ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 10, Issue 2, pp: (1-4), Month: April - June 2022, Available at: www.researchpublish.com

Malware analysis and detection using reverse Engineering

[1]B.Rashmitha, [2]J. Alwina Beauty Angelin, [3]E.R. Ramesh

[1]Department of Information Security and Digital Forensics,
Dr. M.G.R. Educational and Research Institute, Chennai - 600095, Tamil Nadu, India.
[2]Department of Information Security and Digital Forensics,
Dr. M.G.R. Educational and Research Institute, Chennai - 600095, Tamil Nadu, India.
[3]Center of Excellence in Digital Forensics, Chennai - 600096, Tamil Nadu, India.

Abstract: **The exponential growth of the internet and new technology lead today's world in a hectic situation both positive as well as the negative module. Cybercriminals gamble in the dark net using numerous techniques. This leads to cybercrime. Cyber threats like Malware attempt to infiltrate the computer or mobile device offline or internet, chat(online), and anyone can be a potential target. Malware is also known as malicious software is often used by cybercriminals to achieve their goal by tracking internet activity, capturing sensitive information, or blocking computer access. Reverse engineering is one of the best ways to prevent and is a powerful tool to keep the fight against cyber attacks. Most people in the cyber world see it as a black hat—It is said as being used to steal data and intellectual property. But when it is in the hands of cybersecurity experts, reverse engineering dons the white hat of the hero. Looking at the program from the outside in –often by a third party that had no hand in writing the code. It allows those who practice it to understand how a given program or system works when no source code is available. Reverse engineering accomplishing several tasks related to cybersecurity: finding system vulnerabilities, researching malware &analyzing the complexity of restoring core software algorithms that can further protect against theft. It is hard to hack certain software.**

Keywords: **Malware, threat, vulnerablity, detection, reverse engineering, analysis.**

1. INTRODUCTION

Technology has made our lives convenient, it has also allowed a new form of crime, cyber threat. Cybercriminals can attack computers by using malware to track internet activities and capture sensitive information such as usernames and passwords from financial websites. Malicious software is any program or file that is intentionally designed to harm, infiltrate, or damage a computer, server, or computer network. This software can disable or disrupt the operation of a system, allowing hackers to gain access to confidential and sensitive information and to spy on the computer and the owner of the computer itself. Malware is specifically made to be hidden so that it can remain inside a system for a certain period without the knowledge of the system owner. Malware analysis by using the reverse engineering method becomes one solution that can be used to extract data in malware to find out how the malware is working when it attacks the system. Therefore, this study aims to perform malware analysis to know the dangers of malware and how to prevent it and protect our devices against it. In this study, a file named best.exe will be used as a malware sample to find out information about malware contained in it.

In this research, we enhance investigated data analysis using a marble sample to have a brief note on the derived solution data that is extracted, which helps to give a solution by analyzing the resource data evidence.

The purpose of reverse engineering is usually to duplicate or improve the functionality of the original product by detecting the solution.

2. LITERATURE SURVEY

During the research, many paper-based on reverse engineering and sample were there had different techniques like machine learning, AI so on….Another research investigated the developments in Android Malware Detection, which uses both the techniques static and dynamic analysis. The setback in evaluating approaches to Malware attacks is due to the absence of complex Malware datasets. Recent research combines the two techniques of static and dynamic analysis and that has been used as an indicator to analyze Trojan-based behavior. When the Windows operating system gets infected this represents how the behavior of Trojan will be. In the last decade, malware researchers analyzed Malware samples using a static and dynamic method. The outcome of these techniques issued a piece of detailed information on the Malware. But, the problem is that it takes a long process of time to gather complete information about the Malware. The first process of 3 Malware analysis was to identify the distrusted Malware program and to find what type of Malware it is. Here, when it comes to Malware, any type of Malware can shut down Windows systems for any operating system like antivirus, firewall, etc. The work of analyzed Malware has proven that the reverse engineering technique has a huge success rate in analyzing this Malware. But, it causes shoe problems when reversing the Malware. This malicious software has been presently increasing at a startling rate.

