JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

# AN OVERVIEW OF STATIC MALWARE ANALYSIS AND DETECTION

Harpreet Kaur
*Assistant Professor*
*School of Computer Science and Engineering*
*Lovely Professional University*
*Phagwara, Punjab, India*

Sri Vishnu Sai. B
*Student*
*School of Computer Science and Engineering*
*Lovely Professional University*
*Phagwara, Punjab, India*

Venkata Sai Kumar. N
*Student*
*School of Computer Science and Engineering*
*Lovely Professional University*
*Phagwara, Punjab, India*

Prudhvi Ranga Kurakula
*Student*
*School of Computer Science and Engineering*
*Lovely Professional University*
*Phagwara, Punjab, India*

**Abstract-- Malware is posing an increasing number of threats to people all around the world. A product that sneaks to your PC framework without your insight with a harmful purpose to disrupt the system activities. Because of the huge number of malwares, it is difficult to deal with malware by human specialists. As a result, security experts are working tirelessly to develop precise and practical ways for detecting malware. According to studies, malware's impact is diminishing. Two sorts of malware investigation are mainly considered for analysing the behaviour of certain binaries, namely- static and dynamic malware analysis .In this paper, different methods of performing malware analysis are reviewed, which gives the clear understanding of its characteristics and behaviour.**

**Keywords – Malware, Virus, Worm, Rootkit, Spyware, Trojan, Malware Analysis, Static Malware Analysis, Dynamic Malware.**

## I. INTRODUCTION

Now-a-days hackers are strengthening the malwares day by day to attack in the different ways to access data. Scholar people are trying to stop those attacks and standardize the systems to escape from attackers but it getting worse situation by strong malware.

On Internet numerous contributions are reachable and are moreover developing day via day. Web based banking or promoting are the instances of the business contributions of the Internet. Similarly, as in the real world, there are people on the Internet with noxious aims through taking addition of authentic clients each time cash is involved. Malware like programming system of malignant purpose helps these people directing their objectives.

Because of the enormous range of malware, it is beyond the realm of possibilities to expect to adapt to malware through human specialists. Consequently, security analysts use malware discovery designs to become mindful of malware.

Detection structures comprises of two phases: Analysis and Detection. Anti-virus program regularly uses signature-based technique to discover malware. This method is quick and success to realize acknowledged malware with minimal false positive rate. Still, signature-based fails to discover obscure malware and is easily overcome through malware that utilizes confusion techniques or methods. Then again, behaviour based is one more technique that is utilized in malware identification where suspicious archives are performed in a managed environment, monitored, and marked as malicious if their behaviours' suit with acknowledge malware behaviour. Behaviour based is competent to observe unknown malware and malware that utilization of obfuscation strategies, however it is time eating with tremendous false positive rate.

## II. Malware and Types

Malware is a malicious software program that is embedded into the system without client knowledge. It can damage the PC device by means of compromising PC functions, stealing information or evading get entry to controls. The various sorts of malware, which affects the typical activity of PCs, Servers, Mobiles, Tablets, and so on devices. These malware types are as per the following categories:

- Virus: It is a device program, which contaminates another device program by adjusting them to embrace a high-level duplicate of itself. It is a sort of malware, which copies itself and spreads to various devices. Numerous devices can be contaminated in the event that infections connect themselves to different codes and projects. This

contamination can be utilized to harm host devices, make botnets, take information, take money, and render ads

- Trojan: Trojan is a sort of malware that disguises itself as an important record or application to trick user and letting them download/install the software. A Trojan can provide vindictive remote right of entry to a contaminate PC. When the device gets infects by attackers, they will get the full access of the system and can manipulate the system settings and also can download some more malwares and also can-do unsafe activities like taking monetary information, information logins, even electronic cash, altering documents.
- Worm: A worm is a self-replicating malicious computer software that makes use of computer and organisational resources without validating client permission. It uses network capacity in the network of the targeted enterprise. This is a security flaw on the target PC.
- Rootkit: Rootkit is used on targeted system and it works in backend of the infected system. Detection of this rootkit is bit difficult. It is used to configure the system and unable to user to know that the system is got infected by malware.
- Spyware: Spyware is a type of malware used to monitor the activity doing by the user like monitoring activities in system, harvesting the data, and keystrokes. Spyware can also do some activities like changing some security settings to unable the device lo detect any issues in the targeted system.

## III. Malware Analysis

Malware analysis is the method involved with the purpose and character of a given malware test like worms, virus, trojan horse. This is a fundamental step to have the option to develop effective discovery strategies for malicious code. The malware analysis tools are broke into two classes: Static and Dynamic. The Static analysis tools endeavour to investigate a binary without really executing binary.

