infection.

Through the discovery of this kill switch, WannaCry was ultimately defeated. Marcus Hutchins, who registered the domain, first made this discovery, thereby stopping the malware from spreading further. The malware's creators likely embedded this mechanism to halt the malware if it spiralled out of control and to facilitate testing in controlled environments.

In the second variant, WannaCry was analyzed using tools available at Security Onion to verify the malware regarding network communication. After importing the PCAP file, the packets were analyzed using Snort and Zeek. Using the Squert tool, we discovered two types of suspicious software: EternalBlue and DoublePulsar. EternalBlue is an exploit that exploits a vulnerability in the SMB protocol. In this example, WannaCry attempted to exploit the vulnerability using the EternalBlue exploit to spread across the network.

5.2. Remcos RAT Static Analysis

Extracting and searching the malware's strings allowed us to determine early on that we were dealing with an application designed for Windows systems, not, for example, MS-DOS. Using Ghidra, we reverse-engineered the malware. We showed that the malware is a keylogger that records every keystroke, video, and sound, and it does all of this using the `SendInput` function from the `winuser.h` library (see: Figure 5).

However, this was only one functionality of the Trojan. The research also discovered that the Trojan acted as a C2 (Command and Control) server, enabling

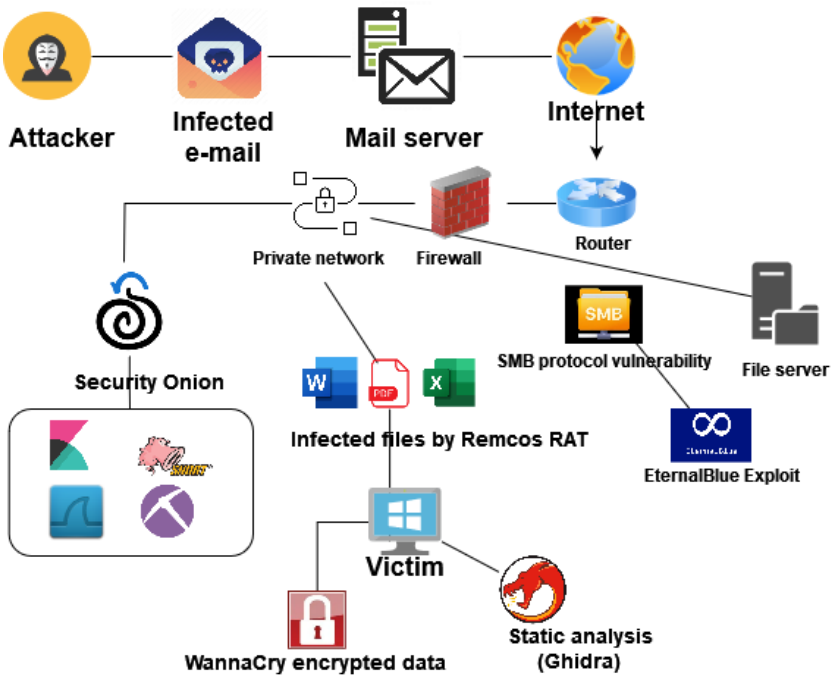

Figure 2. Controlled Lab Environment and Malware Infection Flow

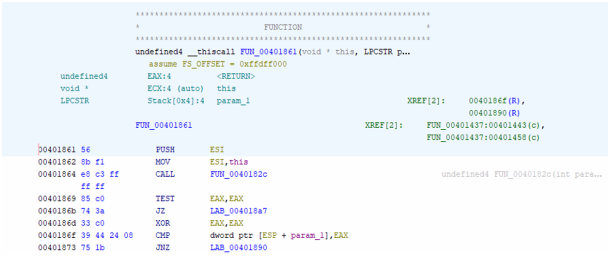

Figure 3. Fragment of disassembled WannaCry code

Figure 4. Key fragment of the decompiled code

it to execute commands in `cmd` and download files (see: Figure 6).

5.3. WannaCry Dynamic Analysis

The primary process spawned three child processes and several additional ones that terminated after completing their tasks. The primary process masked itself within the system, appearing as a standard system process. The ransomware operated using an RSA key pair (public and private) and an AES-128-CBC symmetric key for file encryption.

