MALWARE ANALYSIS

ELABORATED BY :

- NOUÂMANE BAHIJ
- AYMEN MACHAY

Table of Contents

Acknowledgement:

We would like to express our sincere gratitude to all those who have contributed to the completion of this report. Firstly, we would like to thank our beloved family and friends for their unwavering support and encouragement. Your constant belief in us has been a tremendous source of motivation, and we are truly grateful for the love and understanding you have shown throughout this project.

We would also like to extend our thanks to EMSI, The educational environment provided by the institution played a pivotal role in shaping our understanding and providing resources for this report. The opportunities and guidance we received have been instrumental in our growth and development.

Furthermore, we would like to express our deepest appreciation to our teacher Chawki EL BALMANY for their supervision and mentorship. Your expertise, patience, and insightful feedback have been immensely valuable. Your guidance throughout the process has helped us refine our ideas and present them effectively.

Finally, we would like to acknowledge all the individuals who supported us indirectly, whether through their contributions, encouragement, or expertise. Your influence has contributed significantly to the success of this report, and we are truly grateful for your involvement.

Once again, thank you to everyone involved in making this report a reality. Your support, guidance, and encouragement have been indispensable, and we are honored to have had the opportunity to work with such wonderful individuals.

Introduction

Background and Purpose

Malware has become an ever-present threat in today's digital landscape, compromising the security and privacy of individuals and organizations worldwide. The analysis of malware plays a crucial role in understanding its behavior, intent, and impact, ultimately aiding in its detection, prevention, and mitigation. This report presents a comprehensive analysis of a specific malware sample, with the primary goal of dissecting its inner workings, identifying potential threats, and providing insights into its characteristics.

Objectives

The objectives of this malware analysis project are as follows:

1- **Malware Identification:** To identify and obtain a comprehensive understanding of the provided malware sample, known as "Sample_MD5_Hash."

2- **Behavioral Analysis:** To observe and document the behavior of the malware within a controlled environment, including any system modifications, network interactions, and persistence mechanisms.

3- Classification and IOCs: To classify the malware based on its behavior and characteristics and compile a list of Indicators of Compromise (IOCs) for detection and prevention.

4- Recommendations: To provide recommendations and mitigation strategies for protecting against similar malware threats.

This analysis is based on the specific malware sample located at the following VirusTotal URL: https://www.virustotal.com/gui/file/dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56 b8bb9000/details

The insights gained from this analysis will contribute to enhancing cybersecurity practices and better equipping organizations and individuals to defend against evolving malware threats.

Virtual Machine Setup

Virtualization Software

For the purpose of this malware analysis project, a Windows 10 virtual machine was utilized as the analysis environment. VMware Workstation, a robust virtualization software, was employed to create and manage the VM. VMware Workstation offers powerful features for configuring and isolating virtualized environments, making it an ideal choice for malware analysis.

Flare VM Installation
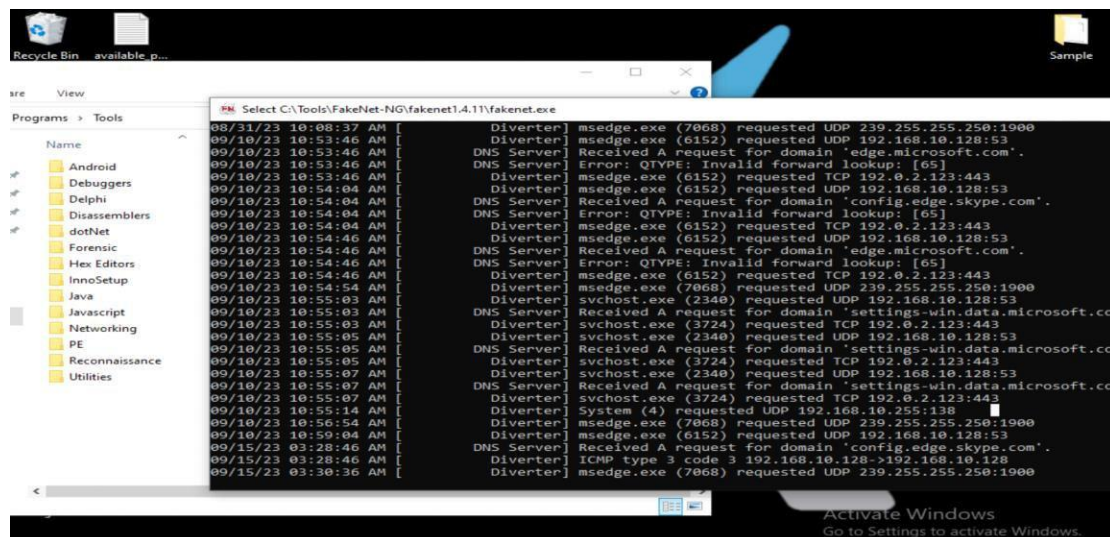
Within the Windows 10 virtual machine, Flare VM was installed. Flare VM is a specialized security distribution for Windows-based analysis, pre-configured with a comprehensive set of malware analysis and reverse engineering tools. This distribution simplifies the setup process and provides a rich toolkit for analysis.

