

School of Design and Informatics

BSc Ethical Hacking, 2022/23

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

**Jigsaw Analysis**

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

Head of Division of Cybersecurity Abertay

Contents

[1 Introduction 1](#_Toc134540208)

[**1.1** **Background** 1](#_Toc134540209)

[**1.2** **Aims** 1](#_Toc134540210)

[2 Procedure 2](#_Toc134540211)

[**2.1** **Overview** 2](#_Toc134540212)

[**2.2** **Static Analysis** 2](#_Toc134540213)

[**2.2.1** **File Hashes Discovery** 2](#_Toc134540214)

[**2.2.2** **String Extraction** 3](#_Toc134540215)

[**2.2.3** **In-depth Inspection** 5](#_Toc134540216)

[**2.2.4** **Dropped Files** 7](#_Toc134540217)

[**2.3** **Dynamic Analysis** 8](#_Toc134540218)

[**2.3.1** **Detonation Symptoms** 8](#_Toc134540219)

[**2.3.2** **Network Symptoms** 10](#_Toc134540220)

[**2.3.3** **Memory Analysis** 11](#_Toc134540221)

[**2.4** **Yara Rule** 13](#_Toc134540222)

[3 Results 15](#_Toc134540223)

[**3.1** **Analysis Results** 15](#_Toc134540224)

[4 Discussion 16](#_Toc134540225)

[**4.1** **General Discussions** 16](#_Toc134540226)

[**4.2** **Countermeasures** 16](#_Toc134540227)

[**4.2.1** **Pre-infection Countermeasures** 16](#_Toc134540228)

[**4.2.1.1** **Frequent Security Updates** 16](#_Toc134540229)

[**4.2.1.2** **Distinguishing Spam** 16](#_Toc134540230)

[**4.2.1.3** **Blacklisting Unknown Applications and Anti-Virus Software** 16](#_Toc134540231)

[**4.2.1.4** **File Scanners and IDS** 16](#_Toc134540232)

[**4.2.2** **Post-infection Countermeasures** 17](#_Toc134540233)

[**4.2.2.1** **Data Backups** 17](#_Toc134540234)

[**4.2.2.2** **Refuse Ransom Payments** 17](#_Toc134540235)

[**4.2.2.3** **Possible Decryption** 17](#_Toc134540236)

[5 Conclusion 18](#_Toc134540237)

[**5.1** **Conclusion** 18](#_Toc134540238)

[**5.2** **Future Work** 18](#_Toc134540239)

[6 References 19](#_Toc134540240)

[7 Appendices 20](#_Toc134540241)

[**Appendix A – Strings** 20](#_Toc134540242)

[**Appendix B – Memory Forensics** 20](#_Toc134540243)

[**Appendix B1 – Command Line** 20](#_Toc134540244)

[**Appendix B2 – Malfind** 20](#_Toc134540245)

[**Appendix B3 – Privileges** 21](#_Toc134540246)

[**Appendix C – Yara Rule** 23](#_Toc134540247)

# **Introduction**

## **Background**

**Jigsaw**, also known as **BitcoinBlackmailer** is a ransomware package that was first identified in April 2016. The malware is known for “**Billy the Puppet**” from the horror movie “**Saw**” and for its behaviour – being aggressive towards the infected yet making the payment process as easy as possible. It is a ransomware that relies a lot on fearmongering by threatening the infected user to not only delete their files if the ransom is not paid but also to publicly reveal any indecent data about the infected. All of this is done with a taunting tone in the ransom message and the image of the horror movie character or with an MP3 sound recording.

Some samples also mocked the user, telling them that at first only a few files will be deleted, gradually increasing with each passed day and hour. Others even had live support for the victims who do not know what bitcoins are, providing them with support in obtaining the currency and paying the requested amount. Files are also deleted if the machine is forcefully restarted, giving a warning to the user before the restarting process occurs. What made Jigsaw different from most other ransomware types was that it deleted the victim’s files rather than simply attempting to scare them (Ashdown, 2021).

Based on the behaviour of the adversaries and their malicious program, researchers believed that they may have not been motivated by the potential monetary gain. Some claim that they may have done it to cause chaos among users.

## **Aims**

The report aims to provide the reader with an analysis of a Jigsaw sample with the simplified Malware Analysis and Digital Forensics Framework and how it can show them information about its capabilities. The information can be used to take precautions until a professional team starts handling the case. It can also be given to the team handling the incident as this could greatly decrease the time to respond to the attack. The report will be split into three major sections to efficiently achieve the goal:

* **Procedure** – Analysis of the sample using the methodology and creation of a Yara rule to detect its signature
* **Results** – Overview and summary of the procedure and its subsections
* **Discussion** – General discussion and appropriate countermeasures

# **Procedure**

## **Overview**

As the developed methodology is aimed at small and medium-sized businesses that may not have a specialised response team due to budget constraints, the procedure will attempt to obtain as much intel as possible with simple techniques and tools. This will ensure that even users without significant technological knowledge can seamlessly follow it without significant difficulties. Considering the beforementioned requirements, an advanced static and dynamic analysis will not be performed on the sample as they require extensive knowledge of how computers operate, as well as high and low-level programming languages (JavaScript, C++, Assembly, etc.)

The hybrid analysis will be conducted on the sample in a safe testing environment. The static analysis will be achieved with a multitude of techniques and tools – obtaining file hashes, inspecting any human-readable strings within the binary (**Strings** (Russinovich, 2021) and **Floss** (Ballenthin, 2016)), analysing the executable’s library imports, functions and file entropy (**PEStudio** (Fox, 2021), and checking whether any known file packers have been used (**Exeinfo PE** (ALS, 2023)).

The Dynamic and Digital Forensic analysis will be combined as both can be done simultaneously. This section will cover the post-execution behaviour of the malicious sample – inspection of system modification (deleted/created/altered files, registries; PowerShell cmdlets execution, detection evasion and persistence mechanisms) with **Procmon**, possible network propagation or attempted communication with a **C&C** (**TCPView** (Russinovich, 2022), **Inetsim** (Hungenberg and Eckert, 2007)and **Wireshark** (Wireshark, 1997 – Present Day)). Additionally, analysing the system’s memory could display hidden processes, stolen data stored in the clipboard, possibly recover encryption keys and many more (**Volatility** (Volatility Foundation, 2020) and **WinPMem** (Cohen, et al., 2019)).

