WannaCry Ransomware Analysis Report

Martin Pavlinov Zhelev

CMP320: Advanced Ethical Hacking

2022/23

Note that Information contained in this document is for educational purposes.

.

Abstract

Malware Is a threat that keep getting more severe with each year. The damage caused to companies all around the world by ransomware attacks keep getting higher year by year. For this report, the analyst examined a malware sample that was provided within a VMWare Virtual Machine running FlareVM. The analyst had the aim of determining the type of the malware sample, analysing its characteristics and components, understanding its behaviour, and discussing relevant countermeasures.

To analyse the malware, the CCDCOE Malware Reverse Engineering Methodology was followed. The methodology outlined the different techniques that should be used for a successful analysis. These techniques included static analysis, disassembling, dynamic analysis, and network analysis. The malware was not copied out of the VM to prevent infection of the host system during analysis. This allowed the safe and secure analysis of the sample. Alongside this by using a Virtual Machine the analyst was able to revert to a “Snapshot” of the freshly booted virtual machine, where the malware was not yet executed to start analysis over again after executing the malware.

By successfully using the tools required for each technique the malware was identified, its behaviour was understood, it had its Indicators of Compromise documented and relevant countermeasures were discussed. It was determined that the malware was simplified version of the WannaCry ransomware with its worming capabilities removed, which meant it was not able to infect other machines on the same network.

.

Contents

.

1 INTRODUCTION

1.1 BACKGROUND

Malware, short for malicious software, is a term that describes malicious program or code that is harmful to computer systems, networks, or devices. There are several types of malwares including ransomware, spyware, worms, trojans and many more. (Malwarebytes, n.d.). Ransomware is a type of malware that has the goal of blocking access to a victim's computer and/or encrypting files until a ransom is paid to the malicious actor (National Cyber Security Centre, n.d.). The USA is the most affected by ransomware country in the world followed by the UK and Canada which place second and third. (Malwarebytes Threat Intelligence Team, 2023)

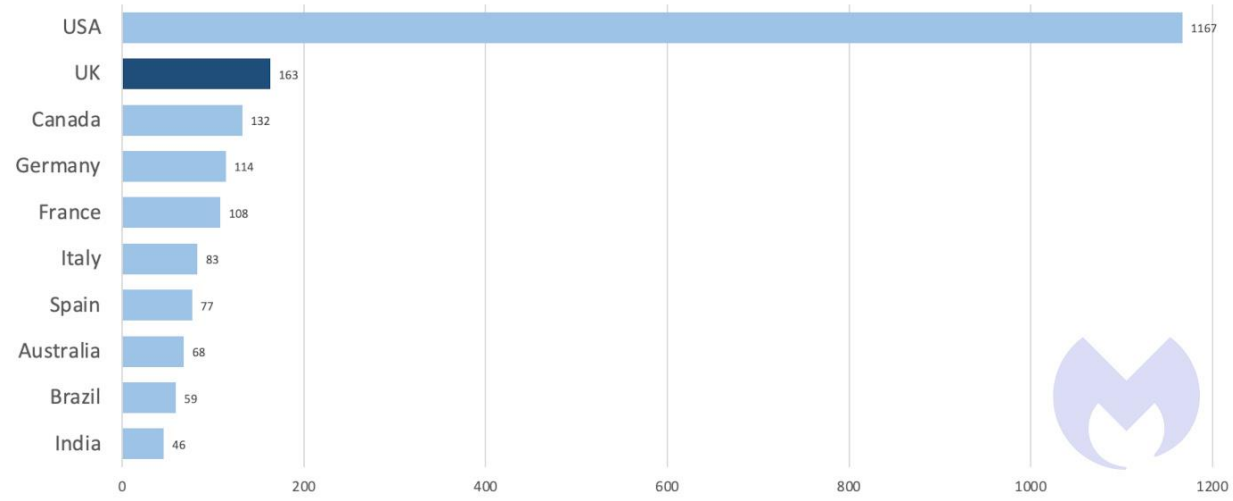


Figure 1.1. Ten most attacked countries (Malwarebytes Threat Intelligence Team, 2023)

In January 2023 ransomware affected Royal Mail in the UK and demanded the biggest ransom ever seen from a ransomware attack – $80 million (Malwarebytes Threat Intelligence Team, 2023). This beats the previous highest of $70 million in 2021 from the attack on the software company Kaseya (Clancy, 2021). It should be noted that ransom payments are just a fraction of the cost of ransomware attack. This is because on average, businesses need 22 days to recover from the ransomware attack which leads to significantly lower productivity. The lost productivity, cost of contractors to fix encountered issue and potential lawsuits which the company might experience if the breach has compromised sensitive customer information all add up. Alongside this the reputational damage also causes loss of revenue. The average ransom payment made is $1 million, while "According to a recent IBM study, the average cost of a ransomware attack is $4.62 million (not including the actual ransom payment)". (Blosil, 2022).

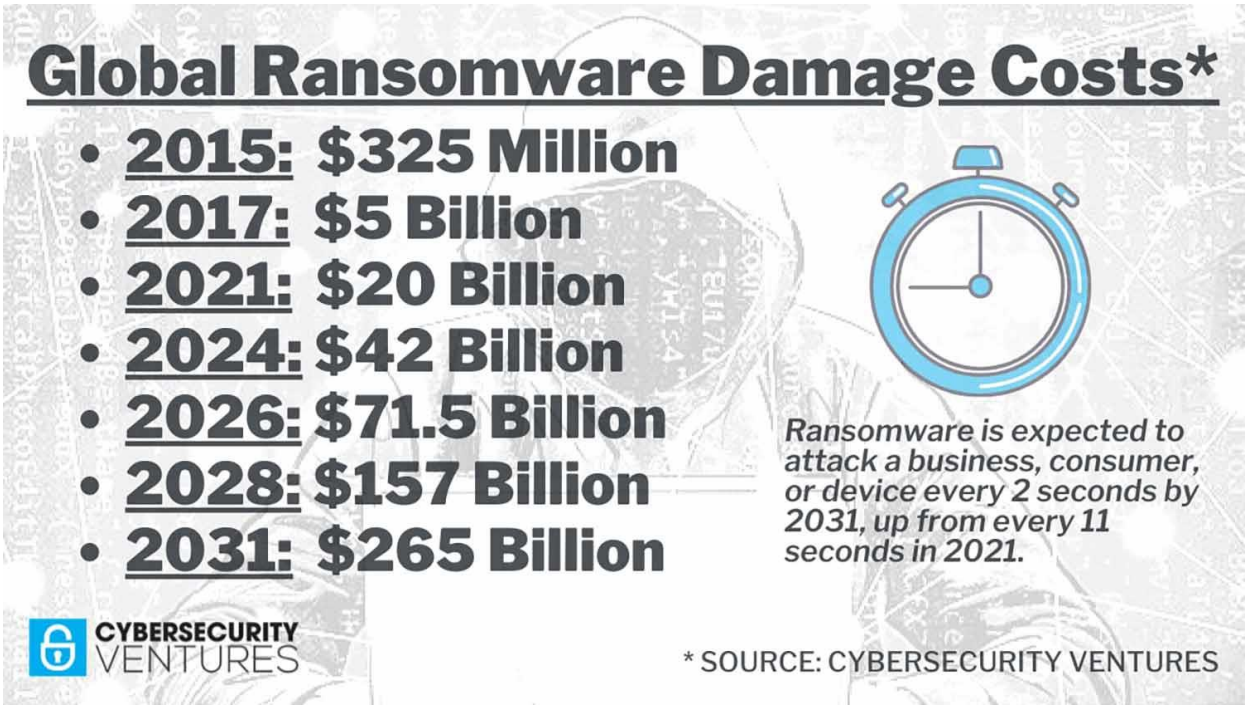


Figure 1.2. Global Ransomware Damage Costs (Braue, 2022)

Global ransomware costs in 2021 were $20 billion, but they keep increasing each year and are to reach $265 billion by 2031 (Figure 1.2). The massive increase of damages year by year shows that ransomware attacks are not going to disappear any time soon and are only going to become more severe year by year (Braue, 2022). Such malware will be examined for this report to determine how it functions and what countermeasures can be taken against it.

1.2 AIM

The aim of this report is to investigate a sample of a malware using appropriate malware analysis techniques. This overall aim has the following sub aims:

- Confirm the malware type.
- Analyse its characteristics.
- Identify components.
- Comprehend the operational behaviour of the malware.
- Discuss countermeasures.

This will be achieved using the following malware analysis methods:

- Static analysis
- Dynamic Analysis
- Disassembly
- Network Traffic Analysis

2 PROCEDURE AND RESULTS

2.1 OVERVIEW OF PROCEDURE

For this report, the CCDCOE Malware Reverse Engineering Handbook was used (CCDOE, 2020) to conduct successful analysis. It provides a framework of best practices for conducting a malware assessment. It was followed closely however certain parts were not performed due to their unsuitability.

The CCDCOE Malware Reverse Engineering Handbook contains the following sections:

- Static Malware Analysis – During static analysis, the malware's executable file is examined without running it. The information that is discovered can be used to determine whether the file is malicious.
- Disassembly – Disassemblers are used to explore the code of the program to get an understanding of what it does. This is done by translating the machine code into assembly language.
- Dynamic Malware Analysis - During dynamic malware analysis the malware is ran in a controlled environment to analyse its behaviour and identify its characteristics.
- Network Traffic Analysis – Network traffic analysis is used to determine the network traffic generated by the malware to understand its behaviour better and identify malicious activity.

Various tools were used to conduct the analysis. (Table 1).

Section used	Tool name	Version used	Reference
Static Malware Analysis	HashMyFiles	Version 2.43	(NirSoft, 2021)
	VirusTotal	Latest Version	(VirusTotal, 2023)
	Strings	Version 2.54	(Microsoft, 2021)
	Flare-Floss	Version 2.20	(Mandiant, 2023)
	PEiD	Version 0.95	(PEiD, 2018)
	CFF Explorer	CFF Explorer 8	(NTCore, n.d.)
	Resource Hacker	Version 5.1.7	(Johnson, 2019)
	PeStudio	Version 9.47	(Winitor, 2023)
Disassembly	IDA Free	Version 8.2	(Hex-rays, 2023)
	Ghidra	Version 10.2.3	(NSA, 2023)
Dynamic Malware Analysis	Process Monitor	Version 3.93	(Microsoft, 2023)
	Process Explorer	Version 17.04	(Microsoft, 2023)
	Regshot	Version 1.9.1	(Regshot, n.d.)
Network Traffic Analysis	Wireshark	Version 4.0.5	(Wireshark, n.d.)
	Fakenet-NG	Version 1.4.11	(Mandiant, 2020)

Table 1: Tools used for analysis.

2.2 STATIC MALWARE ANALYSIS

2.2.1 VirusTotal

First step was to get the MD5 hash of the sample file using HashMyFiles.

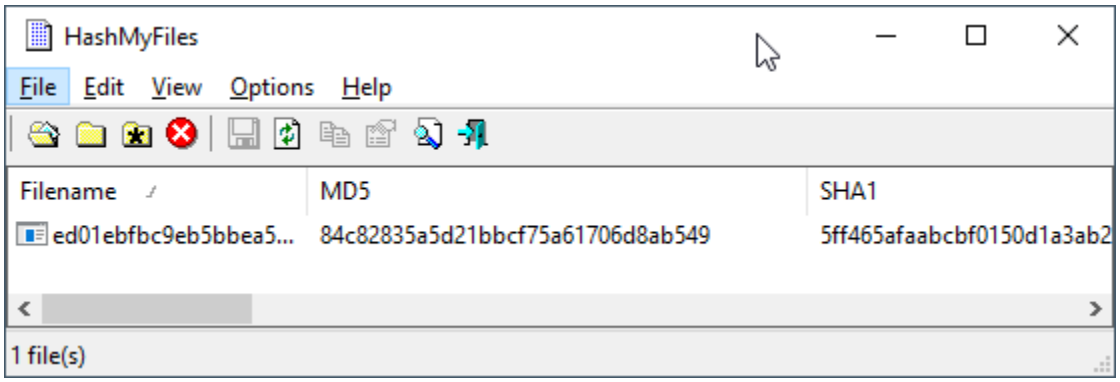


Figure 2.1. Getting MD5 hash using hash my files

The hash was calculated to be "84c82835a5d21bbcf75a61706d8ab549". It was copied and pasted into VirusTotal which gave the results that the file is malicious by cross-referencing it with different antivirus programs.

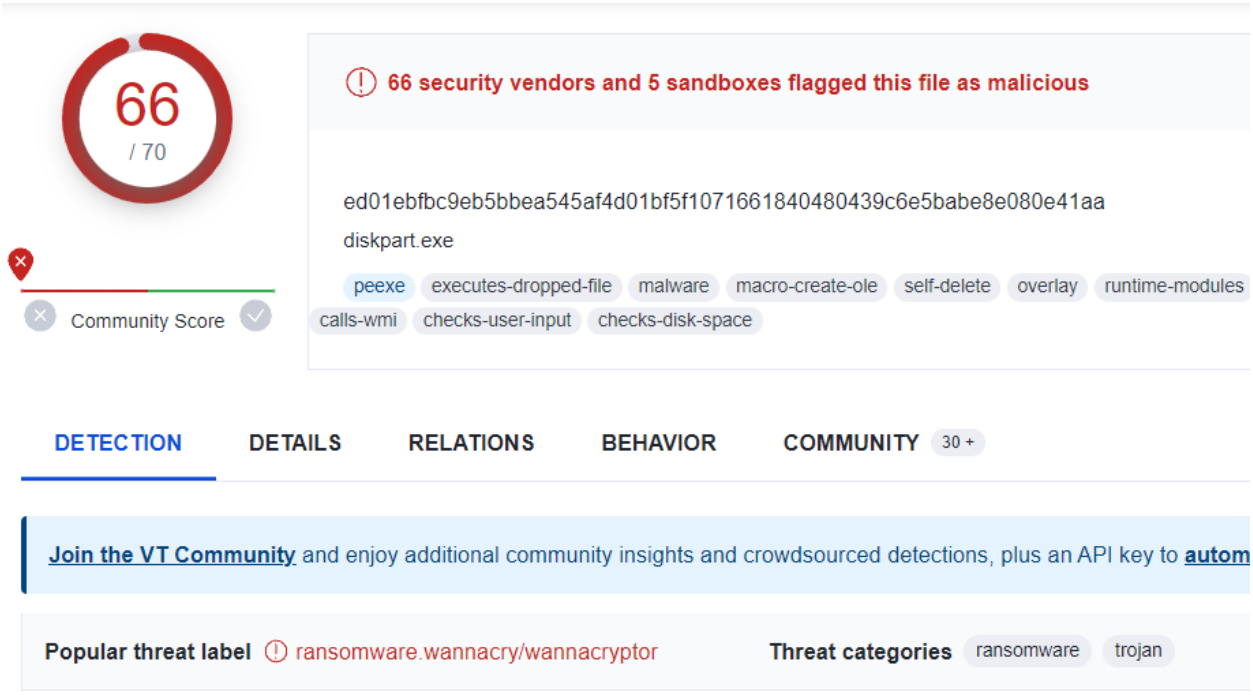


Figure 2.2: VirusTotal result

The malware was identified to be the ransomware WannaCry.

2.2.2 String analysis

Having confirmed that the file is indeed malware the analyst moved onto checking if it contains any useful strings. This is done by using the programs ”Strings” and “Flare-Floss”. They work by reading the binary for any readable Ascii and Unicode characters. Both programs were used successfully as can be seen in Figure 2.3 and Figure 2.4. The output from both programs is incredibly long so it was not included in the Appendices but can be provided upon request.

