Grenze International Journal of Engineering and Technology, June Issue

Analysing the Malware by using Checksum and Signature-Based Detection Techniques

M Manjunath Reddy[1], S Raghava[2], N Naga Tarun[3], S Chetan Reddy[4], Dr. G. Ramakoteswara Rao[5] and Dr. J. Vijaya Chandra[6]
[1,2]CSE-Cloud Computing, Koneru Lakshmaiah Education Foundation, Vijayawada,India
[3,4]CSE-Cyber security, Koneru Lakshmaiah Education Foundation, Vijayawada, India
[5]CSIT- Cloud Security, Koneru Lakshmaiah Education Foundation, Vijayawada, India
[6]CSE- Cloud Security, Koneru Lakshmaiah Education Foundation, Vijayawada, India
E mail: mnreddy.macha@gmail.com, raghavasurakasula07@gmail.com, nagatarun2003@gmail.com, sanivarapuchetanreddy@gmail.com, grkraoganga@gmail.com, vijaychandra.phd@gmail.com

***Abstract*— Malware has become an increasingly widespread security hazard to the internet in today's society. Malware is any harmful program that is designed to cause harm to the user. Malware analysis is critical for detecting, comprehending, and limiting the impacts of harmful software. It uses software and designs to attack weaknesses in computer systems, steal important information, and harm the system or network. It informs us about the features of malware such as viruses, worms, Trojans, and so on. Dynamic analysis and static analysis are the two divisions of malware analysis.**
The purpose of this research is to gain knowledge about the methods and techniques for analyzing and detecting malware effectively. In addition, this research discusses the various methodologies and types of software that are used in malware analysis.

***Index Terms*— Malware, Virus, Worm, Trojan Keylogger, Malware Analysis, Static Malware Analysis, Dynamic Malware Analysis.**

I. INTRODUCTION

Malware is an abbreviation for malicious software. It refers to any type of Intentionally designed software to harm a computer system, network, or mobile device. Malware comes in a variety of forms, including viruses, worms, Trojans, ransomware, spyware, adware, and others. Malware is usually created by cybercriminals with the intention of stealing sensitive information, disrupting computer operations, extorting money, or gaining Access to computer systems without authorization. The impact of malware can vary depending on its type and purpose. Some malware may cause minor annoyances or slow down a computer system, while others can result in serious data breaches, financial losses, or even system crashes. It is important to protect your devices with reputable antivirus software, keep your software up-to-date, and exercise caution when downloading or opening files from unknown sources.

An analysis of malware is a process of examining a piece of malicious software, or malware, For a better understanding of how it works, what it does, and how to detect and remove it. Malware analysis is an important part of cyber security, as it helps identify new and emerging threats, and enables security professionals to develop effective defenses against them. Static analysis involves examining the code of the malware without actually

Grenze ID: 01.GIJET.9.2.48

executing it. This can be done through reverse engineering techniques, such as disassembling or decompiling the code. Static analysis can help identify the functions and features of the malware, as well as any obfuscation or anti-analysis techniques it may use. Behavioral analysis involves observing the actions of the malware on a real system, without modifying its behavior. This can provide insights into how the malware affects the system and any associated networks.

II. METHODOLOGY

From our research, we have chosen two types of malware analysis they are dynamic analysis and static analysis.

A. Dynamic Analysis:

Dynamic analysis is a procedure carried out in a controlled setting to examine the actions of harmful software components. To understand its behaviors and system alterations, the virus is run in a sandbox or virtual machine environment. Security professionals can examine the activity and operation of the infection by using dynamic analysis, which is a key component of malware analysis [6]. For identifying the threat actors responsible for the malware, it can be useful to provide detection and removal tools and offer some information. There are various tools and methodologies for dynamic analysis.

- *Debugger*: A debugger allows researchers to set breakpoints, examine code, and track malware execution.
- *Network monitoring*: Command and control servers are frequently used by malware to send or receive stolen data. This traffic can be recorded and examined with tools for network monitoring.
- *System monitoring:* Malware can edit the system files, registry keys, and other information present in the system. It also provides researchers with the means to capture and understand their impacts.
- *Virtualization*: It is possible for malware to recognize when it is running in a virtual machine and change its behavior accordingly. By using virtualization tools, researchers can avoid detection while simulating various system environments.
- *Memory analysis:* By concealing itself in memory, malware is hard to detect with conventional methods. Memory analysis tools can be used by researchers to detect harmful processes or codes in a system's memory.

