

School of Design and Informatics

BSc Ethical Hacking, 2022/23

**An Evaluation of Modular Incident Response Plans for Efficient Cyber Incident Mitigation in Businesses**

**Malware Analysis Methodology**

**Author:** Martin Georgiev **Supervisor**: Natalie Coull

Head of Division of Cybersecurity Abertay

Contents

[Terminology and Abbreviations ii](#_Toc134475884)

[1 Introduction 1](#_Toc134475885)

[**1.1** **Background** 1](#_Toc134475886)

[**1.2** **Aims** 1](#_Toc134475887)

[2 Methodology 2](#_Toc134475888)

[**2.1** **Testing Environment** 2](#_Toc134475889)

[**2.1.1** **Virtual Machine Set-up** 2](#_Toc134475890)

[**2.1.2** **Network Setup** 2](#_Toc134475891)

[**2.2** **Static Analysis** 8](#_Toc134475892)

[**2.2.1** **Hash Identification and VirusTotal** 8](#_Toc134475893)

[**2.2.2** **String Extraction** 9](#_Toc134475894)

[**2.2.3** **In-Depth Analysis and Packer Identification** 9](#_Toc134475895)

[**2.3** **Dynamic Analysis** 15](#_Toc134475896)

[**2.3.1** **Detonation Symptoms and Conditions** 15](#_Toc134475897)

[**2.3.2** **Network-Level Analysis** 16](#_Toc134475898)

[**2.3.3** **Local-Level Analysis** 18](#_Toc134475899)

[**2.3.4** **RAM Analysis** 22](#_Toc134475900)

[**2.4** **Yara Rule Creation** 27](#_Toc134475901)

[3 Results 29](#_Toc134475902)

[4 Discussion 30](#_Toc134475903)

[**4.1.1** **General Discussion** 30](#_Toc134475904)

[**4.1.2** **Future Work** 30](#_Toc134475905)

[5 Conclusion 31](#_Toc134475906)

[6 References 32](#_Toc134475907)

# **Terminology and Abbreviations**

Malware – Malicious software used to attack computers, cause damage, steal data, etc.

CMD – Command Line, a window used to execute commands

Hash – An encrypted signature of a file showing the amount of data. It is different for each file, even if they are the same size.

BIOS – Basic Input-Output System, a system used by Operating Systems (Linux, Windows, MacOS) to communicate with the physical parts of the computer.

VirtualBox – Virtualising software allows a user to run multiple virtual computers within the physical one.

FlareVM – A version of Windows 10 made for malware analysis.

Remnux – A Linux operating system for malware analysis.

String – A set of characters, usually human-readable data.

VirusTotal – An online service combining multiple Anti-Virus solutions.

File Packer – Tools used to compress files. They may be used for legitimate reasons to save space or for malicious reasons to bypass Anti-Virus tools.

Floss/strings – Tools used to extract strings from the malware samples.

PEStudio – A tool that automates the static analysis of malware.

Dos-header – The start of a file, may reveal crucial information about the packer and what type of file it is.

Sections – The different parts of the file.

Blacklisted strings/functions/libraries – They are often used in malicious applications but may also be used in benign software.

Libraries – Special parts of applications that contain functions created by a user.

Network adapter – Device used for internet connectivity.

Terminal – Linux alternative of the CMD.

Sudo – Allows a user to execute commands as root.

Root – Highest privilege account in the Linux hierarchy.

DNS – Domain Name System, a protocol used to turn IP addresses into names that are easier to remember

IP address – Address of a machine within a network.

HTTP – Hyper-text transfer protocol, a protocol used for web traffic.

Registries – Collections of information within the Windows operating system.

RAM – Random Access Memory, memory used to store data for current processes.

Packet sniffer – A tool used to monitor network traffic.

Network packets – Pieces of data sent through the network and used for communication.

Wireshark – An industry-standard packet analyser.

Inetsim – A tool used to simulate an internet connection, DNS server and DHCP server.

DHCP Server – Protocol used to dynamically configure IP addresses of new machines. It automates the process and allows users to easily connect and disconnect devices.

Plugin – Additional parts of software that are not essential for its normal workflow. They usually enhance the current capabilities or add new functions.

C&C server – Command-and-Control server used to command the malware, store stolen data, or provide it with more features.

Encryption – Concealing information by making it appear as random data. It can be decrypted using specific keys.

Obfuscation – Making data appear unclear and difficult to understand.

# **Introduction**

## **Background**

Malware analysis and digital forensics are sciences dealing with the investigation of cybercrime and weaponised software. Analysts utilise advanced techniques to understand the scope, timeline and damages caused to the organisation during and after an incident. Such techniques will be difficult to understand or even replicate by people with little to no in-depth IT experience.

## **Aims**

The methodology aims to provide business owners with a more accessible approach to identifying and understanding how a malware infection would affect their infrastructure. The methodology will achieve this by providing the users with malware analysis in two phases, employing extensive examples and steps to follow. The analysis will be split into the following phases:

* Static Analysis – obtaining information about a malware’s capabilities without running it.
* Dynamic analysis – running the malware and following its behaviour on a local or network level.

The methodology will also cover how the users could create their testing environment to conduct the investigations. The malware used in the examples was safely obtained from MalwareBazaar (AbuseCH, 2020) and TheZoo (ytsif, 2014). In the end, the guide will introduce the users to **Yara** (VirusTotal, 2013) and provide a thorough explanation of what Yara is, its structure and uses.

# **Methodology**

## **Testing Environment**

## **Virtual Machine Set-up**

The testing environment will comprise of two virtual machines connected in a separate network. This can be achieved with the use of virtualising software such as VMWare and VirtualBox. This methodology will use VirtualBox, and the process may vary if the user decides to use a different software.

VirtualBox can be downloaded from the official website. After installing the software, the user needs to reboot the computer and open the BIOS (usually F1, F2 or F10). The button will vary depending on the motherboard provider, as well as the UI and positioning of the options. Please consult the user guide provided by your motherboard manufacturer to find and enable the virtualisation option. Not turning it on would result in an error whenever the user tries to launch the virtual machines.

In terms of the virtual machines, the testing environment requires FlareVM (Mandiant, 2017) and Remnux (Remnux, 2010). For FlareVM the user first needs to obtain a Windows 10 virtual machine and then run the PowerShell script shown within the GitHub page for the software (please check the reference). The script will automatically install the Windows-based testing environment and all required tools. Remnux can be obtained from the official website as a virtual image which can be directly imported into VirtualBox (please check the reference). Machines can be added through “Machine -> Add…” or with Ctrl+A (Figure 2.1)

Graphical user interface, application

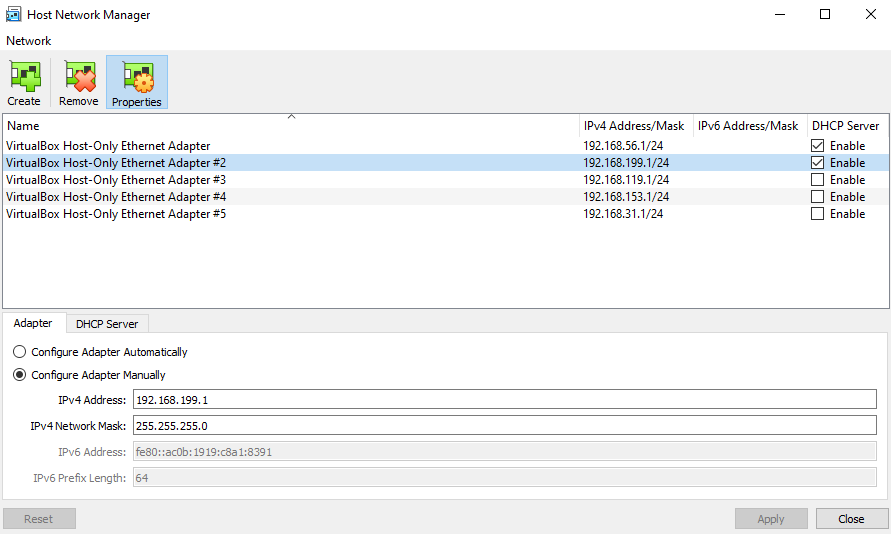
Description automatically generated

***Figure 2.1.1*** *– Adding a machine to VirtualBox.*

Afterwards, the user needs to create the local testing network which can be done from “File -> Host Network Manager…” or with Ctrl+H…. After the network has been created, the user can assign the machines to the network by right-clicking on the machine and then choosing Settings (or Ctrl+S after clicking on the machine) to open the settings. Once in the settings menu, click on the Network tab and choose the newly created network in the “Attached To:” option (the name will vary depending on how you have named it). After this has been completed, the user can proceed with the malware analysis.