Since WannaCry propagated across the network through an SMB protocol vulnerability, filters for

Figure 5. Fragment of a function from the winuser.h library

Figure 6. Executing commands in cmd.exe

`tcp.port == 445`, `ARP`, and `DNS` were applied during packet analysis in Wireshark. In Wireshark, we observed DNS queries containing a domain name acting as a kill switch, which indicated the presence of WannaCry.

5.4. Remcos RAT Dynamic Analysis

During our analysis, we observed that a `notess` directory containing a file named `logs.dat` was created in the system. This file records the victim's activity, providing the attacker with continuous access to data entered by the victim on the computer.

During the data exfiltration phase, we discovered that the trojan communicates with an address functioning as a Command and Control (C2) server. Upon verifying this address, we determined that it is a high-risk IP address that cybercriminals have previously used for malicious activities on the network.

6. Conclusion

This study thoroughly analyzed two types of malware, the ransomware WannaCry and the trojan Remcos RAT, using both static and dynamic analysis methods. The tests and analyses that were conducted provided valuable insights into the effectiveness of these methodologies and the behavior of the analyzed malware.

Static analysis, which focuses on examining binary files, enabled the identification of characteristics such as file structure, string patterns, cryptographic mechanisms, and specific malware behaviors. On the other hand, dynamic analysis allowed real-time observation of the malware's actions, such as process creation and network communication.

The application of both analysis methods offers a comprehensive understanding of malware functionality. It significantly enhances preparedness and the ability to detect, analyze, and neutralize attacks, including known and emerging threats.

Combining static and dynamic analysis methods, a hybrid approach significantly enhances malware detection capabilities. The approach presented in this study demonstrates that in today's threat landscape—where the volume and sophistication of cyber threats are continually on the rise—adequate protection against emerging forms of malicious software is only achievable through a multi-layered defence strategy that leverages a diverse range of techniques.

Future research should focus on automating and integrating these methodologies while incorporating advanced machine learning models for detecting new threats and addressing the challenges of an ever-evolving cyber landscape.

Developing a model designed to correlate various metadata attributes with the observed behavior of malicious software in a controlled environment would enable even more effective threat classification. A promising direction for further research would be to expand the dataset to include additional malware families—such as fileless malware, Advanced Persistent Threats (APTs), and a broader range of ransomware variants.

Considering these enhancements, the proposed solution could be successfully implemented within Endpoint Detection and Response (EDR) systems or Security Information and Event Management (SIEM) platforms in production environments. This would also facilitate a more accurate assessment of the solution's effectiveness under real-world operational conditions.

AUTHORS

Andrzej Mycek* – CUT Doctoral School, Department of Computer Science, Faculty of Computer Science and Mathematics, Cracow University of Technology, Cracow, Poland, ul. Warszawska 24, 31-155 Cracow, Poland, e-mail: andrzej.mycek@pk.edu.pl.

Mirosław Roszkowski – Department of Computer Science, Faculty of Computer Science and Mathematics, Cracow University of Technology, Cracow, Poland, ul. Warszawska 24, 31-155 Cracow, Poland, e-mail: miroslaw.roszkowski@pk.edu.pl.

*Corresponding author

References

[1] V. Srivastava and V. Sharma, "Sandbox technology in a web security environment: a hybrid exploration of proposal and enactment", *International Journal of Scientific Research and Engineering Development*, vol. 5, no. 3, pp. 488–494, 2022.

[2] C. Hoofnagle, B. Van der Sloot, and F. Borgesius, "The European Union general data protection regulation: what it is and what it means", *Information & Communications Technology Law*, vol. 28, no. 1, pp. 65–98, 2019.

[3] A. Bendovschi, "Cyber-attacks – trends, patterns and security countermeasures", *Procedia Economics and Finance*, vol. 28, no. 1, pp. 24–31, 2015.

[4] A. Mycek and M. Łukaczyk, "Security of containerization platforms: threat modelling, vulnerability analysis, and risk mitigation", *ECMS 2024: Proceedings of the 38th ECMS International Conference on Modelling and Simulation*, pp. 585–591, 2024.

