CENTRE FOR DEVELOPMENT OF ADVANCEDCOMPUTING (C-DAC), THIRUVANANTHAPURAM, KERALA

A PROJECT REPORT ON

Wanna Cry Ransomware Malware Analysis

SUBMITTED TOWARDS THE

PG-DCSF March 2024

Group Number – 03

Jadhav Bhagyashree	PRN:240360940011
Kale Sagar	PRN:240360940012
Kanade Priyanka	PRN:240360940013
Kartikey Vaishnav	PRN:240360940014
Manoj Patil	PRN:240360940015

Under The Guidance Of

Mr. Jayaram P.	Dr. Priya S.
Centre Co- Ordinator	Project Guide

CONTENTS

ACKNOWLEDGEMENT

We would like to extend my sincere and heartfelt thanks towards Project Guide Dr.Priya S and Mr. Jayaram P helped us in making this project. It gives us immense pleasure and satisfaction to put forward this report and to express our sincere gratitude to many staff who have helped us directly or indirectly in this project.

First of all, we would like to thank our Project guide, Dr. Priya Sajan, for her support, encouragement, and guidance at every stage of our project.

We would like to thank our course co-ordinator, Mr. Jayaram, for his support and encouragement throughout the course. We would also like to thank all teaching staff along with our parents and friends who has helped us in every little way towards the working of the project.

ABSTRACT

The WannaCry ransomware attack in May 2017 was a major cybersecurity event that affected thousands of organizations worldwide. This malware targeted computers running Microsoft Windows, encrypting users' files and demanding a ransom in Bitcoin for their release. It spread rapidly by exploiting a known vulnerability in the Windows operating system, causing significant disruptions, especially in the healthcare sector, where it severely impacted the UK's National Health Service.

This project focuses on understanding how WannaCry works and why it spread so quickly. By analysing the malware, we learned that it used a method to spread across networks without human interaction, making it especially dangerous. The ransomware also had a "kill switch," a special feature that stopped it from running if it could connect to a specific website, which was accidentally discovered by a security researcher, slowing down its spread.

Through this analysis, we highlight the importance of keeping software up to date and the need for strong cybersecurity practices. The WannaCry attack serves as a reminder of how vulnerable systems can be when they are not properly protected, emphasizing the ongoing need for vigilance in the face of evolving cyber threats.

INTRODUCTION

Wanna Cry is a notorious ransomware strain that first emerged in May 2017, rapidly gaining global attention due to its unprecedented impact on both private and public sector organizations. This ransomware operates as a type of malicious software designed to encrypt a victim’s files, rendering them inaccessible, and subsequently demands a ransom payment to restore access. The Wanna Cry ransomware's significant notoriety stems from its rapid proliferation and the extensive disruption it caused across various sectors worldwide.
Wanna Cry primarily exploited a vulnerability in Microsoft Windows, specifically the SMB (Server Message Block) protocol, which allowed it to spread autonomously across networks. This vulnerability, identified as CVE-2017-0144, was dubbed “**Eternal Blue**” and had been leaked from a trove of cyber tools allegedly developed by the United States National Security Agency (NSA). Once a system was infected, Wanna Cry encrypted files and appended them with a .wncry extension.

The ransomware demanded payment in Bitcoin, threatening permanent data loss if the ransom was not paid within a specified time frame.
The ransomware's rapid spread was facilitated by a worm-like capability, enabling it to infect vulnerable systems without user interaction. This led to widespread disruptions, with numerous organizations experiencing significant operational setbacks.

Notably, the National Health Service (NHS) in the United Kingdom was among the high-profile victims, with the attack impacting numerous healthcare facilities and services.

The WannaCry incident highlighted critical vulnerabilities in cybersecurity practices and the importance of timely software updates and robust security measures. It served as a stark reminder of the evolving nature of cyber threats and the need for ongoing vigilance in the protection of digital assets

Executive Summary

The Wanna Cry ransomware attack was a global epidemic that took place in May 2017. This ransomware attack spread through computers operating Microsoft Windows.

User’s files were held hostage, and a Bitcoin ransom was demanded for their return. Were it not for the continued use of outdated computer systems and poor education around the need to update software, the damage caused by this attack could have been avoided .Wanna Cry is written in C++ language. On executing the malware it checks for a hard coded URL, if it successfully pings that URL malware does not execute.

If the URL was not found then malware execution takes place. Symptoms of the infection include ransomware payment window popup, encryption of the files, new desktop shortcuts and new services created. After executing the malware it creates a file named “C:\Windows\tasksche.exe” which contains the payloads, and then starts encrypting all the files on computer.

Wanna Cry ransomware also tries to spread to other Windows Computers using the Eternal Blue vulnerability.

YARA signature rules are attached in Rules & Signatures. Malware sample and hashes have been submitted to Virus Total for further examination.

High level Technical summary

Wanna Cry consists of two parts:

stage 0 executable and an unpacked

stage 2 encryption and worm program.

It first attempts to contact its kill switch URL

(hxxps://iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com).

If the URL is alive it does not execute. If the URL is not found then the

malware unpacks tasksche.exe and creates a service to start

tasksche.exe on startup. This executable encrypts all the files,

shows the popup ransom window and changes the background of

Desktop. It creates a random folder inside C:\ProgramData to store

all the wanna cry files. It exploits the Eternal Blue vulnerability on port 445 to spread to other computers.

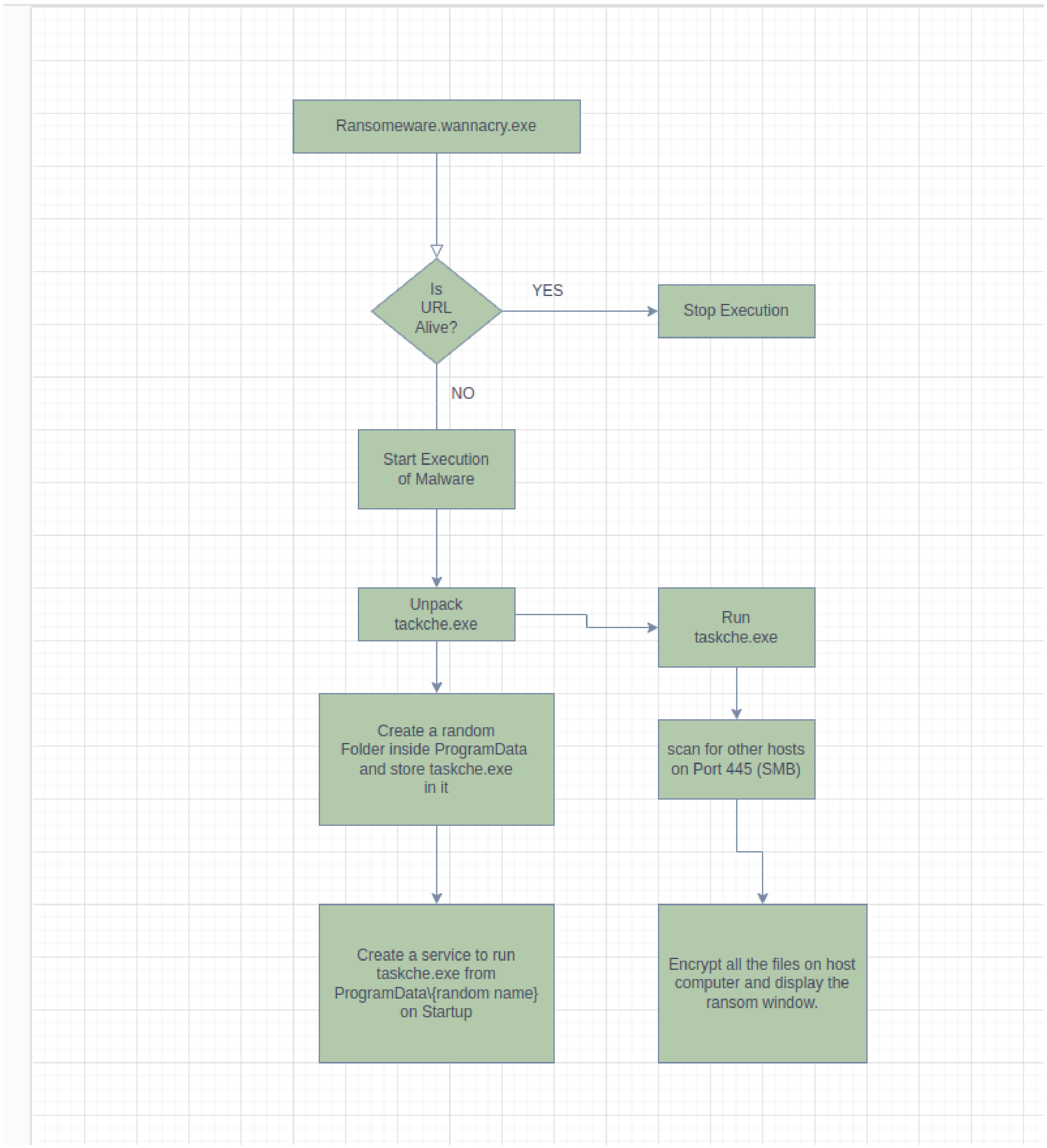
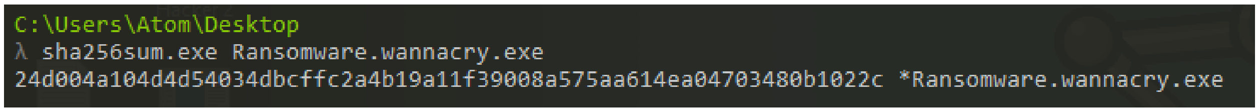
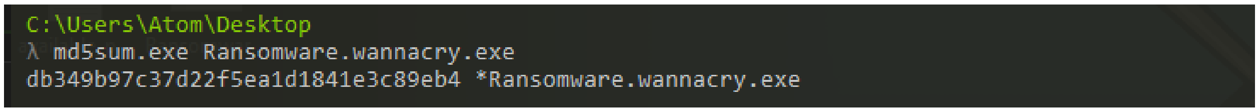

Ransomeware.wannacry.exe
Is URL Alive?
YES
Stop Execution
NO
Start Execution of Malware
Unpack tackche.exe
Run taskche.exe
Create a random Folder inside ProgramData and store taskche.exe in it
scan for other hosts on Port 445 (SMB)
Create a service to run taskche.exe from ProgramData\{random name} on Startup
Encrypt all the files on host computer and display the ransom window.