### A. Static Analysis

Static analysis refers to malware that is examined without being launched in a real-time context. Malware often utilises binary loaders such as UPX and ASP Pack Shell to prevent detection. Malicious script should be extracted and encrypted before analysis can begin. Decompiling a Windows executable file with a disassembler application like IDA Pro or Olly Dbg will reveal assembly instructions, provide information about the infection, and extract patterns to identify the attacker. Analyze the Op-code, control flow graph, string signature, windows API calls, and byte sequence n-grams to detect malware.

To interface with the operating system, almost all programmes use Windows API (Application Programming Interface) calls. For example, the Windows API "OpenFileW" in "Kernel32.dll" generates new files or opens current ones. As a result, API calls identify software behaviour and might be regarded a key metric in malware identification. For example, the Windows API methods "WriteProcessMemory," "LoadLibrary," and "CreateRemoteThread" are suspected of being exploited by malware to inject DLLs into a process, despite the fact that they are seldom used together in a valid context. In the memory analysis part, DLL injection is explored.

A CFG is a directed graph that depicts a program's control flow, with nodes representing code blocks and edges representing control flow pathways. In malware detection, CFG could be used to monitor the actions of a PE file and retrieve the programme structure.

N-grams were all of sequence's N-length continuous subsequences. The word "MALWARE," for example, is a seven-letter sequence that can be split down into the following segments: "MAL," "ALW," "LWA," "WAR," and "ARE. "NGrams have been used to identify API calls and opcodes, among other things. Other characteristics utilised in static analysis include file size and function length, in addition to the previously mentioned ones Networking components such as TCP/ UDP ports, destination IP addresses, and HTTP requests are shown in static analysis. Strings are an indicator that malware is running.

Strings are a sign of the presence of malware. Because strings typically contain significant semantic information, they indicate the attacker's purpose and aims. When a malicious file is located outside of the standard PE header, which is a common element of droppers and installers, the string "This programme cannot be launched in DOS mode" denotes it.

Opcodes are the initial component of a machine language that tells the CPU what operation to do. An opcode and one or more operands make up a full machine language instruction (e.g., "mov eax 7", "add eax ecx" and "sub ebx 1"). By checking opcode frequency or computing the similarity of opcode sequences, opcode may be used as a feature in malware identification.

Static analysis is a technique for examining a binary without having to run it. It is mostly done in a physical manner. If the source file contains, for example, several useful details such as data structures and functions used can be accessed. This data is lost once the source code is turned into a binary executable, prohibiting further investigation. Static malware research employs a variety of methods. A list of several of those is given below.

- File Fingerprinting: Before starting, it's a good idea to create a cryptographic hash value for each file under scrutiny. Although there are various hash functions available, the one that other researchers are most likely to employ for malware analysis is MD5, SHA1, or SHA256. You may use the file hash to see if the software has been updated or if it changes itself on a regular basis after computing it. A number of programmes exist that can calculate hash values for files.
- File Format: Furthermore, relevant information may be retrieved by exploiting metadata of a specific file type. This includes the magic number needed to detect the file format on UNIX systems. A large volume of data may be retrieved from a Windows binary, which is normally in PE format (portable executable), such as compile time, import and export operations, as well as text, menus, and icons.
- Packer Detection: Malware is now commonly disseminated in an obfuscated format, such as encrypted or compressed. A packer is utilised to do this, while other algorithms can be employed to modify the data. From a static analysis standpoint, the programme looks very different after packing, and its logic, as well as other information, is difficult to retrieve. While there

are several unpackers, such as PEiD2, there is no general unpacker, making static malware analysis a big task.

- AV Scanning: If the binary under study is well-known malware, it will almost certainly be detected by one or more antivirus scanners. Using one or more antivirus scanners takes time, but they are sometimes necessary because some anti-virus detects some kind of virus. So, to detect the malware we have to take no chance to leave malware without detection.
- Disassembly: The disassembly of a binary is usually the most important element of static analysis. This is done with the use of programmes like IDA Pro, which can reverse machine code into assembly language. A researcher could therefore investigate the programme logic and thereby assess its purpose using the rebuilt assembly code.

The major benefit of static malware analysis is that it allows for a thorough examination of a particular binary. Such that, this can cover all of a malware sample's unique execution tracks. Furthermore, because the source code is not really evaluated, static analysis is often safer than dynamic analysis. It may, however, be exceedingly time-consuming and so necessitates skill.