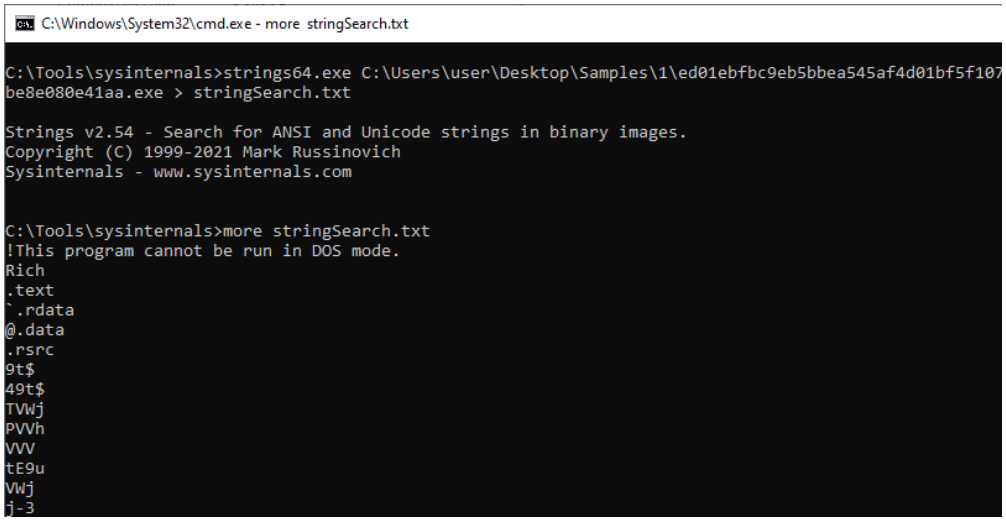


Figure 2.3. Searching for strings using "Strings"

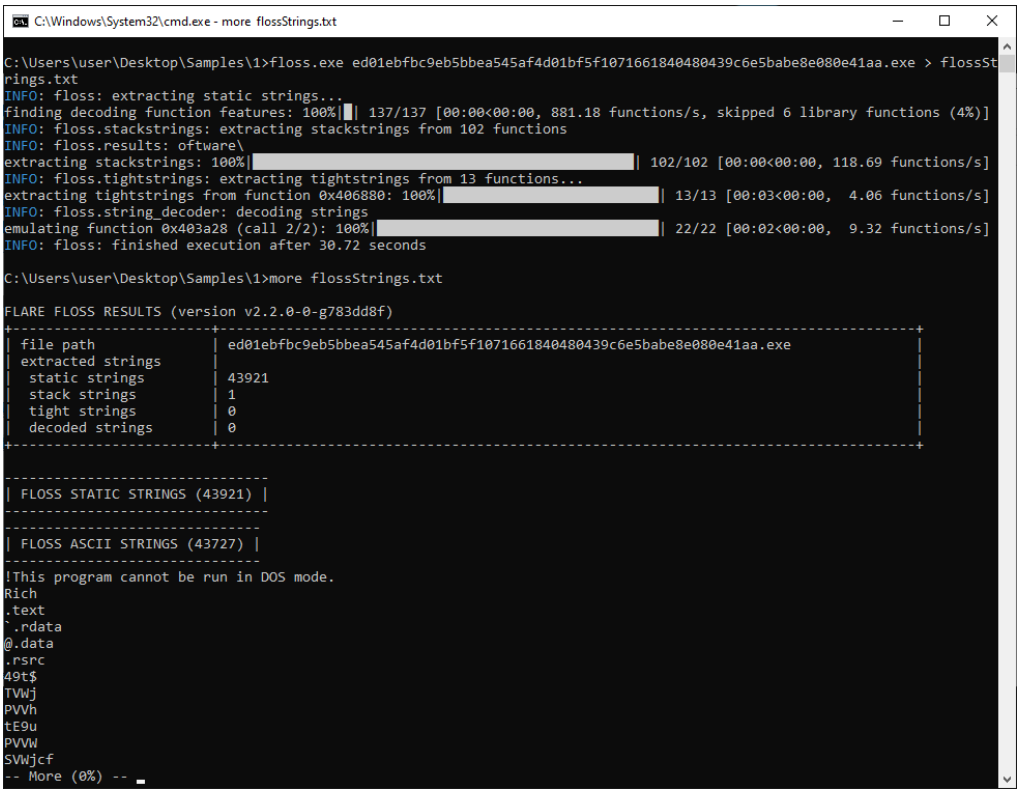


Figure 2.4: Searching for strings using "Flare-Floss”.

2.2.3 PEID Tool

PEID Tool is used to determine several types of information such as whether the file is packed and what compilers was used. It does that be analysing the PE file header and comparing it to the ones contained it its database.

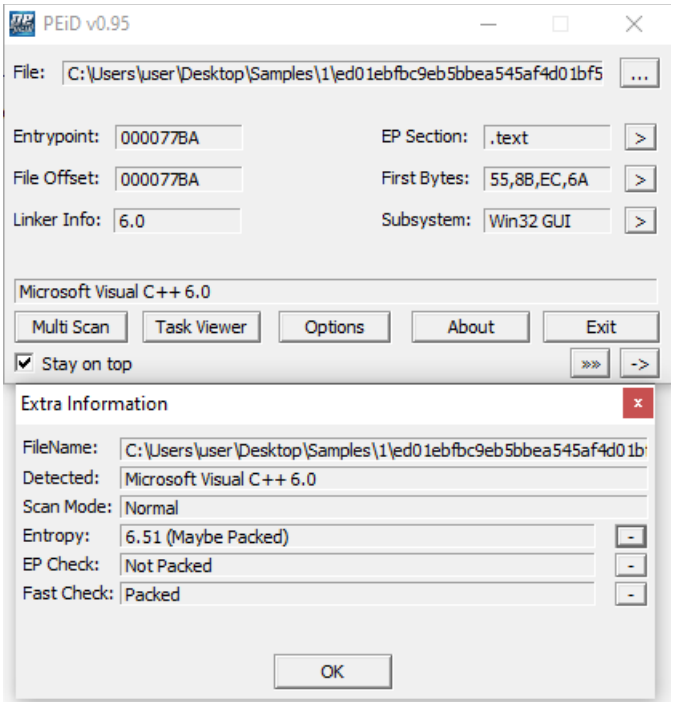


Figure 2.5: PEiD Scan Result

In Figure 2.5 it can be seen that the compiler that was used was Microsoft Visual C++ 6.0. EP Check shows the file as "Not Packed", however the Fast Check shows "Packed". Fast Check is more inaccurate and could be giving a false positive but considering the high entropy value of "6.51" the analyst assumed that the executable is using some kind of obfuscation to hide whether it is packed. Because of that evidence further analysis will have to be performed.

2.2.4 CFF Explorer

Using CFF explorer for malware analysis allows the extraction of valuable information such as compilation date and architecture type from the malware sample. This information is retrieved by looking inside the PE Header of the file. Using PE Header, the analyst was able to confirm some of the information discovered earlier such as the MD5 hash and compiler information. It was also discovered that the file appeared to present itself as genuine "Microsoft Corporation program" called "DiskPart" (Figure 2.6). Following this the File Header was examined (Figure 2.7). It was found that the malware has the "TimeDateStamp" value of "4CE78F41". This is in Epoch Unix Time, so it had to be converted to a human readable date. (Figure 2.8)

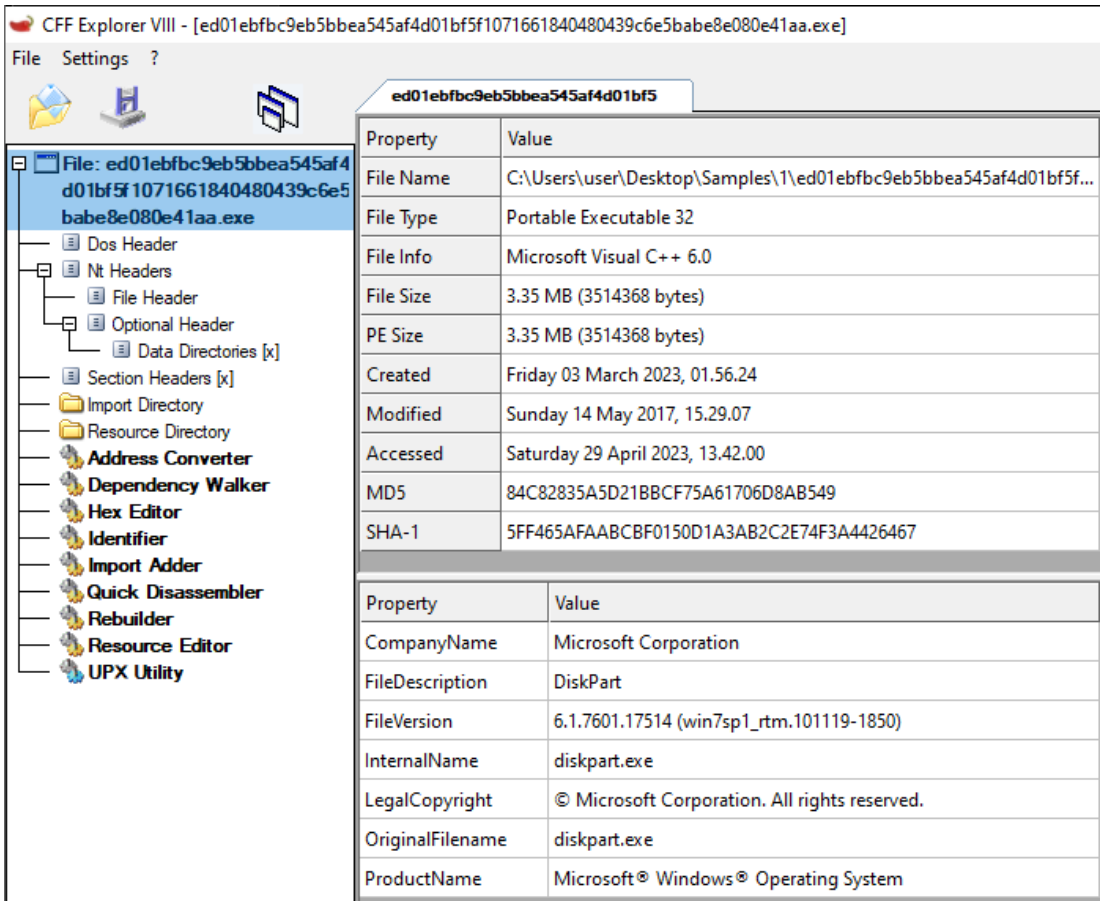


Figure 2.6: CFF Explorer basic file information

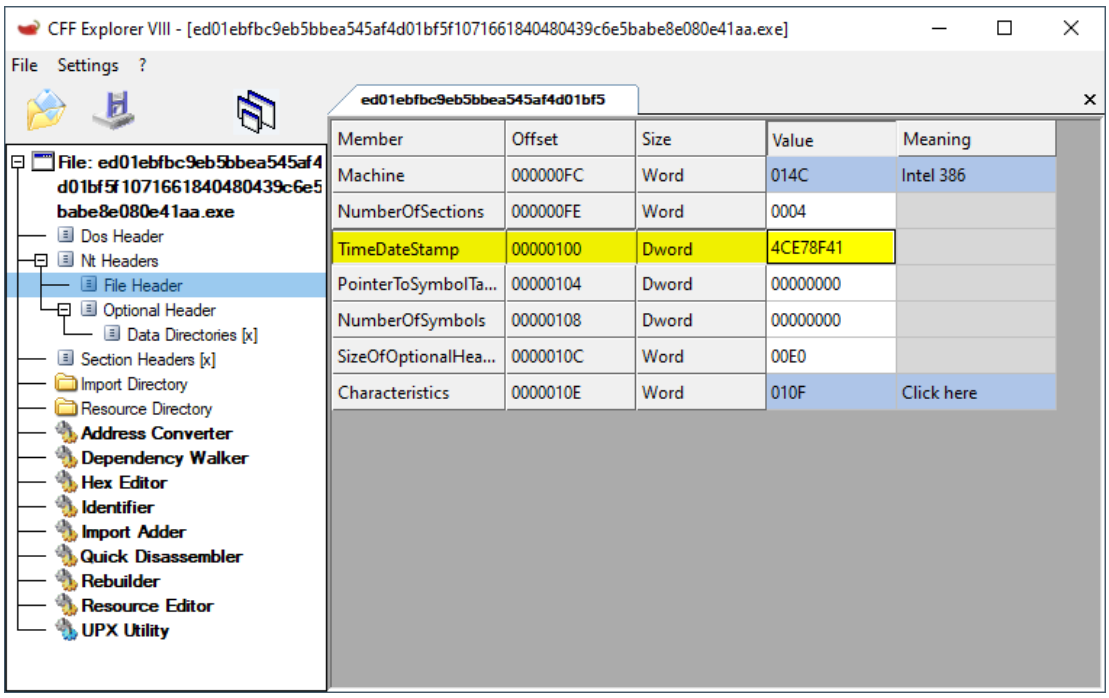


Figure 2.7: TimeDateStamp of malware

Figure 2.8: Converting epoch to human-readable date using (EpochConverter, 2023)

From the output of the website https://www.epochconverter.com/ (EpochConverter, 2023). It was determined that the file was created on 20th of November 2010.

Finally, the analyst moved onto examining the Section Headers of the file (Figure 2.9). Based on the fact the section names are not changed from their default values (.text, .rdata, .data, .rsrc) it was determined that the file is indeed not packed.

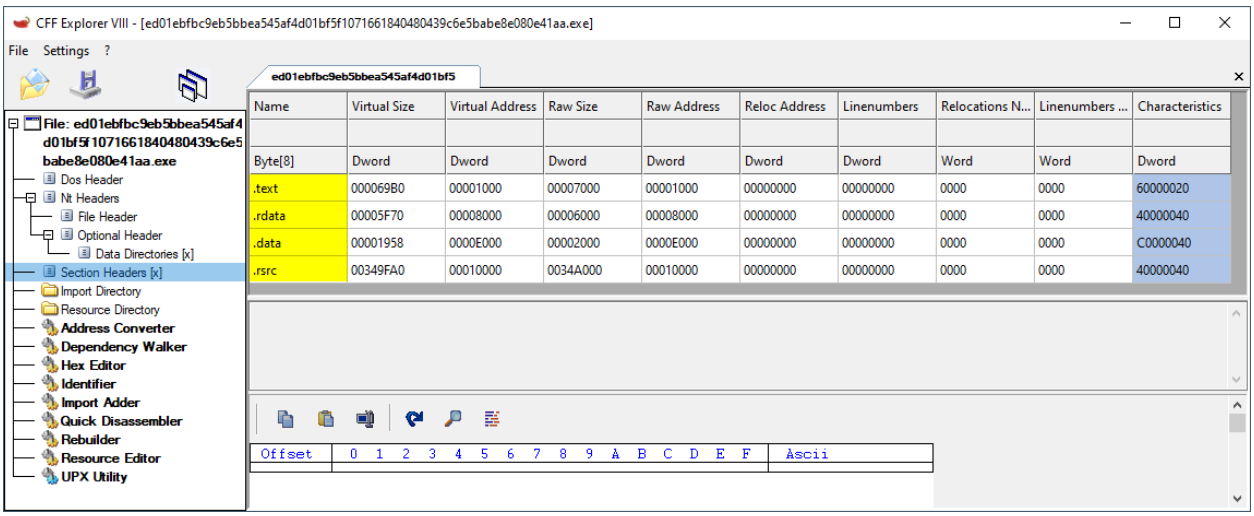


Figure 2.9: Section Headers

2.2.5 Resource Hacker

Resource hacker is a tool that can be used to change the functionality and appearance of an application by modifying or adding resources to Windows binaries. It can also be used for extracting files from the executable file. It is useful for malware analysis to examine the resources found within the malware and potentionally extracting useful file. Using the tool the analyst was able to see an interesting file called XIA. In its hex output it can be seen that it has the magic number 50 4b 03 04, which indicates it is a .zip file. (Figure 2.10)

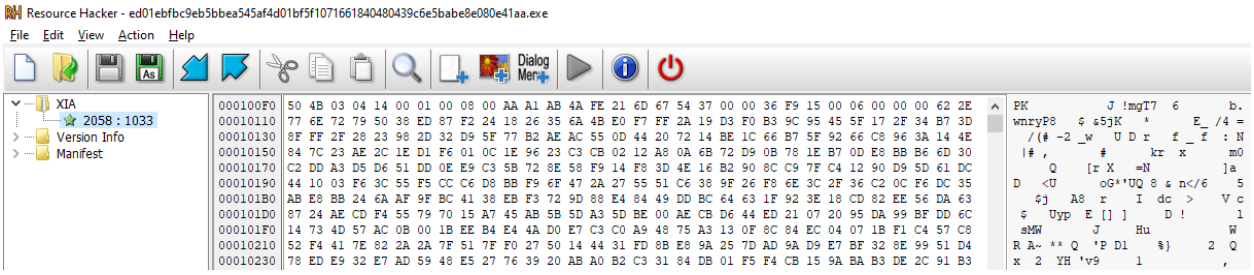


Figure 2.10: XIA bin file

The file was saved and was attempted to be extracted; however, it was password locked which led to the extraction failing. (Figure 2.11, Figure 2.12 and Figure 2.13)