Fig. Malware Analysis Flowchart

Malware Composition



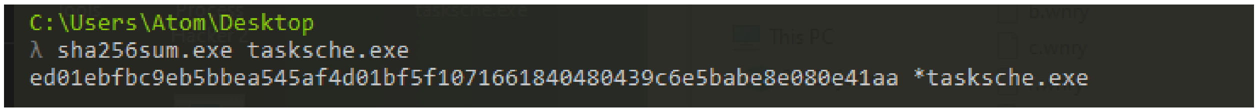
```
C:\Users\Atom\Desktop
λ sha256sum.exe Ransomware.wannacry.exe
24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614ea04703480b1022c *Ransomware.wannacry.exe
```



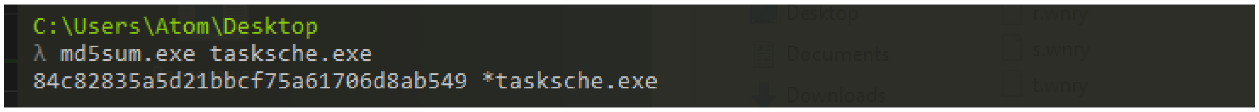
```
C:\Users\Atom\Desktop
λ md5sum.exe Ransomware.wannacry.exe
db349b97c37d22f5ea1d1841e3c89eb4 *Ransomware.wannacry.exe
```

Ransomware.wannacry.exe

The initial executable that runs and checks the kill switch URL. If alive don't run else unpack tasksche.exe.



```
C:\Users\Atom\Desktop
λ sha256sum.exe tasksche.exe
ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa *tasksche.exe
```



```
C:\Users\Atom\Desktop
λ md5sum.exe tasksche.exe
84c82835a5d21bbcf75a61706d8ab549 *tasksche.exe
```

tasksche.exe:

This is used for persistence. It creates a random folder for wannacry staging area inside ProgramData. After execution of malware on host computer it tries to spread itself on other windows computers using SMB port 445. It starts encrypting all the files and after that it displays the ransomware popup and message.

Static Analysis

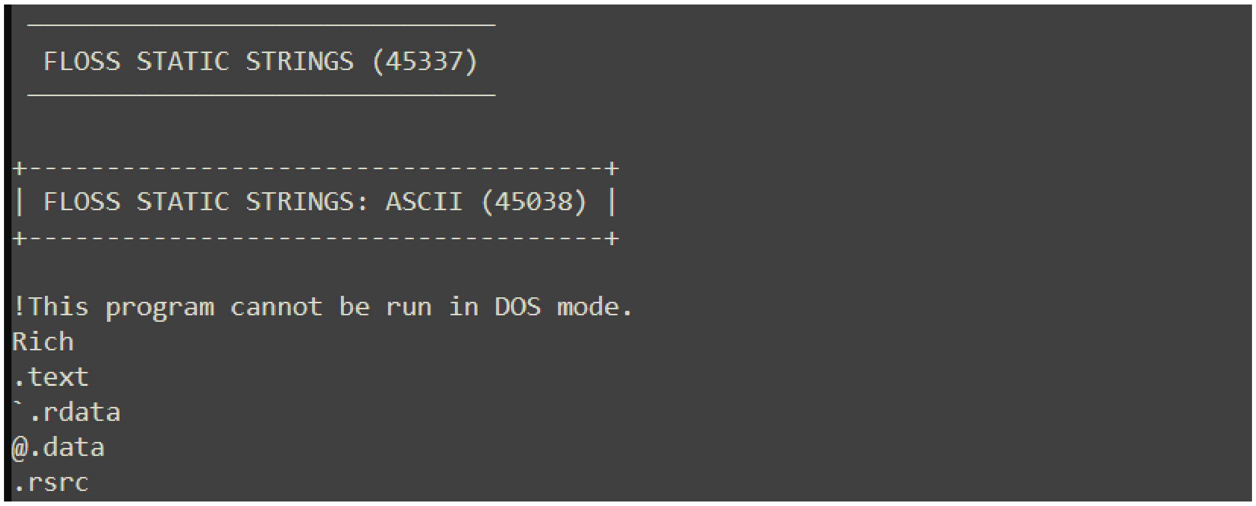
1. Sample Overview

- **Filename:** Ransomware.wannacry.exe (may vary)
- **File Type:** Executable (.exe)
- **File Size:** Approximately 500 KB
- **MD5 Hash:** [db349b97c37d22f5ea1d18413c89eb4]
- **SHA256 Hash:** [24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614e]

2 . **Examine File Metadata**

Extract readable text from the binary.

Tool used :- Floss

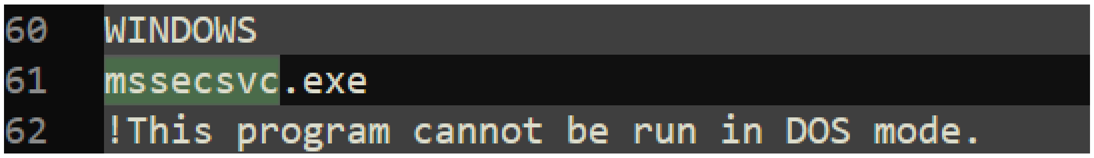


```
 ──────────────────────────────
  FLOSS STATIC STRINGS (45337)
 ──────────────────────────────


+------------------------------------+
| FLOSS STATIC STRINGS: ASCII (45038) |
+------------------------------------+

!This program cannot be run in DOS mode.
Rich
.text
`.rdata
@.data
.rsrc
```

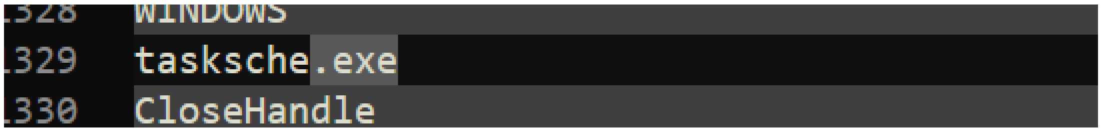
From the strings “!This program cannot be run in DOS mode ” we can conclude that its a portable binary.

We also saw the repetition of the above string, so this might be a packed portable binary.

Some suspicious exe we might want to note down for further analysis



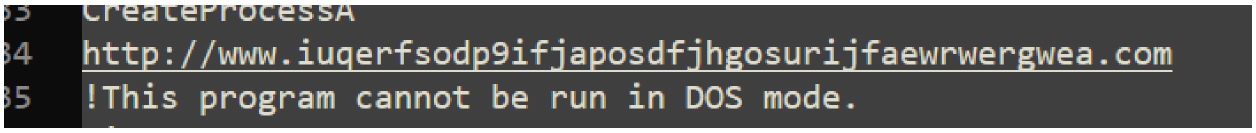
```
WINDOWS
mssecsvc.exe
!This program cannot be run in DOS mode.
```



```
WINDOWS
tasksche.exe
CloseHandle
```

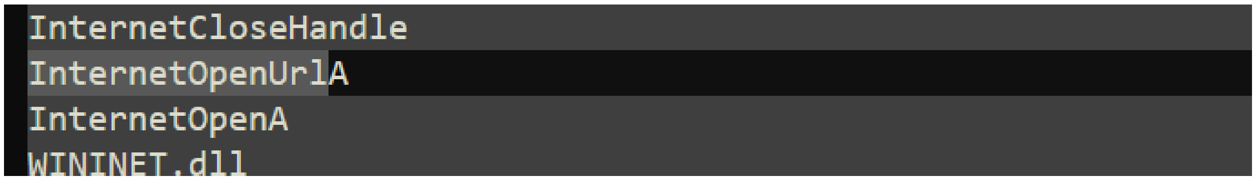
Suspicious URL :

Now this a unregistered domain. However, a security researcher discovered this behavior and registered the domain, which effectively acted as a "kill switch" for the ransomware. Once the domain was registered, any instance of Wanna Cry that could reach the domain would stop executing, significantly reducing the spread and impact of the ransomware.



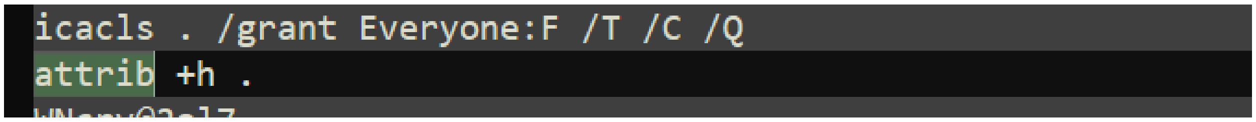
```
CreateProcessA
http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com
!This program cannot be run in DOS mode.
```

Modules used to open suspicious URL



```
InternetCloseHandle
InternetOpenUrlA
InternetOpenA
WININET.dll
```

Now these modules are not itself malicious and are used in many legitimate software but also very common in malware trying to connect to a domain.



```
icacls . /grant Everyone:F /T /C /Q
attrib +h .
```

Icacls is a Windows utility(Displays or modifies discretionary access control lists (DACLs) on specified files)

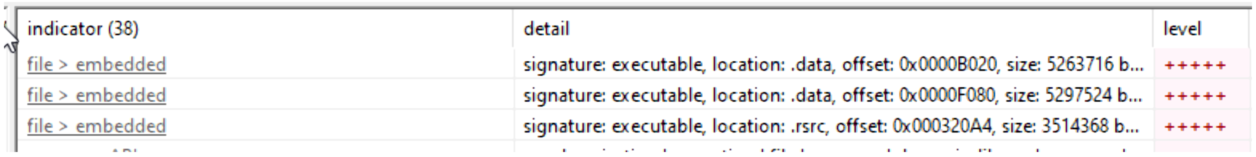
Here it is used to grant permission to everyone in current working directory and this directory is also hidden. Now these two strings raises a lot off suspicion as a normal user wont even know that this hidden directory even exist .