Networking Configuration

To maintain a high level of isolation and security during the analysis, the virtual machine's network settings were configured to "Host-Only" mode. This configuration restricts network communication to the virtual machine exclusively, preventing any direct external connectivity. As a result, the malware sample under analysis cannot interact with the broader internet or other devices on the local network, enhancing the security of the analysis environment.

**No Additional Tools Installed for Extra Security**

To minimize potential security risks, no additional tools or software were installed on the virtual machine beyond the Flare VM distribution. This approach ensures that the analysis environment remains as clean and controlled as possible, reducing the risk of interference or contamination from extraneous software.

The setup of the virtual machine environment adheres to best practices in malware analysis, providing a secure and controlled platform for conducting the subsequent analysis phases. This controlled environment helps safeguard the integrity of the host system while allowing for in-depth examination of the malware sample's behavior and characteristics.

Static Analysis

Static analysis of the malware sample "Sample_MD5_Hash" was conducted to gather insights into its structure, characteristics, and potential functionality without executing the code. This section presents key findings from the static analysis.

File Identification
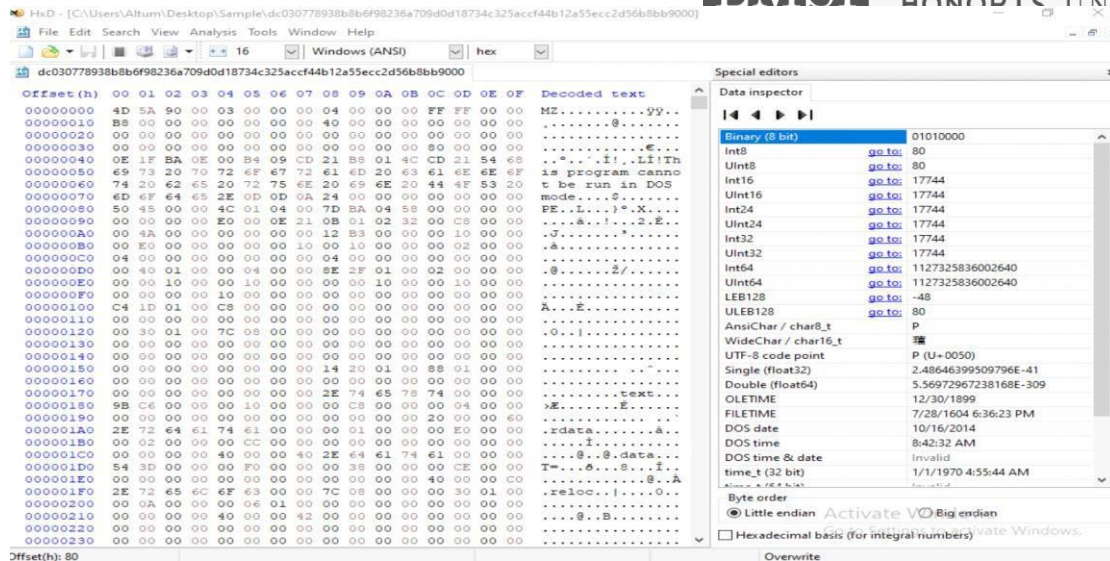
Upon opening the malware sample in a hex editor, the initial bytes of the file header were examined. The first two bytes "4D 5A" were identified, denoting the "MZ" signature, indicating that it is a DOS executable file. Further analysis revealed the presence of "50 45," which marks the start of the PE (Portable Executable) header, confirming the file's status as a Portable Executable.