## **Network Setup**

To conduct the dynamic analysis, the researchers first need to create a host-only network for FlareVM and Remnux. Such a network would prevent it from accessing physical machines and allow them to communicate. The network can be created in VirtualBox from **File -> Host Network Manager…** (Ctrl+H). Opening the network manager will reveal a window with virtual adapters. (**Figure 2.3.1**)

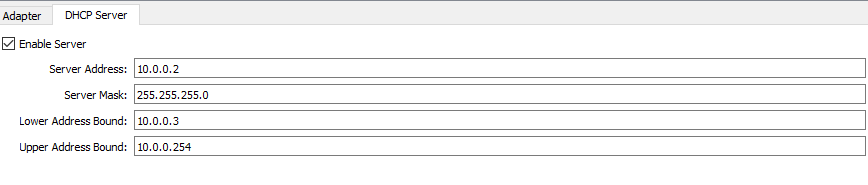


***Figure 2.3.1*** *– Host Network Manager.*

A new adapter can be created from the create button. It should have the following data in the adapter (**Figure 2.3.2**) and DHCP Server (**Figure 2.3.3**) tabs:

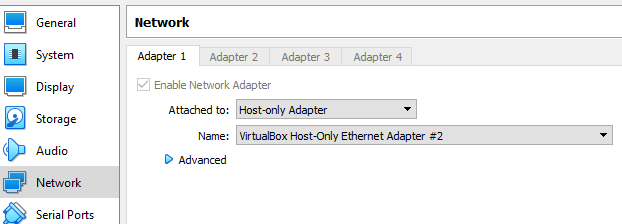


***Figure 2.3.2*** *– Adapter tab.*



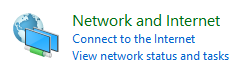
***Figure 2.3.3*** *– DHCP Server tab.*

The analyst should then right-click on the imported FlareVM and Remnux machines and open Settings (the top option). Once the Settings window is opened, open the network tab and change the “Attached to:” setting to “Host-only Adapter” and the “Name:” to the name of the newly created adapter. (**Figure 2.3.4**) This will allow the machines to communicate.



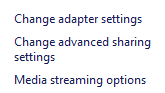
***Figure 2.3.4*** *– Network settings of the machines.*

The final action the analyst needs to take is to launch FlareVM and edit the network adapter’s settings. This can be achieved by opening the Control Panel (press the Windows key and type Control Panel) and then clicking on “View network status and tasks” (**Figure 2.3.5**)



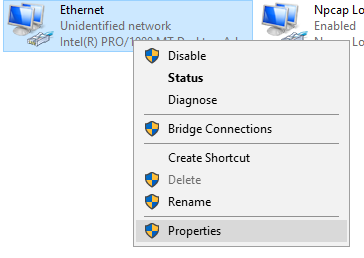
***Figure 2.3.5*** *– Network tab in Control Panel.*

Once the window opens, click on “Change adapter settings” located on the top left. (**Figure 2.3.6**)



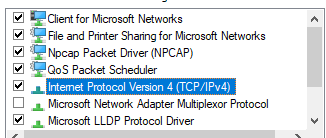
***Figure 2.3.6*** *– Change adapter settings.*

Right-click on the Ethernet adapter and open the properties. (**Figure 2.3.7**)

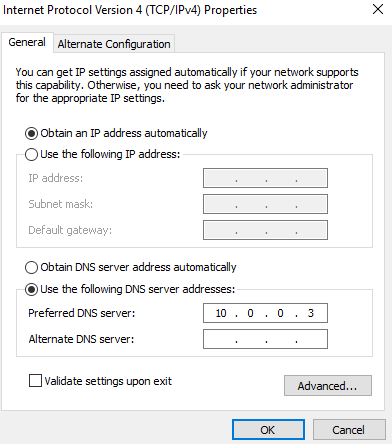


***Figure 2.3.7*** *– Opening properties of the ethernet adapter.*

Afterwards, the analysts need to double-click on the **Internet Protocol Version 4** option and change the preferred DNS address to 10.0.0.3. (**Figure 2.3.8** and **Figure 2.3.9**)

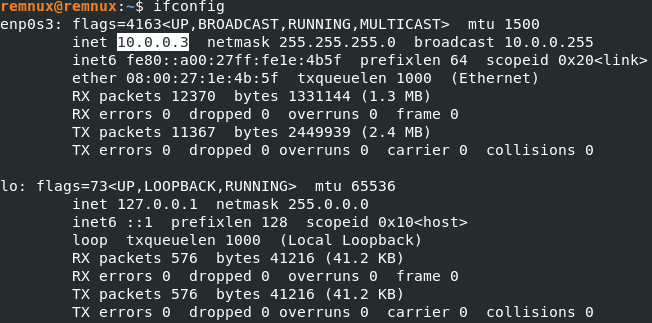


***Figure 2.3.8*** *– IPv4 option.*



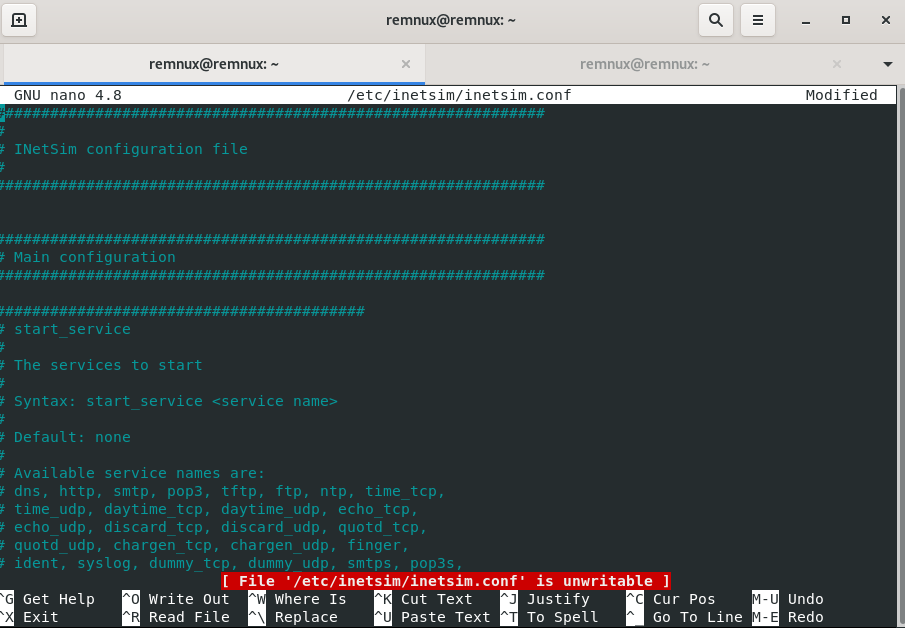
***Figure 2.3.9*** *– Network adapter settings.*

The DNS address may vary, depending on the address of the Remnux machine. Analysts can identify the correct address by launching Remnux, pressing Ctrl+Alt+T to open a terminal window and typing **ifconfig**. This will display the IP address of the machine (**Figure 2.3.10**)



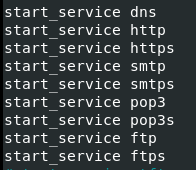
***Figure 2.3.10*** *– Remnux IP Address.*

Finally, the analyst is required to configure Inetsim (Hungenberg and Eckert, 2007). This can be achieved by opening the configuration file and altering a few lines. The config file can be opened inside a terminal window with the following commands: **sudo** **nano /etc/inetsim/inetsim.conf**. Running the command will open the following window: (**Figure 2.3.11**)



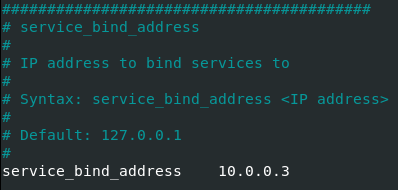
***Figure 2.3.11*** *– Inetsim in Nano.*

Go down using the arrow keys and enable the following services by removing the **#** sign. (**Figure 2.3.12**)

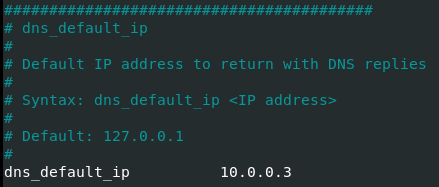


***Figure 2.3.12*** *– Required services.*

Scroll down further and look for the service\_bind\_address and dns\_default\_ip and change it to the machine’s IP (**Figure 2.3.13** and **Figure 2.3.14**).