In further analysis we will discover that this directory in

C:\ProgramData\{hidden directory with random name} is used as a staging area for the malware execution

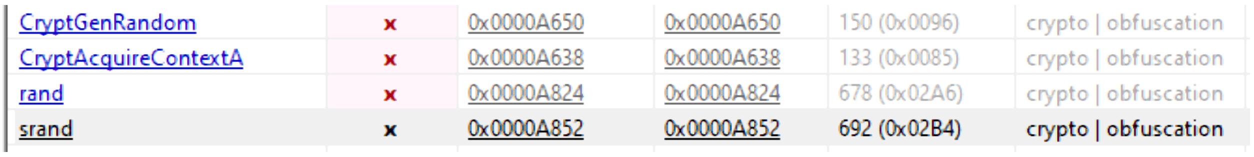
Tool used PEstudio :

Here we can see that there are three packed binaries present inside the file.



indicator (38)	detail	level
file > embedded	signature: executable, location: .data, offset: 0x0000B020, size: 5263716 b...	+++++
file > embedded	signature: executable, location: .data, offset: 0x0000F080, size: 5297524 b...	+++++
file > embedded	signature: executable, location: .rsrc, offset: 0x000320A4, size: 3514368 b...	+++++

Some cryptography libraries

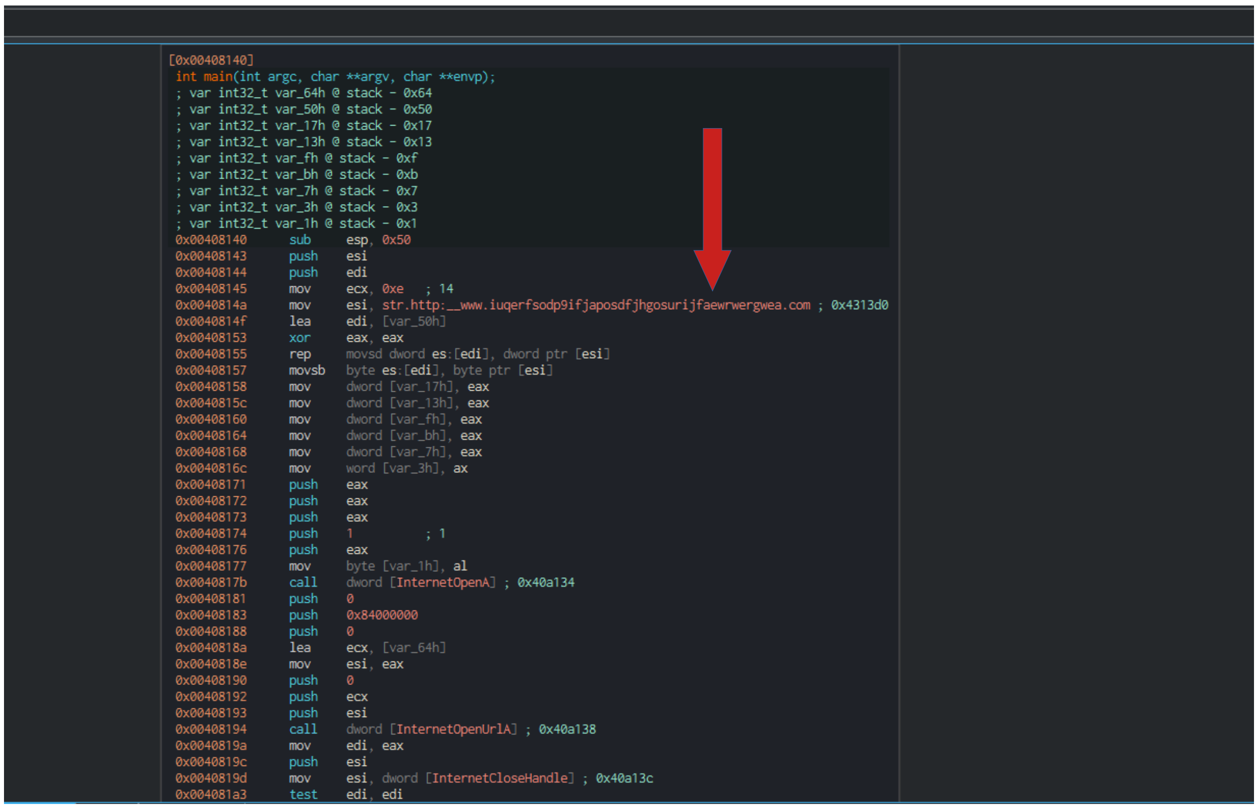


CryptGenRandom	x	0x0000A650	0x0000A650	150 (0x0096)	crypto | obfuscation
CryptAcquireContextA	x	0x0000A638	0x0000A638	133 (0x0085)	crypto | obfuscation
rand	x	0x0000A824	0x0000A824	678 (0x02A6)	crypto | obfuscation
srand	x	0x0000A852	0x0000A852	692 (0x02B4)	crypto | obfuscation

The **CryptGenRandom** function is part of the Windows Cryptography API and is used to generate cryptographically secure random numbers. It fills a buffer with random bytes, which can be used for various cryptographic operations such as key generation, nonce creation, or other purposes where randomness is required.

The **CryptAcquireContext** function is part of the Windows Cryptography API, which is used to acquire a handle to a particular key container within a cryptographic service provider (CSP). This handle is then used in subsequent calls to other cryptographic functions.

Tool used disassembler cutter :

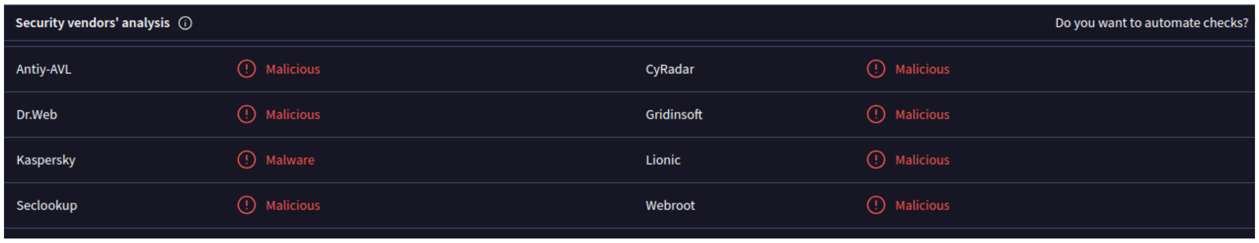


```
[0x00408140]
 int main(int argc, char **argv, char **envp);
 ; var int32_t var_64h @ stack - 0x64
 ; var int32_t var_50h @ stack - 0x50
 ; var int32_t var_17h @ stack - 0x17
 ; var int32_t var_13h @ stack - 0x13
 ; var int32_t var_fh @ stack - 0xf
 ; var int32_t var_bh @ stack - 0xb
 ; var int32_t var_7h @ stack - 0x7
 ; var int32_t var_3h @ stack - 0x3
 ; var int32_t var_1h @ stack - 0x1
 0x00408140      sub     esp, 0x50
 0x00408143      push    esi
 0x00408144      push    edi
 0x00408145      mov     ecx, 0xe   ; 14
 0x0040814a      mov     esi, str.http:__www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com ; 0x4313d0
 0x0040814f      lea     edi, [var_50h]
 0x00408153      xor     eax, eax
 0x00408155      rep     movsd dword es:[edi], dword ptr [esi]
 0x00408157      movsb   byte es:[edi], byte ptr [esi]
 0x00408158      mov     dword [var_17h], eax
 0x0040815c      mov     dword [var_13h], eax
 0x00408160      mov     dword [var_fh], eax
 0x00408164      mov     dword [var_bh], eax
 0x00408168      mov     dword [var_7h], eax
 0x0040816c      mov     word [var_3h], ax
 0x00408171      push    eax
 0x00408172      push    eax
 0x00408173      push    eax
 0x00408174      push    1          ; 1
 0x00408176      push    eax
 0x00408177      mov     byte [var_1h], al
 0x0040817b      call    dword [InternetOpenA] ; 0x40a134
 0x00408181      push    0
 0x00408183      push    0x84000000
 0x00408188      push    0
 0x0040818a      lea     ecx, [var_64h]
 0x0040818e      mov     esi, eax
 0x00408190      push    0
 0x00408192      push    ecx
 0x00408193      push    esi
 0x00408194      call    dword [InternetOpenUrlA] ; 0x40a138
 0x0040819a      mov     edi, eax
 0x0040819c      push    esi
 0x0040819d      mov     esi, dword [InternetCloseHandle] ; 0x40a13c
 0x004081a3      test    edi, edi
```

Now the first thing to notice is that the string referenced to the URL is loaded in ESI (Extended Source Index).

Its the same URL found in String analysis.

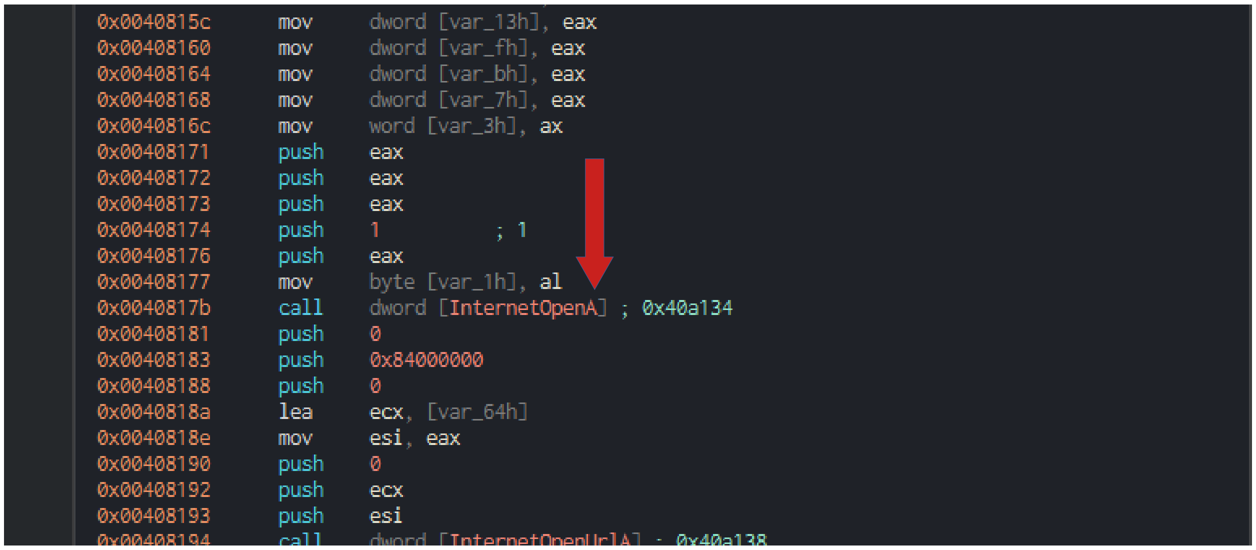
In virus total this string is flagged as a malicious Link.



Security vendors' analysis ⓘ — Do you want to automate checks?

Antiy-AVL	ⓘ Malicious	CyRadar	ⓘ Malicious
Dr.Web	ⓘ Malicious	Gridinsoft	ⓘ Malicious
Kaspersky	ⓘ Malware	Lionic	ⓘ Malicious
Seclookup	ⓘ Malicious	Webroot	ⓘ Malicious

The first API call is **[InternetOpenA]** is part of the Windows API for handling HTTP/HTTPS requests. This function is used to initialize an application's use of the WinINet API, which provides functions for internet access.