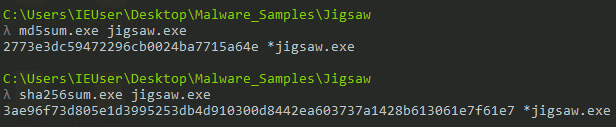
In the end, the obtained intel will be used to create a **Yara** (VirusTotal, 2013) rule for signature scanning that can then be incorporated with a scanner such as Strelka for passive file metadata scanning.

## **Static Analysis**

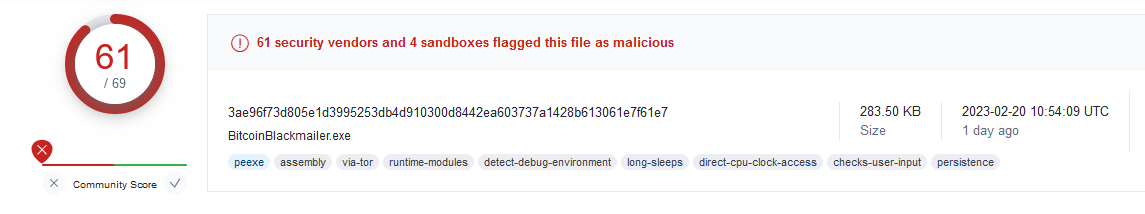
As mentioned in the previous section, the analysis will start with the static analysis of the sample. The basic static analysis provides limited information about the malware’s functionality. This, however, is often enough in introducing analysts with a base overview of the malicious software’s capabilities.

### **File Hashes Discovery**

Tools such as md5sum.exe and sha256sum.exe can be used from the command prompt to obtain the hashes of the malware. They will display the MD5 and SHA256 hashes of the sample respectively. The user can then copy them and check them in VirusTotal. This will check the databases of multiple antivirus vendors and display any matching results. (**Figure 2.2.1** and **Figure 2.2.2**)



***Figure 2.2.1*** *– Obtaining MD5 and SHA256 hashes.*

**

***Figure 2.2.2*** *– VirusTotal indicating the sample is malicious.*

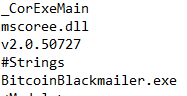
### **String Extraction**

Human-readable strings can be extracted from files with a variety of tools. Some of the most common ones are **strings** and **floss**. In some cases, **floss** may be a better alternative to the former tool as it attempts to de-obfuscate and decode any strings which were intentionally made hard to read. The tool also has a multitude of flags for data filtering. In this case, **-n** was used as this puts a minimum character length of the extracted strings. The analyst set the length to **8** as they were interested in possible sentences, links and/or imported functions/libraries. The output was piped onto a text file (with **>**) to allow easier analysis and further filtering. (**Figure 2.2.3**)



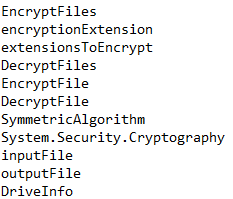
***Figure 2.2.3*** *– Extracting strings with Floss.*

Opening the file ([**Appendix A**](#_Appendix_A_–)) immediately revealed that it was a Windows executable. Scrolling down showed the name of a binary file (**BitcoinBlackmailer.exe**) – the other name the ransomware is known as. (**Figure 2.2.4**)



***Figure 2.2.4*** *– BitcoinBlackmailer.exe name in the strings.*

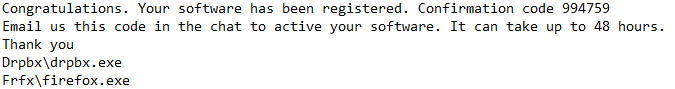
Further inspection showed a multitude of functions related to encryptions and cryptocurrencies (obtaining price, balance, etc.). It also contained function names used for file decryption. One of the strings also revealed that a symmetric algorithm (and **AesCryptoServiceProvider** hinting at AES encryption) may have been used to encrypt the files – this would be of great help when researchers attempt to create a decryption tool. (**Figure 2.2.5**)



***Figure 2.2.5*** *– Strings related to cryptography capabilities.*

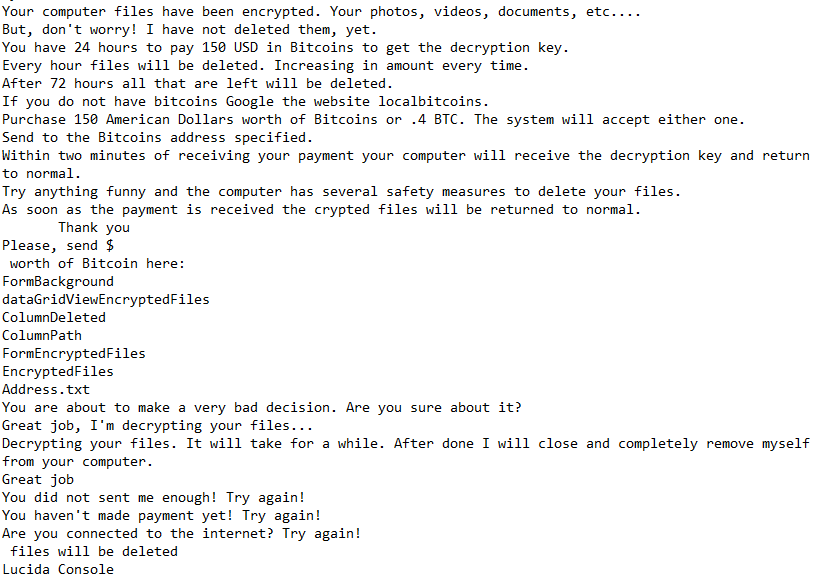
Similar to previously analysed malware, Jigsaw also used costura for embedding various dependencies as resources. It also had functions allowing it to enumerate through the entire system and all available drives (possibly also encrypting any newly plugged drives as well).