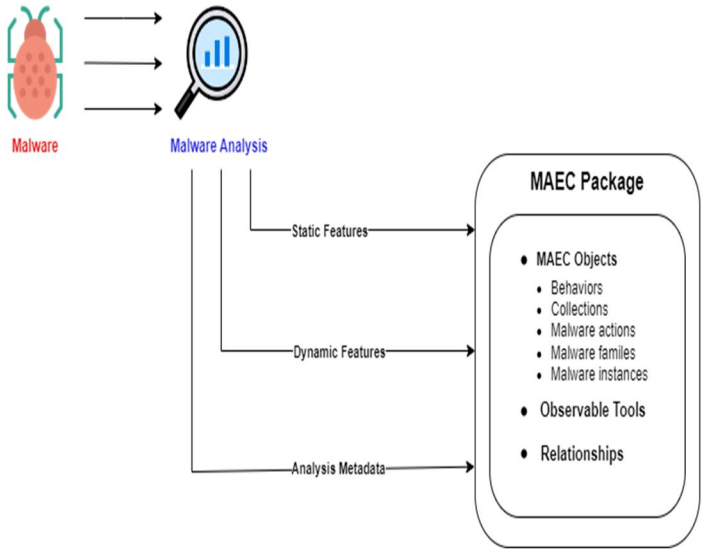
B. *Static Analysis*:

Static analysis is nothing but the collection of information from the application without running it. It consists of elemental analysis and examination of executable files without showing the actual instructions. The fundamental static analysis involves the examination of the type of malware, whether it is malicious or not, and also involves providing information about [4] the functionality of the application. It is a quick process. The first technique we use in the fundamental static analysis is uploading the suspicious executable to Virus Total.

Virus Total is software that shows the detection ratio of uploaded suspicious executable information. It is generally performed by finding the binary file signature, which gives a unique identification for the binary file. It can also be done by performing a cryptographic hash of the file. The executable code that was performed by the machine can be reverse-engineered and converted from assembly-level language code to a form of language that even humans can understand. Four techniques are involved in static analysis; they are Signature-based analysis, heuristic-based analysis, structural analysis, and behavioral analysis.

- *Heuristic*-based analysis: This technique uses a set of rules to find the data's potentially malicious code. The rules are set based on the characteristics of the commonly detected malware, such as obfuscation of the code, usage of packing tools, and also the size of the file.
- *Structural analysis*: This technique consists of analyzing the basic structure of the code to find the anomalies that are common in the malware. This also consists of searching for unused functions, strings that are not in use and also have unusual control flow.
- *Behavioural analysis*: This method involves analysis of the behavior of the code to identify potential malicious activity. This also involves searching for the code that communicates with the known malicious domains, modifying system settings, and also accessing sensitive data.

Nowadays, malware developers are constantly evolving their techniques to evade detection, and static analysis alone cannot detect all types of malware. The malware detection process can't be done alone, but it can be done through the combination of static and dynamic analysis to improve the accuracy of malware detection.



fig(1) : Features of Malware analysis

From the fig(1), To understand the inner workings of malware, static, dynamic, and behavioural analysis are required, as shown in the above image. These analyses can provide information that can be used to find malware, minimize its effects, develop defenses against it, and determine whether further study is necessary.
A malware instance can be examined using MAEC's static analysis tools. The static properties of a malware instance binary, such as how it was packaged, and interesting code fragments found by manually reversing its binary code, can disclose a range of information about the malware instance binary. You can apply MAEC to record specifics of the precise behaviours demonstrated by the malicious binary or code when evaluated dynamically. The level of abstraction can be gradually raised from the lowest level, which is often captured as a native system API call, to the highest level, which defines a specific type of destructive behaviour, such as key logging or exploiting vulnerabilities. It is for this reason that MAEC facilitates the recording of all details of analysis for a malware case in the MAEC Package, a single document containing all the details of the analysis. For the purpose of detecting malware in our research, we have employed checksum and signature-based techniques. By comparing those two ways, the best ethos will be selected that will be more useful in determining the accuracy and efficiency of malware.