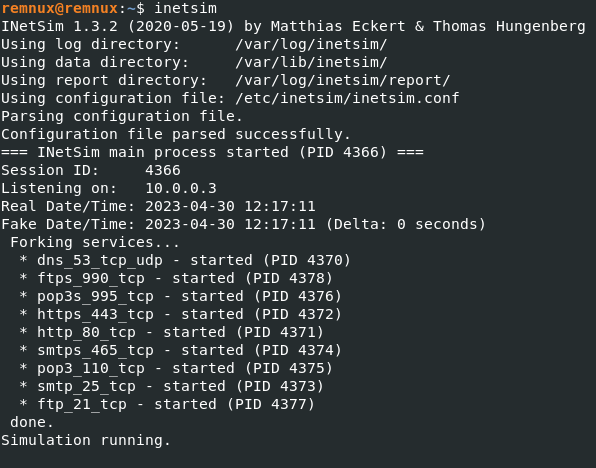


***Figure 2.3.13*** *– Service Bind change.*



***Figure 2.3.14*** *– Default DNS change.*

When done, press Ctrl+S to save and Ctrl+X to close the file. Type in **inetsim** after the file is closed to launch the internet simulator. (**Figure 2.3.15**)



***Figure 2.3.15*** *– Running Inetsim*

## **Static Analysis**

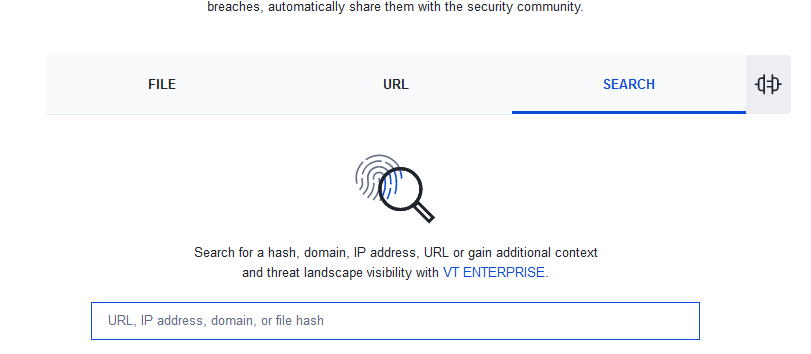
## **Hash Identification and VirusTotal**

Hashes are mathematical functions that convert data into an encrypted output of fixed length. Hashes can be a vital part of a malware analysis as it alone could show possible signs of maliciousness. The hash of a file can be extracted in multiple ways.

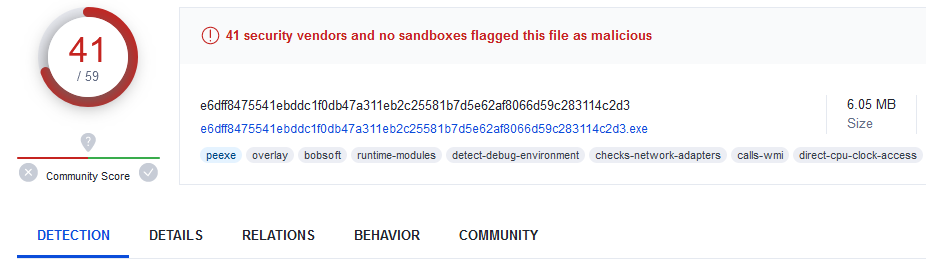
The first way someone can obtain the hash of a file is through the **sha256sum.exe** and the **md5sum.exe** applications within FlareVM. To use them, the user needs to open a command window (Right Click within the malware’s folder and click on “Open command window here”). In the command window, the user can type “**sha256sum.exe [file\_name]**” to obtain the SHA256 hash and “**md5sum.exe [file\_name]**” to obtain the MD5 hash.

Additionally, a user can obtain the hashes using PEStudio – a tool used in [**Section 2.2.3 In-Depth Analysis and Packer Identification**](#_In-Depth_Analysis_and). The tool can be launched by pressing the Windows button and then typing PEStudio within the search field. The malware can then be loaded into the tool through the File menu (Open File…) or with drag and drop (left click on the malware and hold -> drag to the tool -> release left click). The MD5 and SHA256 hashes will be displayed after the tool analyses the malware sample.

Users can then utilise the obtained hashes in tools such as VirusTotal (VirusTotal, 2004 – Present Day). VirusTotal allows a researcher to look for specific file hashes or even upload suspicious files. The service then uses multiple anti-virus vendors, scans them, and shows the results. While this may be useful, vendors may sometimes show false results or not mark it as malicious if the malware was just released in the wild. To use the service, users need to open its web page (check references), click on the “Search” tab, and paste the obtained file hash. (**Figure 2.2.1**) After the scanning is complete, the analyst will be provided with a “Malicious” (**Figure 2.2.2**) or “Non-Malicious”/”No Matches Found” result.



***Figure 2.2.1*** *– Search bar for file hashes.*

******

***Figure 2.2.2*** *– Malicious result (FileTour hash)*

## **String Extraction**

Strings are an object used to store human-readable text. They can be extracted by utilising multiple tools and techniques. This methodology suggests three easy-to-use tools – Strings (Russinovich, 2021), Floss (Ballenthin, 2016), and PEStudio (Fox, 2021).

The first tool, Strings, has its capabilities in its name – it scans a file and looks for human-readable data. The tool is easy to use and can be accessed through the command line. Users can open it by right-clicking in the sample’s directory/location (the folder where it is stored or even the Desktop) and choosing “Open CMD window here” (figure xxx). Strings have the following syntax – **strings [file\_name]**. This will run the software and display any identified strings in the command line window. Users can also employ the following command (**strings [file\_name] > [output\_name].txt**) to write the output in a text file. The text file can then be opened for easier reading/filtering (looking for specific words with **Ctrl+F**, etc.)

Floss works similarly to strings but has additional capabilities. The tool attempts to identify any obfuscated/encrypted strings and then tries to make them readable. The tool can be especially useful as a lot of modern malware is heavily obfuscated. Floss can be used in the same way as **Strings** through a command line window. The tool has the following syntax – **floss [file\_name] > [output\_name].txt**. Floss has built-in filtering that users can employ for more accurate results. An example is the **-n** flag that allows a user to specify the minimum string length (i.e., **floss -n 8 [file\_name] > [output\_name].txt** would write any strings that are 8 characters or longer to a text file). (**Figure 2.2.3**)



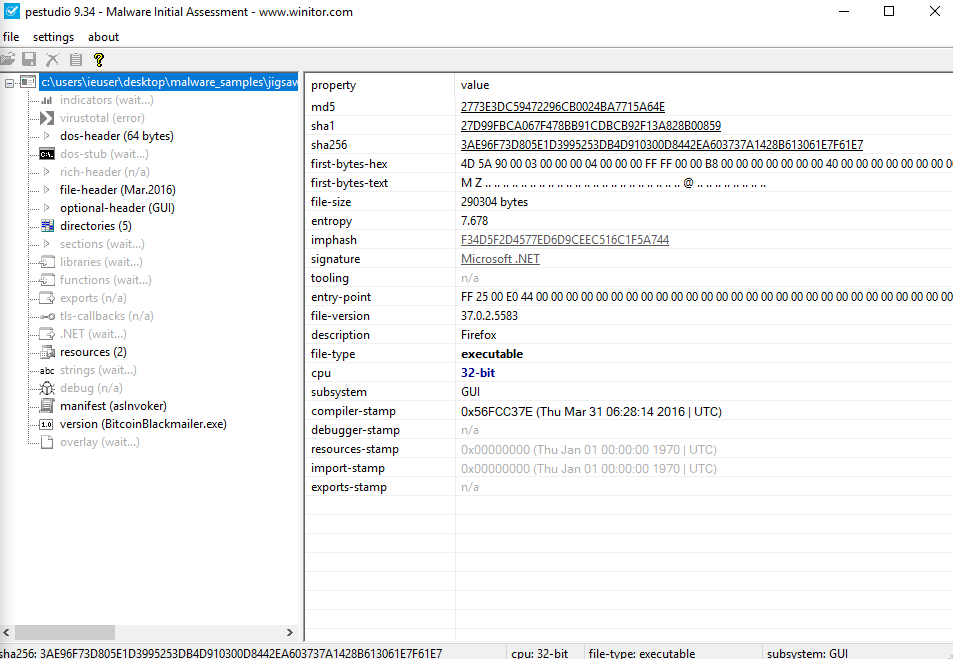
***Figure 2.2.3*** *– Extracting strings with floss.*

PEStudio was already mentioned in the previous section. PEStudio can be especially useful for people who are just starting with malware analysis because it can show various malicious indicators that could point to malicious activity. Users can upload the file through **File -> Open File…** then click on the Strings menu to find any blacklisted strings. ([**Section 2.2.3 In-Depth Analysis and Packer Identification**](#_In-Depth_Analysis_and)**)**

## **In-Depth Analysis and Packer Identification**

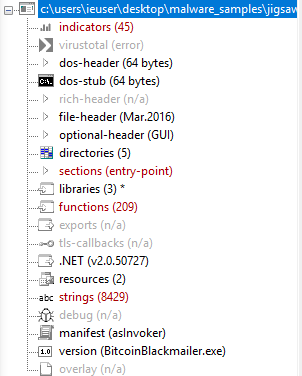
The in-depth analysis and packer identification will be achieved with two separate tools – PEStudio and ExeInfoPE.