Continuing with the string analysis, various messages began to reveal themselves. A congratulatory message was possibly showed after the detonation of the sample, providing a confirmation code. It also indicated that the code must be emailed to activate the software with the procedure taking up to forty-eight hours. Right below it the names of two more executables were revealed – **Drpbx/drpbx.exe** and **Frfx/firefox.exe** (Anyrun, 2019). The names are similar to the legitimate Dropbox and Firefox applications, showing that the malware could be using them to evade detection and/or ensure persistence on the system. (**Figure 2.2.6**)



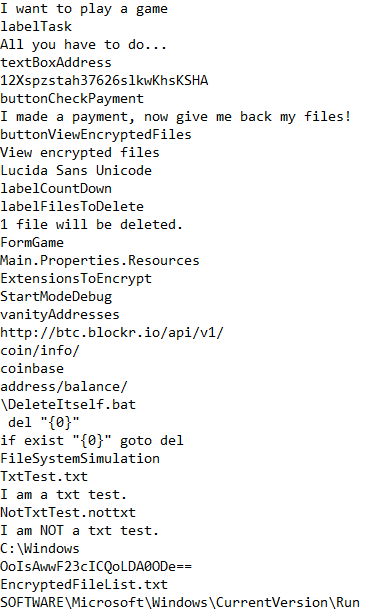
***Figure 2.2.6*** *– Congratulatory message and suspicious names for executable files.*

Below the message shown above was the ransom message, attempting to fearmonger the victim the threatening them to delete their files if a specific amount of BTC was not sent . Additionally, the message mentions that any other actions taken by the user would result in their files being deleted. Further inspection showed a different fearmongering message, telling the victim that they “are about to make a very bad decision” – possibly if they were trying to restart their machine. (**Figure 2.2.7**)



***Figure 2.2.7*** *– Ransom and fearmongering messages.*

At the end of the identified strings the analyst could see various strings and a URL connected to bitcoins, the Windows directory, the Run registry (possibly used for persistence) and a text file listing all encrypted files. (**Figure 2.2.8**)

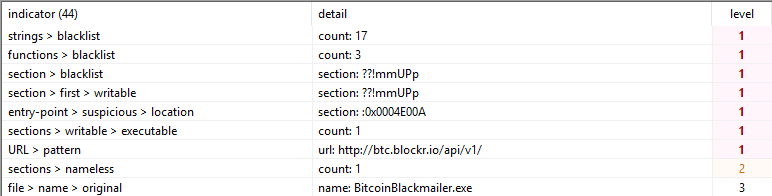


***Figure 2.2.8*** *– Bitcoin related strings and various text files.*

### **In-depth Inspection**

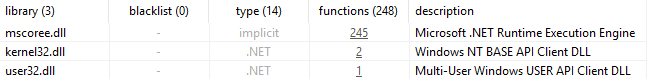
The analysis proceeded with **PEStudio**. **PEStudio** is a tool speeding up the initial malware analysis process and making it easier. The tool conducts a complete static analysis of the file and provides researchers with indicators, imports, libraries, and file entropy (randomness of data hinting at hidden/suspicious data).

The file header indicated that the initial compilation of the ransomware was on the 31st of March 2016 – just a day before the month it started appearing in the wild. The tool then showed seven critical and one serious indicators – blacklisted strings/sections/functions, suspicious entry points/sections, URL patterns, and the original name of the file. Additionally, the file had an entropy value of 4.794 out of 8, showing that it was obfuscated. (**Figure 2.2.9**)



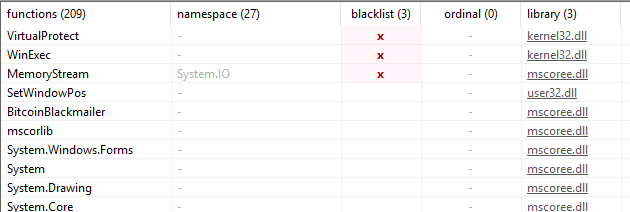
***Figure 2.2.9*** *– Malicious indicators.*

In terms of the libraries, the tool identified three libraries, none of which were blacklisted. Based on the functionalities of the sample and the identified strings in the previous section, the researcher could deduct that not all libraries were displayed. This could be due to obfuscation. (**Figure 2.2.10**)



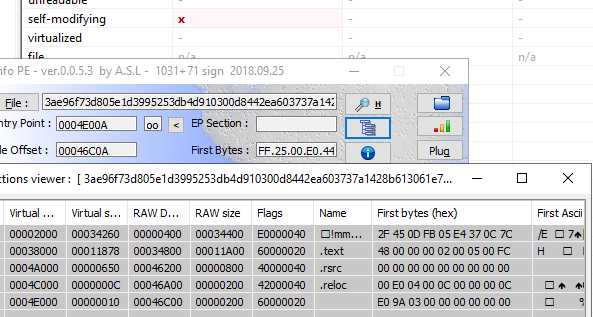
***Figure 2.2.10*** *– List of libraries.*

The tool discovered a total of three blacklisted functions – VirtualProtect, WinExec, and MemoryStream. Those functions would allow an attacker to asynchronously execute applications within memory. Something suspicious in the results was the lack of encryption and decryption functions as they were identified in the strings analysis. The analyst also found a function with an illegible name and a method called Jigsaw in the **mscoree.dll** library. Such function does not exist in the mentioned dynamic link library. This could have been done to bypass Anti-Virus detection at the time with the use of obfuscation and/or other techniques. The only cryptography related functions were **ExtensionsToEncrypt**, **CryptoStream,** and **AesCryptoServiceProvider** – the latter two are standard functions used in non-malicious applications which was why they were not marked as blacklisted. (**Figure 2.2.11**)



***Figure 2.2.11*** *– Partial list of functions utilised by the malware.*

The analyst then checked whether the malware was packed using one of the commonly found packers such as UPX1 to evade detection. This was achieved with **Exeinfo PE**. Loading the file into the tool did not reveal anything regarding the EP section – the entry point, according to **PEStudio** was a custom section with self-modifying code. This was possibly the reason why **Exeinfo PE** could not identify the packer. (**Figure 2.2.12**)

******

***Figure 2.2.12*** *– FileTour using a custom packer within its CODE section.*

### **Dropped Files**

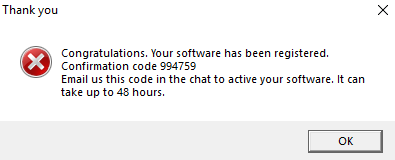
As mentioned in [**2.2.2 String Extraction**](#_String_Extraction), two additional files were identified – **drpbx.exe** and **firefox.exe**. Analysing those files revealed the same information seen in the analysis of the original sample with **Floss** and **PEStudio**. The extracted strings were identical, as well as the intel provided by the latter tool. The malware also does not attempt to connect to any C&C server or domain until the user clicks on the button used to confirm their payment.

## **Dynamic Analysis**

Dynamic analysis of malware is achieved by detonating the sample in a safe environment (or surveying an already compromised environment) to see how it behaves on a local and network level. This may be dangerous if the safe environment is not properly set up as it may allow the malicious software to propagate to the physical machine, the user’s network, and possibly even other connected networks.