pestudio Analysis
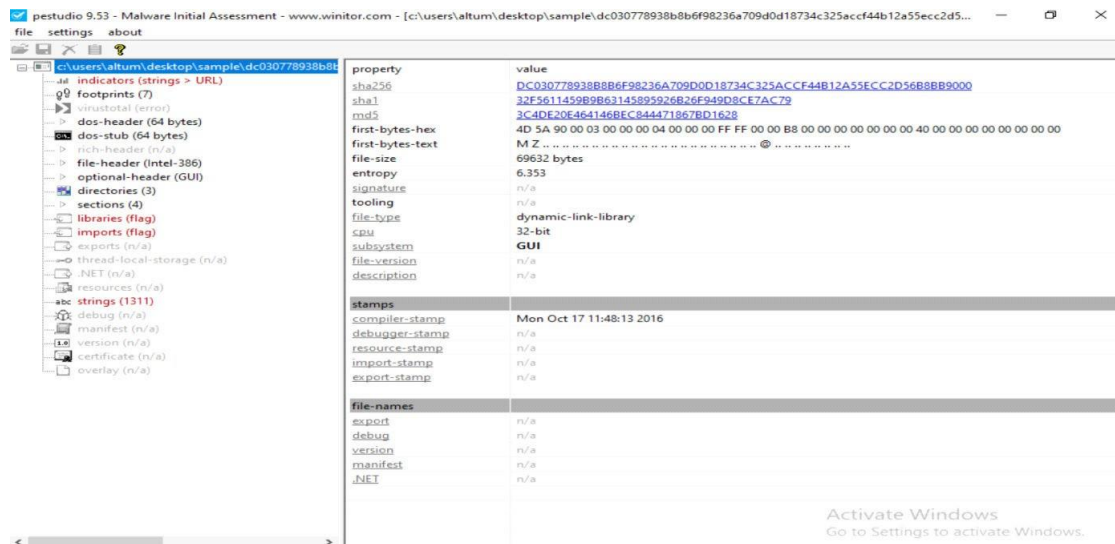
Using pestudio, additional information about the malware sample was gathered. Key findings include:

Compiler Timestamp: The malware sample contained a compiler timestamp dating to October 17, 2016.

File Type: The analysis indicated that the file is a 32-bit DLL using a GUI subsystem.

Entropy Indicator: The malware exhibited a high entropy indicator of 6.353, suggesting potential packing, encryption, or compression.

Hashing

The malware sample was hashed using SHA-256, and the hash value was checked in VirusTotal. It was found that the sample had been previously analyzed on VirusTotal, which was expected due to its prevalence.

Strings Analysis

Strings were extracted from the malware sample, and garbage strings were filtered out. Notable findings from the strings analysis include:

Identification of three URLs that appear to belong to Russian servers, indicating potential communication points for the malware.

Discovery of a registry key associated with the malware's self-uninstallation functionality.

Identification of functionalities such as enumerating users and credentials, interacting with Windows components, and making post requests, suggesting communication with an attacker.

Focus on Internet settings and potential theft of software credentials.

Presence of a URL related to Facebook, along with a corresponding key in the Unicode strings section.

Detailed String Analysis with pestudio

pestudio provided a detailed analysis of the identified strings, including information on the services they invoke and registries they interact with. In total, 63 strings were flagged as suspicious.

exeinfo PE Analysis

Analysis using exeinfo PE indicated that the file was not packed.

Library and Imports Analysis

pestudio 9.53 - Malware Initial Assessment - www.winitor.com - [c:\users\altum\desktop\sample\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d5...
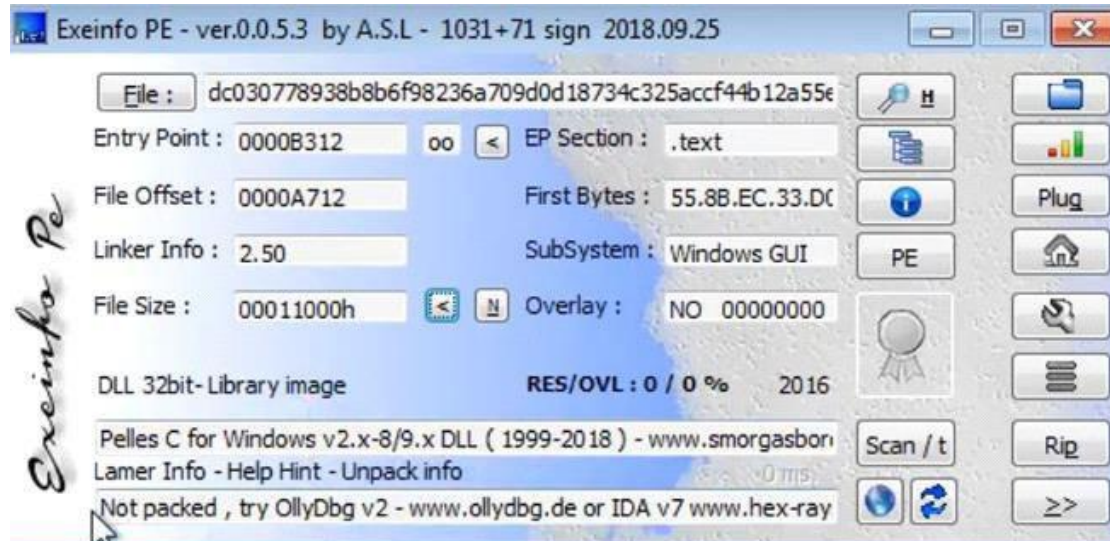
file   settings   about