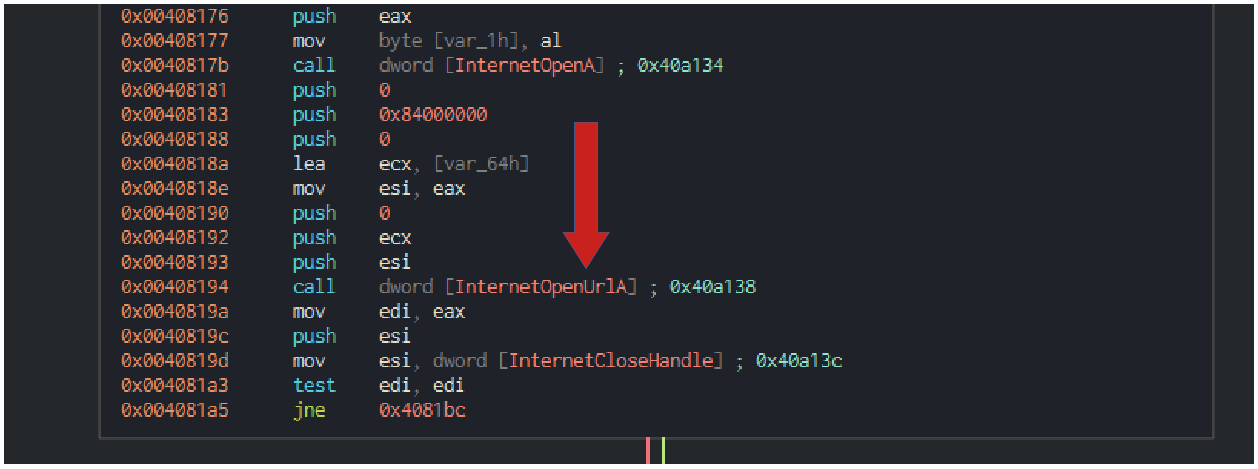


```
0x0040815c      mov     dword [var_13h], eax
0x00408160      mov     dword [var_fh], eax
0x00408164      mov     dword [var_bh], eax
0x00408168      mov     dword [var_7h], eax
0x0040816c      mov     word [var_3h], ax
0x00408171      push    eax
0x00408172      push    eax
0x00408173      push    eax
0x00408174      push    1          ; 1
0x00408176      push    eax
0x00408177      mov     byte [var_1h], al
0x0040817b      call    dword [InternetOpenA] ; 0x40a134
0x00408181      push    0
0x00408183      push    0x84000000
0x00408188      push    0
0x0040818a      lea     ecx, [var_64h]
0x0040818e      mov     esi, eax
0x00408190      push    0
0x00408192      push    ecx
0x00408193      push    esi
0x00408194      call    dword [InternetOpenUrlA] ; 0x40a138
```

In malware analysis, seeing **InternetOpenA** can indicate that the malware is attempting to establish internet connectivity, possibly for:

- Downloading additional payloads.
- Communicating with a command and control (C2) server.
- Sending Exfiltrated data.

InternetOpenA and its usage in both legitimate and malicious contexts is essential for effective network security and malware analysis.



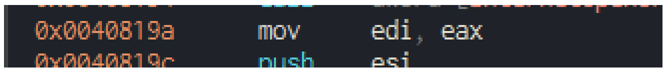
```
0x00408176      push    eax
0x00408177      mov     byte [var_1h], al
0x0040817b      call    dword [InternetOpenA] ; 0x40a134
0x00408181      push    0
0x00408183      push    0x84000000
0x00408188      push    0
0x0040818a      lea     ecx, [var_64h]
0x0040818e      mov     esi, eax
0x00408190      push    0
0x00408192      push    ecx
0x00408193      push    esi
0x00408194      call    dword [InternetOpenUrlA] ; 0x40a138
0x0040819a      mov     edi, eax
0x0040819c      push    esi
0x0040819d      mov     esi, dword [InternetCloseHandle] ; 0x40a13c
0x004081a3      test    edi, edi
0x004081a5      jne     0x4081bc
```

The content of ESI is pushed onto the stack which will be used as a parameter for the API call.Then after that [**InternetOpenUrlA**] API is called with the URL as a parameter.

The **InternetOpenUrlA** function is part of the Windows API and is used to open a URL and obtain a handle to the internet resource. This function is typically used after initializing an internet session with **InternetOpenA**.

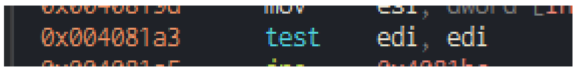
Now if the above API is able to connect to the URL then 0 is loaded in EAX (Extended Accumulator register) else 1 is loaded.

The value of EAX is moved to EDI (Extended Destination Index).



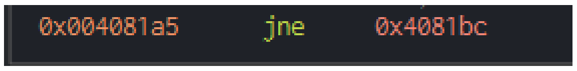
```
0x0040819a      mov     edi, eax
```

TEST edi , edi is ran which means bitwise boolean AND operator is used.



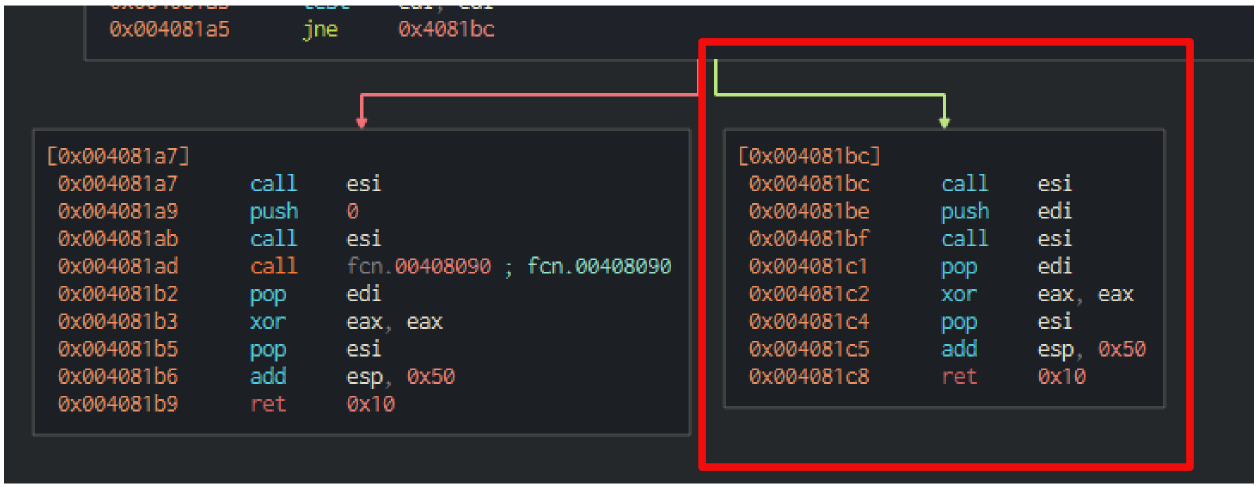
```
0x004081a3      test    edi, edi
```

Next a jne (jump if not equal) instructions ran



```
0x004081a5      jne     0x4081bc
```

IF the API is able to make a connection with the (hxxp://iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com/) then

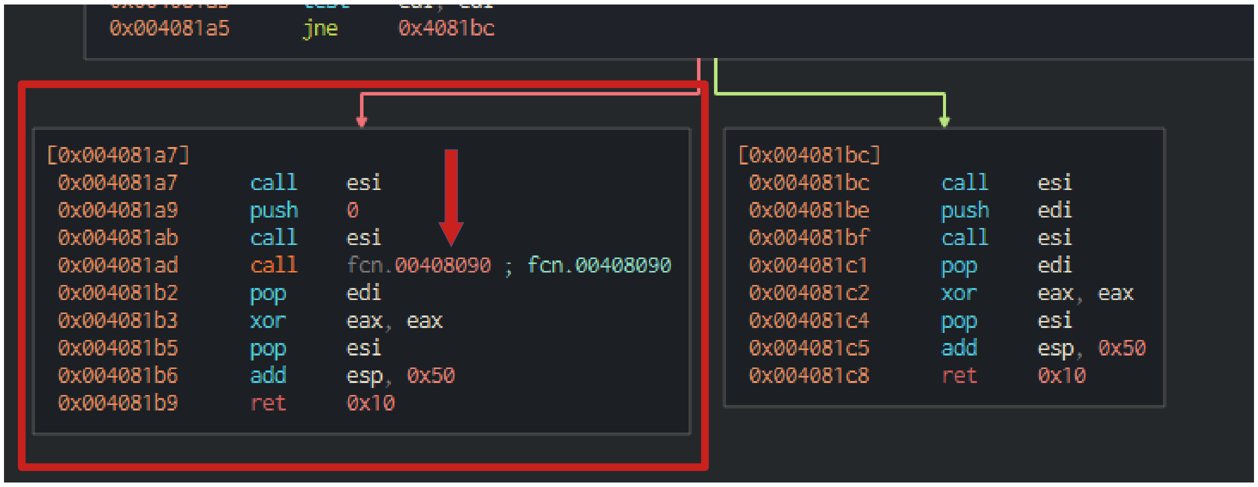


```
0x004081a5       jne      0x4081bc

[0x004081a7]                                               [0x004081bc]
 0x004081a7      call     esi                               0x004081bc      call     esi
 0x004081a9      push     0                                 0x004081be      push     edi
 0x004081ab      call     esi                               0x004081bf      call     esi
 0x004081ad      call     fcn.00408090 ; fcn.00408090       0x004081c1      pop      edi
 0x004081b2      pop      edi                               0x004081c2      xor      eax, eax
 0x004081b3      xor      eax, eax                          0x004081c4      pop      esi
 0x004081b5      pop      esi                               0x004081c5      add      esp, 0x50
 0x004081b6      add      esp, 0x50                         0x004081c8      ret      0x10
 0x004081b9      ret      0x10
```

Then this part of the code runs which basically clear the stack and returns, which means malware does not run further and the victim is safe.

Now this is the kill switch url function if malware is able to connect to the url it acts as a kill switch.

A security researcher known as "MalwareTech" discovered this behavior and registered the domain, inadvertently activating the kill switch. This action significantly slowed the spread of the ransomware and prevented further infections. The domain registration essentially created a sinkhole, preventing the malware from executing its payload on affected systems.