3. METHODOLOGY

Static malware analysis

Description:

Static analysis examines a malware file without actually running the program. This is the safest way to analyze malware, as executing the code could infect your system. In its most basic form, static analysis gleans information from malware without even viewing the code. Metadata such as file name, type, and size can yield clues about the nature of the malware. MD5 checksums or hashes can be compared with a database to determine if the malware has been previously recognized. And scanning with antivirus software can reveal what malware you're dealing with.

Advanced static analysis—also known as code analysis—dissects the binary file to study each component, still without executing it. One method is to reverse engineer the code using a disassembler. Machine code is translated into assembly code, which is readable and understandable. By looking at the assembly instructions, an analyst can tell what the program is meant to do. A file's headers, functions, and strings can provide important details. Unfortunately, modern hackers are adept at evading this technique. By embedding certain syntax errors into their code, they can misdirect disassemblers and ensure the malicious code still runs. Because static malware analysis can be more easily foiled, dynamic malware analysis is also necessary.

PROCEDURE:

Reverse engineering process in software or application can be implemented with the steps:

1. Assembly: Assembly language is used for microprocessors and other programmable devices which any low-level programming language. An assembly language is not just a single language but rather a group of languages and the most basic programming language available for any processor. Assembly language cannot recognize high-level languages like Java and Pascal.

2. Disassembly: Disassembly is used for transforming assembly language into machine code. Disassembly is a reverse assembly process [2].

3. Debugging: Debugging is a method that developers can implement to search bugs, subtract bugs, and isolate the source of the problem. Debugging is used for executing testing from each core process in malware [11].

4. X86 Architecture: The X86 architecture is a design of complex instruction set computer with varying instruction lengths. Basically, on the internal; most modern computer architectures including x86 follow the Von Neumann architecture. In the design of the reconfigurable system, interoperability could also be an issue in the architecture [12].

5. Instruction: Instruction is constructed from an assembly program. An assembly of x86 instruction consists of mnemonic and zero or operands.

6. Hashing: The hash process is executed for verification before and after the malware analysis process. Verification is executed to determine the absence of hash changes in the sample malware after the analysis process.

7. String analysis: Strings in a program are values that will be loaded from a malware sample when executed. A reverse engineering process must be done for string analysis to get strong evidence from malware samples.

ALGORITHM:

String analysis is the process of extracting readable Ascii and Unicode characters from the binary. Not all the strings found are used by the program; attackers may also include fake strings to disrupt the investigation.

Tools used for string analysis:

• Strings2 – command-line utility, Windows 32bit/64bit executable, is used for extracting strings from binary data. This application is an improved version of the classic Sysinternals strings approach and can also dump strings from process address spaces. At the time of writing, Strings2 could be downloaded

from the following link: https://github.com/glmcdona/strings2.

• Flare-Floss (obfuscated string solver) - combines and automates different techniques to perform string decoding. At the time of writing, the Floss tool could be downloaded from the following.

link: https://github.com/fireeye/flare-floss.

Static analysis techniques & tools:

VirusTotal:

By uploading a file to VirusTotal, and cross-referencing it with a list of detections from various antivirus programs, the analyst will discover whether the sample is malicious or not. This process also provides information regarding the file, such as SHA256, MD5, file size, signature info, section details, imports, etc. If it is not possible to upload the sample to VirusTotal, the platform also provides the option to query for an existing sample that was already uploaded on the website by searching after the hash value of your sample.

EXTRACTING DATA:

PEID tool:

PEiD is a tool used for analyzing the PE header to give the analyst more details about the cryptos, packers, and compilers found in the executable files. PEiD makes this identification by using static signatures stored within the application.

CFF Explorer:

CFF Explorer is a tool commonly used to make modifications inside the PE. It runs on Windows OS and has the capability of listing processes or dumping the process to a file.

By using this tool, the analyst can extract the compilation date and architecture type from the analyzed malware sample, based on the information inside the PE Header. The compilation data is presented using EpochUnix Time in the 'TimeDateStamp' rubric. In this case, the date is 'GMT Sunday, July 13, 2008, 6:47:12 PM'.