A. Checksum

Checksum is one of the methods to detect malware files. Checksums are also known as hash sums and hash values. If the checksum values are unlike, then it will suggest that the file has been varied. The checksum is an effective method for detecting certain types of malware, but it is not reliable. Further, checksum alone [3] may not be sufficient to detect all types of malware and other methods such as behavior-based analysis and signature-based detection are also necessary. Five mathematical methods are employed to create a checksum value: MD5, SHA-1, SHA-256, CRC checksum, and Adler-32 checksum.
MD5:- MD5 is a broadly used algorithm that produces a 128-bit hash value. MD5 is commonly used for checksum and it is used to verify the integrity of files and data.
SHA-1:- A hash function that takes an input and produces a 20-byte hash result. This algorithm has been cryptographically broken but is still mostly used. The American National Security Agency created the blueprints for it.
SHA-256:- It's a hashing operation that produces a 256-bit [32 bytes] hash value. It is a family of hashing algorithms. It is no longer endorsed for use in cryptographic applications. It is no longer endorsed for use in cryptographic applications. Instead of a fixed-length hash result, SHA-256 applies a series of mathematical operations to a message of arbitrary length. The main use of SHA-256 is for digital signatures and data integrity checks. Provides shielded communication over networks.
CRC Checksum:- Cyclic Redundancy Check is a sort of checksum algorithm frequently used for data transfer fault detection [11].This algorithm will work by healing the data as a binary polynomial and splitting it by another polynomial called a generator polynomial’s checksum is used in network protocols such as [Ethernet and Wi-Fi],

storage devices such as [hard drives and USB drives] and digital communication systems [such as modems and fax machines].

B. Signature based-detection:

When identifying malware or other harmful behavior, signature-based detection looks for well-known patterns or signatures in files or network traffic. These signatures are often based on unique code sequences or behaviors connected to a certain class of malware or attack.

The limitations of signature-based detection include that it can only identify known threats for which signatures have been developed. Fresh and emerging dangers may not yet have known signatures, allowing them to elude detection. Moreover, attackers can alter their code [2] to evade detection by signature-based systems by using strategies like polymorphism or encryption. As a result, a lot of security professionals support a multi-layered strategy for security that combines behavior-based analysis and machine learning with signature-based detection. Security firms keep a database of dangerous signatures, which comprises the digital fingerprints of known viruses and malware. Similar signatures will be found when they are compared to the database. When a malicious file or piece of code is identified, the system can take action to prevent it from causing harm, such as quarantining or destroying it.

First, some risky programming examples. Extract malicious signatures that can define the attributes of each type of malicious software. Save the signature to a database so that it may be scanned again. After that, analyze the target program and search the database for anything suspect. We'll go through feature extraction, code pre-processing, and matching algorithms in great depth.

C. Differences between Signature-based detection and checksum detection:

- Signature-based detection approach searches code or files for specific patterns or signatures that match known malware signatures. Checksum detection method that computes a file's checksum and compresses it with the known checksum value of a clean file.
- Signature-based detection method, browse for exact matches of familiar malware signatures. Checksum audits for changes in file content that reshape the checksum value.
- Signature-based is highly effective. Checksum is less effective.
- Signature-based detection requires frequent updates of the signature database. Checksum is less resource-intensive, as it only requires a one-one-time checksum calculation.
- More false positives resulted in signature-based detection. Checksum detection has less likely of false positives.
- Signature-based detection is more accurate than the checksum.
- When a new file is encountered in signature-based and checksum detection methods, its signatures and checksums are compared w.r.t to the database of known malware signatures and checksums respectively. If a match is found, the file is flagged as malicious.
- Ultimately, all strategies have advantages and disadvantages, and many anti-malware solutions employ a combination of these and other techniques to provide effective malware protection.

III. Proposed Work

Finally, we choose the signature-based detection method because it works efficiently and compares the malicious code with the database; it will be considered malware, and the firewall will stop running the software [11] that contains malicious code. The reasons to choose this method are because of,

- Signature-based detection is effective in detecting the malware in the system, and it is pre-configured with a signature database that can be easily deployed.
- Signature-based detection does not require huge computation power and runs in the background without affecting system performance.
- This method is relatively low-cost and available as an anti-virus software solution.

A. Drawbacks of signature-based detection:

Signature-based detection is a conventional approach used to diagnose and prevent known hazards. However, there are a lot of drawbacks to this method, below are some of them:

Limited scope: Signature-based detection is only a powerful counter to known threats for which a signature has been assembled. It cannot identify fresh or undiscovered hazards.