c:\users\altum\desktop\sample\dc030778938b8t
- indicators (strings > URL)
- footprints (7)
- virustotal (error)
- dos-header (64 bytes)
- dos-stub (64 bytes)
- rich-header (n/a)
- file-header (Intel-386)
- optional-header (GUI)
- directories (3)
- sections (4)
- libraries (flag)
- imports (flag)
- exports (n/a)
- thread-local-storage (n/a)
- .NET (n/a)
- resources (n/a)
- strings (1311)
- debug (n/a)
- manifest (n/a)
- version (n/a)
- certificate (n/a)
- overlay (n/a)

property	value	value	value
headers	header[0]	header[1]	header[2]
name	.text	.rdata	.data
md5	0475EEBBDF0316A70495FCA...	62B50172BB46AC5AD3E41D...	0A81FF1A5E311E7608C826B...
entropy	6.173	3.048	5.391
file-ratio (98.53%)	73.53 %	0.74 %	20.59 %
raw-address	0x00000400	0x0000CC00	0x0000CE00
raw-size (68608 bytes)	0x0000C800 (51200 bytes)	0x00000200 (512 bytes)	0x00003800 (14336 bytes)
virtual-address	0x00001000	0x0000E000	0x0000F000
virtual-size (68971 bytes)	0x0000C69B (50843 bytes)	0x00000100 (256 bytes)	0x00003D54 (15700 bytes)
characteristics			
value	0x60000020	0x40000040	0xC0000040
writable	-	-	×
executable	×	-	-
shareable	-	-	-
self-modifying	-	-	-
virtualized	-	-	-
items			
directory > import	-	-	0x00011DC4
directory > relocation	-	-	
directory > import-address	-	-	0x00012014
optional-header > base-of-code	0x00001000	-	
optional-header > base-of-data	-	0x0000E000	
optional-header > entry-point	0x0000B312	-	

Activate Windows
Go to Settings to activate Windows.

sha256: DC030778938B8B6F98236A709D0D18734C325ACCF44B12A55ECC2D56B8BB9000   Mail   32-bit   file-type: dynamic-link-library   subsystem: GUI   entr

---

pestudio 9.53 - Malware Initial Assessment - www.winitor.com - [c:\users\altum\desktop\sample\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d5...

file   settings   about

c:\users\altum\desktop\sample\dc030778938b8t
- indicators (strings > URL)
- footprints (7)
- virustotal (error)
- dos-header (64 bytes)
- dos-stub (64 bytes)
- rich-header (n/a)
- file-header (Intel-386)
- optional-header (GUI)
- directories (3)
- sections (4)
- libraries (flag)
- imports (flag)
- exports (n/a)
- thread-local-storage (n/a)
- .NET (n/a)
- resources (n/a)
- strings (1311)
- debug (n/a)
- manifest (n/a)
- version (n/a)
- certificate (n/a)
- overlay (n/a)

value	value	value	value
header[0]	header[1]	header[2]	header[3]
.text	.rdata	.data	.reloc
0475EEBBDF0316A70495FCA...	62B50172BB46AC5AD3E41D...	0A81FF1A5E311E7608C826B...	B152A3B5C3C8C66494DCD0...
6.173	3.048	5.391	6.118
73.53 %	0.74 %	20.59 %	3.68 %
0x00000400	0x0000CC00	0x0000CE00	0x00010600
0x0000C800 (51200 bytes)	0x00000200 (512 bytes)	0x00003800 (14336 bytes)	0x00000A00 (2560 bytes)
0x00001000	0x0000E000	0x0000F000	0x00013000
0x0000C69B (50843 bytes)	0x00000100 (256 bytes)	0x00003D54 (15700 bytes)	0x0000087C (2172 bytes)
0x60000020	0x40000040	0xC0000040	0x42000040
-	-	×	-
×	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	0x00011DC4	-
-	-		0x00013000
-	-	0x00012014	
0x00001000	-		
-	0x0000E000		
0x0000B312	-		

Activate Windows
Go to Settings to activate Windows.