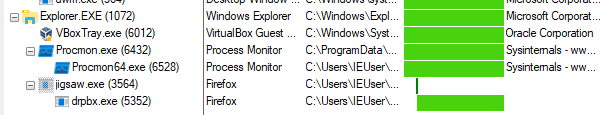
### **Detonation Symptoms**

The dynamic analysis started with the initial detonation of the sample and inspection of the symptoms of the infected virtual machine. They attempted to detonate it both with and without administrator rights to see if there would be any differences. The sample executed itself in both cases. It first displayed the product registration message found in the extracted strings and dropped the other two binaries in their respective folders - %User%\AppData\Roaming\Frfx\firefox.exe and %User%\AppData\Local\Drpbx\dropbox.exe (**Figure 2.3.1**) After less than a minute, visible changes began to appear – files changing their extension. Using Procmon, they saw that



***Figure 2.3.1*** *– Product registration alert.*

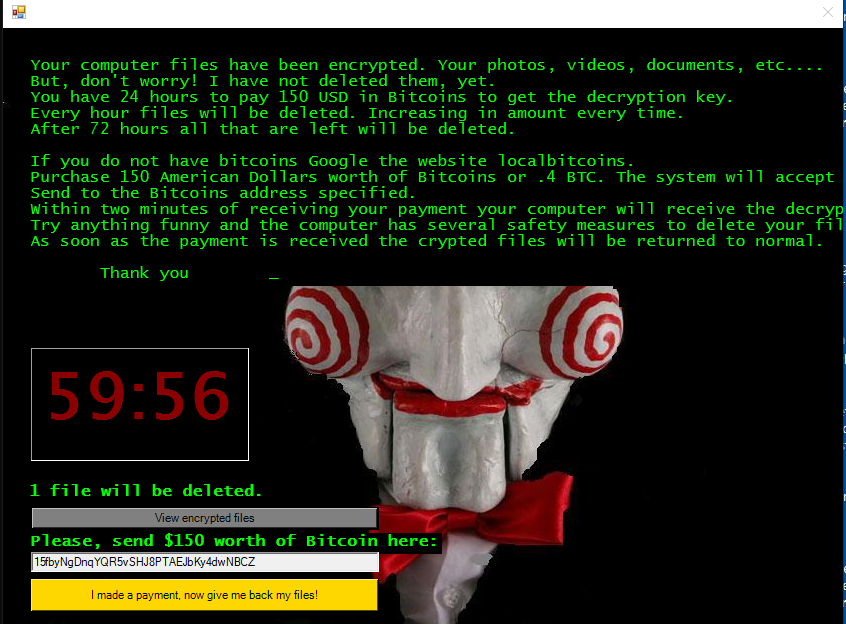
Examining the behaviour in **Procmon** revealed that the initial sample was simply used as a detonator. The encryption process was then forwarded to **drpbx.exe** which had a considerably longer lifetime compared to the detonator. (**Figure 2.3.2**)



***Figure 2.3.2*** *– Lifetime of the initial sample and drpbx.exe*

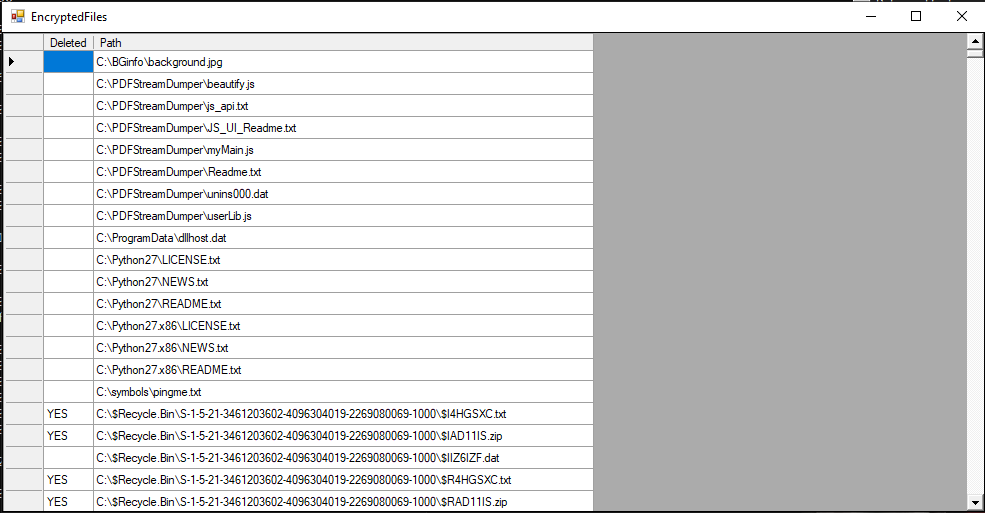
**Jigsaw.exe**, the initial sample, stopped at a total of 10,258 events. In comparison, **drpbx.exe** had a total of 3,025,860 events. This essentially proved that the first dropped binary handled the encryption process of the victim’s machine. The process connected to **firefox.exe** did not have any events and it did not appear in the process tree.

A few seconds later a custom window appeared and displayed the ransom message. The message slowly output itself on the screen word by word, with the infamous villain in **Saw** being in the background. Once the entire message displayed itself, the users were provided with two buttons, a timer, and a bitcoin wallet address. (**Figure 2.3.3**)



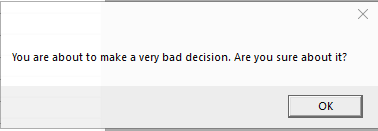
***Figure 2.3.3*** *– Ransom message.*

The button below the timer allowed the infected user to check which files on their system have been encrypted. There was also a field indicating whether the files were deleted or not. (**Figure 2.3.3**)



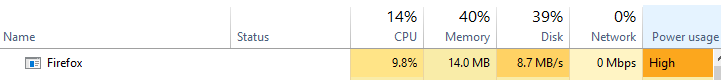
***Figure 2.3.3*** *– List of encrypted files.*

As the ransomware threatened the victim to delete their files, the analyst conducted two tests. They first waited for the timer to run out. Launching the list revealed that a significant number of files were deleted, and the action was confirmed by inspecting the directories where the files should have existed. The second test required the computer to be restarted. The analyst first saw the warning message that prevented the computer from restarted itself. (**Figure 2.3.4**)



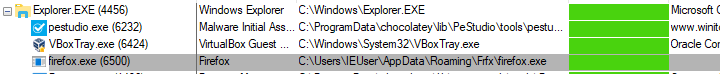
***Figure 2.3.4*** *– Warning before the machine is restarted.*

The analyst then forcefully rebooted the machine. After they logged into the machine they opened the task manager to see if any persistence task would appear. They identified a suspicious version of Firefox using a lot of power and resources – the persistence mechanism of the ransomware. (**Figure 2.3.5**)



***Figure 2.3.5*** *– Launching the firefox.exe process.*

Opening **Procmon** revealed that **firefox.exe** was the only file of the three to have any processes. **Jigsaw.exe** and **drpbx.exe** did not have any processes related to them. With this, the analyst identified that **firefox.exe** was used as the persistence mechanism for the sample. (**Figure 2.3.6**) It was possible that **drpbx.exe** deleted the files before the restart, as **firefox.exe** had no events connected to deletion of files.