As mentioned, PEStudio is a powerful tool that automates most static analyses. Opening a file within the tool automatically starts the analysis process. Samples can be loaded into PEStudio from File -> Open File (Ctrl+O) or by drag-and-drop. Analysts will then see a similar window, indicating that the file is being analysed. (**Figure 2.2.4**)



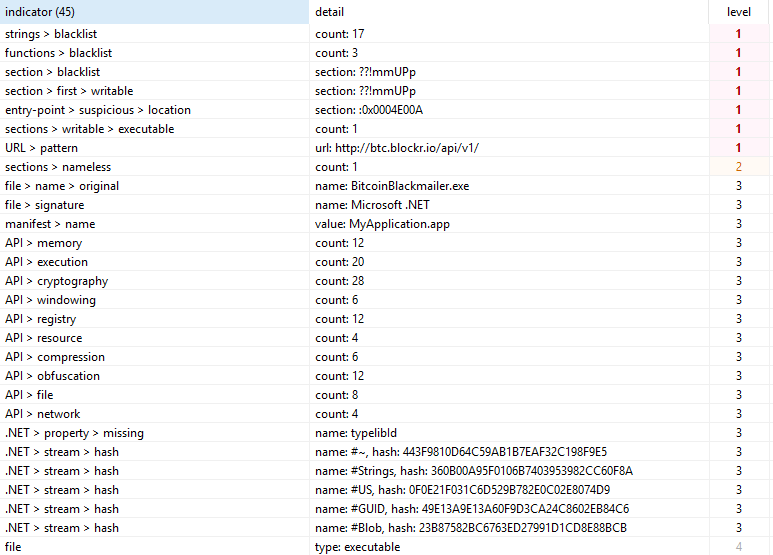
***Figure 2.2.4*** *– PEStudio analysing the sample.*

After the analysis has been concluded, some tabs will change depending on whether malicious indicators were identified or not. Those tabs can be recognised by their distinct red font. (**Figure** **2.2.5**) In this example, the Indicators, Sections, Functions and Strings tabs display malicious indicators. It should be noted that some non-malicious functions/libraries/strings may be used for offensive purposes and vice versa. It is recommended to always check all tabs to obtain detailed information.



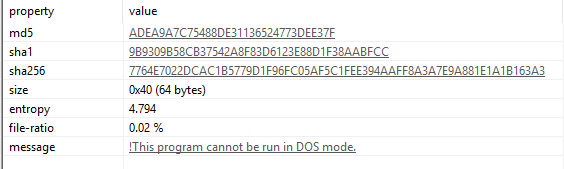
***Figure 2.2.5*** *– Tabs showing malicious indicators.*

Analysts can get an overview of various details from the analysis and their level of severity. The tab can be used as a first insight into the malware’s capabilities. (**Figure 2.2.6**) The severity level can range from 1 (critical) to 4 (possibly benign). PEStudio can also automatically look for the hash in VirusTotal if the machine is provided with an Internet connection (As seen in the VirusTotal tab below the indicators)



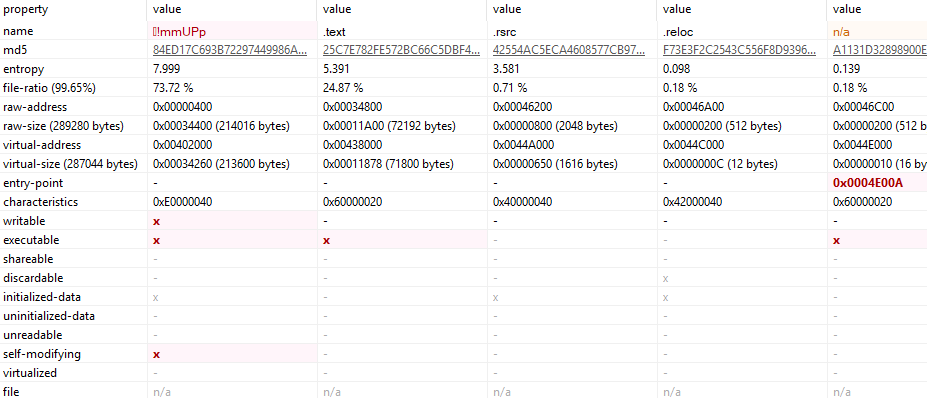
***Figure 2.2.6*** *– Indicators tab example.*

The DOS-Stub tab (**Figure 2.8**) is another useful feature that shows information about the file such as its entropy. Entropy shows the randomness within the data and can indicate obfuscation and encryption. High entropy levels are common for malicious software as they attempt to bypass Anti-Virus detection.



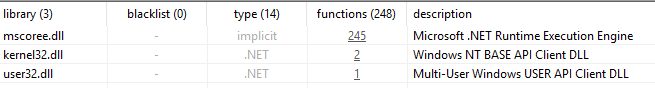
***Figure 2.2.7*** *– DOS-Stub tab.*

The next important tab shows the sections within the file. In this tab, analysts can identify each section and its capabilities. Any writeable, executable, self-modifying and virtualised sections and custom names may indicate malicious behaviour within an application. (**Figure 2.2.8**)

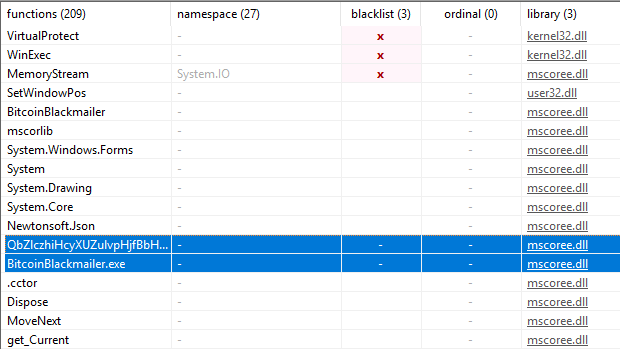


***Figure 2.2.8*** *– Sections tab for Jigsaw.*

The libraries tab (**Figure 2.2.9**) shows what DLL files were utilised for the malware’s capabilities and the functions tab (**Figure 2.2.10**) shows their respective methods. Similarly, to the previous tab, the possibly malicious libraries/functions will be marked with a red X symbol. However, this does not indicate whether it is malicious as it may be used for legitimate purposes (encryption, internet communication, etc. Additionally, some non-marked (unknown or usually benign) functions/libraries may be employed in malicious activities.



***Figure 2.2.9*** *– Libraries tab for Jigsaw.*

**

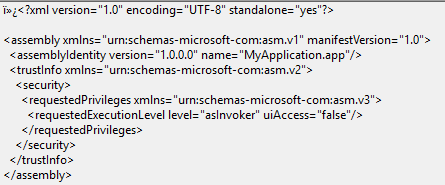
***Figure 2.2.10*** *– Functions tab for Jigsaw with two non-blacklisted malicious functions.*

The strings tab shows human-readable text within the file. Blacklisted values in this tab can be embedded software names, functions commonly used in malicious applications, URLs, and more. It will show similar results to what was discovered in [**Section 2.2.2 String Extraction**](#_String_Extraction). (**Figure 2.2.11**)



***Figure 2.2.11*** *– Strings tab for Jigsaw.*

Finally, the manifest and version tabs can show additional information about its behaviour and other suspicious information. The manifest tab displays the privileges a file requires to execute. While most request the execution level to be “asInvoker” (the same level of privileges as the user who executed it, some actively try to escalate their privileges to an administrator (FileTour (NetEnrich, 2021 and Stamus Labs, 2022), etc.). (**Figure 2.2.12**)



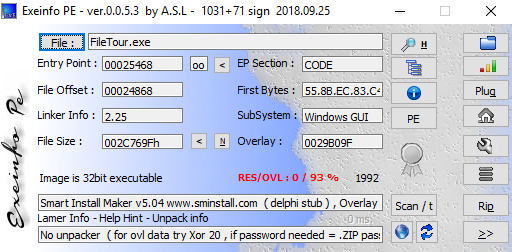
***Figure 2.2.12*** *– Jigsaw Manifest.*

The version can reveal suspicious information about the file such as its original name and descriptions. In Jigsaw’s (Ashdown, 2021) case, the original file name is BitcoinBlackmailer.exe, while the legal copyright and file description are associated with Firefox. (**Figure 2.2.13**)



***Figure 2.2.13*** *– Jigsaw version tab.*

ExeInfoPE (ASL, 2023) is another tool which can display the packer used in potential malware. The application can be launched by pressing the Windows key and typing ExeInfoPE. The packer can be found inside the “**EP Section:**” part of the window and it may serve as useful information about possible obfuscation and encryption in the malware. (**Figure 2.2.14**)



***Figure 2.2.14*** *– ExeInfoPE results for FileTour.*

## **Dynamic Analysis**

## **Detonation Symptoms and Conditions**

The dynamic analysis requires the researcher to execute the malware sample and observe its behaviour. Some samples may require specific conditions to execute. Such conditions can be (but are not limited to):

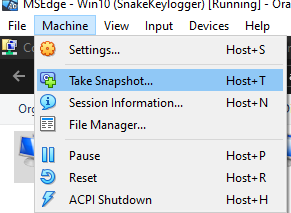
* Administrator privileges
* Lack of Internet connection (WannaCry)
* Need of Internet connection (multi-stage payloads)
* Specific functions (NotPetya)

Once the malware is executed, the analysts will be able to observe various behaviour which would depend on the type of malware (**Table 2.1**).