---

pestudio 9.53 - Malware Initial Assessment - www.winitor.com - [c:\users\altum\desktop\sample\dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d5...

file   settings   about

c:\users\altum\desktop\sample\dc030778938b8t
- indicators (strings > URL)
- footprints (7)
- virustotal (error)
- dos-header (64 bytes)
- dos-stub (64 bytes)
- rich-header (n/a)
- file-header (Intel-386)
- optional-header (GUI)
- directories (3)
- sections (4)
- libraries (flag)
- imports (flag)
- exports (n/a)
- thread-local-storage (n/a)
- .NET (n/a)
- resources (n/a)
- strings (1311)
- debug (n/a)
- manifest (n/a)
- version (n/a)
- certificate (n/a)
- overlay (n/a)

library (9)	flag (3)	first-thunk-orig...	first-thunk (IAT)	type (1)	import...	group	description
wsock32.dll	×	0x00011E8C	0x00012014	implicit	10	network	Windows Socket 32-
urlmon.dll	×	0x00011F84	0x0001210C	implicit	1	network	OLE32 Extensions for
wininet.dll	×	0x00011FEC	0x00012174	implicit	2	network	Internet Extensions f
kernel32.dll	-	0x00011EB8	0x00012040	implicit	50	-	Windows NT BASE A
userenv.dll	-	0x00011F8C	0x00012114	implicit	2	-	User Environment Lil
ole32.dll	-	0x00011F98	0x00012120	implicit	6	-	Microsoft OLE for W
user32.dll	-	0x00011FB4	0x0001213C	implicit	1	-	Multi-User Windows
advapi32.dll	-	0x00011FBC	0x00012144	implicit	11	-	Advanced Windows
shlwapi.dll	-	0x00011FF8	0x00012180	implicit	6	-	Shell Light-weight U-

Activate Windows
Go to Settings to activate Windows.

sha256: DC030778938B8B6F98236A709D0D18734C325ACCF44B12A55ECC2D56B8BB9000   cpu: 32-bit   file-type: dynamic-link-library   subsystem: GUI

Notable libraries invoked by the malware include:

advapi32.dll: Suggests modification of Windows registries.

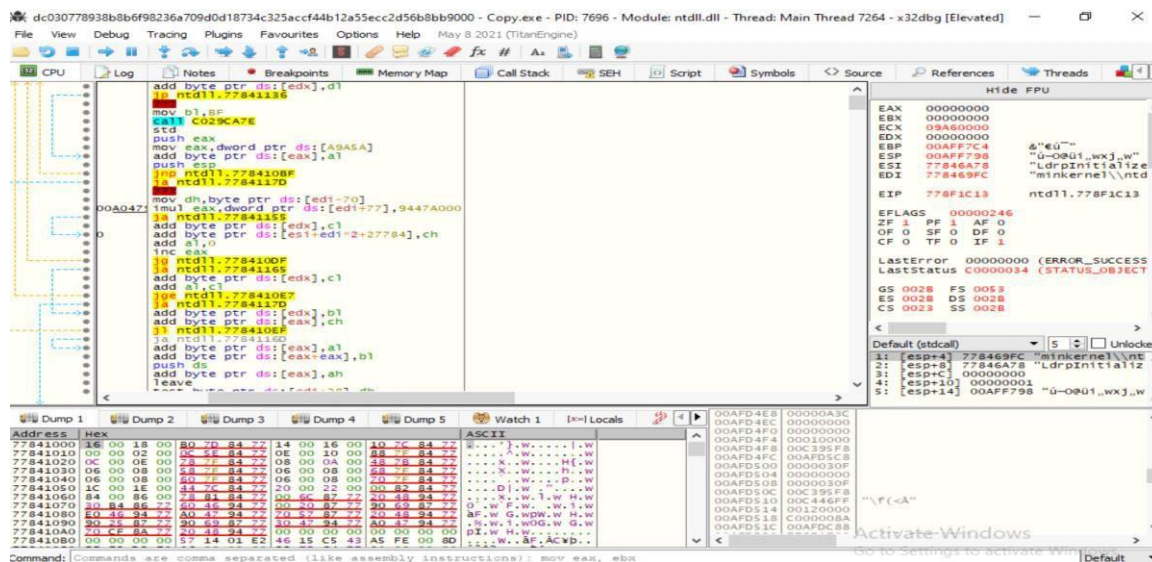wsock32.dll: Indicates interaction with the internet and data transmission.
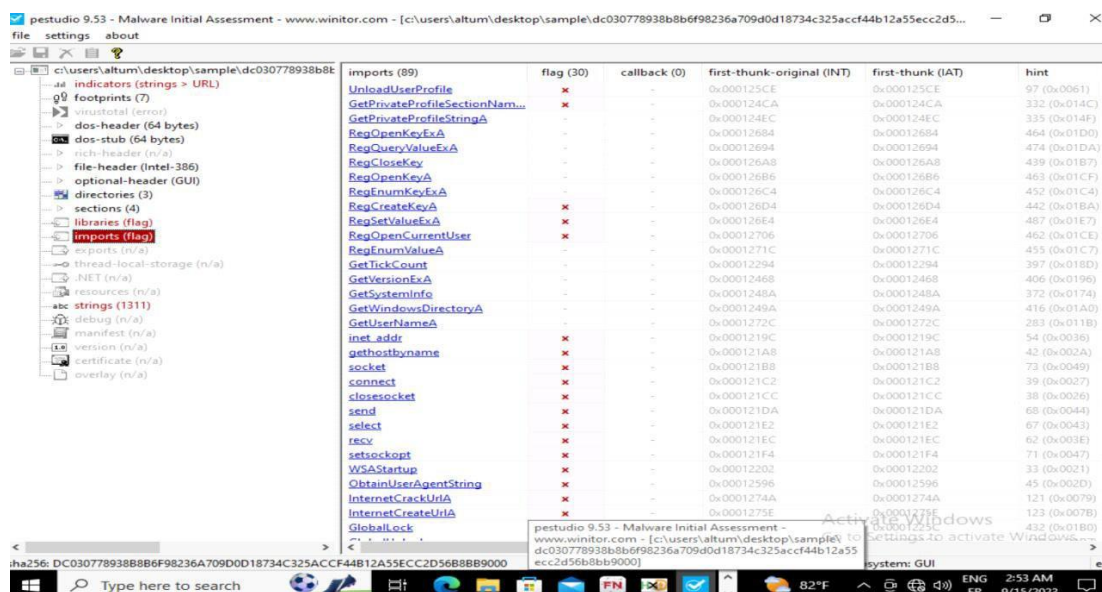
Additionally, a check of imports revealed various functionalities such as creating and deleting files, exiting processes, finding files, getting file attributes, interacting with processes, reading files, connecting to sockets, sending data, parsing URLs, initializing sleep events, and dynamically loading libraries. This dynamic loading capability implies the potential for the malware to call other DLLs for additional functionality, which cannot be extracted statically.