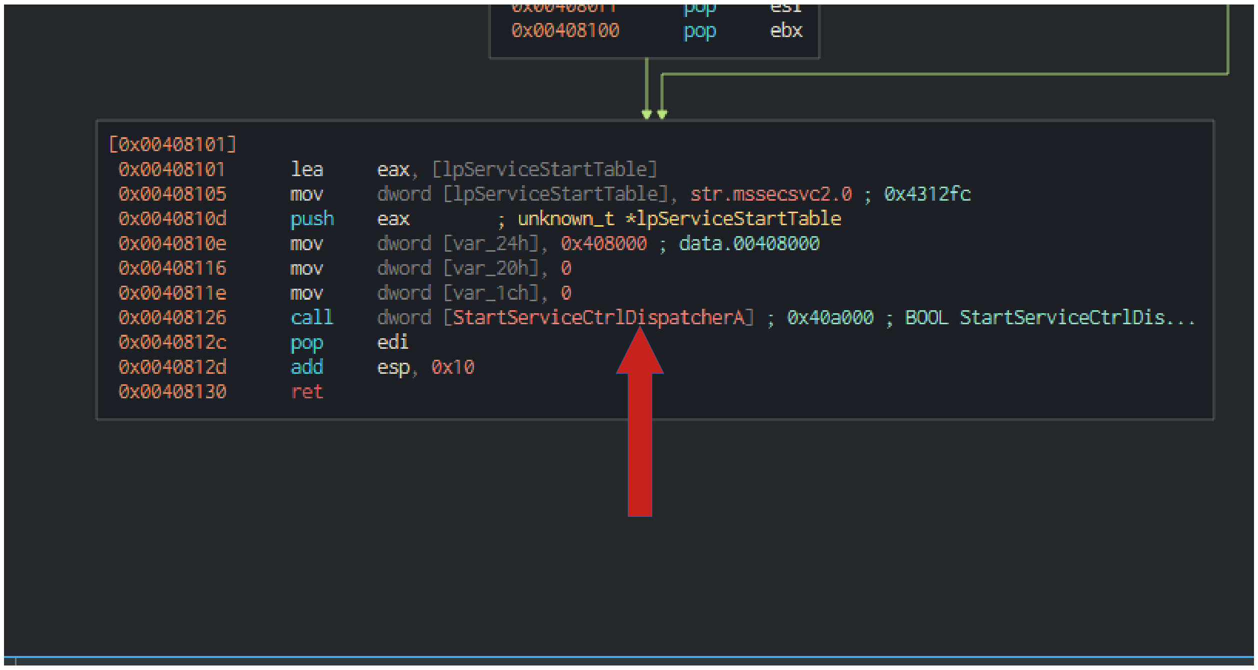
```
0x004081a5      jne     0x4081bc

[0x004081a7]                                          [0x004081bc]
 0x004081a7      call    esi                           0x004081bc      call    esi
 0x004081a9      push    0                             0x004081be      push    edi
 0x004081ab      call    esi                           0x004081bf      call    esi
 0x004081ad      call    fcn.00408090 ; fcn.00408090   0x004081c1      pop     edi
 0x004081b2      pop     edi                           0x004081c2      xor     eax, eax
 0x004081b3      xor     eax, eax                      0x004081c4      pop     esi
 0x004081b5      pop     esi                           0x004081c5      add     esp, 0x50
 0x004081b6      add     esp, 0x50                     0x004081c8      ret     0x10
 0x004081b9      ret     0x10
```

If the other part of code runs it calls a **function** and clears the stack.

This code is the rest of the encryption payload it open up and unpack the rest of the Unpacked Portable executable in a hidden directory.

It then install itself as a service and becomes persistent.

Every time a victim restart windows machine the service also starts up and encrypt any new files added to the machine.



```
0x00408100      pop     ebx

[0x00408101]
 0x00408101      lea     eax, [lpServiceStartTable]
 0x00408105      mov     dword [lpServiceStartTable], str.mssecsvc2.0 ; 0x4312fc
 0x0040810d      push    eax         ; unknown_t *lpServiceStartTable
 0x0040810e      mov     dword [var_24h], 0x408000 ; data.00408000
 0x00408116      mov     dword [var_20h], 0
 0x0040811e      mov     dword [var_1ch], 0
 0x00408126      call    dword [StartServiceCtrlDispatcherA] ; 0x40a000 ; BOOL StartServiceCtrlDis...
 0x0040812c      pop     edi
 0x0040812d      add     esp, 0x10
 0x00408130      ret
```

Dynamic Analysis

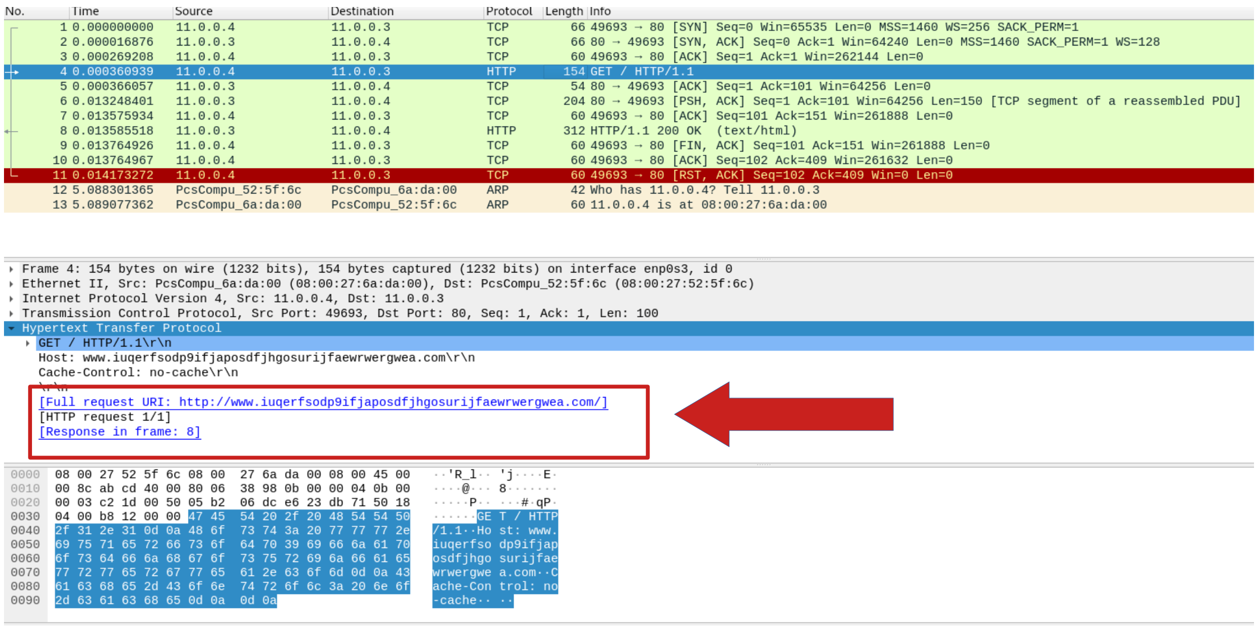
Tool used :- inetsim

First we will detonate the malware with inetsim on

When the malware is executed with inetsim set as our DNS resolver, the malware does not execute. It tries to connect to

“hxxp://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com”.

On successful connection it does not infect the system.



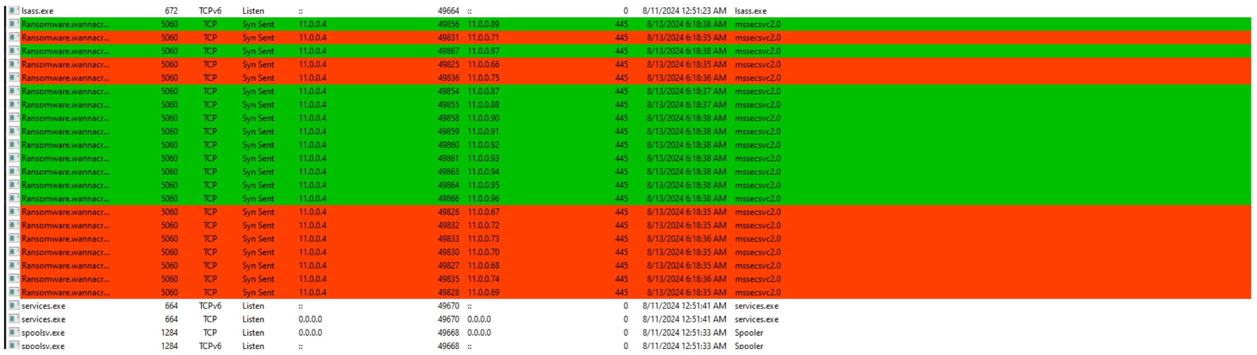
```
No.  Time         Source             Destination        Protocol Length Info
   1 0.000000000  11.0.0.4           11.0.0.3           TCP        66 49693 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
   2 0.000016876  11.0.0.3           11.0.0.4           TCP        66 80 → 49693 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
   3 0.000269208  11.0.0.4           11.0.0.3           TCP        60 49693 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
   4 0.000360939  11.0.0.4           11.0.0.3           HTTP      154 GET / HTTP/1.1
   5 0.000366057  11.0.0.3           11.0.0.4           TCP        54 80 → 49693 [ACK] Seq=1 Ack=101 Win=64256 Len=0
   6 0.013248401  11.0.0.3           11.0.0.4           TCP       204 80 → 49693 [PSH, ACK] Seq=1 Ack=101 Win=64256 Len=150 [TCP segment of a reassembled PDU]
   7 0.013575934  11.0.0.4           11.0.0.3           TCP        60 49693 → 80 [ACK] Seq=101 Ack=151 Win=261888 Len=0
   8 0.013585518  11.0.0.3           11.0.0.4           HTTP      312 HTTP/1.1 200 OK  (text/html)
   9 0.013764926  11.0.0.4           11.0.0.3           TCP        60 49693 → 80 [FIN, ACK] Seq=101 Ack=151 Win=261888 Len=0
  10 0.013764967  11.0.0.4           11.0.0.3           TCP        60 49693 → 80 [ACK] Seq=102 Ack=409 Win=261632 Len=0
  11 0.014173272  11.0.0.4           11.0.0.3           TCP        60 49693 → 80 [RST, ACK] Seq=102 Ack=409 Win=0 Len=0
  12 5.088301365  PcsCompu_52:5f:6c  PcsCompu_6a:da:00  ARP        42 Who has 11.0.0.4? Tell 11.0.0.3
  13 5.089077362  PcsCompu_6a:da:00  PcsCompu_52:5f:6c  ARP        60 11.0.0.4 is at 08:00:27:6a:da:00
```

```
▸ Frame 4: 154 bytes on wire (1232 bits), 154 bytes captured (1232 bits) on interface enp0s3, id 0
▸ Ethernet II, Src: PcsCompu_6a:da:00 (08:00:27:6a:da:00), Dst: PcsCompu_52:5f:6c (08:00:27:52:5f:6c)
▸ Internet Protocol Version 4, Src: 11.0.0.4, Dst: 11.0.0.3
▸ Transmission Control Protocol, Src Port: 49693, Dst Port: 80, Seq: 1, Ack: 1, Len: 100
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    Host: www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com\r\n
    Cache-Control: no-cache\r\n
    \r\n
    [Full request URI: http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com/]
    [HTTP request 1/1]
    [Response in frame: 8]
```

```
0000  08 00 27 52 5f 6c 08 00  27 6a da 00 08 00 45 00   ··'R_l·· 'j····E·
0010  00 8c ab cd 40 00 80 06  38 98 0b 00 00 04 0b 00   ····@··· 8·······
0020  00 03 c2 1d 00 50 05 b2  06 dc e6 23 db 71 50 18   ·····P·· ···#·qP·
0030  04 00 b8 12 00 00 47 45  54 20 2f 20 48 54 54 50   ······GE T / HTTP
0040  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 77 77 77 2e   /1.1··Ho st: www.
0050  69 75 71 65 72 66 73 6f  64 70 39 69 66 6a 61 70   iuqerfso dp9ifjap
0060  6f 73 64 66 6a 68 67 6f  73 75 72 69 6a 66 61 65   osdfjhgo surijfae
0070  77 72 77 65 72 67 77 65  61 2e 63 6f 6d 0d 0a 43   wrwergwe a.com··C
0080  61 63 68 65 2d 43 6f 6e  74 72 6f 6c 3a 20 6e 6f   ache-Con trol: no
0090  2d 63 61 63 68 65 0d 0a  0d 0a                     -cache·· ··
```