|  |  |
| --- | --- |
| Type | Functions |
| Ransomware | A type of malware that encrypts files. Some ransomware samples make visible changes by changing the extension of affected files, while others remain “hidden” and encrypt them without altering the extension. Most ransomware also displays a ransom message (with cryptocurrency wallets) that is often accompanied by fearmongering and other social engineering attempts. An example of such malicious software can be Jigsaw. |
| Spyware | Spyware is a type of malware that remains silent on the victim’s machine and collects their data. The information is then sent to the adversary using the software. Spyware can be used in phishing/spearfishing campaigns or to steal user credentials and session tokens. An example of such malware is SnakeKeylogger (Zhang, 2021). |
| Trojan | The Trojan is a reference to the Trojan horse used by the Greeks to infiltrate the city of Troy. Such malware disguises itself as legitimate software. It is often used as a backdoor, allowing attackers to access the machine without the victim’s knowledge. |
| Adware | Adware monitors victims’ activity and generates pop-up ads based on their browser data, which can cause bandwidth issues. It was initially considered spyware and requires identifying its accessed information and usage to effectively differentiate. An example of such malicious software is FileTour. |
| Worm | Worms are one of the oldest types of malware that replicate on other machines, potentially across networks. Nowadays they are often combined with additional capabilities to cause as much damage as possible. Good examples are WannaCry and NotPetya (Ivanov and Mamedov, 2017). |
| Wiper | Wipers are designed to destroy the data on a victim’s system. They may either delete the files or encrypt them without any way to retrieve a decryption key. An example of such malware is NotPetya. |
| Fileless Malware | Fileless Malware is a new strain which utilises functions native to Windows by using Powershell. Such malware is difficult to identify and monitor. Most samples use Windows Office files (Word, Excel, etc) and ask the user to enable the macros. Once they have been enabled, the script will run and infect the machine. An example of fileless malware is Emotet and Duku 2.0 |

***Table 2.1*** *– Types of malware with functionalities.*

It is advised to create snapshots of the virtual machine. Snapshots can be used to revert the system to before it was infected with the malware. This will ensure that the analysts can safely revert any of the damage and continue testing the malware. A snapshot can be taken from the Machine tab (top left of the virtual machine window) and clicking on the “Take Snapshot…” option. (**Figure 2.3.16**)

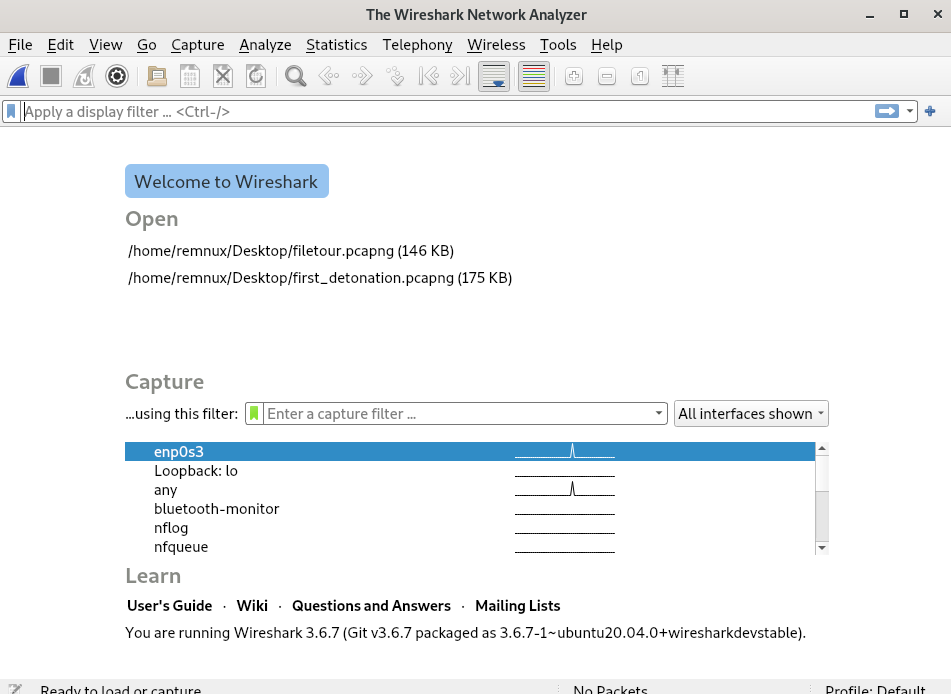


***Figure 2.3.16*** *– Taking a snapshot of the machine.*

## **Network-Level Analysis**

Network-Level analysis can show if malware attempts to access other machines. This could mean downloading multi-stage payloads, attempting to move to other uninfected machines or accessing links to send/receive data.

Analysts will need both machines to monitor the network. FlareVM will be used to detonate the malware, while Remnux will keep track of the activity. Open a terminal window with Ctrl+Alt+T and run Inetsim by typing **Inetsim**. Open a second tab from the **+** sign in the top left corner and type **wireshark** (Wireshark, 1997 – Present Day) to open the packet sniffer. It will then open a new window for the application. Click on the blue shark fin (top left) to start recording the network traffic. (**Figure 2.3.17**)



***Figure 2.3.17*** *– Wireshark.*

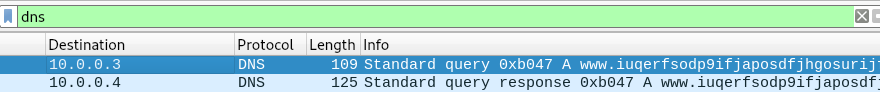
After running the traffic sniffer, the user can then detonate the malware sample. For this example, the analyst will execute WannaCry. Traffic data was immediately captured after launching the malware.

As a lot of packets may appear, the analysts will have to sort the data to obtain useful information. While many types of protocols may be used, the following ones are the most common – DNS, HTTP and TCP. DNS will show any websites the malware attempts to connect to, HTTP will show the web traffic and TCP can show a variety of data. The port filters can be applied from the text bar above the captured traffic (**Figure 2.3.18**)



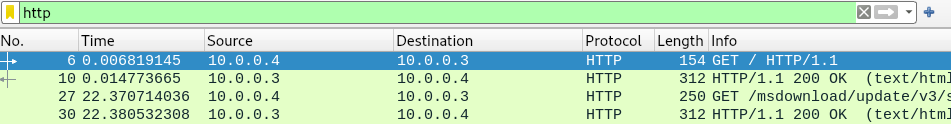
***Figure 2.3.18*** *– Box for filtering data.*

Filtering the DNS packets will show the website the malware tries to connect to. (**Figure 2.3.19**)



***Figure 2.3.19*** *– DNS packets.*

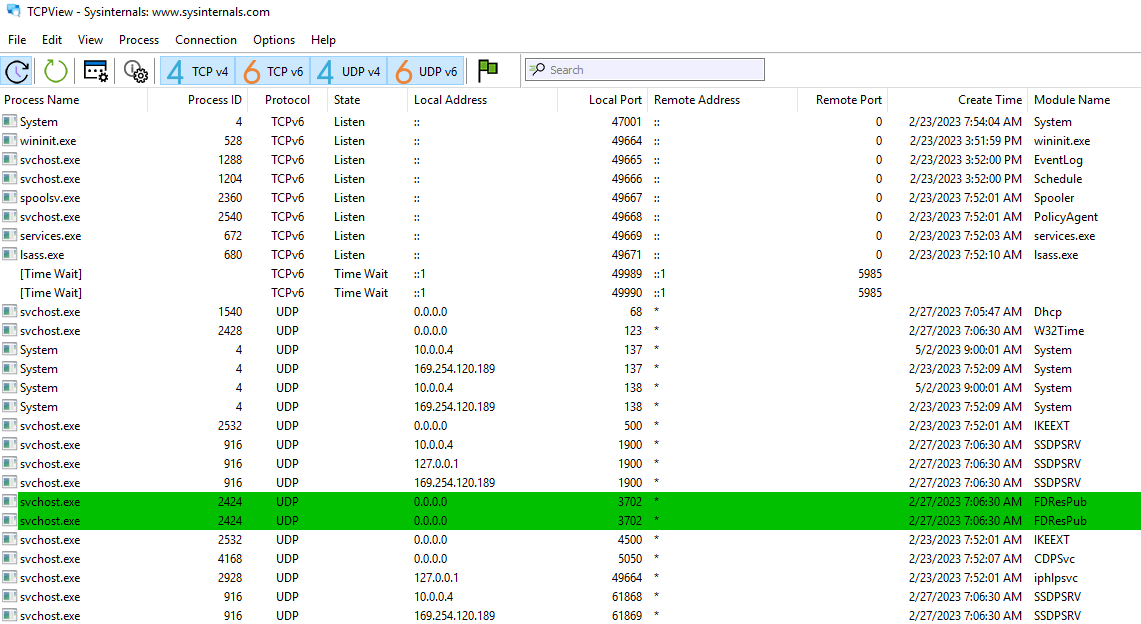
The HTTP packets would show successful/unsuccessful connections, and various requests to obtain the geolocation, IP, etc. (**Figure 2.3.20**) As this link is the malware’s killswitch, a successful connection will fully prevent it from executing.



***Figure 2.3.20*** *– HTTP packets.*

The TCP packets show a more in-depth version of multiple other packets (HTTP, etc) and will be difficult to analyse without previous experience. Such data should be provided to professionals.