This static analysis provides valuable insights into the structure and potential functionality of the malware, laying the foundation for further analysis stages, including code analysis and dynamic analysis.

13

Dynamic Analysis

Code Analysis

Upon utilizing the x32dbg debugging tool, we immediately noticed a distinct pattern. The DLL in question was accessing a particular function in ntdll, a key system library in Windows. However, rather than referring to a standard function within ntdll, the DLL was using a hexadecimal number as a reference. This hexadecimal number is representative of the specific functions accessed within the ntdll library.
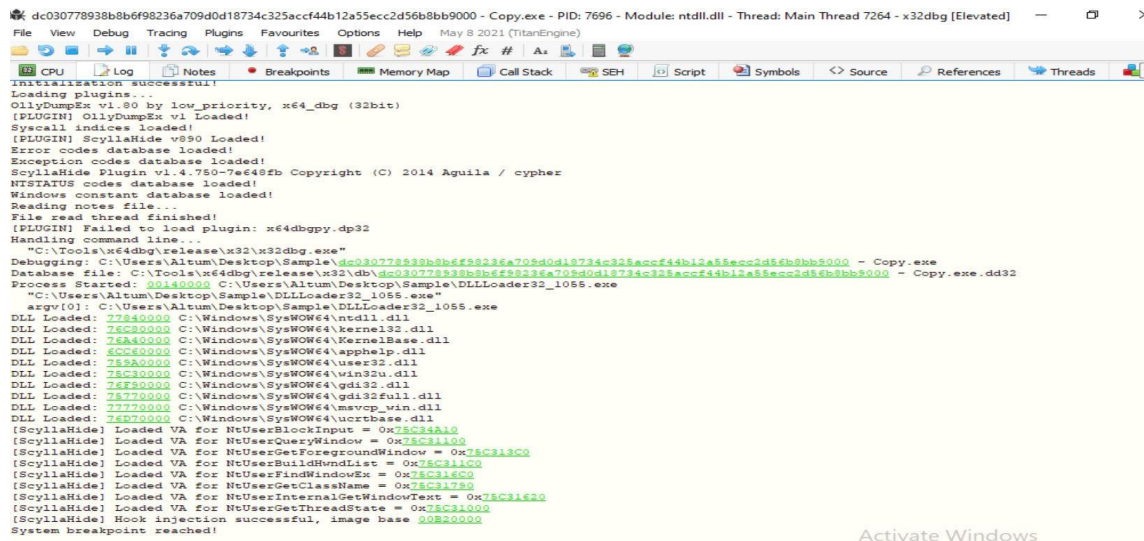
The use of a hexadecimal number rather than a standard function within ntdll is not typical. This suggests that the function is being resolved dynamically at runtime. Dynamic resolution is a technique where the address of a function is determined during the execution of a program, rather than at compile-time. This method is frequently employed when the location of a function is not known in advance, such as when it is contained within a dynamically loaded library.