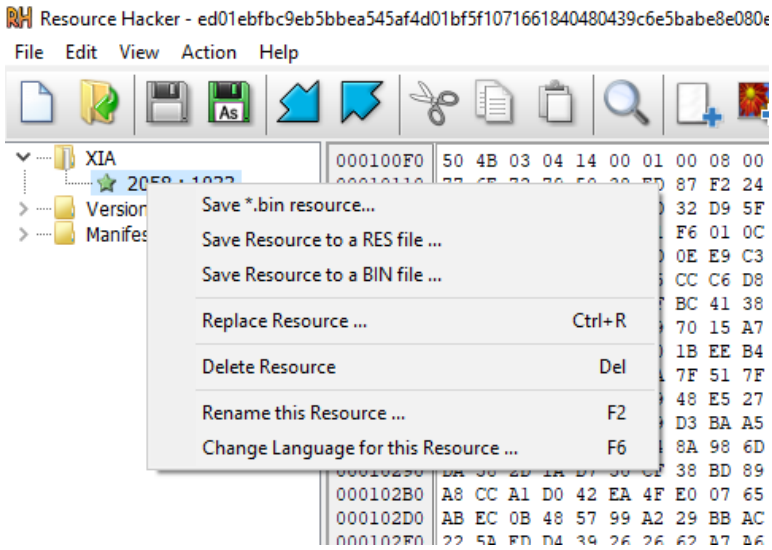


Figure 2.11: Saving File

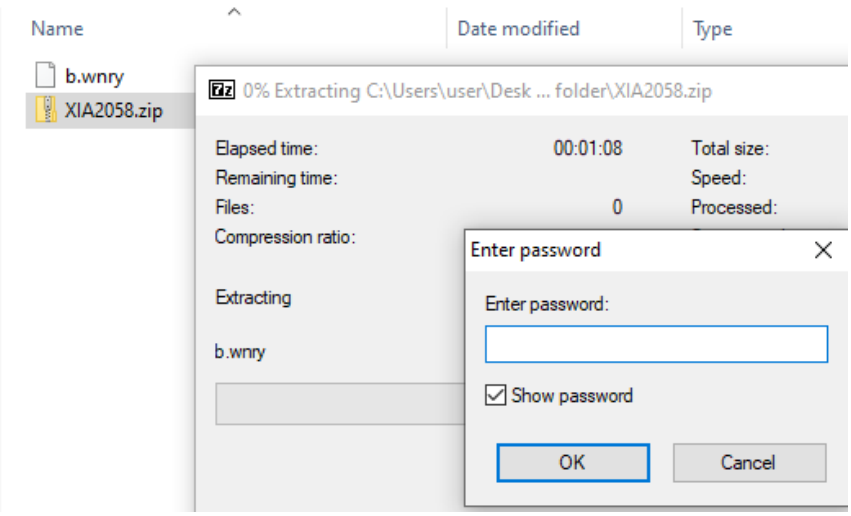


Figure 2.12: Password prompt

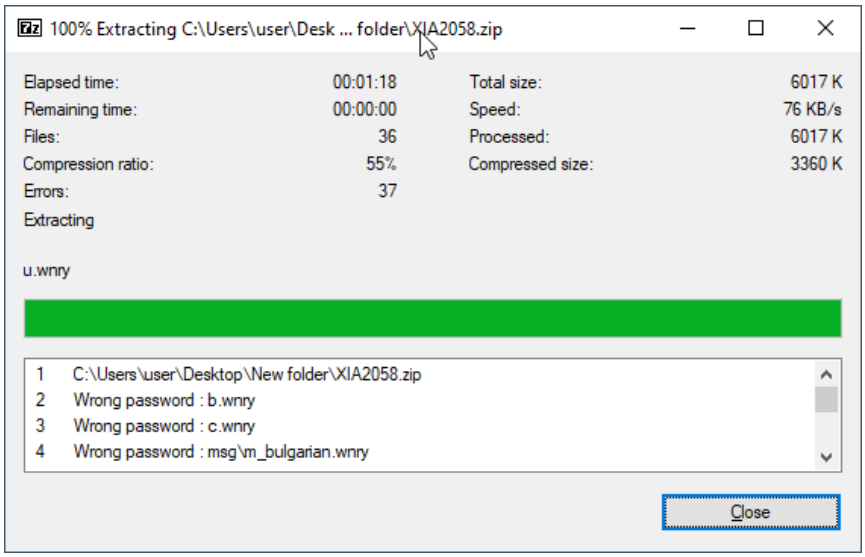


Figure 2.13: Extract failing.

Next step was to examine the "Version info". The output contains the value which makes it identify as "diskpart.exe" and as a legitimate "Microsoft Corporation" program. (Figure 2.14). However, based on the information from the VirusTotal scan we know that the program is not legitimate Microsoft application, because it has not been signed (Appendix A – VirusTotal signature info)

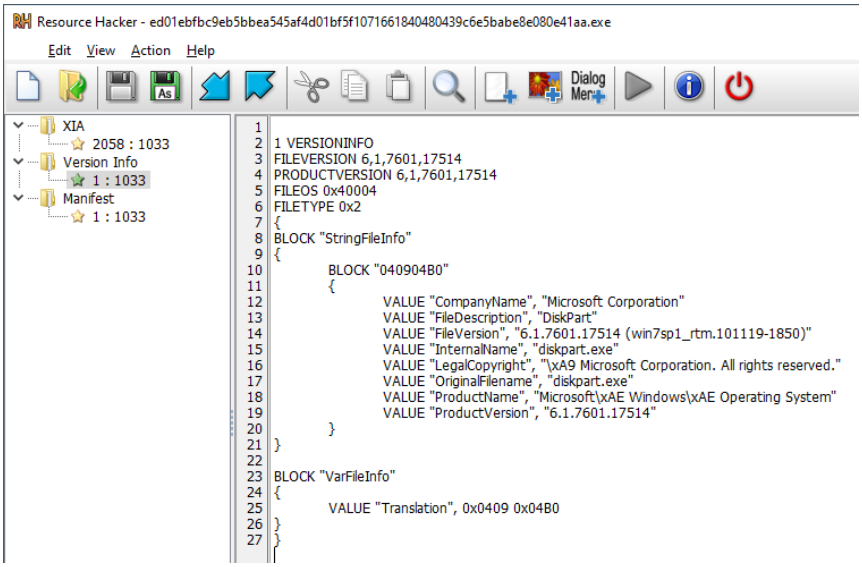


Figure 2.14: Version info of file

The contents of the "Manifest" show that the file requests to be executed with the "asInvoker" execution level, which means it wants to run with the privilege of the user who starts it. This is the normal execution level for most applications. (Figure 2.15)

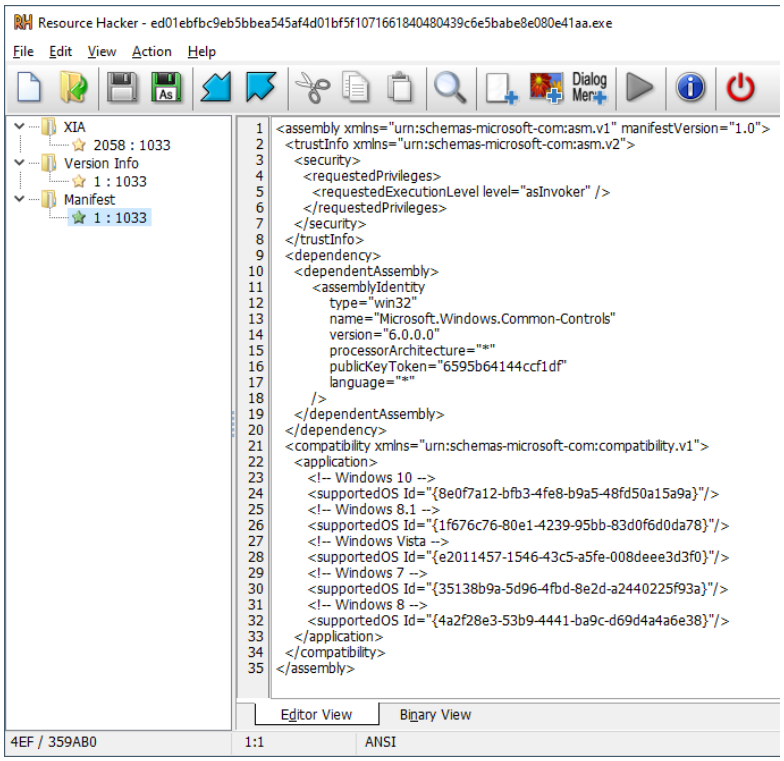


Figure 2.15: Manifest of file

2.2.6 PeStudio

PeStudio can be used by the analyst to find suspicious functionalities in the executable. In the imports section of the file there were fourteen imports flagged as suspicious (Figure 2.16):

- CreateServiceA – used for the creation of a service object and its addition to the service control manager database. Could potentially be getting used for persistence.
- RegCreateKeyW – used for the creation of registry keys.
- RegSetValueExA – used for the editing of registry keys.
- WriteFile – used for writing date into files.
- SetFileAttributesW – used for changing file attributes, such as whether a file is hidden or read only.
- TerminateProcess – used for ending a process and all its threads.
- GetExitCodeProcess – used to determine whether a file has successfully terminated by retrieving its termination status.
- rand/srand– used for generating random numbers.
- SetCurrentDirectoryW/SetCurrentDirectoryA – used for changing directories.

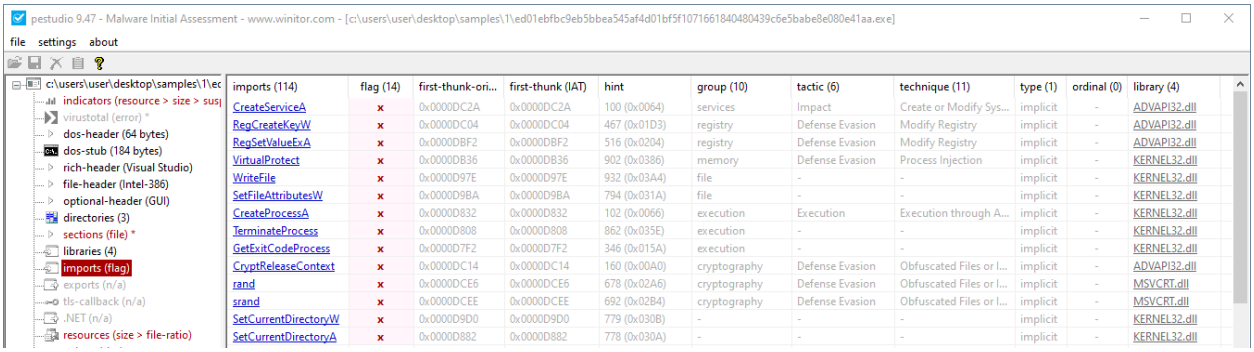


Figure 2.16: Suspicious imports

The next step was to examine the “strings”. By analysing its contents, we can see that PeStudio automatically flagged thirty-two suspicious strings. From the flagged strings it could be seen there was some related to encryption (CryptGenKey, CryptDecrypt, CryptEncrypt, CryptDestroyKey, CryptImportKey, CryptAcquireContext) which gives more evidence that the type of malware being analysed could be ransomware.

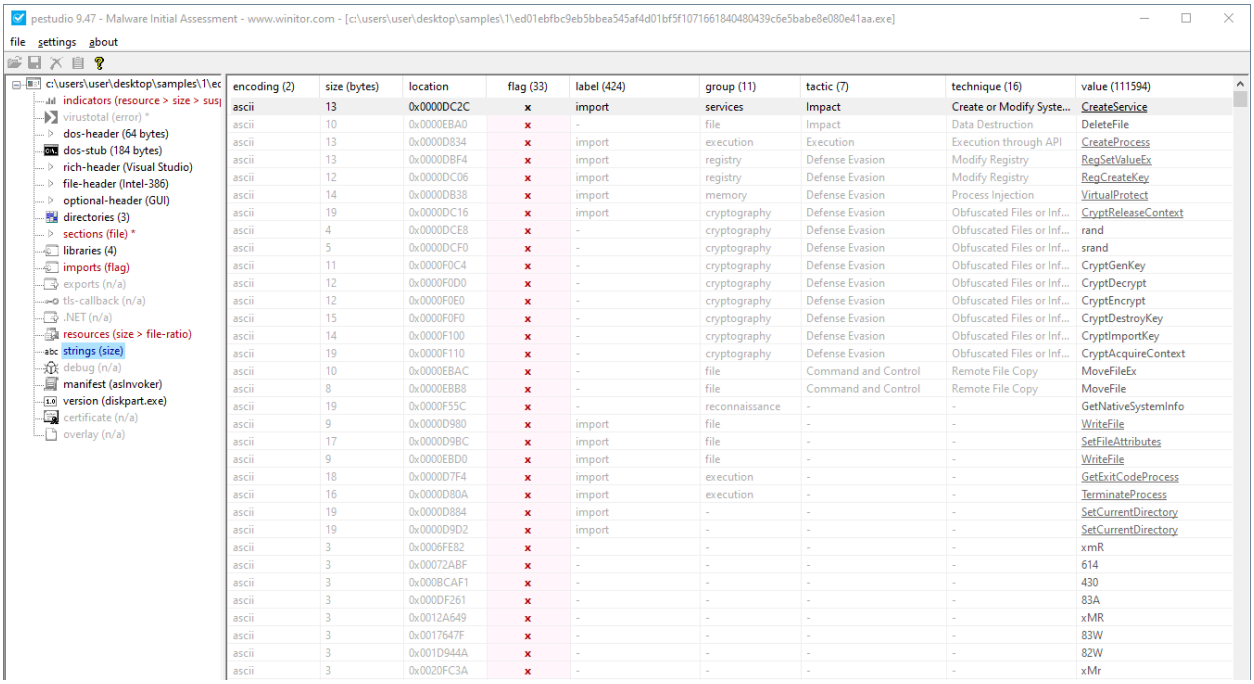


Figure 2.17: Suspicious strings

2.3 DISASSEMBLY (IDA & GHIDRA)

2.3.1 IDA free

Using IDA, the code of the program can be examined. In Figure 2.18 the malware potentially has code that is used to encrypt data.

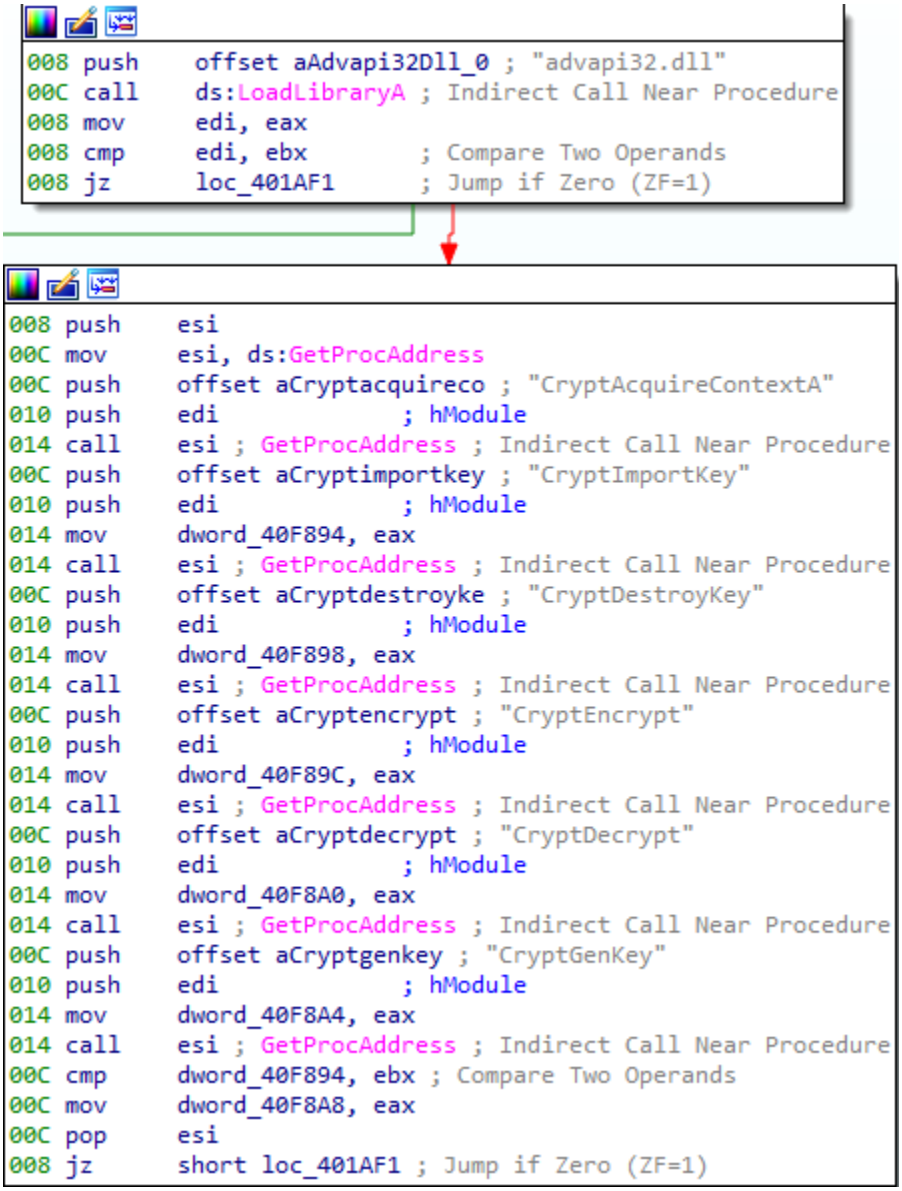


Figure 2.18: IDA Cryptography

There also appears to be code related to file manipulation that could potentially be used to delete files after they have been encrypted. (Figure 2.19)