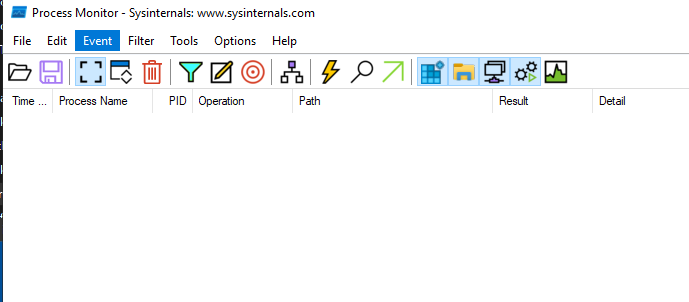
Researchers can also examine the open network connections of the machine by using TCPView (Russinovich, 2022). The tool can be opened by pressing the Windows key and typing TCPView in FlareVM. Launching the tool will reveal a window that monitors any inbound and outbound connections live. Analysts can then look for the name of the detonated malware to find possible results. (**Figure 2.3.21**)



***Figure 2.3.21*** *­– TCPView live connection monitoring.*

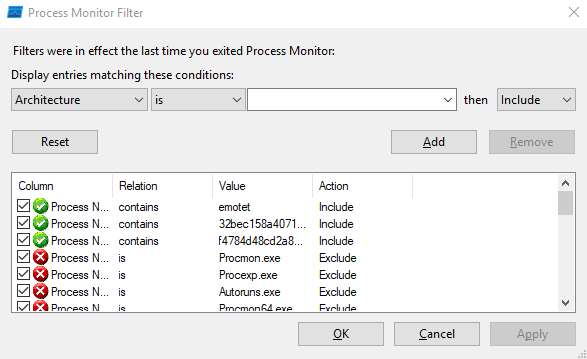
## **Local-Level Analysis**

In this section, analysts will learn about identifying how the malware can affect the system locally. This will be achieved with a tool called Procmon (Russinovich, 2022). To open it, users need to open the FlareVM machine, press the Windows key and type **procmon**. Launching the application will open the following window (**Figure 2.3.22**):



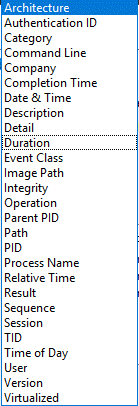
***Figure 2.3.22*** *– Procmon.*

A lot of processes will appear as the tool records every action that was taken by the applications. Analysts can sort the data by pressing on the blue funnel icon under the tool’s menu. Once the menu is opened, the analysts will see the following window (**Figure 2.3.23**):



***Figure 2.3.23*** *– Procmon sorting window.*

At the top, they will be able to provide the tool with specific conditions to appropriately sort the data. Analysts can choose between multiple types (Architecture, Path, Process Name, Date & Time, etc.) (**Figure 2.3.24**), inclusion conditions (is, contains, begins with, less than, etc) and whether the condition should be included or excluded from the results.



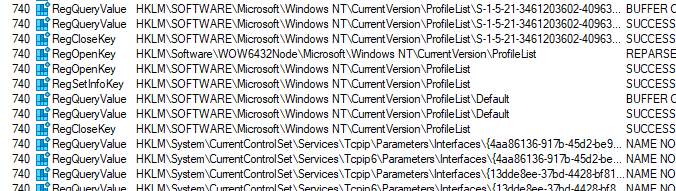
***Figure 2.3.24*** *– Types of filters in procmon.*

To easily identify the malware’s actions, the researcher recommends using the following filters (**Table 2.2**):

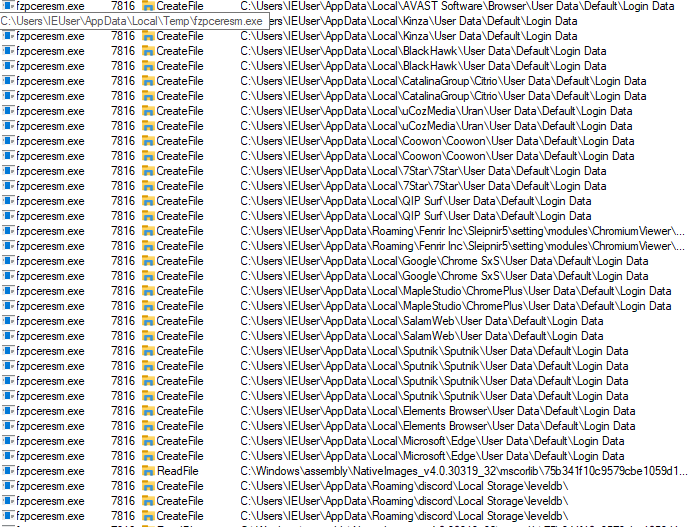
|  |  |
| --- | --- |
| Type | Functions |
| Type Filter | The best way to easily filter out the malware is by using the **Process Name** filter and typing its name within the search bar. This will reveal only actions taken by the sample. Additionally, it can be combined with **Path** to filter specific folders. Examples of such folders are C:\Users\*Username*\AppData\Local, C:\Users\*Username*\AppData\Roaming or any folders found after filtering the process name. The Local folder usually contains browser data within folders with the same name as the browser. The latter can also contain browser data but also information from applications such as Discord. |
| Inclusion Condition | The easiest way to filter the conditions is to use the “contains” filter. This will ensure that the tool will filter out any processes that contain a specific name/word and will not look for a definite string. It can be used when the analyst is not completely certain about the name/string. |
| Search Bar | The search bar allows a user to refine the sorting. Analysts can look for the following:   * CreateFile – Any created files/accessed locations by the malware * CloseFile – Any deleted files * RegQueryKey – Any checked registries (credentials, browser data, profile lists, etc.) * RegOpenKey/RegCloseKey – Any created/removed registries (persistence, causing damage, etc.) |

***Table 2.2*** *– Common filters.*

To show how such results may appear in the tool, the researcher will use screenshots from SnakeKeylogger’s (spyware) analysis as an example. The malware attempted to identify the number of users and their names by examining related registries (**Figure 2.3.25**), then attempted to steal data from Browser folders and Discord – surfing data, login credentials, etc. (**Figure 2.3.26**)



***Figure 2.3.25*** *– Profile list check.*



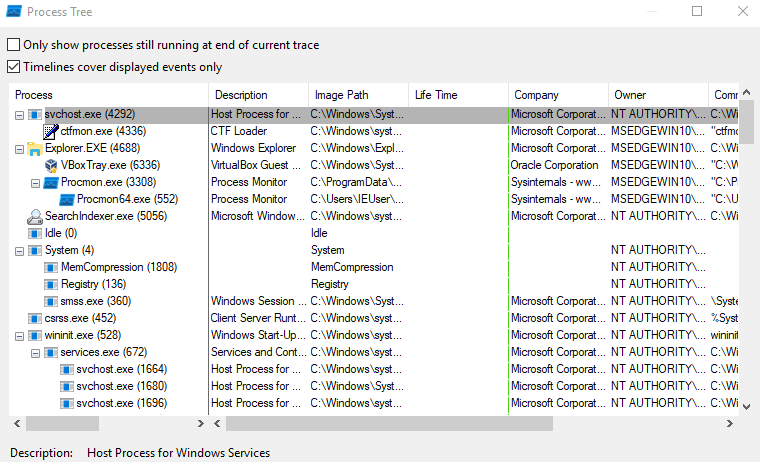
***Figure 2.3.26*** *– Browser and Discord data theft.*

Finally, the tool can also be used to display the process trees. A process tree shows applications with their parent/child processes in a tree-like structure. It can be used to easily identify what other processes were called by the malware and how they can affect the system. The identified process/software can then be used in the sorting tab to refine the research. The tree can be opened by clicking on the following icon (**Figure 2.3.27**):



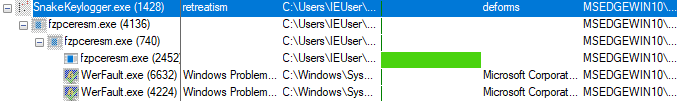
***Figure 2.3.28*** *– Opening the process tree.*

Once the window was opened, the analyst will see the following (**Figure 2.3.29**):



***Figure 2.3.29*** *– Process Tree window.*

Analysts can then scroll through the results and look for the name of the suspected malware. Once again, an example will be given with SnakeKeylogger. The tree showed that the malware called a second-stage payload (fzpcresm.exe). Looking at the lifetime, (column between Image Path and Company) revealed that these processes continued after SnakeKeylogger.exe was executed. This indicated that this file carried out malicious activities. (**Figure 2.44**)



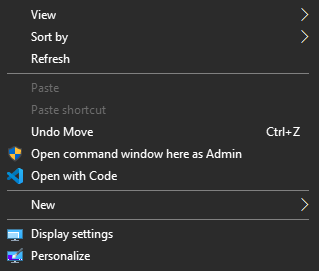
***Figure 2.3.30*** *– SnakeKeylogger activity.*

## **RAM Analysis**

RAM (or Random Access Memory) is a part of the machine which stores data that certain applications require to run. It speeds up the process as the processor has more direct access to it compared to constantly accessing a hard drive. Analysing the RAM could reveal a lot of information about the malware and possible persistence mechanisms.