## IV. Static Malware Detection

During static analysis, the signature of the malware binary file is evaluated, which is a unique identifier for the binary file. A disassembler, such as IDA, can be used to convert machine-executable code to assembly language code, allowing the binary file to be read by humans. Static analysis employs techniques such as file fingerprinting, virus scanning, memory dumping, packer identification, and debugging.

Static analysis is a **"Signature-based"** approach for detecting and analysing malware. A signature is a byte sequence that acts as a one-of-a-kind identifier for malware. Signatures may be scanned in a number of different methods. Signature-based antimalware programmes are effective against the majority of malware, but they are ineffective against complex and advanced malware.

The static detection method looks for malware, discovers it, and destroys it using signatures. Malware that poses a security risk to computer networks and systems is regularly detected using this technique. Aside from data mining, machine learning, and other heuristic techniques for malware detection, a new method known as opcode extraction has just been created. The strategies were utilised as part of the feature determination methodology to minimise the number of features. The model discovered malware with a sensitivity of approximately 98 percent and an accuracy of around 99 percent, according to the findings of the research. The researchers have created a database with the viral types that have been identified as well as the regulations that govern them.

Signature-based detection matches the series of symbols contained in a file to known virus variants kept in a database after it has been examined. If an element is found, the signature-based detection algorithm classifies the file as a virus. The signature detection approach is based on a database of known virus types, which is worth emphasising. The database was built by researching known viruses, extracting instruction sequences from them, and removing any sequences that were not characteristic of worthwhile programmes.

A number of major security issues have been discovered using signature-based malware detection approaches in several commercial antivirus programmes. Their methods generated a significant number of obfuscated types of known bugs, which were tested on a range of antivirus software, proving that these methods were unsuccessful in discovering these viruses. This emphasises the vital need of developing a malware signature generation algorithm, which is commonly used in antivirus software.

Signature-based detection is a method of identifying malicious code in which the signature is based on a section of the virus's code that has been deleted. Using this way, the scanner will scan messages, programmes, files, emails, and other data. To compare these files to the signatures in the data, procedures are utilised.

```
Data: Contenting Directory Files, Virus List Data Base
Result: Detected Files

1  Begin
2  | For each Directory Files do
3  | | If Directory Files Name not equal Null then
4  | | | For each Virus List Name do
5  | | | | If Directory Files Name equal Virus List Name then
6  | | | | | ++ Detected Files [Name];
7  | | | | end
8  | | | end
9  | | end
10 | end
11 end
```

Signature Detection methodology

Meanwhile, the detection takes into account two criteria that suggest a malicious activity: When the "Directory File Name" is not null, and when the "Virus List Name" is equivalent to the "Directory File Name," Similarly, this algorithm's quality features were rated simple (one nested loop) and reusable.

```
Data: Contenting Directory Files, Virus List Data Base
Result: Detected Files

1  Begin
2  | For each Directory Files do
3  | | If Directory Files Name not equal Null then
4  | | | For each Virus List Name do
5  | | | | If Directory Files Name equal Virus List Name then
6  | | | | | ++ Detected Files [Name];
7  | | | | | Remove File by Name
8  | | | | end
9  | | | end
10 | | end
11 | end
12 end
```

Signature Removal Technique

The elements that affect removal are comparable to those that affect detection. The removal algorithm finds and eliminates malware in less than four seconds when the

"Directory File Name" is not null and the "Directory File Name" equals "Virus List Name." Similarly, this algorithm's quality features were rated simple (one nested loop) and reusable.

## V. Limitations in Static Malware Analysis

In general, malware samples' source code is not usually available. As a result, static analysis techniques for malware analysis that retrieve information from the malware's binary representation may be utilized. Consider the fact that the IA32 instruction set is used by the majority of malware activities. Disassembly of such products may provide misleading results if the binary employs self-modifying code mechanisms.

## VI. Summary on analysis of Tools

In our Survey on Static Malware Analysis tools are different but at the end the result will be same. To improve identification, use as many antivirus analysis algorithms as necessary. Mostly used tool for analysis was Virus Total (Virus Total, 2008) Look for such string with in malware's code. For strings use tool Strings (Microsoft, 2008c)

| Technique | Definition |
|---|---|
| 1.File Fingerprinting | For each file under investigation, you might wish to generate a cryptographic hash value. Malware analysis is most likely to employ MD5, SHA1, or SHA256. |
| 2.File Formatting | A Significant Type of data is going to be extracted from a windows binary and using metadata some data is accessed. On UNIX systems, this includes the magic number required to detect the file format. |
| 3.Packet Detection | Programme looks different after packing and difficult to retrieve. We do some unpacking using some tools such as PEiD2, there is no general unpacker. |
| 4.AV Scanning | We use multi-Antivirus tools to find the malware, If it is a well-known malware. |
| 5. Disassembly | Machine code is converted to Assembly Language using IDA Pro. By conversion, Researcher might understand the logic and estimates the purpose. |

## VII. References


[1] Anson, Steve. *Applied Incident Response*. John Wiley & Sons, 2020.