[5] A. Mallik et al., "Man-in-the-middle-attack: understanding in simple words", *International Journal of Data and Network Science*, vol. 3, no. 2, pp. 77–92, 2019.

[6] L. Bošnjak, J. Sreš, and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords", *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, vol. 41, no. 1, pp. 1161–1166, 2018.

[7] A. Singh and B. Gupta, "Distributed denial-of-service (DDoS) attacks and defense mechanisms in various web-enabled computing platforms: issues, challenges, and future research directions", *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 18, no. 1, pp. 1–43, 2022.

[8] A. Namanya et al., "The world of malware: an overview", *2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud)*, vol. 6, no. 1, pp. 420–427, 2018.

[9] I. Vayansky and S. Kumar, "Phishing – challenges and solutions", *Computer Fraud & Security*, vol. 1, no. 1, pp. 15–20, 2018.

[10] T. Chen and J. Robert, *The Evolution of Viruses and Worms*, CRC Press, 2004.

[11] M. Manjramkar and K. Jondhale, *Cyber Security Using Machine Learning Techniques*, Atlantis Press, 2023.

[12] E. Filiol, *Computer viruses: from theory to applications*, Springer, 2005.

[13] D. Venugopal and G. Hu, "Efficient signature based malware detection on mobile devices", *Mobile Information Systems*, vol. 4, no. 4, pp. 33–49, 2008.

[14] Z. Bazrafshan et al., "A survey on heuristic malware detection techniques", *IKT 2013 - 2013 5th Conference on Information and Knowledge Technology*, vol. 5, no. 1, pp. 113–120, 2013.

[15] H. Galal, "Behavior-based features model for malware detection", *Journal of Computer Virology and Hacking Techniques*, vol. 12, no. 2, pp. 59–67, 2016.

[16] V. Fasna and S. Swamy, "Sandbox: a secured testing framework for applications", *Journal of Technology & Engineering Sciences*, vol. 4, no. 1, pp. 3–11, 2020.

[17] V. Sathya et al., *An Obfuscation Technique for Malware Detection and Protection in Sandboxing*, volume 972, Springer, 2021.

[18] S. Megira, R. Pangesti, and F. Wibowo, "Malware analysis and detection using reverse engineering technique", *Journal of Physics: Conference Series*, vol. 1140, no. 1, pp. 1–12, 2018.

[19] M. Hossain Faruk et al., "Malware detection and prevention using artificial intelligence techniques", *2021 IEEE International Conference on Big Data (Big Data)*, pp. 5369–5377, 2021.

[20] L. Hughes and G. DeLone, "Viruses, worms, and Trojan horses: serious crimes, nuisance, or both?", *Social Science Computer Review*, vol. 25, no. 1, pp. 78–98, 2007.

[21] M. Akbanov and V. Vassilakis, "WannaCry ransomware: analysis of infection, persistence, recovery prevention and propagation mechanisms", *Journal of Telecommunications and Information Technology*, vol. 1, no. 1, pp. 113–124, 2019.

[22] A. Mycek, "Monitoring, management, and analysis of security aspects of IaaS environments", *Journal of Telecommunications and Information Technology*, vol. 94, no. 4, pp. 108–116, 2023.

[23] R. Heenan and N. Moradpoor, "Introduction to security onion", *PGCS 2016: The First Post Graduate Cyber Security Symposium - The Cyber Academy, Edinburgh Napier University*, pp. 1–4, 2016.

[24] Z. Liu et al., "Working mechanism of Eternalblue and its application in ransomworm". In: *Cyberspace Safety and Security*, pp. 178–191, 2022.

[25] V. Valeros and S. Garcia, "Growth and commoditization of remote access trojans". In: *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pp. 454–462, 2020.

[26] J. Chen and R. Deng, "Similarity-based malware classification using graph neural networks", *Applied Sciences*, vol. 12, no. 21, pp. 10837–10853, 2016.

[27] D. Priambodo et al., "Observe-Orient-Decide-Act (OODA) for cyber security education", *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 10, pp. 246–255, 2022.