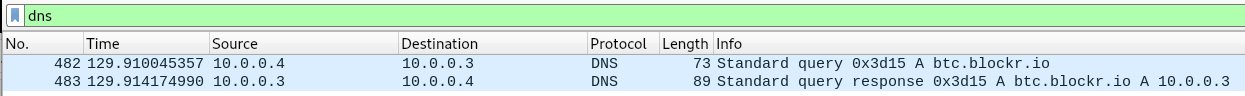


***Figure 2.3.6*** *– Only firefox.exe showing in the process list.*

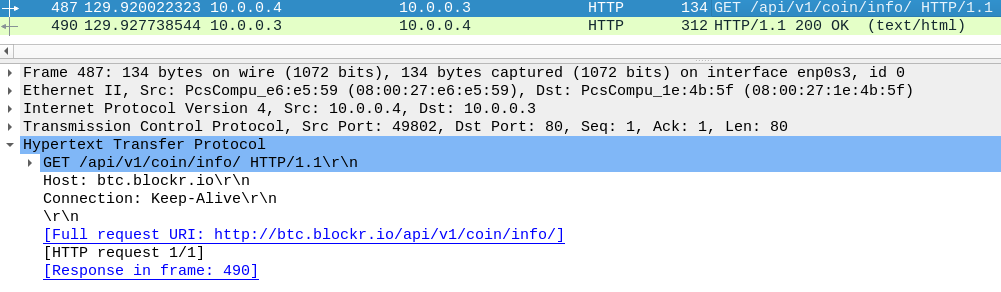
### **Network Symptoms**

To examine the network symptoms, the analyst used tools such as **TCPView** and **Wireshark** while the machine was connected to a second one (**Remnux**) that had **Inetsim** running in the background. **Inetsim** is a tool which simulates an internet connection and a DHCP server, allowing it to capture all traffic in a host-only environment. **TCPView** allows the analyst to examine all connections going from and to the machine, while **Wireshark** is used to capture and examine the traffic towards **Inetsim**.

Opening **TCPView** did not reveal any connections performed by the malware. Afterwards, the analyst examined the network traffic in Wireshark, but the sample remained static. It only attempted to connect a specific domain whenever the user clicked on the payment button. The domain was also seen in the strings analysis. (**Figure 2.3.7** and **Figure 2.3.8**)



***Figure 2.3.7*** *– DNS traffic of the malware.*

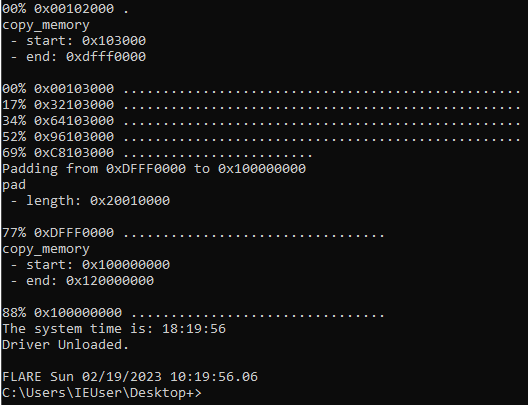
**

***Figure 2.3.8*** *– Http traffic of the sample.*

### **Memory Analysis**

Memory analysis can be a tedious task, but simple checks may reveal malicious behaviour in different processes. To analyse the memory of the infected virtual machine, the analyst used **Volatility 3.0** and **WinPMem** to dump the memory.

Dumping the memory could be easily achieved with the beforementioned tool. The analyst ran the following command in a **CMD** window to acquire the memory data in raw format: **winpmem\_mini\_x64\_rc2.exe memory.raw**. This command wrote the contents of the RAM in a file called **memory.raw**. (**Figure 2.3.9**)

****

***Figure 2.3.9*** *– Dumping the memory using WinPMem.*

Afterwards, the researcher analysed different data from the raw memory dump. (**Figure 2.3.18**) This was achieved with various plugins, in this case specifically ([**Appendix B**](#_Appendix_B_–)):

* windows.info – basic information about the windows machine
* windows.cmdline – executed cmd commands
* windows.malfind – looking for applications/services with injected malicious code
* windows.privileges – display the privileges for all dumped processes
* windows.pstree – lists the visible processes

The analysis was achieved using the following command: **py vol.py -f C:\Users\IEUser\dDesktop\memory.raw** **windows.***plugin\_name*. The commands list contained only one entry connected to the malware – **PID 4424**. The process was connected to **drpbx.exe** and it simply showed how it was executed after **jigsaw.exe**. (**Figure 2.3.10**)

**

***Figure 2.3.10*** *– Drpbx.exe executing itself.*

The analyst used the connections plugin to see if there were any hidden connections that were not displayed on **TCPView**. The results, however, did not show any hidden connections related to the malicious process. Using the windows.privileges plugin revealed malicious behaviour from the **drpbx.exe** executable. It had multiple privileges which could be used for malicious purposes (**Figure 2.3.11**):