Time-consuming: Updates to signatures can take a lot of time and resources, especially in the case of large businesses with a lot of data to monitor. Here is where the signature-based detection approach will take up time.
Ineffectiveness to detect polymorphic threats: Polymorphic hazards are malware that can change their code to evade signature-based detection. These threats cannot be detected with signature-based detection.
Incorrect positives: Signature-based detection can bring out false positives, which take place when can arise when an innocent file or behavior is falsely classified as malicious. And this will lead to Investigating false alarms is a waste of time and money.

IV. Architecture Of Signature-Based Detection

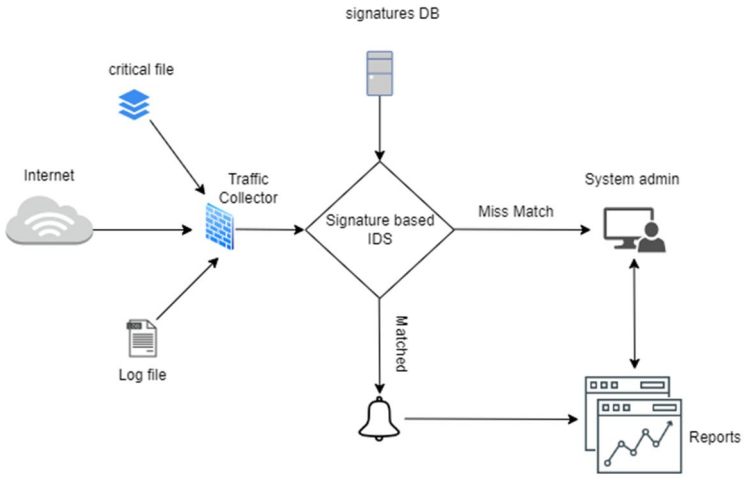


Fig (2):- Architecture of signature-based detection:

From the fig (2), the function of the traffic collector in signature-based detection, is to capture network traffic and organize it for scanning by the signature-based IDS block. This demonstrates the traffic collector is referred to be the system's "heart." Locating intrusions is the aim of the signature-based IDS [9]. When the signature-based IDS receives traffic from the traffic collector, it checks to determine whether any of the signatures already stored in the database match the traffic. When the signatures don't match, the system will inevitably report an incursion.
Generally, signature-based methods can be discovered in the HTTP headers of network packets as well as in data sequences that resemble viruses. A series of packets or a particularly precise data sequence can also contain an attack signature, as can the source or destination network addresses. These IOCs often include well-known byte sequences, malicious domains, and specialized network attack characteristics. They may also include email line subjects and file hashes. Signature-based detection uses a well-known set of IOCs.
This strategy has the fundamental disadvantage that it cannot detect attack signatures that have not been observed or stored in the past, which is its fundamental shortcoming. If malicious attackers wish to escape detection by the IDS, they may simply adjust the attack sequences within the malware they are attempting to inject or use other methods of attack. This result means the shield provided by an IDS can easily be breached if attackers adopt novel attack techniques that are completely different from those employed traditionally.

V. Result

According to our analytical results, malware assaults have grown more sophisticated, and malware developers are employing more strategies to escape detection and persist. The investigation and comprehension of malware require a mix of static and dynamic analysis techniques. In order to defend against malware attacks and improve the security posture of businesses and individuals, we may be able to use the data we have collected. Through the use of a virtual computer, we ran a dynamic analysis on malware samples. Certain malware strains also avoided analysis by detecting virtual machines and escaping sandboxes, as well as detecting virtual machines in the first place. Finally, we have concluded that Signature-based detection is a popular method for detecting malware compared to other malware detection methods because it is highly accurate and effective.

VI. Conclusion

The overall perspective of this research illustrates malware detection employing firewalls as well as methodologies that explain malware detection. According to the research, there are two forms of malware analysis, and these analyses reflect the identification of harmful software, as well as the varied approaches to detect malware utilizing firewalls. This study also discusses the many types of software used to identify malware, such as Virus Total and others. Although each malware's behaviour differs from one another, and their detection methods differ from one virus to the next.