[2] Aslan, Ömer. "Performance comparison of static malware analysis tools versus antivirus scanners to detect malware." In *International Multidisciplinary Studies Congress (IMSC)*. 2017.

[3] Bermejo Higuera, Javier, Carlos Abad Aramburu, Juan-Ramon Bermejo Higuera, Miguel Angel Sicilia Urban, and Juan Antonio Sicilia Montalvo. "Systematic approach to malware analysis (SAMA)." *Applied Sciences* 10, no. 4 (2020): 1360.

[4] Bhojani, Nirav. "Malware Analysis." *Malware Analysis* (2014): 1-5.

[5] Chikapa, Macdonald, and Anitta Patience Namanya. "Towards a fast off-line static malware analysis framework." In *2018 6th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pp. 182-187. IEEE, 2018.

[6] Dutta, Nitul, Nilesh Jadav, Sudeep Tanwar, Hiren Kumar Deva Sarma, and Emil Pricop. *Cyber Security: Issues and Current Trends*. Springer, 2022.

[7] Gopaldinne, Sanjeev Reddy, Harpreet Kaur, Pawandeep Kaur, and Gunseerat Kaur. "Overview of PDF Malware Classifiers." In *2021 2nd International Conference on Intelligent Engineering and Management (ICIEM)*, pp. 337-341. IEEE, 2021.

[8] Harris, Shon, Allen Harper, Chris Eagle, Jonathan Ness, and Michael Lester. *Gray hat hacking: the ethical hacker's handbook*. McGraw-Hill Osborne Media, 2004.

[9] Jawad, Ahmad Ridha, Khaironi Yatim Sharif, and Ammar Khalel Abdulsada. "N/a and signature analysis for malwares detection and removal." *Indian Journal of Science and Technology* 12 (2019): 25.

[10] Kaur, Gursimran, and Bharti Nagpal. "Malware analysis & its application to digital forensic." *International Journal on Computer Science and Engineering (IJCSE)* 4, no. 04 (2012): 622-626.

[11] Kris, Kendall, and McMillan Chad. "Practical malware analysis." In *Black Hat Conference, USA*, p. 10. 2007.

[12] Kunwar, Rakesh Singh, and Priyanka Sharma. "Malware Analysis: Tools and Techniques." In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, pp. 1-4. 2016.

[13] Mohamed, Gamal Abdel Nassir, and Norafida Bte Ithnin. "Survey on representation techniques for malware detection system." *Am. J. Appl. Sci* 11 (2017): 1049-1069.

[14] Monnappa, K. A. *Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware*. Packt Publishing Ltd, 2018.

[15] Sharif, Monirul, Vinod Yegneswaran, Hassen Saidi, Phillip Porras, and Wenke Lee. "Eureka: A framework for enabling static malware analysis." In *European Symposium on Research in*

*Computer Security*, pp. 481-500. Springer, Berlin, Heidelberg, 2008.

[16] Shijo, P. V., and A. J. P. C. S. Salim. "Integrated static and dynamic analysis for malware detection." *Procedia Computer Science* 46 (2015): 804-811.

[17] Sihwail, Rami, Khairuddin Omar, and KA Zainol Ariffin. "A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis." *Int. J. Adv. Sci. Eng. Inf. Technol* 8, no. 4-2 (2018): 1662-1671.

[18] Talukder, Sajedul, and Zahidur Talukder. "A survey on malware detection and analysis tools." *International Journal of Network Security & Its Applications (IJNSA) Vol* 12 (2020).

[19] Vidyarthi, Deepti, C. R. S. Kumar, Subrata Rakshit, and Shailesh Chansarkar. "Static malware analysis to identify ransomware properties." *International Journal of Computer Science Issues (IJCSI)* 16, no. 3 (2019): 10-17.

[20] Z. Tzermias, G. Sykiotakis, M. Polychronakis, and E. P. Markatos, "Combining static and dynamic analysis for the detection of malicious documents," in Proceedings of European Workshop on System Security (EUROSEC), 2011.