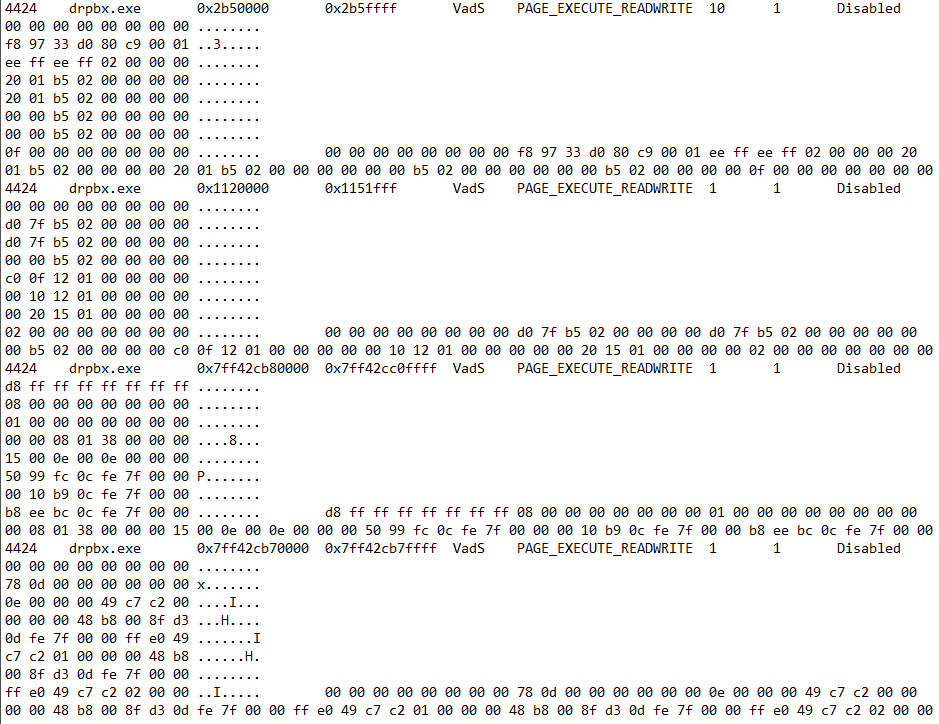
* System shutdown
* Program debugging
* Load/unload device drivers
* Change system time
* Manage security logs
* Act as a part of the OS
* Process-level tokens
* Lock pages in memory
* Manage files on volumes
* Obtain an impersonation token for another user in the same session
* Client impersonation after authentication
* Memory allocation for user—space applications



***Figure 2.3.11*** *– Process privileges for* **drpbx.exe**.

This effectively shows that the payload has complete access to the resources of the infected machine, as well as possibly impersonating all users in the session and any authenticated clients.

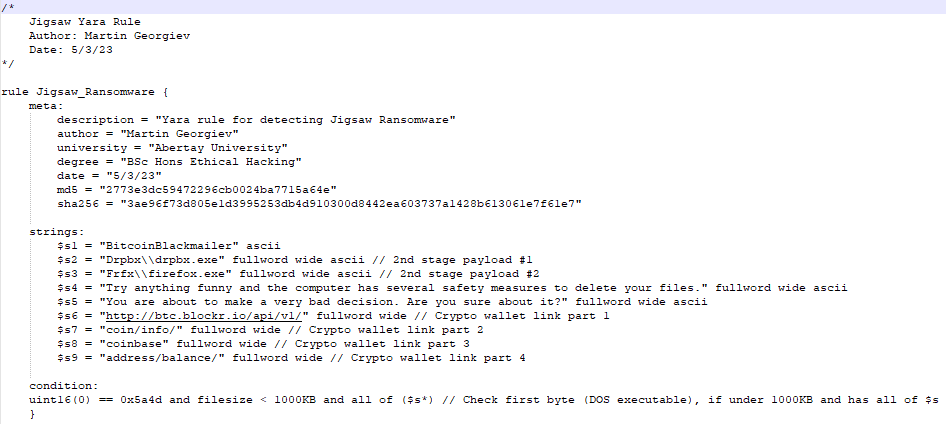
Finally, the analyst used the windows.malfind plugin to look for possible injected malicious code within applications. It must be noted that the plugin may show false-positive results, but it can still be a great indication of possible malicious processes. Out of five processes four were connected to **drpbx.exe**, with the locations having **PAGE\_EXECUTE\_READWRITE** permissions. (**Figure 2.3.12**)



**Figure 2.3.12** – Instances of **drpbx.exe** in malfind.

## **Yara Rule**

The Yara rule was created with notable data from the analysed strings. This is the easiest method to create simple Yara rules which can detect the malware – file names, specific commands and other strings related to this specific malware. (**Figure 2.4.1**) The Yara rule can be found in [**Appendix C**](#_Appendix_C_–).



***Figure 2.4.1*** *– Jigsaw Yara Rule.*

# **Results**

## **Analysis Results**

By conducting both static and dynamic analysis, the researcher identified a lot of valuable information regarding the behaviour of the ransomware and its capabilities. The static analysis revealed what functions have been used in the sample, the ransom message, and names of local second-stage payloads. It also showed how the ransomware possibly attempts to evade Anti-Virus systems using a custom, self-modifying entry point.

The dynamic analysis showed how the ransomware works – does it really contact any web servers if the user has paid, how does it persist in the system, and does it delete any files or are those attempts to simply fearmonger the victims. The analysis proved the beforementioned points and revealed the detonation process of the sample, as well as which processes are called if the user does not comply with the threats.

# **Discussion**

## **General Discussions**

Analysing the sample revealed that it successfully encrypted the user’s files. Compared to other ransomware variants, Jigsaw does delete the files of users if they do not comply with its demands. The static and dynamic analysis showed the behaviour of the sample, as well as how it would behave in both local and external environments. The lack of a propagation mechanism was expected as it is known that the ransomware mainly propagated through phishing emails.

The sample did not use any payloads downloaded over the internet, nor did it have any communications with a C&C server. Some of the other strains may have different behaviour as there have been identified variants with live support chats for the victims controlled by the attackers.

## **Countermeasures**

### **Pre-infection Countermeasures**

The most effective way to protect a system from infection would be before it becomes infected. As some of the modules contain cryptography functions, a fully working sample could encrypt the victim’s files. This, in most cases, would not allow them to retrieve their files.

#### **Frequent Security Updates**

One of the reasons why malware is successful is the lack of security patches or users refusing to apply the newest updates to their operating systems and/or anti-virus applications Keeping your system and anti-virus software up to date would ensure that publicly known vulnerabilities could not be exploited, and the AV may have updated signature databases to detect the sample.

#### **Distinguishing Spam**

As the malware is primarily distributed through social engineering, users must be able to distinguish spam emails from real ones. This also applies to legitimate and fake websites and/or files. Users should not open any links or execute files unless they know the sender and the nature of the link or file. Additionally, users should look for bad grammar, fearmongering, rushed actions or similar addresses to legitimate ones.