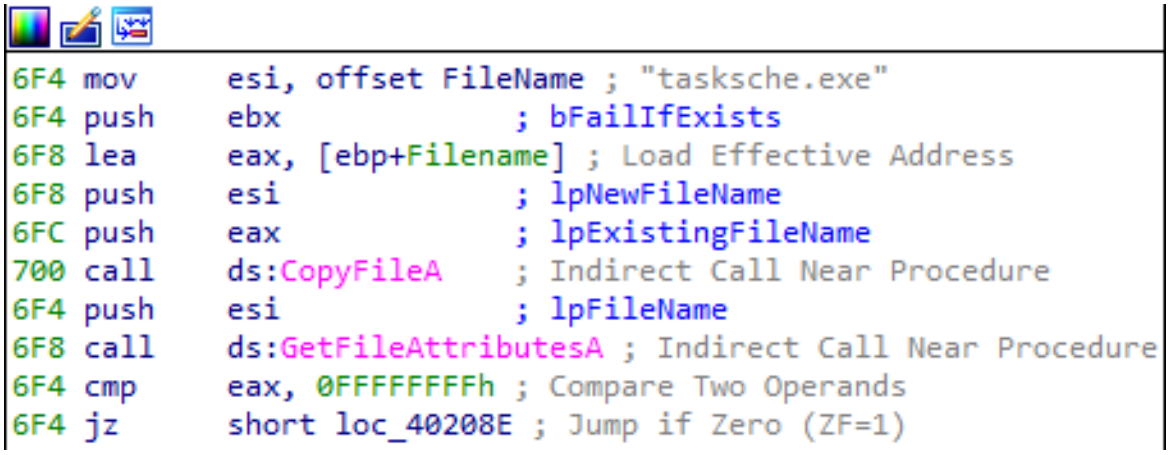
```
push    esi
mov     esi, ds:GetProcAddress
push    offset ProcName ; "CreateFileW"
push    edi             ; hModule
call    esi ; GetProcAddress
push    offset aWritefile ; "WriteFile"
push    edi             ; hModule
mov     dword_40F878, eax
call    esi ; GetProcAddress
push    offset aReadfile ; "ReadFile"
push    edi             ; hModule
mov     dword_40F87C, eax
call    esi ; GetProcAddress
push    offset aMovefilew ; "MoveFileW"
push    edi             ; hModule
mov     dword_40F880, eax
call    esi ; GetProcAddress
push    offset aMovefileexw ; "MoveFileExW"
push    edi             ; hModule
mov     dword_40F884, eax
call    esi ; GetProcAddress
push    offset aDeletefilew ; "DeleteFileW"
push    edi             ; hModule
mov     dword_40F888, eax
call    esi ; GetProcAddress
push    offset aClosehandle ; "CloseHandle"
push    edi             ; hModule
mov     dword_40F88C, eax
call    esi ; GetProcAddress
cmp     dword_40F878, ebx
mov     dword_40F890, eax
pop     esi
jz      short loc_4017D8
```

Figure 2.19: File manipulation

In Figure 2.20 it is visible that there is some sort of service creation that could potentially be for persistence. Moreover in Figure 2.21 there is code for creating a process called “tasksche.exe”.

```
loc_401D45:
push    [ebp+arg_0]
lea     eax, [ebp+Buffer]
push    offset Format   ; "cmd.exe /c \"%s\""
push    eax             ; Buffer
call    ds:sprintf
add     esp, 0Ch
lea     eax, [ebp+Buffer]
push    edi             ; lpPassword
push    edi             ; lpServiceStartName
push    edi             ; lpDependencies
push    edi             ; lpdwTagId
push    edi             ; lpLoadOrderGroup
push    eax             ; lpBinaryPathName
push    1               ; dwErrorControl
push    2               ; dwStartType
push    10h             ; dwServiceType
push    ebx             ; dwDesiredAccess
push    esi             ; lpDisplayName
push    esi             ; lpServiceName
push    [ebp+hSCManager] ; hSCManager
call    ds:CreateServiceA
mov     esi, eax
cmp     esi, edi
jz      short loc_401D98
```

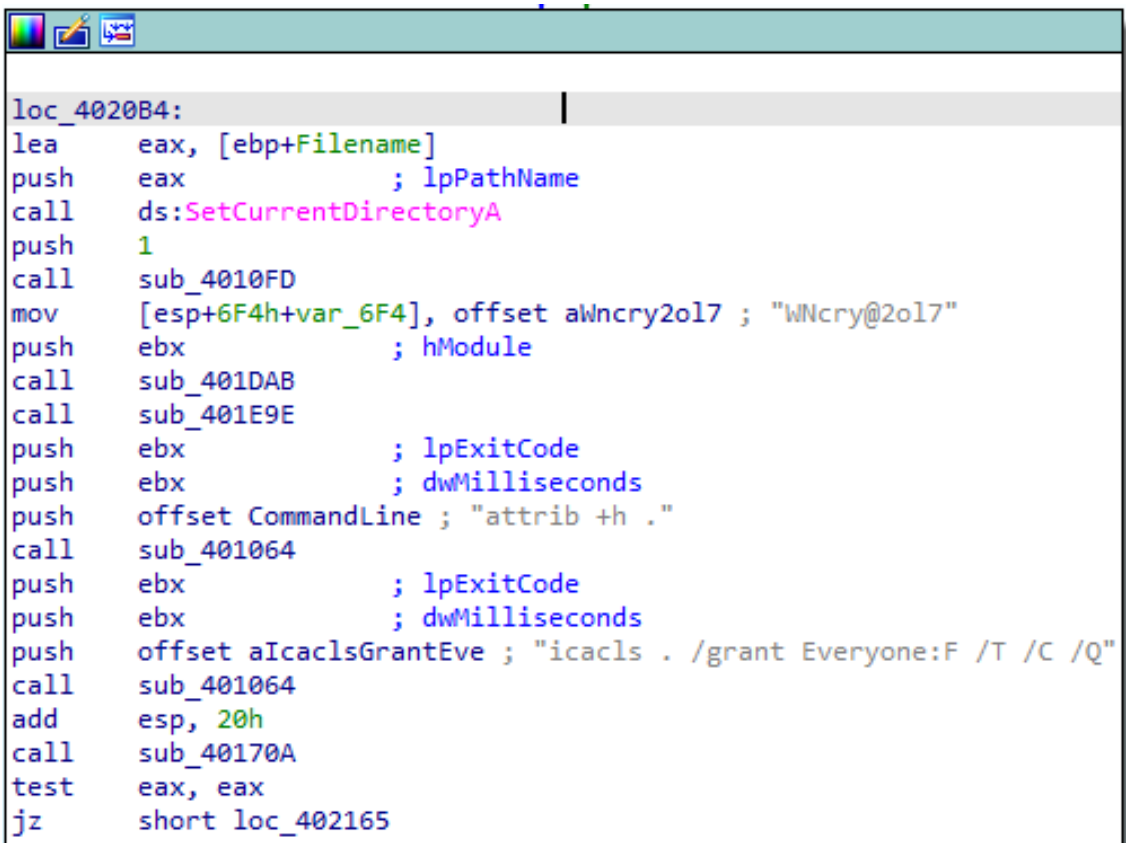
Figure 2.20: IDA service creation



```
6F4 mov     esi, offset FileName ; "tasksche.exe"
6F4 push    ebx                  ; bFailIfExists
6F8 lea     eax, [ebp+Filename] ; Load Effective Address
6F8 push    esi                  ; lpNewFileName
6FC push    eax                  ; lpExistingFileName
700 call    ds:CopyFileA         ; Indirect Call Near Procedure
6F4 push    esi                  ; lpFileName
6F8 call    ds:GetFileAttributesA ; Indirect Call Near Procedure
6F4 cmp     eax, 0FFFFFFFFh ; Compare Two Operands
6F4 jz      short loc_40208E ; Jump if Zero (ZF=1)
```

Figure 2.21: IDA "tasksche.exe"

After “tasksche.exe” it appears to do some operation using the string “WNcry@2ol7” (Figure 2.22).



```
loc_4020B4:
lea     eax, [ebp+Filename]
push    eax             ; lpPathName
call    ds:SetCurrentDirectoryA
push    1
call    sub_4010FD
mov     [esp+6F4h+var_6F4], offset aWncry2ol7 ; "WNcry@2ol7"
push    ebx             ; hModule
call    sub_401DAB
call    sub_401E9E
push    ebx             ; lpExitCode
push    ebx             ; dwMilliseconds
push    offset CommandLine ; "attrib +h ."
call    sub_401064
push    ebx             ; lpExitCode
push    ebx             ; dwMilliseconds
push    offset aIcaclsGrantEve ; "icacls . /grant Everyone:F /T /C /Q"
call    sub_401064
add     esp, 20h
call    sub_40170A
test    eax, eax
jz      short loc_402165
```

Figure 2.22: Malware performing operations with an interesting string.

The analyst decided to use that string to attempt to unzip the archive discovered using Resource Hacker in Section 2.2.5. This proved successful as can be seen in Figure 2.23, Figure 2.24 and Figure 2.25. Because of this discovery the analyst determined that the code in Figure 2.22 is used for extracting the archive.

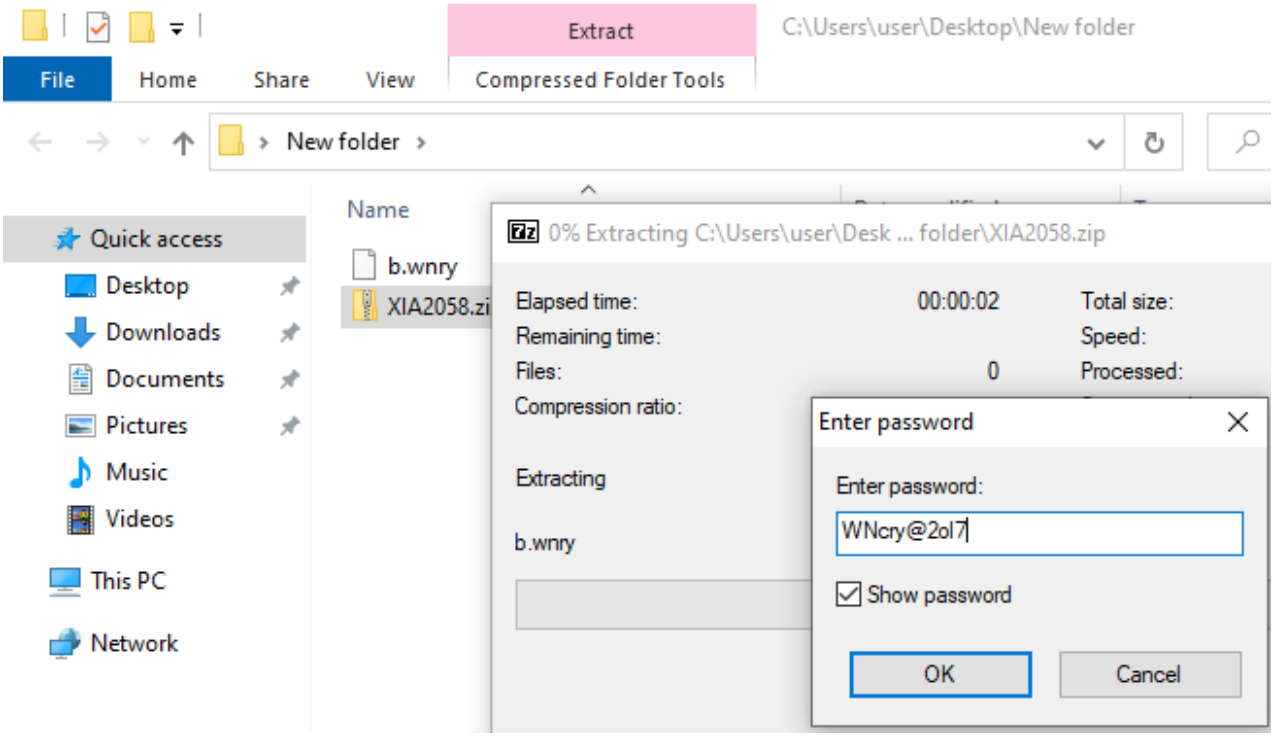


Figure 2.23: Unzipping of archive using discovered string.

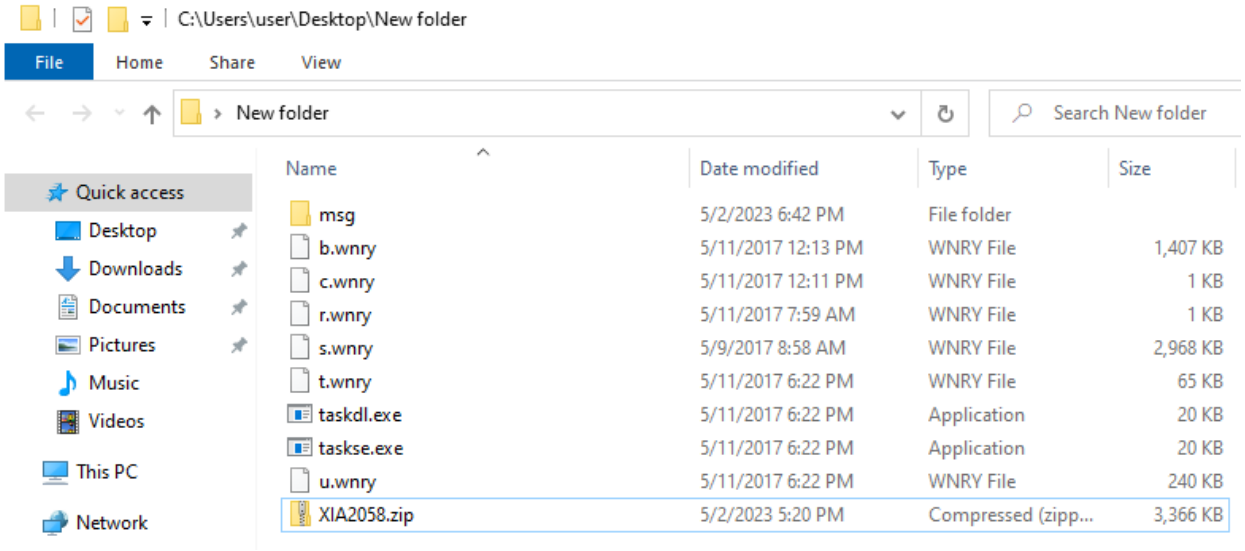


Figure 2.24: Successfully unzipped archive.

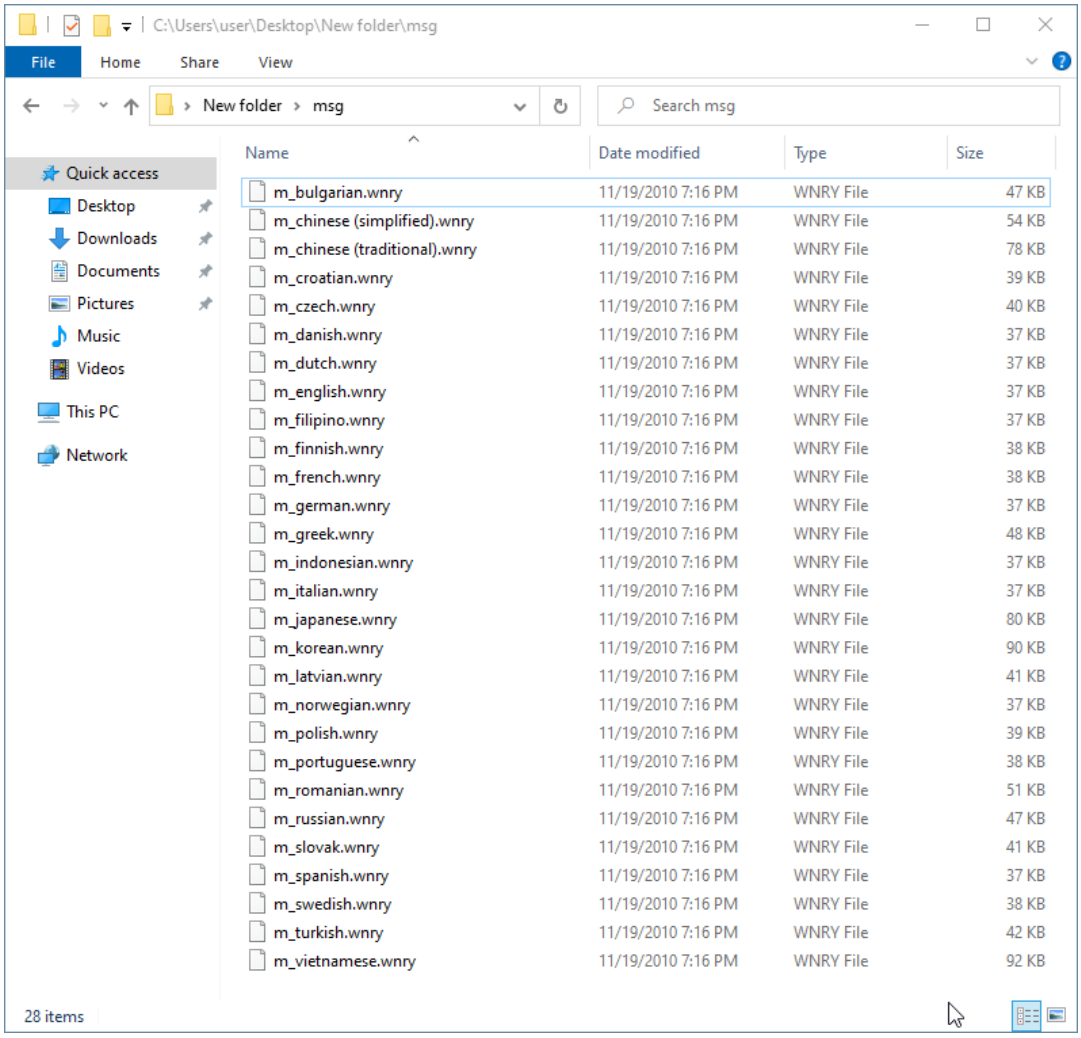


Figure 2.25: Contents of msg folder.