Another possible explanation for the use of a hexadecimal number instead of a standard function is that it could be an indicator of obfuscation. Obfuscation is a technique used in software protection to make code harder to understand, thereby making it more difficult for unauthorized individuals to reverse engineer or tamper with the software. By using a hexadecimal number instead of a standard function name, the developers could be attempting to disguise the true functionality of the DLL, which can be a

To summarize, the use of a hexadecimal number as a reference to a function within the ntdll library, as observed when using x32dbg, could be indicative of either dynamic resolution at runtime or a potential obfuscation technique. Both possibilities provide insight into the underlying mechanisms of the DLL and its interaction with the ntdll library.
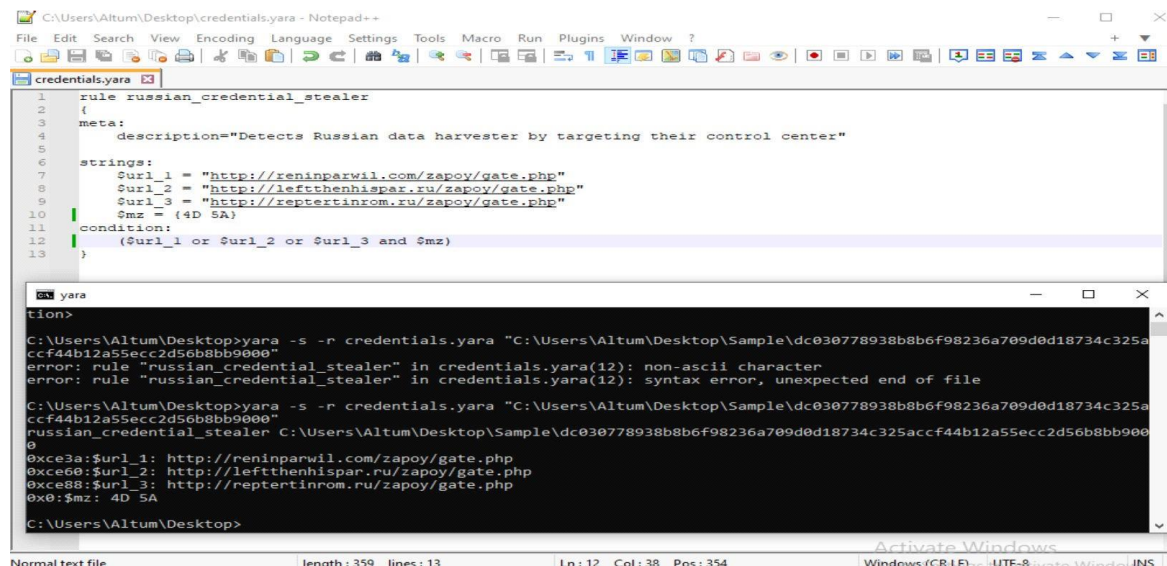
Log Analysis

Upon analyzing the log provided, we can gain further clarity on the issue at hand and uncover the functions being accessed. The log reveals that the DLL in question begins with the function "NtUserBlockInput." This particular function is utilized to block all mouse and keyboard events. It is commonly employed to disable debuggers by preventing user input, making it challenging for the debugger to interact with the target process.

The malware detected the use of the debugger and implemented a technique known as hook injection. Hook injection involves adding a breakpoint each time the x32dbg debugger attempts to execute the "NtUserBlockInput" function. By doing so, the malware effectively obfuscates its behavior and hinders the debugger's ability to analyze the malware's actions .

In summary, the presence of the "NtUserBlockInput" function and the subsequent hook injection strategy used by the malware indicate its awareness of the debugger's presence and its deliberate attempts to obfuscate its behavior. This highlights the sophistication of the malware and the measures it takes to evade detection and analysis.

Detection

While we could rely on the file hashes to detect the presence of malwares and while this is a legitimate way to keep track of the file despite changes in the name or extention scheme. There is a flaw in this approach, The attacker need only change a few lines of code, most likely the garbage strings they add to change the file hash signature.