#### **Blacklisting Unknown Applications and Anti-Virus Software**

System administrators could put restrictions on users by blacklisting unknown software. This way they would not be able to execute suspicious applications and provide the system/network with damage control to prevent any harm. It could be achieved with Anti-Virus software and integrated browser protection.

An updated Anti-Virus software could be used to perform system scans or simply scan newly generated files. Some may even prevent the malicious software from executing itself if they recognise specific code patterns or behaviours.

#### **File Scanners and IDS**

Intrusion Detection Systems will alert security analysts if they detect any suspicious behaviour – phishing emails, specific signatures, etc. Some of them can also be combined with file scanners such as Strelka for greater detection accuracy. This way the internal SOC team could notice the threat before it causes any harm to the system.

### **Post-infection Countermeasures**

#### **Data Backups**

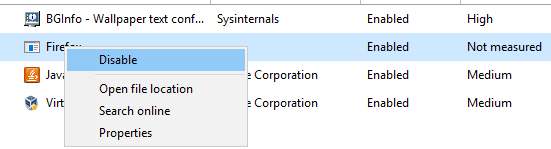
As some of the payloads contained cryptography-related libraries and functions, it would be beneficial to keep data backups. With such, the company could wipe the infected drive and simply replace it with the information they have stored elsewhere. It is also advised to keep such data in physical storage if possible or in locations which are not directly connected to the network of the infected machine as some malware could propagate to it and destroy the backups.

#### **Refuse Ransom Payments**

Ransom payments should **NOT** be considered even in dire situations. In the case of encrypted files, the adversary may attempt to fearmonger the victim by threatening them to publicly post their data or delete it. Paying the ransom does not guarantee that the data can be recovered as the attacker may send a fake decryption key or they may not send one at all.

#### **Possible Decryption**

Multiple analysts discovered that the ransomware could be decrypted for free. After reverse-engineering the sample and identifying how its encryption algorithm works, they created a decryptor. To use it, the infected users must first terminate the **firefox.exe** and **drpbx.exe** processes from the Task Manager to prevent further deletion of files. The victims must then open the start-up tab within the Task Manager to disable **firefox.exe** (**Figure 4.2.1**)



***Figure 4.2.1*** *– Disabling the start-up process.*

Afterwards, the users can download the decryptor from websites such as **BleepingComputer**. Depending on the strain, such decryptors may also result in further deletion or they may not even work. Victims are suggested to first consult themselves with a professional. Using such tools must be done at their own risk.

# **Conclusion**

## **Conclusion**

Jigsaw is ransomware which toys with its victims to scare them. Compared to other samples, the BitcoinBlackmailer deleted the users’ files if they refused to comply with the attackers. Based on the design of the ransomware, previous research conducted by professionals and the behaviour of the authors, it can be deduced that the ransomware’s main goal was to cause chaos among infected users, rather than trying to obtain money from them.

## **Future Work**

With more time, the analyst would attempt to reverse-engineer the sample to learn more about the encryption mechanism. With the obtained intel they would create an easy-to-use decryptor which can be used by victims to decrypt their files without being forced to pay the attackers for a decryption key they might not even receive.

# **References**

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

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

Anyrun (2019). Jigsaw Analysis. [online] Available at: https://any.run/report/3ae96f73d805e1d3995253db4d910300d8442ea603737a1428b613061e7f61e7/90db558b-a6cc-4d20-9024-6617bc6b00db [Accessed 17 Feb. 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].

# **Appendices**

## **Appendix A – Strings**

*Note: The output was truncated to remove a lot of illegible data.*

As the size of the strings remained substantial, they were not included in the report. They can be found in the folders of each binary in .txt files.

## **Appendix B – Memory Forensics**

### **Appendix B1 – Command Line**

644 532 winlogon.exe 0x9082ad5e9240 6 - 1 False 2023-02-21 12:49:48.000000 N/A

\* 864 644 fontdrvhost.ex 0x9082ae43f080 5 - 1 False 2023-02-21 12:49:48.000000 N/A

\* 564 644 dwm.exe 0x9082ae47f080 25 - 1 False 2023-02-21 12:49:48.000000 N/A

\* 872 644 userinit.exe 0x9082af35d080 0 - 1 False 2023-02-21 12:50:24.000000 2023-02-21 12:50:40.000000

\*\* 1072 872 explorer.exe 0x9082af368080 99 - 1 False 2023-02-21 12:50:24.000000 N/A

\*\*\* 6464 1072 cmd.exe 0x9082acfa5540 3 - 1 False 2023-02-21 13:54:33.000000 N/A

\*\*\*\* 5648 6464 conhost.exe 0x9082afa16080 7 - 1 False 2023-02-21 13:54:33.000000 N/A

\*\*\*\* 1488 6464 winpmem\_mini\_x 0x9082b15a62c0 4 - 1 False 2023-02-21 13:54:48.000000 N/A

\*\*\* 2600 1072 Procmon.exe 0x9082aee93080 3 - 1 True 2023-02-21 13:49:09.000000 N/A

\*\*\*\* 1408 2600 Procmon64.exe 0x9082ab250080 8 - 1 False 2023-02-21 13:49:09.000000 N/A

\*\*\* 6012 1072 VBoxTray.exe 0x9082af4e5540 11 - 1 False 2023-02-21 12:50:32.000000 N/A

\*\*\* 4688 1072 msdt.exe 0x9082ab34c080 0 - 1 False 2023-02-21 12:55:19.000000 2023-02-21 12:55:29.000000

\*\*\* 2172 1072 chrome.exe 0x9082ae38e540 0 - 1 False 2023-02-21 12:55:03.000000 2023-02-21 13:08:01.000000

1480 7112 software\_repor 0x9082a7d71340 0 - 1 False 2023-02-21 12:55:06.000000 2023-02-21 12:59:36.000000

848 7112 software\_repor 0x9082a7d742c0 0 - 1 False 2023-02-21 12:55:06.000000 2023-02-21 12:59:36.000000

4424 4516 drpbx.exe 0x9082ab0a9080 4 - 1 False 2023-02-21 13:49:47.000000 N/A

### **Appendix B2 – Malfind**

4424 drpbx.exe 0x2b50000 0x2b5ffff VadS PAGE\_EXECUTE\_READWRITE 10 1 Disabled

00 00 00 00 00 00 00 00 ........

f8 97 33 d0 80 c9 00 01 ..3.....

ee ff ee ff 02 00 00 00 ........

20 01 b5 02 00 00 00 00 ........

20 01 b5 02 00 00 00 00 ........

00 00 b5 02 00 00 00 00 ........