The files that were extracted from the archive were then examined by the analyst. The file called c.wnry appeared to contain some .onion addresses. (Figure 2.26)

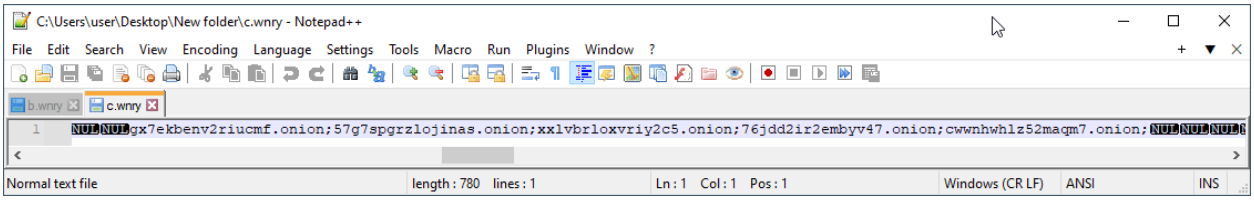


Figure 2.26: Onion address followed inside c.wrny

The file r.wnry contained a ransom message, which confirms that the malware is indeed ransomware. While the files inside the “msg” folder contained the ransom message in different languages. (Figure 2.27 and Figure 2.28)

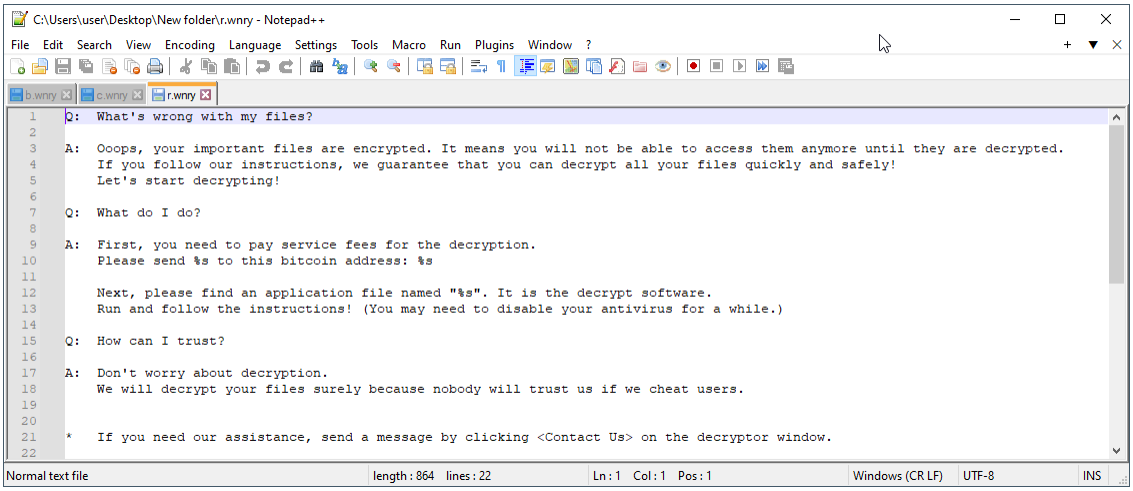


Figure 2.27: Ransom message founds inside r.wnry

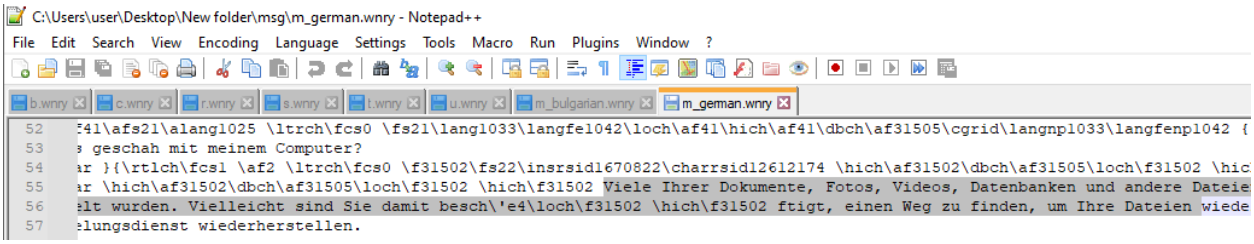


Figure 2.28: Reading of file contained inside msg folder.

The rest of the files did not contain anything of interest, so the analyst returned to analysing the assembly code in IDA Free.

After extracting the archive, the code runs the subfunction 401E9E. Figure 2.29

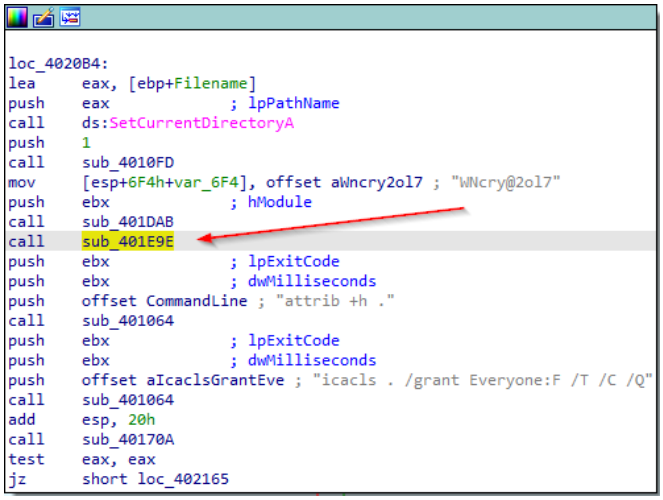


Figure 2.29: Subfunction being ran.

The subfunction appear to read/write 3 random 20-character strings (Figure 2.30):

- 13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94
- 12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
- 115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn

```
    ; Attributes: bp-based frame

    sub_401E9E proc near

    Buffer= byte ptr -318h
    Destination= byte ptr -266h
    Source= dword ptr -0Ch
    var_8= dword ptr -8
    var_4= dword ptr -4

000 push    ebp
004 mov     ebp, esp
004 sub     esp, 318h       ; Integer Subtraction
31C lea     eax, [ebp+Buffer] ; Load Effective Address
31C push    1               ; int
320 push    eax             ; Buffer
324 mov     [ebp+Source], offset a13am4vw2dhxygx ; "13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94"
324 mov     [ebp+var_8], offset a12t9ydpgwuez9n ; "12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw"
324 mov     [ebp+var_4], offset a115p7ummngoj1p ; "115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn"
324 call    sub_401000      ; Call Procedure
324 pop     ecx
320 test    eax, eax        ; Logical Compare
320 pop     ecx
31C jz      short locret_401EFD ; Jump if Zero (ZF=1)
```

Figure 2.30: Subfunction 401E9E

After it successfully completes that operation the malware runs the command “append +h” which after doing research the analyst determined was used to make a folder hidden and it uses “icacls . /grant Everyone:F /T /C /Q” to grant everyone full access to all files and folders in the directory Figure 2.31.

```
loc_4020B4:
lea     eax, [ebp+Filename]
push    eax             ; lpPathName
call    ds:SetCurrentDirectoryA
push    1
call    sub_4010FD
mov     [esp+6F4h+var_6F4], offset aWncry2ol7 ; "WNcry@2ol7"
push    ebx             ; hModule
call    sub_401DAB
call    sub_401E9E
push    ebx             ; lpExitCode
push    ebx             ; dwMilliseconds
push    offset CommandLine ; "attrib +h ."
call    sub_401064
push    ebx             ; lpExitCode
push    ebx             ; dwMilliseconds
push    offset aIcaclsGrantEve ; "icacls . /grant Everyone:F /T /C /Q"
call    sub_401064
add     esp, 20h
call    sub_40170A
test    eax, eax
jz      short loc_402165
```

Figure 2.31: Changing of file attributes.

2.3.2 Ghidra

Ghidra was used by the analyst, but no new data was discovered. (Figure 2.32)

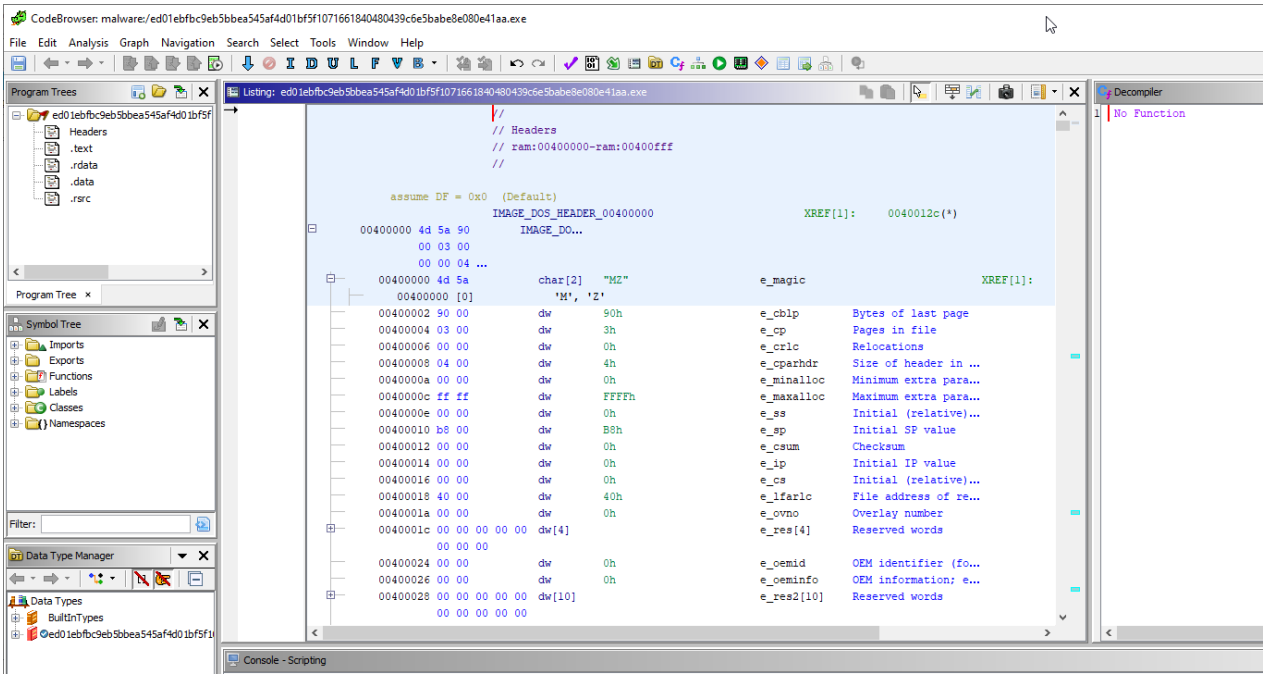


Figure 2.32: Examining malware with Ghidra.

2.4 Dynamic Analysis

2.4.1 Behaviour analysis tools

2.4.1.1 Process Monitor

Process Monitor is a tool used for monitoring the creation or termination of processes and allows the analyst to get information about the actions performed by each process, such as thread creation, file system and registry operations.

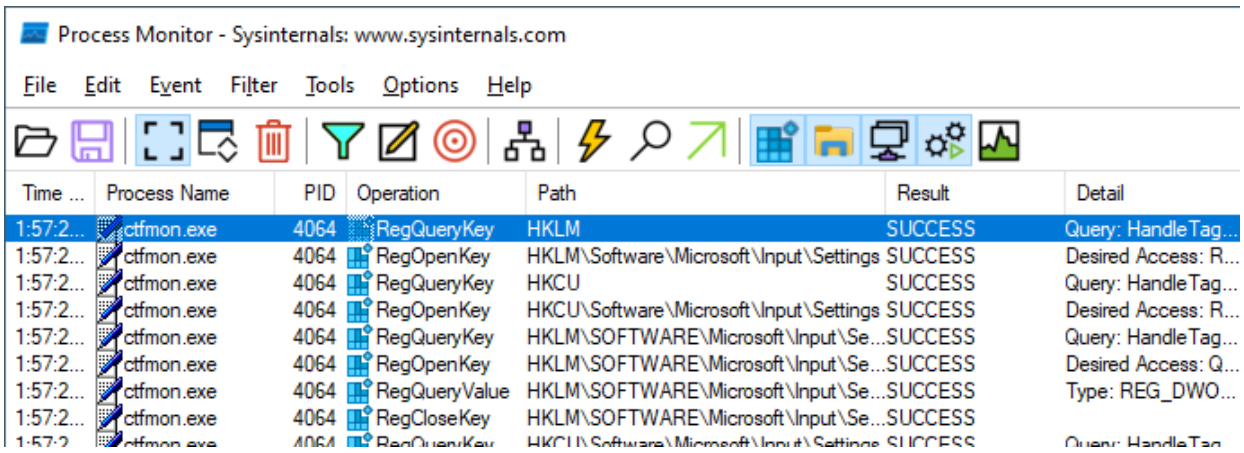


Figure 2.33: Process monitor capturing traffic.

After process monitor was started the malware was executed to examine the operations it performs.

The program initially starts off with loading its DLL's it needs to function. Then it creates the HKLM\Software\WanaCrypt0r registry key and sets the HKLM\Software\WanaCrypt0r\wd registry key to the current directory. After that it starts to extract the XIA files to the current directory. Figure 2.34 and Figure 2.35.

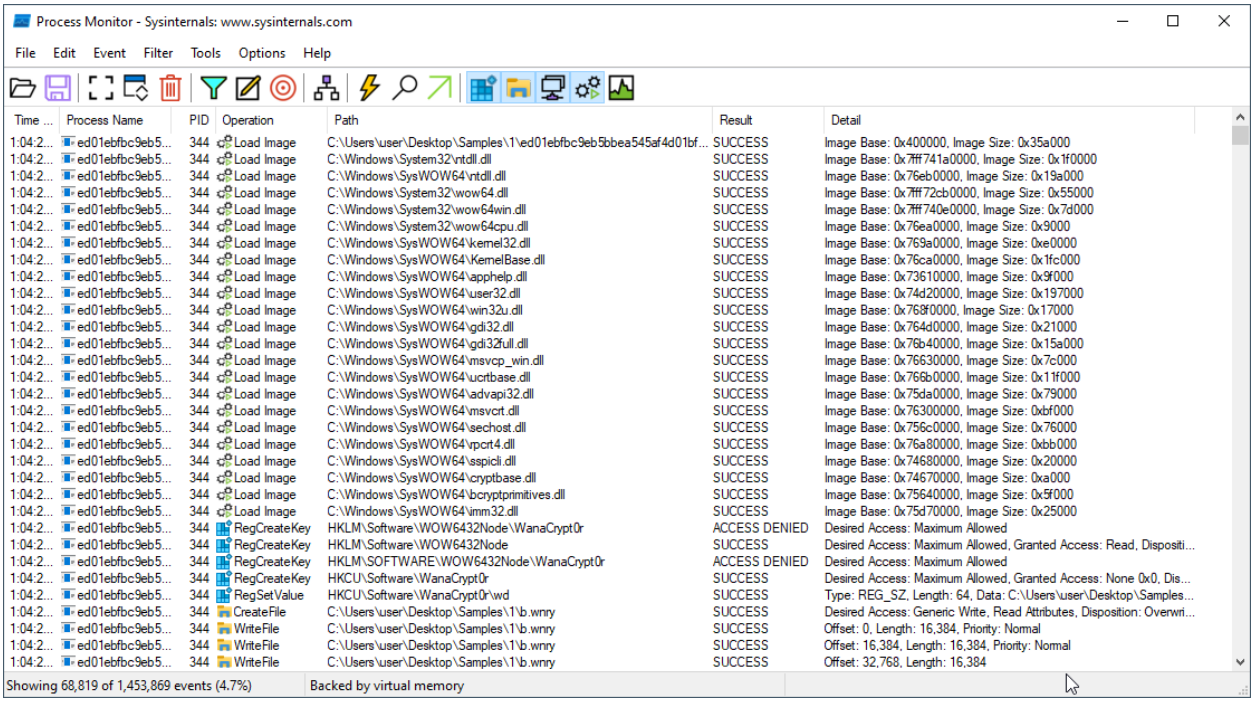


Figure 2.34: Malware start.