Now if we turn off the inetsim then the malware infects our system

Tool Tcp view :



```
lsass.exe               672   TCPv6  Listen    ::        49664  ::           0    8/11/2024 12:51:23 AM  lsass.exe
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49856  11.0.0.89    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49831  11.0.0.71    445  8/13/2024 6:18:35 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49867  11.0.0.97    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49825  11.0.0.66    445  8/13/2024 6:18:35 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49836  11.0.0.75    445  8/13/2024 6:18:36 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49854  11.0.0.87    445  8/13/2024 6:18:37 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49855  11.0.0.88    445  8/13/2024 6:18:37 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49858  11.0.0.90    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49859  11.0.0.91    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49860  11.0.0.92    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49861  11.0.0.93    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49863  11.0.0.94    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49864  11.0.0.95    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49866  11.0.0.96    445  8/13/2024 6:18:38 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49826  11.0.0.67    445  8/13/2024 6:18:35 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49832  11.0.0.72    445  8/13/2024 6:18:35 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49833  11.0.0.73    445  8/13/2024 6:18:36 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49830  11.0.0.70    445  8/13/2024 6:18:35 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49827  11.0.0.68    445  8/13/2024 6:18:35 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49835  11.0.0.74    445  8/13/2024 6:18:36 AM   mssecsvc2.0
Ransomware.wannacr...   5060  TCP    Syn Sent  11.0.0.4  49828  11.0.0.69    445  8/13/2024 6:18:35 AM   mssecsvc2.0
services.exe            664   TCPv6  Listen    ::        49670  ::           0    8/11/2024 12:51:41 AM  services.exe
services.exe            664   TCP    Listen    0.0.0.0   49670  0.0.0.0      0    8/11/2024 12:51:41 AM  services.exe
spoolsv.exe             1284  TCP    Listen    0.0.0.0   49668  0.0.0.0      0    8/11/2024 12:51:33 AM  Spooler
spoolsv.exe             1284  TCPv6  Listen    ::        49668  ::           0    8/11/2024 12:51:33 AM  Spooler
```

Here we see that wanna cry is trying to target Port 445, which is primarily used for SMB its a network file sharing protocol that allows applications to read and write to files and request services from server programs.

Wanna Cry exploited a vulnerability in the Server Message Block (SMB) protocol. This vulnerability, known as Eternal Blue.

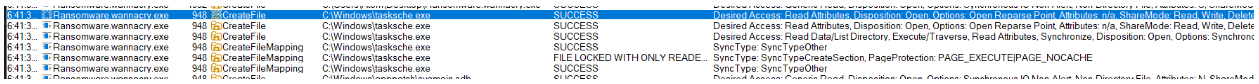
Eternal Blue allowed the ransomware to spread quickly across networks by taking advantage of unpatched Windows systems.

Worm-Like Behavior:

Once Wanna Cry infected a system, it used the SMB vulnerability to scan and infect other vulnerable systems on the same network. This worm-like behavior enabled it to spread rapidly from one infected machine to others within the same network or across connected networks.

CVE-2017-0144: This is the specific CVE identifier for the vulnerability.

Tool used procmon

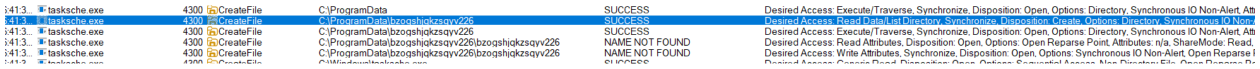


```
6:41:3...  Ransomware.wannacry.exe  948  CreateFile         C:\Windows\tasksche.exe  SUCCESS                       Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete
6:41:3...  Ransomware.wannacry.exe  948  CreateFile         C:\Windows\tasksche.exe  SUCCESS                       Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete
6:41:3...  Ransomware.wannacry.exe  948  CreateFile         C:\Windows\tasksche.exe  SUCCESS                       Desired Access: Read Data/List Directory, Execute/Traverse, Read Attributes, Synchronize, Disposition: Open, Options: Synchron
6:41:3...  Ransomware.wannacry.exe  948  CreateFileMapping  C:\Windows\tasksche.exe  SUCCESS                       SyncType: SyncTypeOther
6:41:3...  Ransomware.wannacry.exe  948  CreateFileMapping  C:\Windows\tasksche.exe  FILE LOCKED WITH ONLY READE...  SyncType: SyncTypeCreateSection, PageProtection: PAGE_EXECUTE|PAGE_NOCACHE
6:41:3...  Ransomware.wannacry.exe  948  CreateFileMapping  C:\Windows\tasksche.exe  SUCCESS                       SyncType: SyncTypeOther
```

we see a file named tasksche.exe is created which was also found in static analysis.

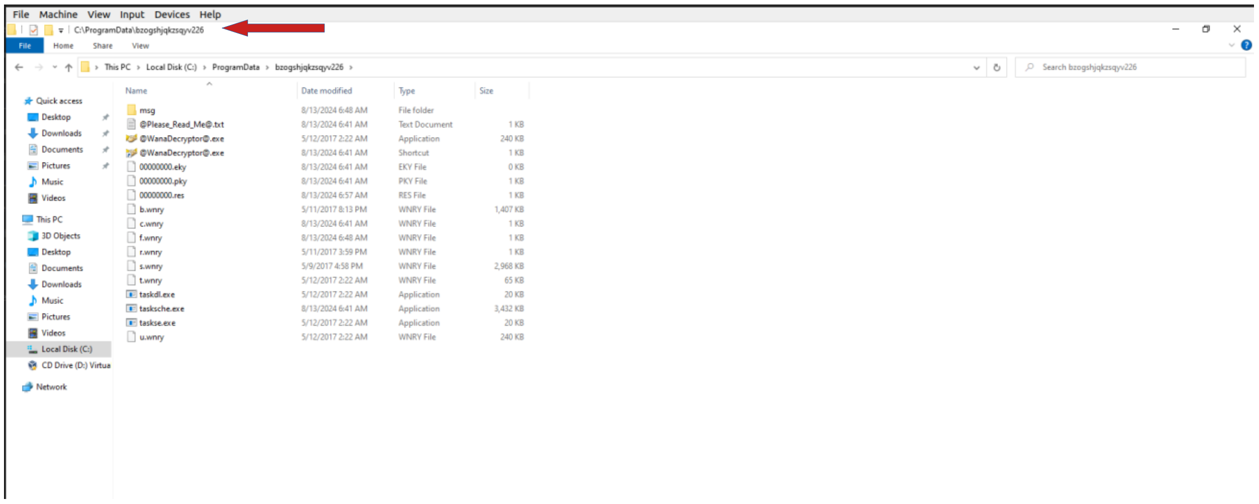
By applying filters in procmon for parent id we will see all child processes spawned by our malware.

Here we notice that a directory is created so lets check its content.



```
:41:3... tasksche.exe   4300 CreateFile   C:\ProgramData                                        SUCCESS          Desired Access: Execute/Traverse, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Att
:41:3... tasksche.exe   4300 CreateFile   C:\ProgramData\bzogshjqkzsqyv226                      SUCCESS          Desired Access: Read Data/List Directory, Synchronize, Disposition: Create, Options: Directory, Synchronous IO Non-
:41:3... tasksche.exe   4300 CreateFile   C:\ProgramData\bzogshjqkzsqyv226                      SUCCESS          Desired Access: Execute/Traverse, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Att
:41:3... tasksche.exe   4300 CreateFile   C:\ProgramData\bzogshjqkzsqyv226\bzogshjqkzsqyv226    NAME NOT FOUND   Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read,
:41:3... tasksche.exe   4300 CreateFile   C:\ProgramData\bzogshjqkzsqyv226\bzogshjqkzsqyv226    NAME NOT FOUND   Desired Access: Write Attributes, Synchronize, Disposition: Open, Options: Synchronous IO Non-Alert, Open Reparse F
```

This is the staging area of wanna cry



Name	Date modified	Type	Size
msg	8/13/2024 6:48 AM	File folder	
@Please_Read_Me@.txt	8/13/2024 6:41 AM	Text Document	1 KB
@WanaDecryptor@.exe	5/12/2017 2:22 AM	Application	240 KB
@WanaDecryptor@.exe	8/13/2024 6:41 AM	Shortcut	1 KB
00000000.eky	8/13/2024 6:41 AM	EKY File	0 KB
00000000.pky	8/13/2024 6:41 AM	PKY File	1 KB
00000000.res	8/13/2024 6:57 AM	RES File	1 KB
b.wnry	5/11/2017 8:13 PM	WNRY File	1,407 KB
c.wnry	8/13/2024 6:41 AM	WNRY File	1 KB
f.wnry	8/13/2024 6:48 AM	WNRY File	1 KB
r.wnry	5/11/2017 3:59 PM	WNRY File	1 KB
s.wnry	5/9/2017 4:58 PM	WNRY File	2,968 KB
t.wnry	5/12/2017 2:22 AM	WNRY File	65 KB
taskdl.exe	5/12/2017 2:22 AM	Application	20 KB
tasksche.exe	8/13/2024 6:41 AM	Application	3,432 KB
taskse.exe	5/12/2017 2:22 AM	Application	20 KB
u.wnry	5/12/2017 2:22 AM	WNRY File	240 KB

b.wnry is an image file used for displaying instructionsfor the decryption of user files. It starts with 42 4Dstrings, which indicates that this file is a bitmap image

c.wnry contains a list of Tor addresses with .onionextension and a link to a zipped installation file of theTor browser from Tor Project

r.wnry is a text file in English with additional de-cryption instructions to be used by the decryptioncomponent

s.wnry file is a ZIP archive (HEX signature 50 4B 0304) which contains the Tor software executable. Thisexecutable has been obtained with the assistance of the WinHex tool [12] by saving raw binary data with.zip extension.