To combat that and ensure detection we use the tool Yara to check any file using a preset of rule we already put in place based on our previous analysis. In the case the condition for detection is the file type being a portable executable and having access to one of the three control centers specified.

Given that the core functionality of the credential stealer is to send personal informations to their command center. This rule ensure dynamic detection of the malware.

Conclusion

Summary of Findings

The analysis of the malware sample, identified as "Sample_MD5_Hash," has yielded critical insights into its characteristics and behavior. This conclusion summarizes the key findings of the analysis:

Malware Classification

Based on the static analysis and behavior observed during this analysis, "Sample_MD5_Hash" has been classified as a credential stealer. This type of malware is specifically designed to compromise and exfiltrate sensitive user information, including login credentials, from infected systems.

Static Analysis

Static analysis of the malware provided essential information about its structure and potential functionality. The presence of the "MZ" signature, confirmation of it being a Portable Executable (PE) file, and the compilation timestamp dating to October 17, 2016, were noteworthy findings. The malware operates as a 32-bit DLL within a GUI subsystem.

Strings Analysis

The analysis of extracted strings revealed clear indications of credential-stealing capabilities. Notably, the malware was observed interacting with Russian servers via URLs, suggesting communication with malicious entities. Additionally, a registry key associated with self-uninstallation was identified, indicating one of its functionalities.

Library and Imports Analysis

The examination of invoked libraries and imports pointed to various actions, including the modification of Windows registries (advapi32.dll) and interaction with the internet and data transmission (wsock32.dll). Further analysis revealed a range of functionalities, such as file manipulation, process management, and socket communication.

High Entropy Indicator

The malware exhibited a high entropy indicator (6.353), suggesting the potential use of packing, encryption, or compression techniques to obfuscate its code, a common strategy employed by malware authors.

Obfruscation Indicator

The Dynamic Analysis exposes the attempts at obfruscating it's behavior, while the malware managed to detect the vm enviroment despite using tools like fakenet to create a simulated network, It nonetheless noticed the debbuger and used a hook injection upon interception to ensure it could sabotage any analysis attempts, while that might hinder our behavioral analysis. It's strings, most notably the

registeries it accesses and the link to their control center exposes the malware behavior. Further analysis is simply not needed.

Confirmation of Credential Stealer

The culmination of findings from static analysis, string extraction, and library analysis strongly supports the conclusion that "Sample_MD5_Hash" is indeed a credential stealer with the capacity to compromise and extract sensitive user data.

Recommendations and Mitigations

Given the confirmed credential-stealing nature of the malware, it is imperative to take immediate actions to mitigate its impact and prevent future infections:

- **Endpoint Security:** Deploy robust endpoint security solutions, including up-to-date antivirus and anti-malware software, to detect and quarantine credential-stealing malware.

- **User Education:** Educate users about the dangers of downloading and executing unknown files and the importance of maintaining strong, unique passwords.

- **Network Monitoring:** Implement network traffic monitoring and intrusion detection systems to identify suspicious activities, such as data exfiltration attempts.

- **Regular Updates:** Keep operating systems, software, and security solutions up-to-date to patch vulnerabilities frequently targeted by malware.

- **Email Filtering:** Utilize email filtering systems to detect and block phishing emails that often deliver credential-stealing malware.

- **Incident Response Plan:** Develop and regularly update an incident response plan to swiftly respond to and contain security incidents involving credential theft.

- **Data Encryption:** Encourage the use of encryption for sensitive data to protect it from being easily stolen in case of a breach.

Lessons Learned

This analysis underscores the importance of proactive cybersecurity measures and continuous vigilance in defending against credential-stealing malware. By staying informed about evolving threats and adopting best practices in cybersecurity, organizations and individuals can better safeguard their sensitive information and digital assets.

The insights gained from this analysis contribute to the ongoing effort to combat malware threats and enhance cybersecurity practices, reinforcing the need for a multi-layered defense strategy in an ever-evolving threat landscape.

This conclusion section summarizes the findings, offers recommendations for mitigation, and highlights the importance of cybersecurity measures in defending against credential-stealing malware.

References

FlareVM, FakeNet, Pestudio,Hexeditor,CFF explore,Strings,peid.delphi,AnyRun,x32dbg,yara.

https://docs.bmc.com/docs/bcm122/microsoft-windows-registry-642931495.html

https://www.virustotal.com/gui/file/dc030778938b8b6f98236a709d0d18734c325accf44b12a55ecc2d56b8bb9000/details

https://en.wikipedia.org/wiki/Windows_Registry