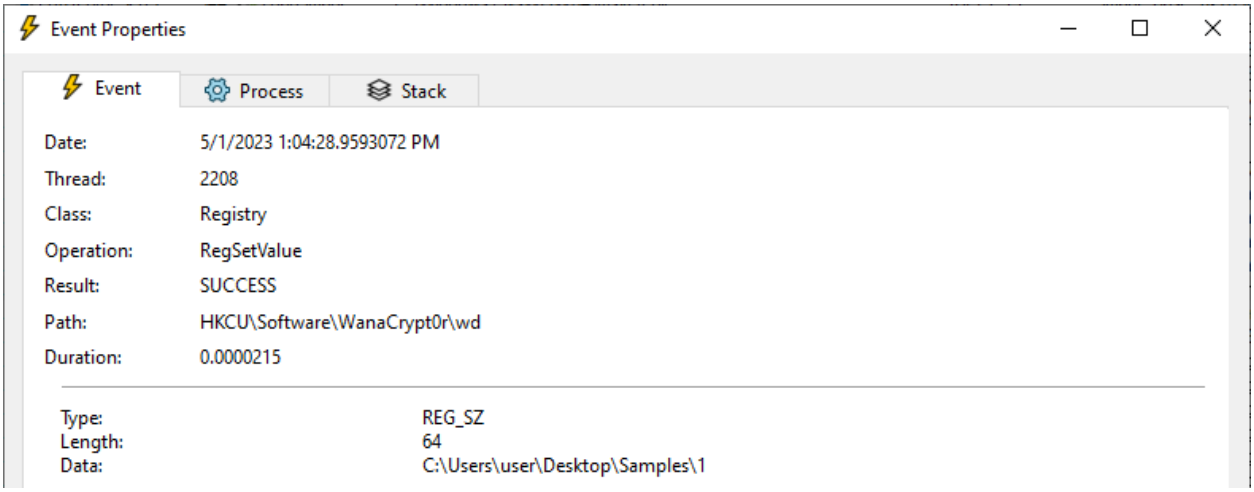


Figure 2.35: Editing HKCU\Software\WanaCrypt0r\wd registry.

The next step the malware takes is to hide its working directory by executing attrib +h. (Figure 2.36)

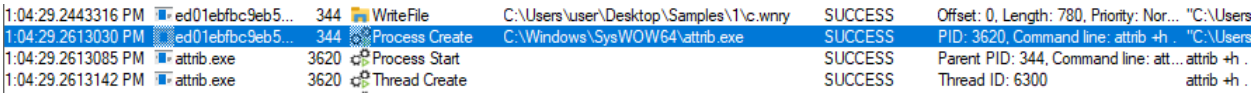


Figure 2.36: Hiding working directory.

Once it is done exporting all the files the malware then grans all user permissions in the current directory and all its subdirectories

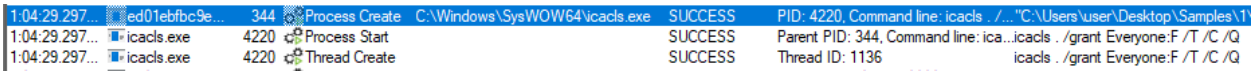


Figure 2.37: Granting all user permissions.

After doing that successfully it proceeds to create two filles called 00000000.pky and 00000000.eky, which are used for encryption. Then, it starts a new thread that writes 136 bytes into a file called 000000000.res every 25 second. (Figure 2.38 and Figure 2.39)

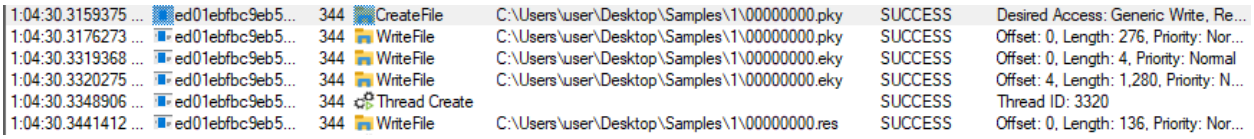


Figure 2.38: Creation of encryption files.

Time of Day	Process Name	PID	Operation	Path	Result
1:04:30.3441412 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:04:30.8888875 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:04:55.6877447 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:05:21.0400847 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:05:46.3836116 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:06:11.7552532 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:06:37.2522986 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:06:49.6224406 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS
1:07:02.7301014 PM	ed01ebfbc9eb5...	344	WriteFile	C:\Users\user\Desktop\Samples\1\00000000.res	SUCCESS

Figure 2.39: Thread writing into 00000000.res.

The next operation it does is to create a thread that launches taskdl.exe every 30 seconds. It is used for deletion of temporary files. (Figure 2.40 and Figure 2.41)

Time of Day	Process Name	PID	Operation	Path	Result	Detail	Command Line
1:04:30.674...	ed01ebfbc9e...	344	WriteFile	C:\Users\user\Desktop\Sa...	SUCCESS	Offset: 0, Length: 20,480, I/O Flags...	"C:\Users\user\Desktop\Samples\1\ed01...
1:04:30.685...	ed01ebfbc9e...	344	Process Create	C:\Users\user\Desktop\Sa...	SUCCESS	PID: 6548, Command line: taskdl.exe	"C:\Users\user\Desktop\Samples\1\ed01...
1:04:30.685...	taskdl.exe	6548	Process Start		SUCCESS	Parent PID: 344, Command line: tas...	taskdl.exe
1:04:30.685...	taskdl.exe	6548	Thread Create		SUCCESS	Thread ID: 4332	taskdl.exe
1:05:00.822...	ed01ebfbc9e...	344	Process Create	C:\Users\user\Desktop\Sa...	SUCCESS	PID: 3060, Command line: taskdl.exe	"C:\Users\user\Desktop\Samples\1\ed01...
1:05:00.822...	taskdl.exe	3060	Process Start		SUCCESS	Parent PID: 344, Command line: tas...	taskdl.exe
1:05:00.822...	taskdl.exe	3060	Thread Create		SUCCESS	Thread ID: 5068	taskdl.exe
1:05:30.884...	ed01ebfbc9e...	344	Process Create	C:\Users\user\Desktop\Sa...	SUCCESS	PID: 3820, Command line: taskdl.exe	"C:\Users\user\Desktop\Samples\1\ed01...

Figure 2.40: Taskdl thread

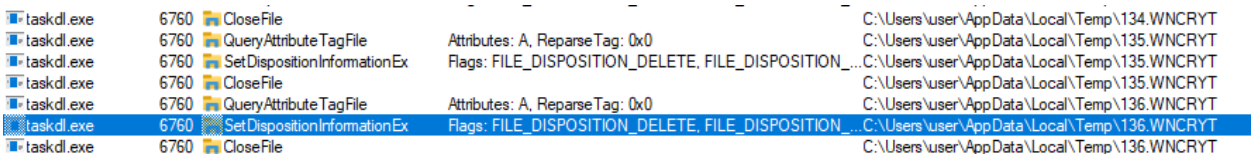


Figure 2.41: Taskdl deleting temp files.

After this it creates a file called @WanaDecryptor@.exe and 32341682975150.bat. (Figure 2.42)

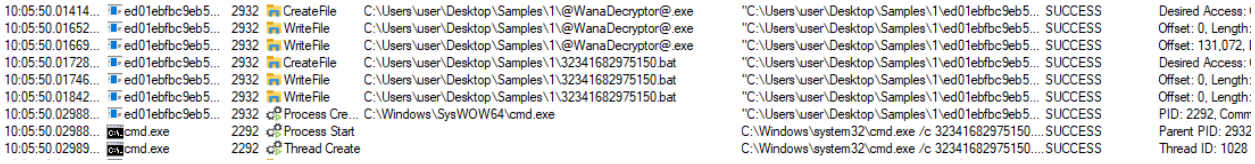


Figure 2.42: Creation of @WanaDecryptor@.exe and 32341682975150.bat.

Once those files are created it starts encrypting all the files and writing a file called @Please_Read_Me@.txt and @WanaDecryptor@.exe in every directory where it encrypts files. It encrypts files by first creating ".tmp" files that have names starting with "~SD" and might contain information about the contents of the directory they are created in.

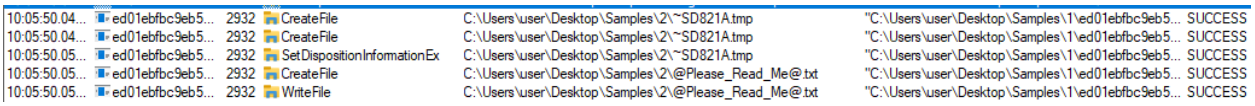


Figure 2.43: Creating ".tmp" files.

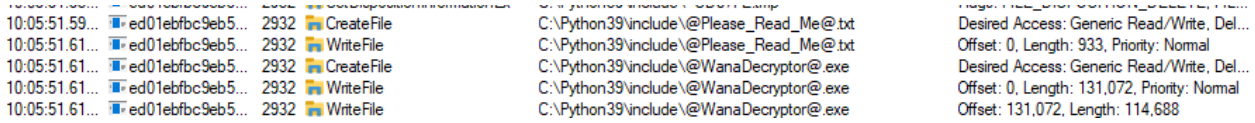


Figure 2.44: Files getting created in directory with encrypted files/

When it encrypts files it adds the extension ".WNCRYT" or ".WNCRY" to the name.

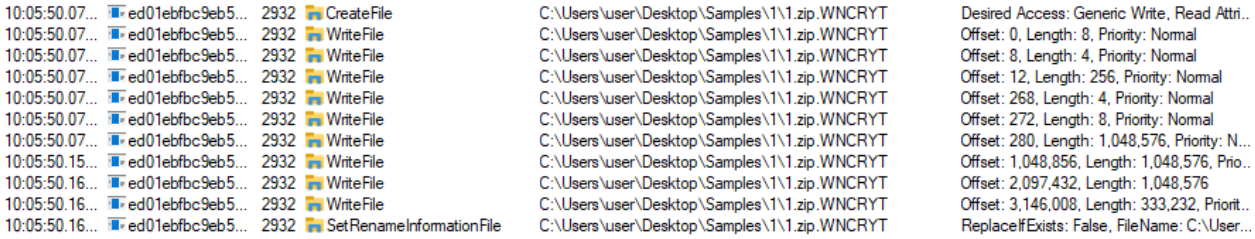


Figure 2.45: Creating ".WNCRYT" files.

When the hex of the files is examined it contains the string "WANACRY!" in its hex.

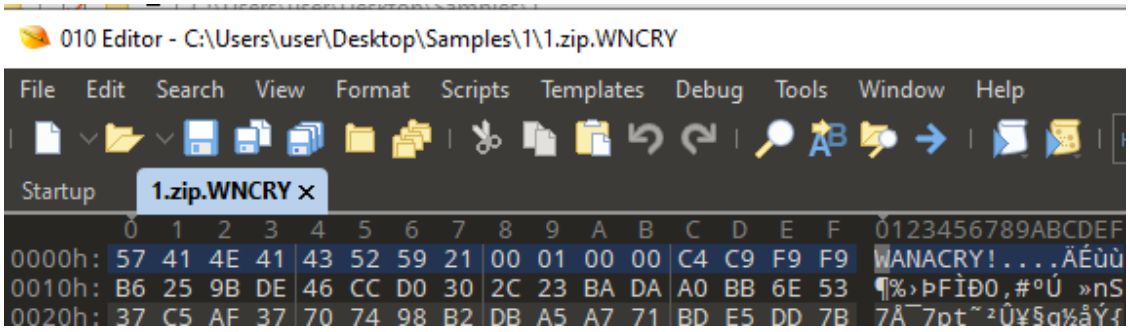


Figure 2.46: Encrypted file hex

To achieve persistence, the malware stars cmd and runs a command to add a registry key to "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\osnhnowfratdjot119". The "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" is used for program that are ran during startup. (Figure 2.47 and Figure 2.48)

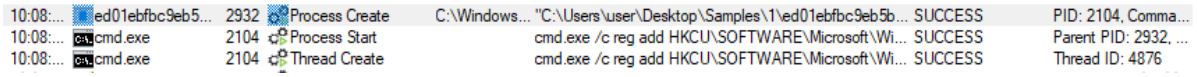


Figure 2.47: Starting cmd.

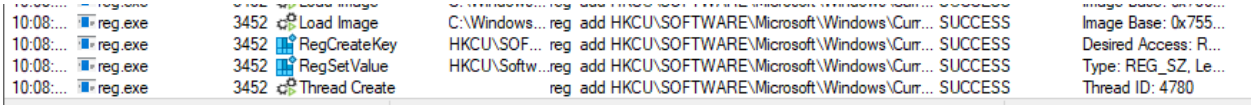


Figure 2.48: Creating registry key for persistence.

The key it adds has the location of the "tasksche.exe" file in its contents. (Figure 2.49)

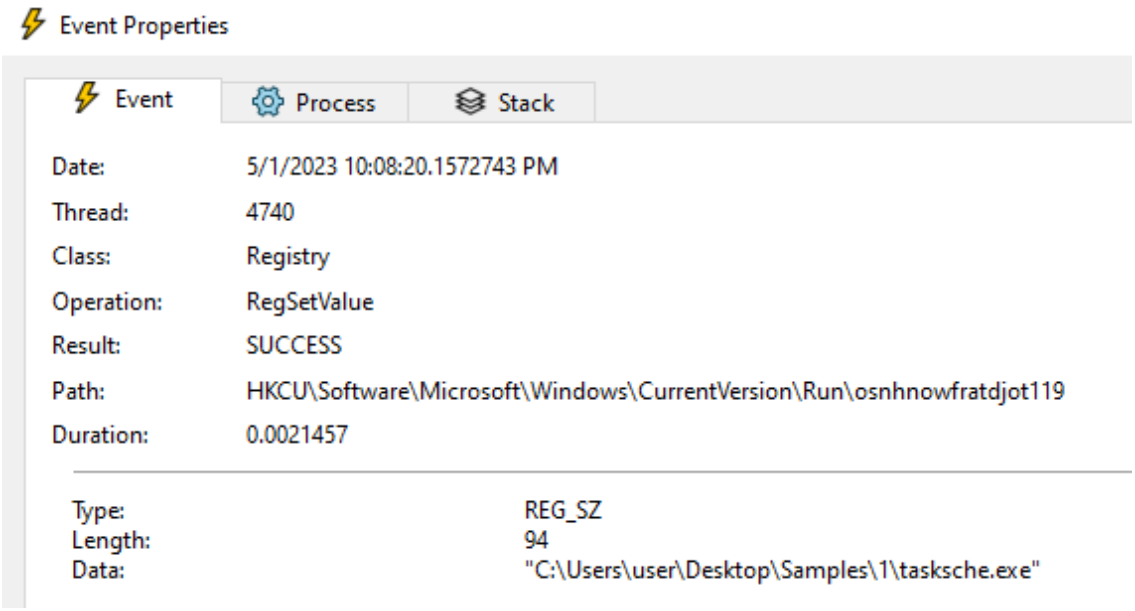


Figure 2.49: Value added in key for persistence.

Another operation performed by the malware is the placing of a copy of @WanaDecryptor@.bmp and @WanaDecryptor@.exe on the desktop of all users.

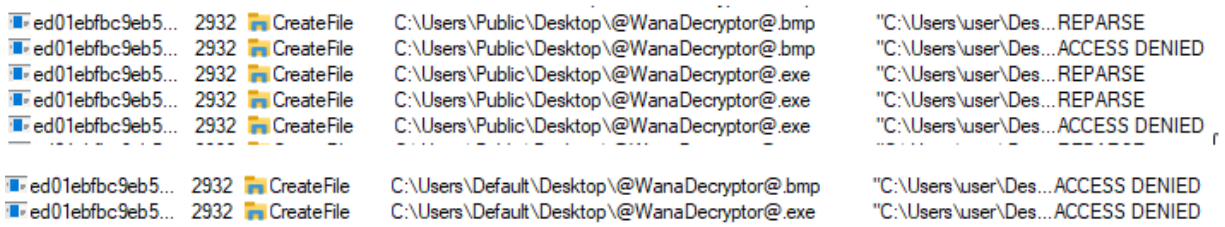


Figure 2.50: Creating a copy of @WanaDecryptor@.bmp/exe on the desktop of all users.