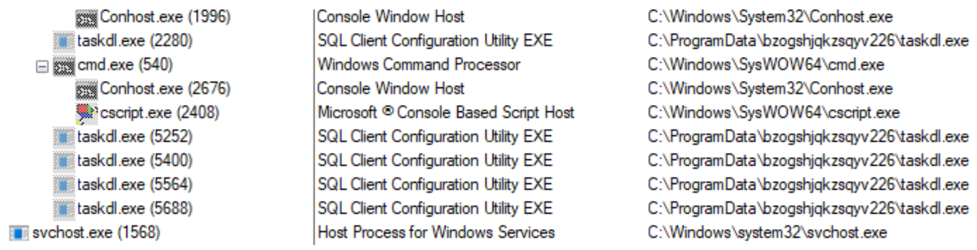
t.wnry — Encrypted DLL containing file-encryption functionality.

u.wnry — Main module of the WCry ransomware “decryptor”.

taskse.exe — Program that displays decryptor window to RDP sessions.

msg — Directory containing Rich Text Format (RTF) ransom demands in multiple languages.

taskdl.exe — WNCRYT temporary file cleanup program.



```
        Conhost.exe (1996)      Console Window Host                   C:\Windows\System32\Conhost.exe
      taskdl.exe (2280)         SQL Client Configuration Utility EXE  C:\ProgramData\bzogshjqkzsqyv226\taskdl.exe
    ⊟ cmd.exe (540)             Windows Command Processor             C:\Windows\SysWOW64\cmd.exe
        Conhost.exe (2676)      Console Window Host                   C:\Windows\System32\Conhost.exe
        cscript.exe (2408)      Microsoft ® Console Based Script Host C:\Windows\SysWOW64\cscript.exe
      taskdl.exe (5252)         SQL Client Configuration Utility EXE  C:\ProgramData\bzogshjqkzsqyv226\taskdl.exe
      taskdl.exe (5400)         SQL Client Configuration Utility EXE  C:\ProgramData\bzogshjqkzsqyv226\taskdl.exe
      taskdl.exe (5564)         SQL Client Configuration Utility EXE  C:\ProgramData\bzogshjqkzsqyv226\taskdl.exe
      taskdl.exe (5688)         SQL Client Configuration Utility EXE  C:\ProgramData\bzogshjqkzsqyv226\taskdl.exe
  svchost.exe (1568)            Host Process for Windows Services     C:\Windows\system32\svchost.exe
```

Here we can see that taskdl.exe is trying to end SQL Client Configuration utility executable so sql data can also be encrypted.

Similarly WCry terminates several services so that their data stores can be encrypted:

taskkill.exe /f /im mysqld.exe

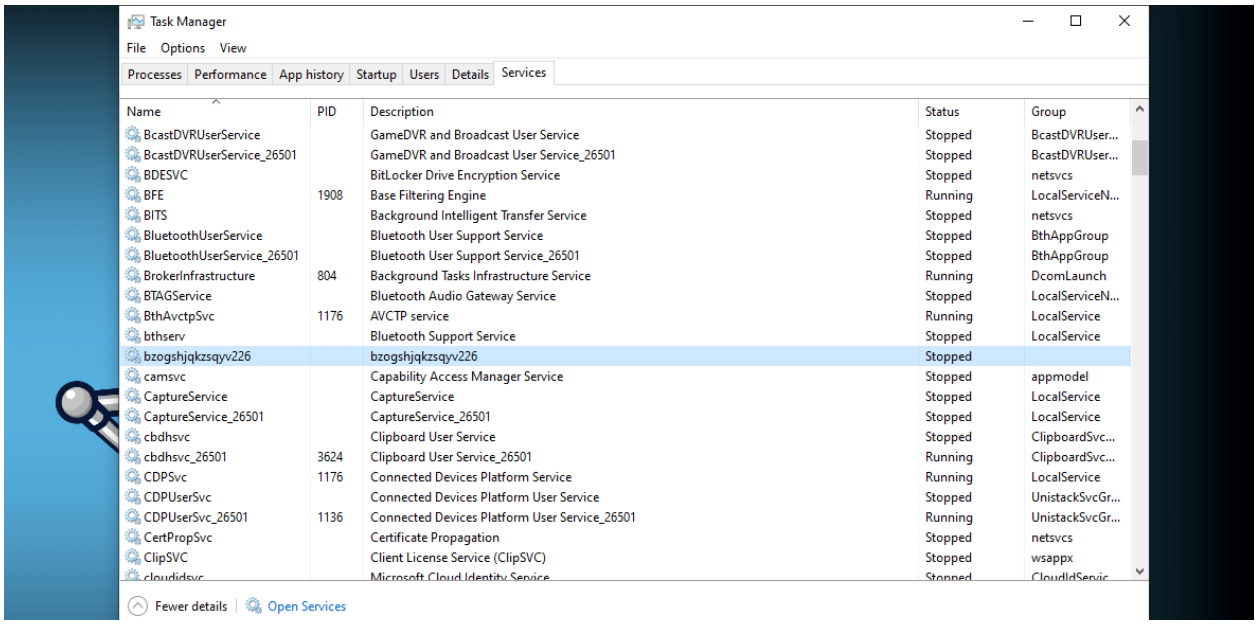
taskkill.exe /f /im sqlwriter.exe

taskkill.exe /f /im sqlserver.exe

taskkill.exe /f /im MSExchange

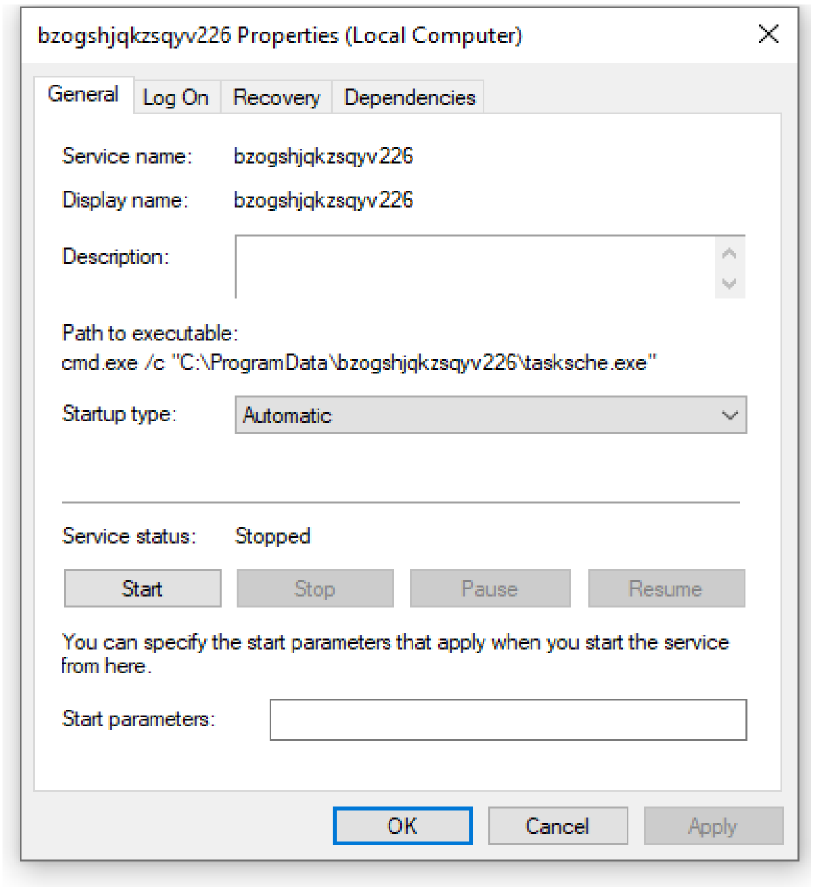
taskkill.exe /f /im Microsoft.Exchange.

A service with the same name is the directory is also created its a persistence mechanism so if the victim adds new files to the system it will also be encrypted.



Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Fewer details
Open Services

Name	PID	Description	Status	Group
BcastDVRUserService		GameDVR and Broadcast User Service	Stopped	BcastDVRUser...
BcastDVRUserService_26501		GameDVR and Broadcast User Service_26501	Stopped	BcastDVRUser...
BDESVC		BitLocker Drive Encryption Service	Stopped	netsvcs
BFE	1908	Base Filtering Engine	Running	LocalServiceN...
BITS		Background Intelligent Transfer Service	Stopped	netsvcs
BluetoothUserService		Bluetooth User Support Service	Stopped	BthAppGroup
BluetoothUserService_26501		Bluetooth User Support Service_26501	Stopped	BthAppGroup
BrokerInfrastructure	804	Background Tasks Infrastructure Service	Running	DcomLaunch
BTAGService		Bluetooth Audio Gateway Service	Stopped	LocalServiceN...
BthAvctpSvc	1176	AVCTP service	Running	LocalService
bthserv		Bluetooth Support Service	Stopped	LocalService
bzogshjqkzsqyv226		bzogshjqkzsqyv226	Stopped	
camsvc		Capability Access Manager Service	Stopped	appmodel
CaptureService		CaptureService	Stopped	LocalService
CaptureService_26501		CaptureService_26501	Stopped	LocalService
cbdhsvc		Clipboard User Service	Stopped	ClipboardSvc...
cbdhsvc_26501	3624	Clipboard User Service_26501	Running	ClipboardSvc...
CDPSvc	1176	Connected Devices Platform Service	Running	LocalService
CDPUserSvc		Connected Devices Platform User Service	Stopped	UnistackSvcGr...
CDPUserSvc_26501	1136	Connected Devices Platform User Service_26501	Running	UnistackSvcGr...
CertPropSvc		Certificate Propagation	Stopped	netsvcs
ClipSVC		Client License Service (ClipSVC)	Stopped	wsappx
cloudidsvc		Microsoft Cloud Identity Service	Stopped	CloudIdServic



bzogshjqkzsqyv226 Properties (Local Computer)
General Log On Recovery Dependencies
Start Stop Pause Resume
OK Cancel Apply

Service name: bzogshjqkzsqyv226

Display name: bzogshjqkzsqyv226

Description:

Path to executable:
cmd.exe /c "C:\ProgramData\bzogshjqkzsqyv226\tasksche.exe"

Startup type: Automatic

Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters:

The startup type of this service is automatic.