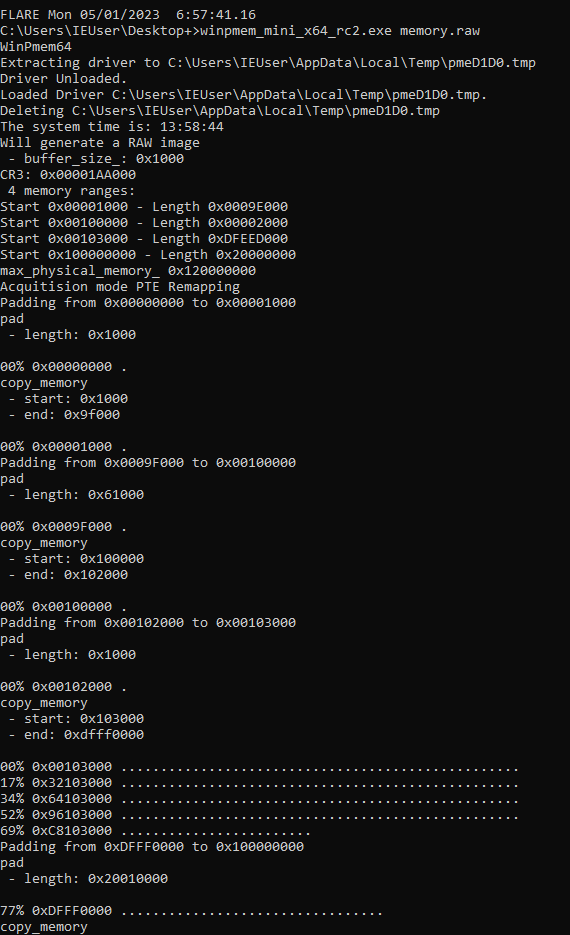
RAM can be analysed with the help of two tools - WinPMem (Cohen, et al., 2019) and Volatility 3.0 (Volatility Foundation, 2020). Both tools can be downloaded from their Github pages (**https://github.com/Velocidex/WinPmem/releases** - **winpmem\_mini\_x64\_rc2.exe** and **https://github.com/volatilityfoundation/volatility3**). The former does not need to be installed. It can be immediately used after the download is complete. The latter, on the other hand, requires installation and an appropriate table of symbols. Its GitHub page contains an in-depth guide on how the tool should be installed.

After installation is complete, the analyst can run the malware. Afterwards, they need to open a command prompt where the tools are located by right-clicking and pressing on the “Open command window here as admin…” (**Figure 2.3.31**)



***Figure 2.3.31*** *– Opening a command window.*

Once the window is opened, the user is required to create a memory dump. Memory dumps contain all data held within RAM. This can be easily achieved with WinPMem by writing the following command – “winpmem\_mini\_x64\_rc2.exe *file\_name*.raw” (**Figure 2.3.32**). The tool will then extract the data and save it within a file in the same folder.



***Figure 2.3.32*** *– Memory extraction.*

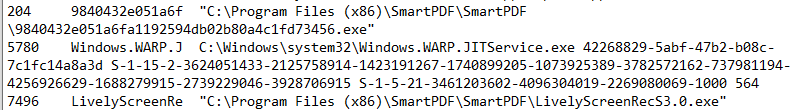
Once the RAM has been extracted, the user can start analysing it with Volatility 3.0. Volatility is a tool specifically created for RAM analysis. It automates the process and allows the user to save the results on files for further investigation. The tool contains a variety of plugins which can be used to filter out specific data:

* windows.info – basic information about the Windows machine
* windows.cmdline – shows the executed command line functions
* windows.malfind – looking for applications/services with injected malicious code
* windows.privileges – display the privileges (what they are allowed to do on the machine) for all dumped processes.
* windows.pstree – lists the visible processes – similar to Procmon process tree
* windows.netscan – lists the network connections.

To use the tool, the analyst is required to open a command line window inside Volatility’s folder by right-clicking and pressing on “Open command window here as admin…”. Once the window has opened, the user can then use the following command to analyse the memory dump – **py vol.py -f ­*memory\_dump\_location plugin\_name* > *file.txt***. The location of the memory dump should be **C:\Users\IEUser\Desktop\memory.raw** if **WinPMem** was saved on the Desktop as suggested. The ***file.txt*** should be changed with an appropriate file name. An example of a complete command is as follows:

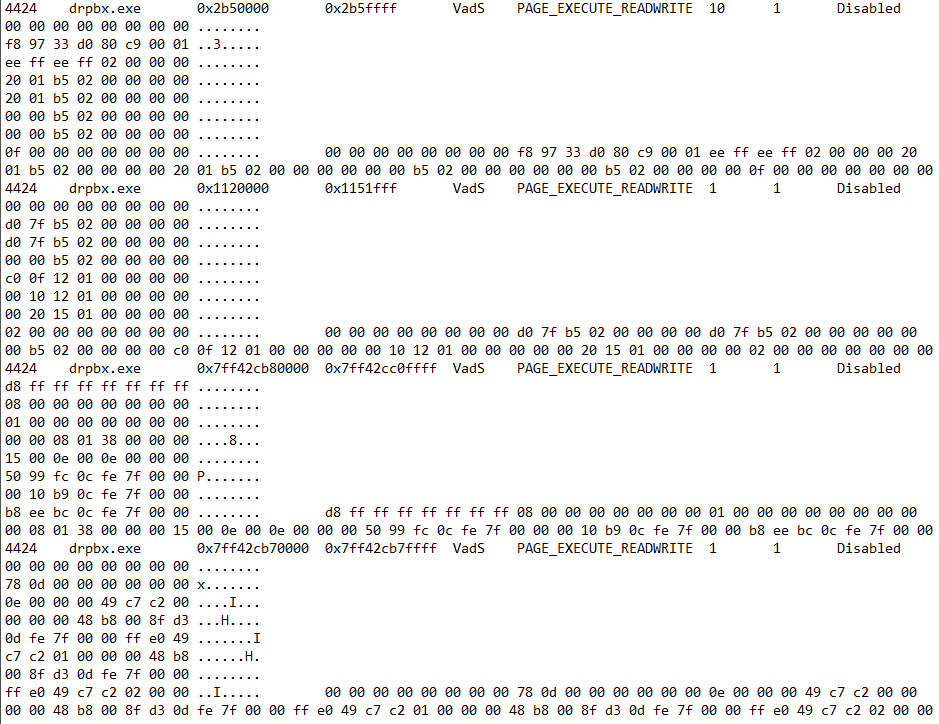
* **py vol.py -f C:\Users\IEUser\Desktop\memory.raw windows.cmdline > cmdline.txt**

The plugins will reveal various information as mentioned above. Cmdline will show the executed commands and the ID of the Process. It is important to identify the ID as it can be used for easier filtering. The ID can be found by looking for the malware’s name. (**Figure 2.3.33**)



***Figure 2.3.33*** *– Cmdline results for FileTour – process 204 and 7496.*

Malfind will identify applications with malicious code within them. It may display false-positive results so they should be provided to a professional for analysis. (**Figure 2.3.34**)



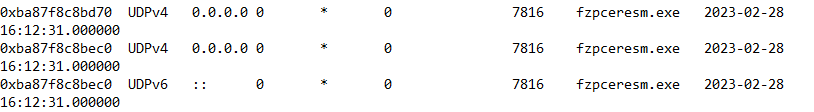
***Figure 2.3.34*** *– Injected malicious code in one of Jigsaw’s second-stage payloads.*

The privileges plugin will reveal all processes and their respective privileges. Most malware will commonly have a variety of privileges, allowing them to take a wide variety of actions within the system. (**Figure 2.3.35**)



***Figure 2.3.35*** *– Privileges of the drpbx second-stage payload.*

The process tree plugin will reveal similar information as Procmon. Sometimes it can reveal hidden data which was not visible, or the analyst may have missed during the previous stage. Finally, the netscan plugin can show the active network connections, possibly indicating that malware is trying to communicate with a C&C server. (**Figure 2.3.36**)



***Figure 2.3.36*** *– Open network connections on all IPs and ports seen in SnakeKeylogger.*

## **Yara Rule Creation**

Yara, also called the Swiss knife used by malware analysts, is a way to detect malicious software. The most common way to detect malware is through signature scanning. Previously, antivirus solutions utilised file hashes as their signatures, but this became inefficient with the evolution of malicious applications. They hashes can change due to encryption and obfuscation, making it appear as benign when the AV scans it.

Yara rules employs a different type of signature scanning – binary and text patterns. But how does it work? Analysts add different strings within the rule, which were found within the malicious software. Such strings may be distinctive file names, various commands, links, etc. Afterwards, researchers create conditions which would trigger the rule.