After successfully copying @WanaDecryptor@.bmp it edits the value of the "HKCU\Windows\SysWOW64\uxtheme.dll" to display the ransom message wallpaper.

10:08:01.60656...	ed01ebfbc9eb5...	2932	CreateFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.60994...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.61010...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.61028...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.61046...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.61067...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.61077...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.bmp	"C:\Users\user\Des... SUCCESS
10:08:01.61202...	ed01ebfbc9eb5...	2932	Load Image	C:\Windows\SysWOW64\uxtheme.dll	"C:\Users\user\Des... SUCCESS
10:08:01.61342...	ed01ebfbc9eb5...	2932	RegSetValue	HKCU\Control Panel\Desktop\Wallpaper	"C:\Users\user\Des... SUCCESS
10:08:01.62305...	ed01ebfbc9eb5...	2932	CreateFile	C:\Users\user\Desktop\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS
10:08:01.62424...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS
10:08:01.62443...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS

Figure 2.51: Settings of ransom wallpaper.

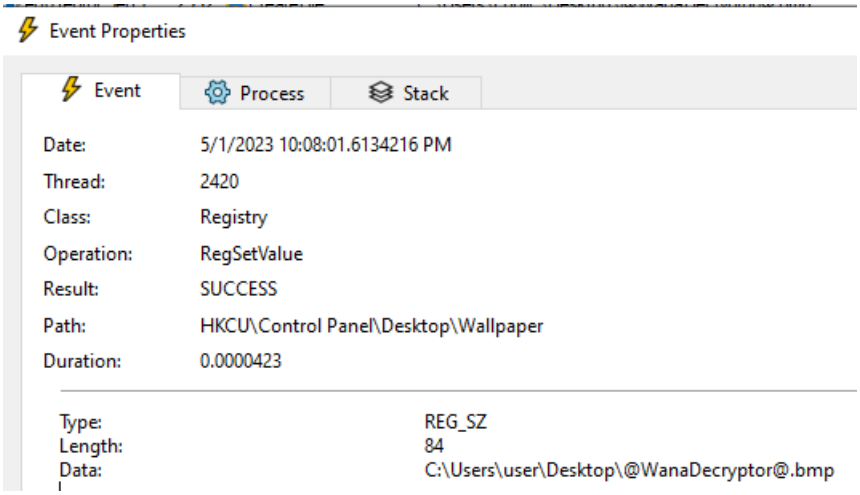


Figure 2.52: Value added to Wallpaper key.

The last step it takes is to start wanadecryptor.exe to display the ransom interface.

10:08:01.61202...	ed01ebfbc9eb5...	2932	Load Image	C:\Windows\SysWOW64\uxtheme.dll	"C:\Users\user\Des... SUCCESS
10:08:01.61342...	ed01ebfbc9eb5...	2932	RegSetValue	HKCU\Control Panel\Desktop\Wallpaper	"C:\Users\user\Des... SUCCESS
10:08:01.62305...	ed01ebfbc9eb5...	2932	CreateFile	C:\Users\user\Desktop\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS
10:08:01.62424...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS
10:08:01.62443...	ed01ebfbc9eb5...	2932	WriteFile	C:\Users\user\Desktop\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS
10:08:01.62654...	ed01ebfbc9eb5...	2932	Process Create	C:\Users\user\Desktop\Samples\1\@WanaDecryptor@.exe	"C:\Users\user\Des... SUCCESS
10:08:01.62655...	@WanaDecryp...	1760	Process Start		@WanaDecryptor@... SUCCESS
10:08:01.62656...	@WanaDecryp...	1760	Thread Create		@WanaDecryptor@... SUCCESS

Figure 2.53: Starting ransom interface.

2.4.1.2 Process Explorer

Process Explorer is used to monitor the process and threads that are running on the teams. It is useful for malware analysis to assist with the identification of malicious processes and their subprocess in a tree format. The analyst ran Process Explorer together with Process monitor before executing the malware to perform dynamic analysis. (Figure 2.54)

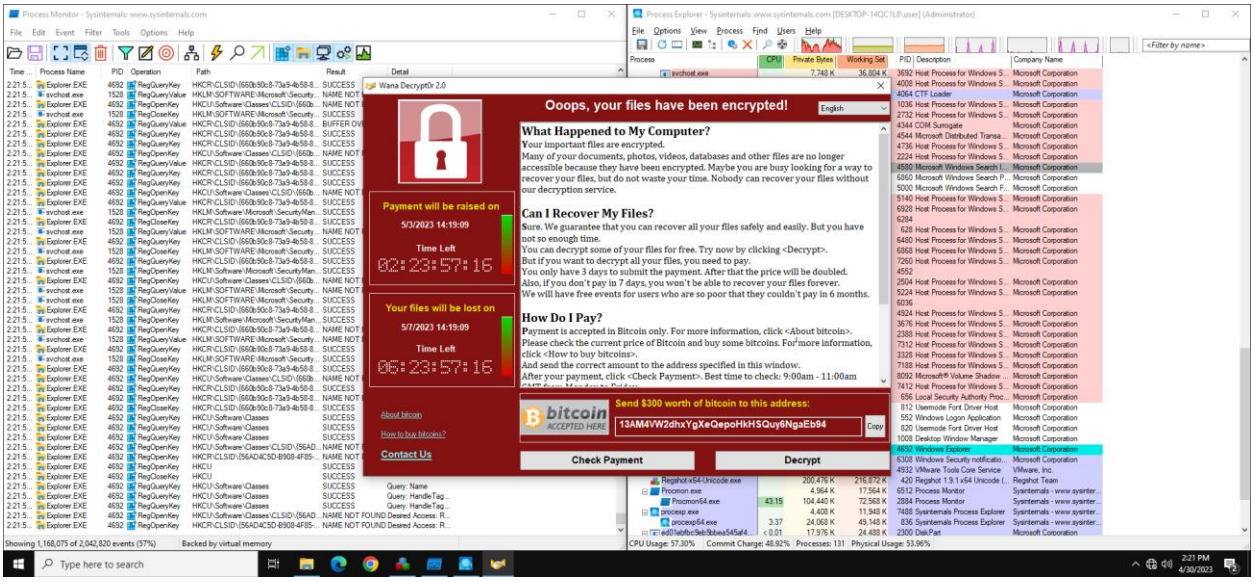


Figure 2.54: Process monitor and Process Explorer running.

Once the malware was successfully finished executing some suspicious looking process appeared. (Figure 2.55)

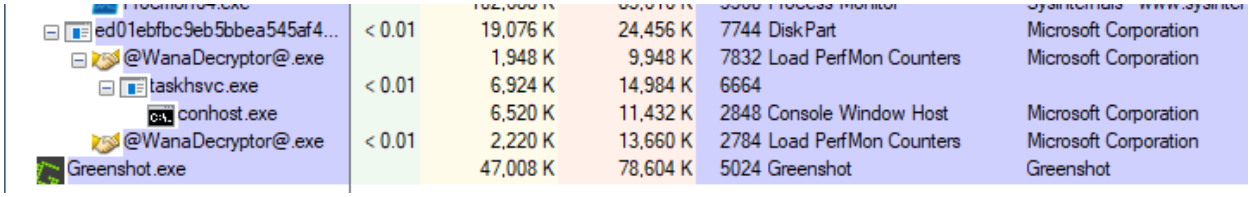


Figure 2.55: Suspicious Processes

- ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe
- @WanaDecryptor@.exe
- Taskhsvc.exe
- Conhost.exe

From the information discovered using process monitor it was determined they are all related to the malware.

2.4.1.3 Regshot

Regshot can be used to take too snapshots of the Registry. One is taken before the infection (Figure 2.56) and then another after (Figure 2.57). Afterwards the two files are compared which to see any changes that were made by the malware. (Figure 2.58)

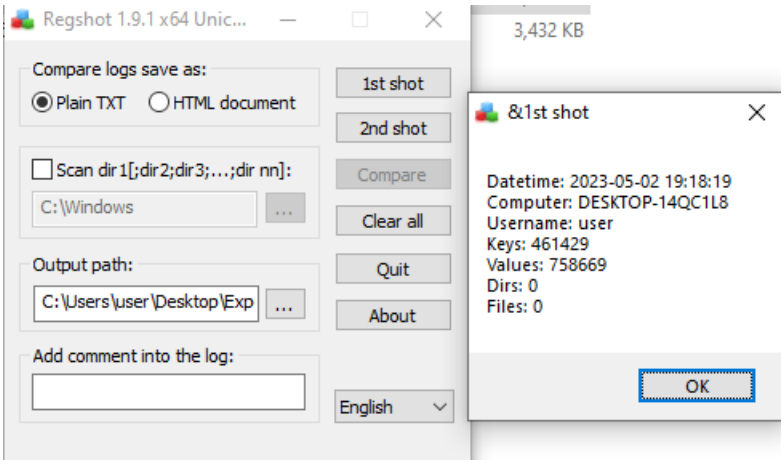


Figure 2.56: Shot before infection.

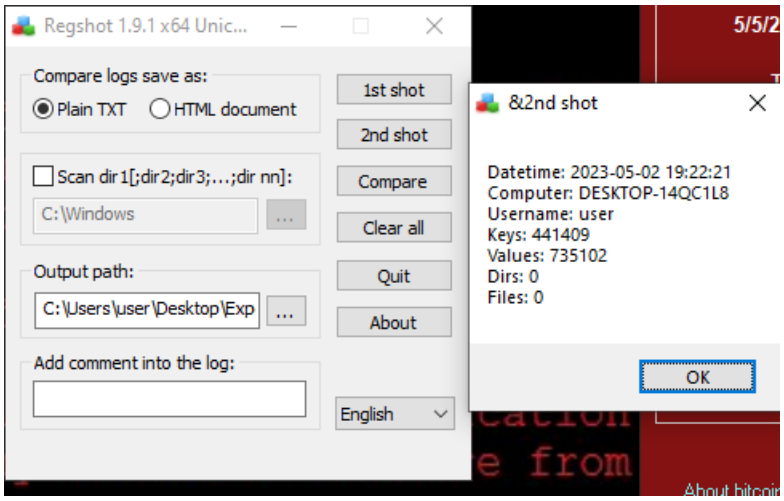


Figure 2.57: Shot after infection.

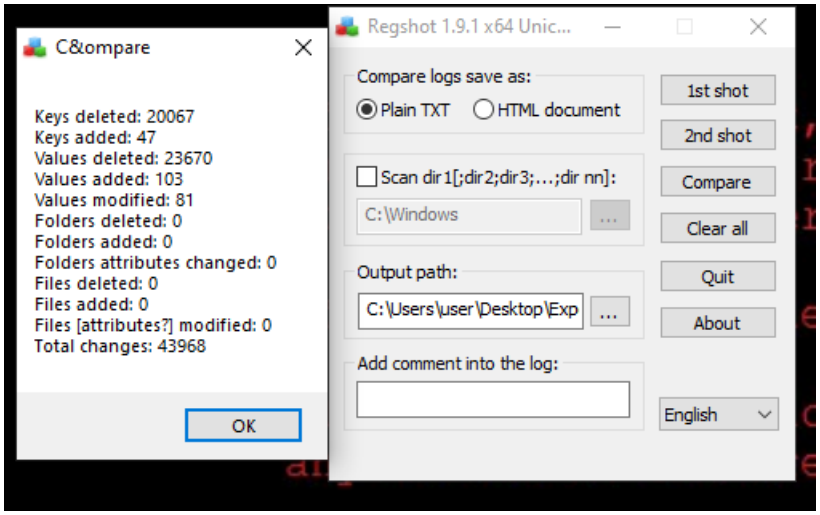


Figure 2.58: Comparison of shots

The keys that were found to have been edited by the malware after examining the comparison output confirmed the findings made when using process monitor in Section 2.4.1.1.

The editing of the working directory can be seen in Figure 2.59.

```
HKU\S-1-5-21-2169232433-3398496680-935370409-1000\Software\WanaCrypt0r\wd: "C:\Users\user\Desktop\Samples\1"
```

Figure 2.59: Regshot comparison WD.

The creation of key for persistence can be seen in Figure 3.59.

```
HKU\S-1-5-21-2169232433-3398496680-935370409-1000\Software\Microsoft\Windows\CurrentVersion\Run
\osnhnowfratdjot119: ""C:\Users\user\Desktop\Samples\1\tasksche.exe""
```

Figure 2.60: Regshot comparison persistence key.

The wallpaper change can be seen in Figure 3.60.

```
HKU\S-1-5-21-2169232433-3398496680-935370409-1000\Control Panel\Desktop\WallPaper: "C:\Users\user
\Desktop\@WanaDecryptor@.bmp"
```

Figure 2.61: Regshot comparison wallpaper.

2.5 NETWORK TRAFFIC ANALYSIS

Wireshark is a network protocol analysis program that is useful for capturing network packets and traffic that is passing through a specified network interface. Wireshark was used to conduct network traffic analysis but there was nothing discovered using that. (Figure 2.62)

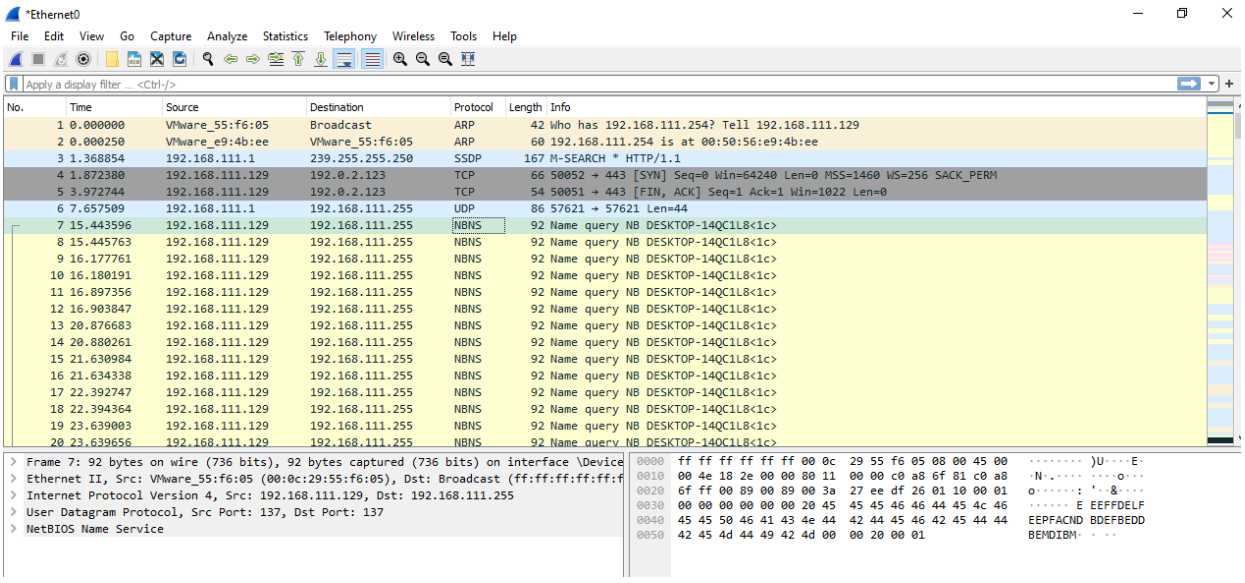


Figure 2.62: Network Traffic Analysis with Wireshark

The next tool that was used was FakeNet-NG. It is a dynamic network analysis tool that works in simulates Internet services and analyses traffic to them. (Figure 2.63)

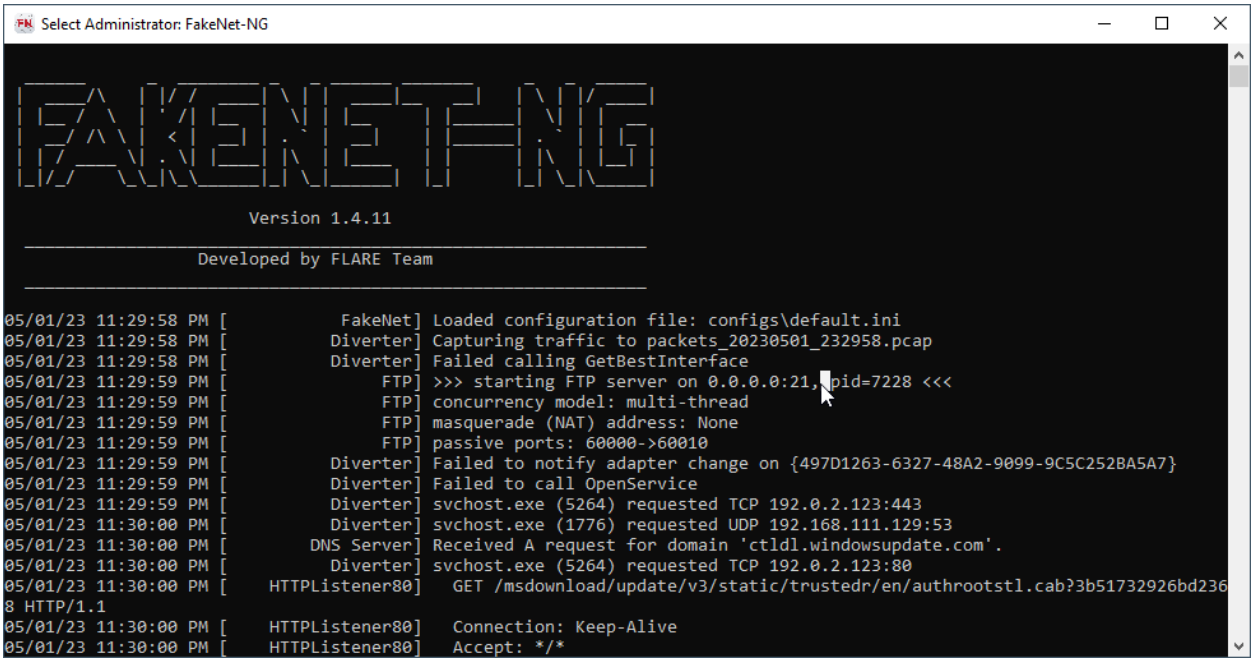


Figure 2.63- FakeNet-NG being ran.