Resource Hacker:

Resource Hacker is a free application that can be used for extracting, modifying, or adding resources(images, dialogs, menus, etc.) from Windows binaries.

Using Resource Hacker can help in analyzing dropper samples that have an additional PE file inside their resources. The tool can also be accessed from the command line without having to open the Resource Hacker GUI.

PEStudio:

PEStudio is a tool used to find suspicious artifacts within executable files to accelerate the initial malware assessment. By using this tool, the analyst can easily spot the functionalities that are commonly used for malicious activities by the malware creators.

When the analyst opens the malicious sample inside the program, general information regarding the file, such as MD5 hash and entropy, is obtained. The hash value of the sample will then be checked on Virus Total and the result of The lookup will be listed inside the program.

4. SOLUTION

Thus, In the field of cyber security, reverse engineering can be used to identify the details of a breach that how the attacker entered the system & what steps were taken to breach the system. Therefore, it is used to enable us to identify their techniques to prevent it in the future. Malware samples help to screw deeper into the dark web and what is the possibility where the threat can play a role. The main purpose of this research is to analyze the data and investigate the evidence.

The purpose of reverse engineering is usually to duplicate or improve the functionality of the original product by detecting the solution.

5. CONCLUSION

In this paper, we focus on the implementation of malware analysis using static analysis methods to provide guides and an overview of how to analyze malware. On the analysis of malware using the basic method of static analysis, the first thing to do is doing identification at the program which is alleged malware or not, besides that on this method also detects packed/obfuscated technique used by malware, as well as finding malware creation time. Meanwhile, on the malware analysis with advanced static analysis methods capable of providing more complete information about characteristics of malware, such as the information of malware to infect another program, as well as modifying the registry and creating new files and folders. It enhances that dynamic analysis takes a long process and the risk factor is high.

Based on this research, the merging of the two methods of malware analysis which is a static analysis and dynamic analysis can provide a more complete picture of the characteristics of malware TT.exe. Further issues for malware analysis with static and dynamic analysis require a long time in the process. In the future need to minimize the time for doing malware analysis but still obtain the detailed result from the malware.

REFERENCES

[1] Hex-Rays SA. 2020. IDA Pro – Accessed May 2020.

[2] Hex-Rays SA. 2020. F.L.I.R.T. – Hex-Rays. Accessed May 2020.

[3] Microsoft. 2020. InternetOpenA function (wininet.h) – Win32 apps | Microsoft Docs. Accessed May 2020.

[4] Hex-Rays SA. 2020. IDA Technology: Open Plug-In Architecture – Hex-Rays. Accessed May 2020.

[5] National Security Agency. 2020. Ghidra. Accessed May 2020.

[6] Microsoft. 2020. Debugging Tools for Windows (WinDbg, KD, CDB, NTSD) – Windows drivers | Microsoft Docs. Accessed May 2020.

[7] X64dbg Community. 2020. X64dbg. Accessed May 2020.

[8] Immunity Inc. 2020. Immunity Debugger. Accessed May 2020.

[9] Oleh Yuschuk. 2014. OllyDbg v1.10. Accessed May 2020.

[10] Microsoft. 2020. ShellExecuteExA function (shellapi.h) – Win32 apps | Microsoft Docs. Accessed May 2020.

[11] info. 2020. Detect It Easy. Accessed May 2020.

[12] FireEye Labs. Obfuscated String Solver. Github. Accessed May 2020.

[13] Strings2. Accessed May 2020.

[14] Practical Binary Analysis. 2018. Dennis Andriesse. No Starch Press (December 18, 2018)

[15] Mastering Malware Analysis. 2019. Alexey Kleymenov. Packt Publishing; 1 edition (June 6, 2019)

[16] Procmon. Accessed May 2020.

[17] Process Monitor for Dynamic Malware Analysis. Windows Sandbox Hari Pulapaka. Accessed May 2020.

[18] Practical Malware Analysis. 2012. Michael Sikorski and Andrew Honig. No Starch Press; 1 edition (February 1, 2012).