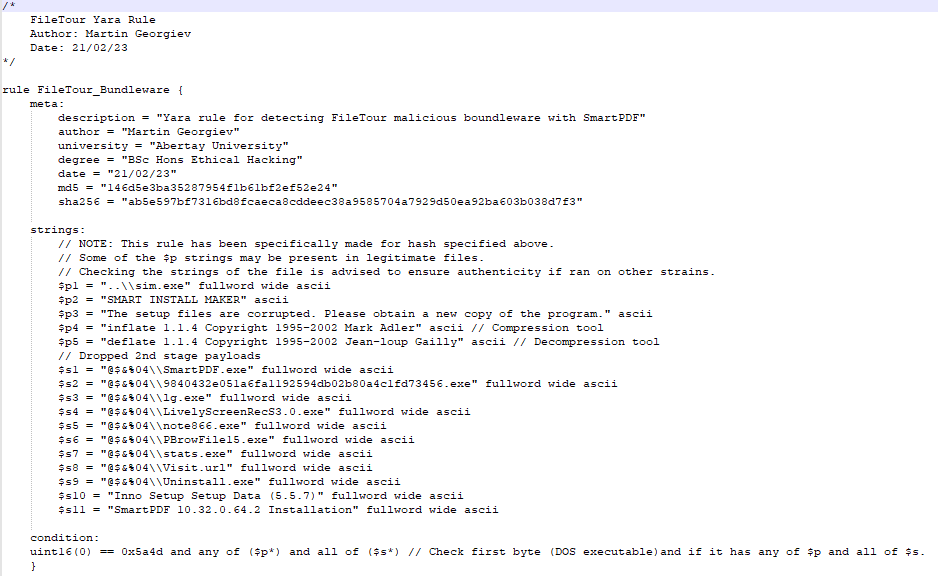
Yara rules have the following structure and syntax:

* Rule – rules must begin with the string “rule” and its name. This part encapsulates the entire structure within curly brackets ({})
* Meta – this is an optional section. It contains information about the rule, author, creation date, supported malware, etc. Researchers recommend its addition for easier identification.
* Strings – this section is not mandatory, but it is vital to customise rules for specific malware. Yara supports three different string types – text strings, hexadecimal strings, and regular expressions (Regex). Text strings will be the easiest to implement, while Regex requires more advanced knowledge. Each string must begin with the symbol **$**.
* Condition – this is the final section in a Yara rule. Here analysts can specify the conditions of the signature scan – include or exclude specific strings, identify first bytes (is it executable), specify file size, etc.

Additionally, strings can be provided with specific modifiers (VirusTotal, 2014) :

* **nocase** – is triggered by both capital and small characters.
* **wide** – UTF16 characters.
* **ascii** – ASCII characters (only if **wide** was used).
* **xor** – exclusive or (only one of the strings but not both).
* **base64** – Base64 encoded strings.
* **base64wide** – wide version of base64.
* **fullword** – any matches are not preceded or followed by alphanumeric characters.
* **private** – string match is not included in the output.

Each modifier can be used in specific cases. It is recommended to use “**fullword wide ascii**” for strings with special characters (**%**,**#**, etc.), sentences and directory paths. “**ascii**” is recommended when the string is comprised of a single alphanumeric phrase. An example will be given with a Yara rule for FileTour which employs various embedded file names, compression and decompression software names and messages. The conditions checked whether the first byte matched **0x5a4d** (Windows executable) and if it had 3 of **$s** strings or any of **$p**. (**Figure 2.4.1**) Furthermore, comments can be added after **//** (single row) or **/\* *text* \*/** (multiple rows).



***Figure 2.4.1*** *– FileTour Yara rule example.*

# **Results**

The report successfully provided analysts with a simple methodology for malware analysis. Following the static and dynamic analysis, they could quickly identify the capabilities of malware based on its behaviour.

The static analysis will reveal the malware’s functions, hidden strings, original data (name, etc.) and possibly when it was created. The dynamic analysis, on the other hand, can show how the malware will work after it was executed. It allows the researchers to monitor its behaviour on a local and network level, showing possible communications and local modifications.

# **Discussion**

## **General Discussion**

While the methodology may be considered as easy, users without any experience could experience difficulties following it. The researcher attempted to create it more accessible, but this could not be possible without automation, which was not the focus of the honours project.

To identify any issues and possible changes, the researcher created a questionnaire which was given to professionals in the Digital Forensics and Incident Response industry. Despite the low number of respondents, the answers provided valuable insight for the future development of the methodology.

Based on the responses, the current methodology would be difficult to follow for executives without technical knowledge. It could also be difficult to maintain due to the rapid development of malware variants and the surge of Fileless Malware (Sudhakar and Kumar, 2020). Additionally, the responses concluded that it could be successful in real-world environments if the analysis and incident response is carried out by a service provider. The combination of the methodology and modular IRPs could enhance the efficiency of the delivered response.

## **Future Work**

Knowing that the methodology is still too technical for inexperienced users, the researcher will attempt to automate the process, possibly creating a free-to-use cloud service. While such services already exist, most of them require the users to pay for in-depth analysis. The analysis will commence after the user uploads the suspicious file. In the end, they will receive a report, indicating what the malware may be capable of, allowing them to decide if they would like to escalate it to an external service provider or law enforcement.

# **Conclusion**

The methodology can provide analysts with an easy-to-follow guide on malware analysis. The users will be required to have basic IT knowledge to be able to effectively follow the guide. In the end, they will identify a lot of information about the malware’s capabilities – what it does, how it achieves it, does it only affect the local system or the whole network, etc.

# **References**

Sudhakar and Kumar, S. (2020). An emerging threat Fileless malware: a survey and research challenges. Cybersecurity, [online] 3(1). doi:https://doi.org/10.1186/s42400-019-0043-x. [Accessed 11 Oct. 2022]

Remnux (2010). *Remnux*. [online] Available at: https://remnux.org/#distro [Accessed 15 Oct. 2022].

Mandiant (2017). *FlareVM*. [online] Available at: https://github.com/mandiant/flare-vm [Accessed 15 Oct. 2022].

ytsif (2014). TheZoo. [online] Available at: https://github.com/ytisf/theZoo [Accessed 15 Oct. 2022].

Ivanov, A and Mamedov, O. (2017). ExPetr/Petya/NotPetya is a Wiper, Not Ransomware. Available: https://securelist.com/expetrpetyanotpetya-is-a-wiper-not-ransomware/78902/. [Accessed 16 Oct. 2022].

VirusTotal (2004 – Present Day). VirusTotal. [online] Available at: https://www.virustotal.com/gui/home/upload [Accessed 15 Nov. 2022].

Ballenthin, W. (2016). Floss. [online] Available at: https://github.com/mandiant/flare-floss. [Accessed 20 Nov. 2022].

Russinovich, M. (2021). Strings v2.54. [online] Available at: https://learn.microsoft.com/en-us/sysinternals/downloads/strings [Accessed 20 Nov. 2022].

ASL (2023). ExeInfoPE. [online] Available at: http://www.exeinfo.byethost18.com/?i=1 [Accessed 24 Nov. 2022].

Hungenberg, T and Eckert, M. (2007). INetSim. [online] Available at: https://www.inetsim.org/. [Accessed 27 Nov. 2022].

Russinovich, M. (2022). TCPView v4.17. [online] Available at: https://docs.microsoft.com/en-us/sysinternals/downloads/tcpview. [Accessed 27 Nov. 2022].

Wireshark. (1997 - Present Day). Wireshark Documentation. [online] Available at: https://www.wireshark.org/docs/. [Accessed 27 Nov. 2022].

Russinovich, M. (2022). Process Monitor v3.89. [online] Available at: https://docs.microsoft.com/en-us/sysinternals/downloads/procmon. [Accessed 28 Nov. 2022].

Fox, N. (2021). PeStudio Overview: Setup, Tutorial and Tips. [online] Available at: https://www.varonis.com/blog/pestudio. [Accessed 28 Nov. 2022].

Volatility Foundation (2020). Volatility 3.0. [online] Available at: https://www.volatilityfoundation.org/3 [Accessed 15 Dec. 2022].

Cohen, M., et al. (2019). WinPmem. [online] Available at: https://github.com/Velocidex/WinPmem [Accessed 15 Dec. 2022].

Stamus Labs (2022). Threat Detection Update. [online] Available at: https://www.stamus-networks.com/stamus-labs/detection-update-2022-05-31 [Accessed 8 Feb. 2023].

Netenrich (2021). FileTour. [online] Available at: https://know.netenrich.com/threatintel/malware/FileTour [Accessed 8 Feb. 2023].

Ashdown, D. (2021). Jigsaw Ransomware Analyses. [online] Available at: https://www.cyberdonald.com/post/jigsaw-ransomware-analyses [Accessed 12 Feb. 2023].

Zhang, X. (2021). Deep Dive into a Fresh Variant of Snake Keylogger Malware. [online] Available at: https://www.fortinet.com/blog/threat-research/deep-dive-into-a-fresh-variant-of-snake-keylogger-malware [Accessed 4 Mar. 2023].

AbuseCH (2020). MalwareBazaar. [online] Available at: https://bazaar.abuse.ch/ [Accessed 24 Mar. 2023].

VirusTotal (2013). Yara - The pattern matching Swiss knife for malware researchers (and everyone else). [online] Available at: https://virustotal.github.io/yara/ [Accessed 29 Mar. 2023].

VirusTotal (2014). Yara Documentation. [online] Available at: https://yara.readthedocs.io/en/stable/writingrules.html?highlight=modifier#string-modifier-summary [Accessed 23 Apr. 2023].