References

[1] Al-Asli, M. and Ghaleb, T.A., 2019, April. Review of signature-based techniques in antivirus products. In 2019 International Conference on Computer and Information Sciences (ICCIS) (pp. 1-6). IEEE

[2] Jin, S., Chung, J.G. and Xu, Y., 2021, May. Signature-based intrusion detection system (IDS) for in-vehicle CAN bus network. In 2021 IEEE International Symposium on Circuits and Systems (ISCAS) (pp. 1-5). IEEE.

[3] Huang, C.T. and Gouda, M.G., 2005, June. State checksum and its role in system stabilization. In 25th IEEE International Conference on Distributed Computing Systems Workshops (pp. 29-34). IEEE.

[4] M. Ijaz, M. H. Durad and M. Ismail, "Static and Dynamic Malware Analysis Using Machine Learning," 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST), Islamabad, Pakistan, 2019, pp. 687-691, doi: 10.1109/IBCAST.2019.8667136.

[5] A. Jain and A. K. Singh, "Integrated Malware analysis using machine learning," 2017 2nd International Conference on Telecommunication and Networks (TEL-NET), Noida, India, 2017, pp. 1-8, doi: 10.1109/TEL-NET.2017.8343554.

[6] Aman, W., 2014. A framework for analysis and comparison of dynamic malware analysis tools. arXiv preprint arXiv:1410.2131

[7] Kaur, G. and Nagpal, B., 2012. Malware analysis & its application to digital forensic. International Journal on Computer Science and Engineering (IJCSE), 4(04), pp.622-626

[8] Egele, M., Scholte, T., Kirda, E. and Kruegel, C., 2008. A survey on automated dynamic malware-analysis techniques and tools. ACM computing surveys (CSUR), 44(2), pp.1-42

[9] Padmaja, B., Sravan, K.S., Patro, E.K.R. and Sekhar, G.C., 2021, November. A System to automate the development of anomaly-based network intrusion detection model. In Journal of Physics: Conference Series (Vol. 2089, No. 1, p. 012006). IOP Publishing.

[10] Sathyanarayan, V.S., Kohli, P. and Bruhadeshwar, B., 2008. Signature generation and detection of malware families. In Information Security and Privacy: 13th Australasian Conference, ACISP 2008, Wollongong, Australia, July 7-9, 2008. Proceedings 13 (pp. 336-349). Springer Berlin Heidelberg.

[11] Maxino, T.C. and Koopman, P.J., 2009. The effectiveness of checksums for embedded control networks. IEEE Transactions on dependable and secure computing, 6(1), pp.59-72.

[12] Yan, L.K. and Yin, H., 2012, August. Droidscope: seamlessly reconstructing the os and dalvik semantic views for dynamic android malware analysis. In USENIX security symposium (pp. 569-584).

[13] Wagner, M., Fischer, F., Luh, R., Haberson, A., Rind, A., Keim, D.A. and Aigner, W., 2015. A survey of visualization systems for malware analysis. In Eurographics conference on visualization (EuroVis) (pp. 105-125).

[14] Jovanovic, N., Kruegel, C. and Kirda, E., 2006, May. Pixy: A static analysis tool for detecting web application vulnerabilities. In 2006 IEEE Symposium on Security and Privacy (S&P'06) (pp. 6-pp). IEEE.

[15] Moser, A., Kruegel, C. and Kirda, E., 2007, December. Limits of static analysis for malware detection. In Twenty-third annual computer security applications conference (ACSAC 2007) (pp. 421-430). IEEE.

[16] Bisheh-Niasar, M., Azarderakhsh, R. and Kermani, M.M., 2021. Area-time efficient hardware architecture for signature based on Ed448. IEEE Transactions on Circuits and Systems II: Express Briefs, 68(8), pp.2942-2946.

[17] Edge, M.E. and Sampaio, P.R.F., 2009. A survey of signature based methods for financial fraud detection. computers & security, 28(6), pp.381-394.

[18] Stone, J. and Partridge, C., 2000. When the CRC and TCP checksum disagree. ACM SIGCOMM computer communication review, 30(4), pp.309-319.

[19] Gandotra, E., Bansal, D. and Sofat, S., 2014. Malware analysis and classification: A survey. Journal of Information Security, 2014.

[20] Idika, N. and Mathur, A.P., 2007. A survey of malware detection techniques. Purdue University, 48(2), pp.32-46.