After FakeNet-NG was ran the malware was executed and a couple of interesting packets were captured which were related, however their purpose was not discovered. It is attempting to send TCP data to different addresses however the purpose of that is unknown.

```
05/01/23 11:33:24 PM [          Diverter] @WanaDecryptor@.exe (1416) requested TCP 127.0.0.1:9050
05/01/23 11:33:24 PM [          Diverter] System (4) requested UDP 192.168.111.255:137
05/01/23 11:33:24 PM [          Diverter] @WanaDecryptor@.exe (1416) requested TCP 127.0.0.1:9050
05/01/23 11:33:24 PM [          Diverter] System (4) requested UDP 192.168.111.255:137
05/01/23 11:33:29 PM [          Diverter] taskhsvc.exe (6716) requested TCP 127.0.0.1:50125
05/01/23 11:33:29 PM [          Diverter] taskhsvc.exe (6716) requested TCP 127.0.0.1:50126
05/01/23 11:33:29 PM [          Diverter] taskhsvc.exe (6716) requested TCP 127.0.0.1:50125
05/01/23 11:33:31 PM [          Diverter] taskhsvc.exe (6716) requested TCP 213.61.66.117:9002
05/01/23 11:33:31 PM [          Diverter] taskhsvc.exe (6716) requested TCP 128.31.0.39:9101
05/01/23 11:33:31 PM [          Diverter] taskhsvc.exe (6716) requested TCP 213.61.66.117:9002
05/01/23 11:33:31 PM [          Diverter] taskhsvc.exe (6716) requested TCP 128.31.0.39:9101
```

Figure 2.64: Packets captured by FakeNet-NG

3 DISCUSSION

3.1 GENERAL DISCUSSION

After completing the CCDCOE Malware Reverse Engineering methodology, the provided malware sample was successfully identified as the ransomware WannaCry. It was uploaded to Virus Total where it was identifies as malware by 66 out of 70 security vendors.

During static analysis it was discovered that it was not packed and would pose a legitimate Microsoft process called “diskpart.exe”. When it was examined using Resource Hacker a password protected “.zip” archive was discovered, which would successfully be extracted by the analyst after discovering the password is “WNcry@2ol7”. Using PeStudio suspicious imports would be discovered:

- CreateServiceA – used for the creation of a service object and its addition to the service control manager database. Could potentially be getting used for persistence.
- RegCreateKeyW – used for the creation of registry keys.
- RegSetValueExA – used for the editing of registry keys.
- WriteFile – used for writing date into files.
- SetFileAttributesW – used for changing file attributes, such as whether a file is hidden or read only.
- TerminateProcess – used for ending a process and all its threads.
- GetExitCodeProcess – used to determine whether a file has successfully terminated by retrieving its termination status.
- rand/srand– used for generating random numbers.
- SetCurrentDirectoryW/SetCurrentDirectoryA – used for changing directories.

Using PE studios multiple strings related to the malware encryption function were also found (CryptGenKey, CryptDecrypt, CryptEncrypt, CryptDestroyKey, CryptImportKey, CryptAcquireContext).

When examining the executable’s assembly code using IDAFree and Ghidra, the analyst was able to determine how its functions operate and was able to discover the password that was needed to extract the archive that was discovered using Resource Hacker.

Finally, during dynamic analysis the malware’s way of functioning was further understood and it was discovered that is successful gains persistence by editing the registry.

After being executed the malware would start by extracting an archive with different files it needs to function by using the password “WNcry@2ol7” in the directory in which it is located. Afterwards it would hide its working directory and begin spreading throughout the system. It would start encrypting each file it finds into files with “.wncry” and “.wncryt” extension. In

every directory where it encrypts files it would place a "@Please_Read_Me@.txt" and a "@WanaDecryptor@.exe" file. Finally, it would get persistence by writing the location of tasksche.exe in HKCU\Software\Microsoft\Windows\CurrentVersion\Run\osnhnowfratdjot119 registry key, change the desktop wallpaper to ransomware message and run "@WanaDecryptor@.exe" to display the ransomware application called "Wana Decryptor 2.0". The ransomware application would display information about what has happened and demand a ransom from the victim(Appendix B – Malware interface and Appendix C – Ransom Message Wallpaper)

From the different sections of this report, it was determined that the ransomware had the following characteristics which can be used as Indicators of Compromise (IoC).

- All files on the computer will get encrypted into a file with a ".wnrcryt" or ".wncry" extension.
- Files called "@WanaDecryptor@.exe" and "@WanaDecryptor@.bmp" get created on all user Desktops.
- Inside each folder where files are encrypted a file called "@Please_Read_Me@.txt" and "@WanaDecryptor@.exe" gets created.
- Processes called "diskpart.exe" and "taskhsvc.exe" get launched. Process called "taskdl.exe" gets launched every 30 seconds to delete temporary files.
- Directory where malware is located gets hidden and multiple files are extracted onto it.
- Registry key is created at HKCU\Software\Microsoft\Windows\CurrentVersion\Run\osnhnowfratdjot119 for persistence.
- An application called "Wana Decryptor 2.0" gets repeatedly opened.

In conclusion, the malware is a simplified version of the "WannaCry" malware that is missing features related to its original worming capabilities that would allow it to spread to other connected devices, but is nevertheless still dangerous, because of it its ability to encrypt every file on the infected system. All the aims of this report were met following the procedure set out by the CCDCOE Malware Reverse Engineering methodology. The malware was successfully identified, its characteristics and components were analysed, the operation behaviour was understood, and countermeasures were discussed.

3.2 COUNTERMEASURES

To mitigate against ransomware attacks it is recommended the following actions are taken:

- Regular backup should be made to allow the restoration of files in case of their encryption.
- Antivirus software should be installed, because it would prevent known malware like the one from the sample in this report from being executed on the system, by immediately quarantining it and/or deleting it.
- Staff training should be frequent and up to date to prevent social engineering leading to malware getting on the system.
- Any untrusted application should not be executed, because they could potentially be disguised malware.

3.3 FUTURE WORK

As future work the analyst could do some more work in relation to analysis of the Network traffic of the malware, since it is possible that there is network activity that was missed, because of misconfiguration of INetSim and lack of time to do longer analysis using Wireshark and FakeNet-NG. Alongside this certain section of the CCDCOE Malware Reverse Engineering methodology that were cut due to lack of time and experience using the tools by the analyst could be added to the report. Those sections include:

- Using Sandboxes to do memory dump analysis.
- Examining code in Debuggers such as WinDbg and OllyDbg to further understand the workflow of the malware using breakpoint as well as to make changes to its code to see the effect on its functions.
- More in-depth analysis of whether the file is packed and documenting the process of unpacking it.
- Creation of Yara rules based on the IoCs that were discovered and using them to scan the malware.
- More in depth analysis using IDAFree or Ghidra after the analyst learns more assembly coding.

4 REFERENCES

Blosil, J., 2022. *Measuring the true cost of a ransomware attack.* [Online]
Available at: https://www.netapp.com/blog/ransomware-cost/
[Accessed 28 April 2023].

Braue, D., 2022. *Global Ransomware Damage Costs Predicted To Exceed $265 Billion By 2031.* [Online]
Available at: https://cybersecurityventures.com/global-ransomware-damage-costs-predicted-to-reach-250-billion-usd-by-2031/
[Accessed 28 April 2023].

CCDOE, 2020. *Malware Reverse Engineering Handbook.* [Online]
Available at:
https://ccdcoe.org/uploads/2020/07/Malware_Reverse_Engineering_Handbook.pdf
[Accessed 26 April 2023].

Clancy, M., 2021. *The True Cost of Ransomware.* [Online]
Available at: https://www.backblaze.com/blog/the-true-cost-of-ransomware/
[Accessed 2023 April 28].

EpochConverter, 2023. *Epoch & Unix Timestamp Conversion Tools.* [Online]
Available at: https://www.epochconverter.com/
[Accessed 26 April 2023].

Hex-rays, 2023. *IDA Free.* [Online]
Available at: https://hex-rays.com/ida-free/
[Accessed 26 April 2023].

Johnson, A., 2019. *Resource Hacker v5.1.7.* [Online]
Available at: http://www.angusj.com/resourcehacker/
[Accessed 26 April 2023].

Malwarebytes Threat Intelligence Team, 2023. *Ransomware in the UK, April 2022–March 2023.* [Online]
Available at: https://www.malwarebytes.com/blog/threat-intelligence/2023/04/ransomware-review-uk
[Accessed 2023 April 28].

Malwarebytes, n.d. *What is malware?.* [Online]
Available at: https://www.malwarebytes.com/malware
[Accessed 28 April 2023].

Mandiant, 2020. *FakeNet-NG.* [Online]
Available at: https://github.com/mandiant/flare-fakenet-ng/releases
[Accessed 26 April 2023].

Mandiant, 2023. *FLARE Obfuscated String Solver v2.20.* [Online]
Available at: https://github.com/mandiant/flare-floss
[Accessed 26 April 2023].

Microsoft, 2021. *String v2.54.* [Online]
Available at: https://learn.microsoft.com/en-us/sysinternals/downloads/strings
[Accessed 26 April 2023].

Microsoft, 2023. *Process Explorer v17.04.* [Online]
Available at: https://learn.microsoft.com/en-us/sysinternals/downloads/process-explorer
[Accessed 26 April 2023].

Microsoft, 2023. *Process Monitor v3.93.* [Online]
Available at: https://learn.microsoft.com/en-us/sysinternals/downloads/procmon
[Accessed 26 April 2023].

National Cyber Security Centre, n.d. *A guide to ransomware.* [Online]
Available at: https://www.ncsc.gov.uk/ransomware/home
[Accessed 2023 April 28].

NirSoft, 2021. *HashMyFiles.* [Online]
Available at: https://www.nirsoft.net/utils/hash_my_files.html
[Accessed 26 April 2023].

NSA, 2023. *Ghidra.* [Online]
Available at: https://github.com/NationalSecurityAgency/ghidra/releases
[Accessed 26 April 2023].

NTCore, n.d. *CFF Explorer 8.* [Online]
Available at: https://ntcore.com/?page_id=388
[Accessed 2023 April 26].

PEiD, 2018. *PEiD v0.95 (Windows).* [Online]
Available at: https://www.softpedia.com/get/Programming/Packers-Crypters-Protectors/PEiD-updated.shtml
[Accessed 26 April 2023].

Regshot, n.d. *Regshot Download.* [Online]
Available at: https://sourceforge.net/projects/regshot/
[Accessed 26 April 2023].

VirusTotal, 2023. *VirusTotal.* [Online]
Available at: https://www.virustotal.com/gui/home/upload
[Accessed 2023 April 26].

Winitor, 2023. *PEStudio.* [Online]
Available at: https://www.winitor.com/download2
[Accessed 26 April 2023].

Wireshark, n.d. *Wireshark.* [Online]
Available at: https://www.wireshark.org/download.html
[Accessed 26 April 2023].

5 APPENDICES

APPENDIX A – VIRUSTOTAL SIGNATURE INFO

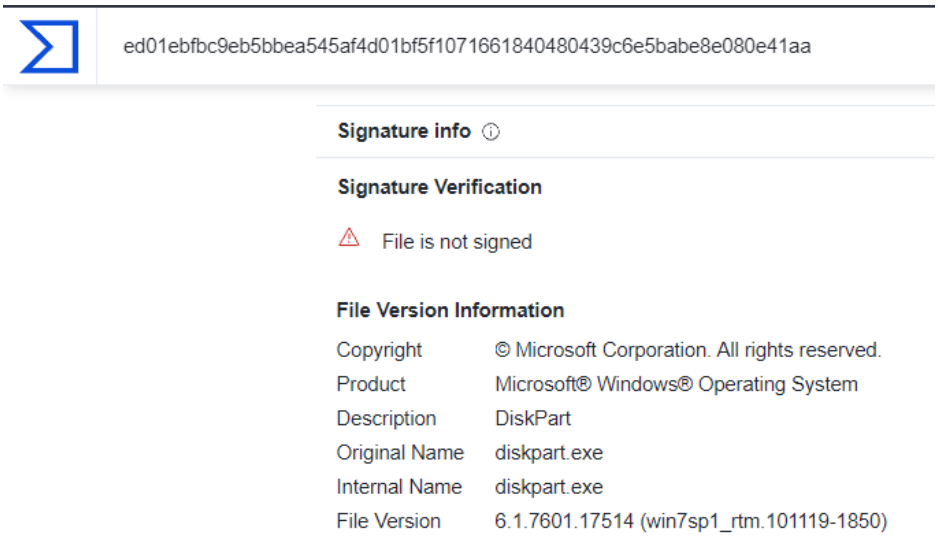


Figure 5.1: VirusTotal signature info

APPENDIX B – MALWARE INTERFACE

Figure 5.2: Malware Interface

APPENDIX C – RANSOM MESSAGE WALLPAPER

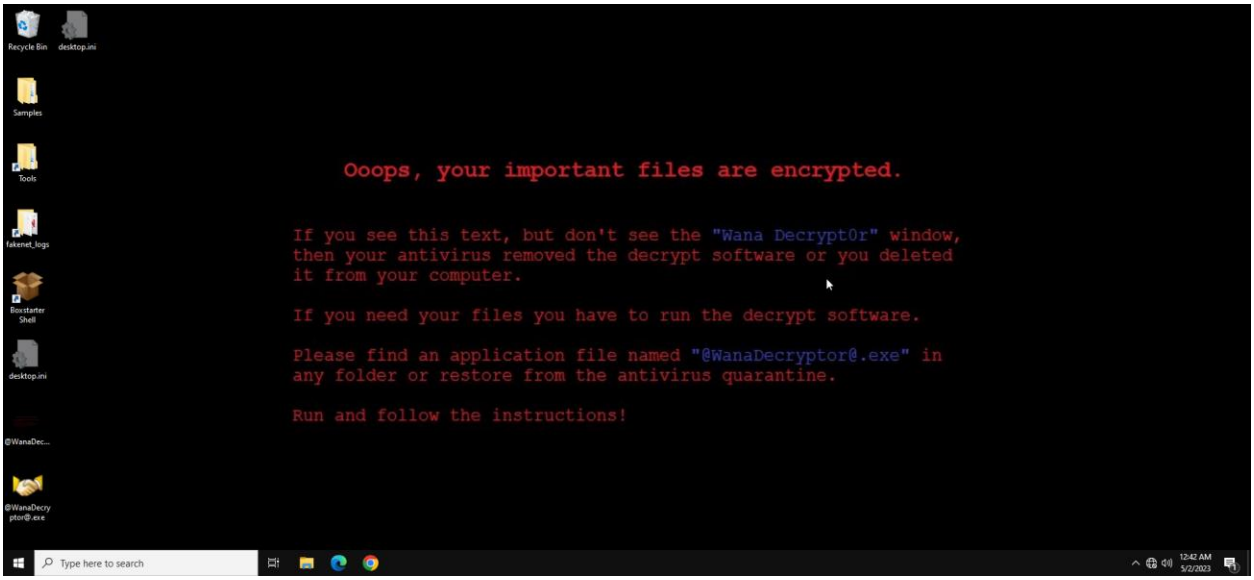


Figure 5.3: Ransom Message Wallpaper