00 00 b5 02 00 00 00 00 ........

0f 00 00 00 00 00 00 00 ........ 00 00 00 00 00 00 00 00 f8 97 33 d0 80 c9 00 01 ee ff ee ff 02 00 00 00 20 01 b5 02 00 00 00 00 20 01 b5 02 00 00 00 00 00 00 b5 02 00 00 00 00 00 00 b5 02 00 00 00 00 0f 00 00 00 00 00 00 00

4424 drpbx.exe 0x1120000 0x1151fff VadS PAGE\_EXECUTE\_READWRITE 1 1 Disabled

00 00 00 00 00 00 00 00 ........

d0 7f b5 02 00 00 00 00 ........

d0 7f b5 02 00 00 00 00 ........

00 00 b5 02 00 00 00 00 ........

c0 0f 12 01 00 00 00 00 ........

00 10 12 01 00 00 00 00 ........

00 20 15 01 00 00 00 00 ........

02 00 00 00 00 00 00 00 ........ 00 00 00 00 00 00 00 00 d0 7f b5 02 00 00 00 00 d0 7f b5 02 00 00 00 00 00 00 b5 02 00 00 00 00 c0 0f 12 01 00 00 00 00 00 10 12 01 00 00 00 00 00 20 15 01 00 00 00 00 02 00 00 00 00 00 00 00

4424 drpbx.exe 0x7ff42cb80000 0x7ff42cc0ffff VadS PAGE\_EXECUTE\_READWRITE 1 1 Disabled

d8 ff ff ff ff ff ff ff ........

08 00 00 00 00 00 00 00 ........

01 00 00 00 00 00 00 00 ........

00 00 08 01 38 00 00 00 ....8...

15 00 0e 00 0e 00 00 00 ........

50 99 fc 0c fe 7f 00 00 P.......

00 10 b9 0c fe 7f 00 00 ........

b8 ee bc 0c fe 7f 00 00 ........ d8 ff ff ff ff ff ff ff 08 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 00 08 01 38 00 00 00 15 00 0e 00 0e 00 00 00 50 99 fc 0c fe 7f 00 00 00 10 b9 0c fe 7f 00 00 b8 ee bc 0c fe 7f 00 00

4424 drpbx.exe 0x7ff42cb70000 0x7ff42cb7ffff VadS PAGE\_EXECUTE\_READWRITE 1 1 Disabled

00 00 00 00 00 00 00 00 ........

78 0d 00 00 00 00 00 00 x.......

0e 00 00 00 49 c7 c2 00 ....I...

00 00 00 48 b8 00 8f d3 ...H....

0d fe 7f 00 00 ff e0 49 .......I

c7 c2 01 00 00 00 48 b8 ......H.

00 8f d3 0d fe 7f 00 00 ........

ff e0 49 c7 c2 02 00 00 ..I..... 00 00 00 00 00 00 00 00 78 0d 00 00 00 00 00 00 0e 00 00 00 49 c7 c2 00 00 00 00 48 b8 00 8f d3 0d fe 7f 00 00 ff e0 49 c7 c2 01 00 00 00 48 b8 00 8f d3 0d fe 7f 00 00 ff e0 49 c7 c2 02 00 00

### **Appendix B3 – Privileges**

4424 drpbx.exe 2 SeCreateTokenPrivilege Create a token object

4424 drpbx.exe 3 SeAssignPrimaryTokenPrivilege Replace a process-level token

4424 drpbx.exe 4 SeLockMemoryPrivilege Lock pages in memory

4424 drpbx.exe 5 SeIncreaseQuotaPrivilege Increase quotas

4424 drpbx.exe 6 SeMachineAccountPrivilege Add workstations to the domain

4424 drpbx.exe 7 SeTcbPrivilege Act as part of the operating system

4424 drpbx.exe 8 SeSecurityPrivilege Manage auditing and security log

4424 drpbx.exe 9 SeTakeOwnershipPrivilege Take ownership of files/objects

4424 drpbx.exe 10 SeLoadDriverPrivilege Load and unload device drivers

4424 drpbx.exe 11 SeSystemProfilePrivilege Profile system performance

4424 drpbx.exe 12 SeSystemtimePrivilege Change the system time

4424 drpbx.exe 13 SeProfileSingleProcessPrivilege Profile a single process

4424 drpbx.exe 14 SeIncreaseBasePriorityPrivilege Increase scheduling priority

4424 drpbx.exe 15 SeCreatePagefilePrivilege Create a pagefile

4424 drpbx.exe 16 SeCreatePermanentPrivilege Create permanent shared objects

4424 drpbx.exe 17 SeBackupPrivilege Backup files and directories

4424 drpbx.exe 18 SeRestorePrivilege Restore files and directories

4424 drpbx.exe 19 SeShutdownPrivilege Present Shut down the system

4424 drpbx.exe 20 SeDebugPrivilege Debug programs

4424 drpbx.exe 21 SeAuditPrivilege Generate security audits

4424 drpbx.exe 22 SeSystemEnvironmentPrivilege Edit firmware environment values

4424 drpbx.exe 23 SeChangeNotifyPrivilege Present,Enabled,Default Receive notifications of changes to files or directories

4424 drpbx.exe 24 SeRemoteShutdownPrivilege Force shutdown from a remote system

4424 drpbx.exe 25 SeUndockPrivilege Present Remove computer from docking station

4424 drpbx.exe 26 SeSyncAgentPrivilege Synch directory service data

4424 drpbx.exe 27 SeEnableDelegationPrivilege Enable user accounts to be trusted for delegation

4424 drpbx.exe 28 SeManageVolumePrivilege Manage the files on a volume

4424 drpbx.exe 29 SeImpersonatePrivilege Impersonate a client after authentication

4424 drpbx.exe 30 SeCreateGlobalPrivilege Default Create global objects

4424 drpbx.exe 31 SeTrustedCredManAccessPrivilege Access Credential Manager as a trusted caller

4424 drpbx.exe 32 SeRelabelPrivilege Modify the mandatory integrity level of an object

4424 drpbx.exe 33 SeIncreaseWorkingSetPrivilege Present Allocate more memory for user applications

4424 drpbx.exe 34 SeTimeZonePrivilege Present Adjust the time zone of the computer's internal clock

4424 drpbx.exe 35 SeCreateSymbolicLinkPrivilege Required to create a symbolic link

4424 drpbx.exe 36 SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for another user in the same session.

## **Appendix C – Yara Rule**