You can disable this service from windows services

Tool used x32 debugger :

Key Steps in Malware Analysis Using x32dbg

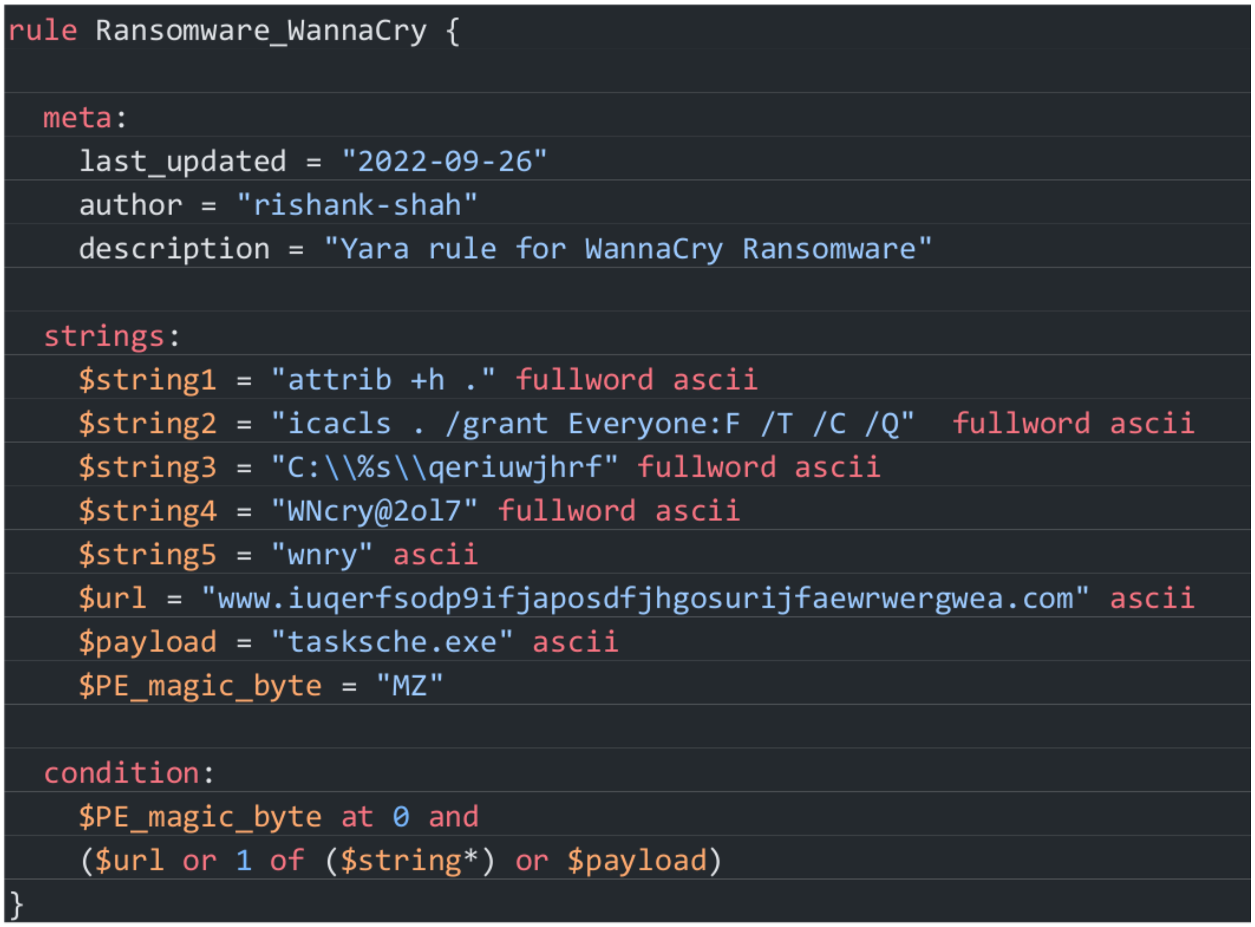
1. **Setting Up a Safe Environment**:
   - **Virtual Machine**: Always analyze malware in a virtual machine (VM) to prevent accidental infection of your primary system. Tools like VMware or VirtualBox are commonly used.
   - **Snapshots**: Take snapshots of your VM before starting the analysis so you can easily revert to a clean state.
2. **Initial Examination**:
   - **Static Analysis**: Before launching x32dbg, use static analysis tools like PEiD, PEview, or CFF Explorer to gather basic information about the malware, such as packers, imports, and section headers.
   - **Identify Entry Points**: Use these tools to identify the entry point of the executable, which will be useful when setting initial breakpoints.
3. **Loading Malware into x32dbg**:
   - Load the malware sample into x32dbg. Ensure that the process starts paused so you can set breakpoints before any malicious code executes.
4. **Setting Breakpoints**:
   - **Entry Point**: Set a breakpoint at the entry point of the malware to start analyzing its behavior from the very beginning.
   - **Imports**: Set breakpoints on important API calls that malware often uses, such as CreateProcess, WriteProcessMemory, RegSetValueEx, and networking functions like send or connect.
5. **Dynamic Analysis**:

- **Step Through Code**: Use single-step execution (F7) to walk through the code line by line. This helps in understanding the flow of execution and observing any suspicious behavior.
- **Function Calls**: Pay close attention to function calls, especially those that interact with the operating system or manipulate files, memory, or the registry.
- **Observing Registers**: Monitor CPU registers closely as they often contain important information like addresses, return values, or parameters passed to functions.

6. **Memory Inspection**:
   - **Memory Dump**: Inspect the memory dump to look for decrypted strings, injected code, or other indicators of malicious activity.
   - **Heap and Stack**: Analyze the heap and stack for any anomalies or patterns that might indicate malicious behavior, like unusual data being pushed onto the stack.
7. **Code Patching and Manipulation**:
   - **Bypass Anti-Debugging**: Many malware samples use anti-debugging techniques. x32dbg allows you to patch out these checks so that you can continue your analysis without the malware detecting the debugger.
   - **Modify Execution Flow**: You can patch the code to alter the execution flow, such as skipping over harmful code or forcing specific outcomes to see how the malware reacts.
8. **Behavior Analysis**:
   - **API Monitoring**: Observe how the malware interacts with system APIs, focusing on actions like file manipulation, registry changes, or network communications.
   - **Network Traffic**: If the malware connects to a remote server, analyze the traffic it generates. You can use tools like Wireshark alongside x32dbg for this purpose.

9. **Identifying Persistence Mechanisms**:
    - Malware often tries to establish persistence by modifying startup entries or dropping files in system directories. Track these activities by setting breakpoints on relevant functions like RegCreateKeyEx or WriteFile.
10. **Logging and Reporting**:
    - **Trace Logs**: Use x32dbg's logging features to keep a record of important function calls, register changes, and memory modifications.
    - **Document Findings**: Create detailed reports of your analysis, including screenshots, code snippets, and explanations of how the malware operates.

YARA RULES



```
rule Ransomware_WannaCry {

  meta:
    last_updated = "2022-09-26"
    author = "rishank-shah"
    description = "Yara rule for WannaCry Ransomware"

  strings:
    $string1 = "attrib +h ." fullword ascii
    $string2 = "icacls . /grant Everyone:F /T /C /Q"  fullword ascii
    $string3 = "C:\\%s\\qeriuwjhrf" fullword ascii
    $string4 = "WNcry@2ol7" fullword ascii
    $string5 = "wnry" ascii
    $url = "www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com" ascii
    $payload = "tasksche.exe" ascii
    $PE_magic_byte = "MZ"

  condition:
    $PE_magic_byte at 0 and
    ($url or 1 of ($string*) or $payload)
}
```

YARA rules check files and networks for patterns, scripts and signatures that indicate the presence of malicious software, which is often written in a simple but unique format. When a rule finds a characteristic or pattern that indicates a piece of malware, it alerts the appropriate person, who can then isolate or delete it

Conclusion

- Wanna Cry is an opportunistic ransomware family whose propagation methods allow it to spread quickly. CTU researchers recommend that clients implement the following best practices to mitigate the threat:
- Apply the Microsoft security updates for MS17-010, including the updates for the Windows XP and Windows Server 2003 legacy operating systems.
- Disable SMBv1 on systems where it is not necessary (e.g., hosts that do not need to communicate with Windows XP and Windows 2000 systems). Carefully evaluate the need for allowing SMBv1-capable systems on interconnected networks compared to the associated risks.
- Segment networks to isolate hosts that cannot be patched, and block SMBv1 from traversing those networks.
- Scan networks for the presence of the DoublePulsar backdoor using plugins for tools such as Nmap.
- Use network auditing tools to scan networks for hosts that are vulnerable to the vulnerabilities described in MS17-010.
- Filter emails containing potentially dangerous file types such as executables, scripts, or macro-enabled documents.
- Implement a backup strategy that includes storing data using offline backup media. Backups to locally connected, network-attached, or cloud-based storage are often insufficient because ransomware frequently accesses and encrypts files stored on these systems.

References

Books:

- Skoudis, E., & Liston, T. (2006). *Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses* (2nd ed.). Prentice Hall.
- Sikorski, M., & Honig, A. (2012). *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. No Starch Press.
- Stuttard, D., & Pinto, M. (2011). *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws*. Wiley.

Journal Articles:

- S Megira[1], A R Pangesti[1] and F W Wibowo[1] Published under licence by IOP Publishing Ltd
  Journal of Physics: Conference Series, Volume 1140, International Conference on Electrical, Electronic, Info+rmatics and Vocational Education (ICE-ELINVO 2018)13 September 2018, Yogyakarta Special Province, Republic of Indonesia**Citation** S Megira *et al* 2018 *J. Phys.: Conf. Ser.* **1140** 012042**DOI** 10.1088/1742-6596/1140/1/012042

Conference Paper:

- Malware Analysis October 2014

  DOI:10.13140/2.1.4750.6889 , Conference: Ethical Hacking , At: Nirma University

  https://www.researchgate.net/publication/267777154_Malware_Analysis

- A Survey on Malware Analysis Techniques: Static, Dynamic, Hybrid and Memory Analysis , September 2018 , International Journal on Advanced Science Engineering and Information Technology 8(4-2):1662 8(4-2):1662
  DOI:10.18517/ijaseit.8.4-2.6827 License CC BY 4.0
  https://www.researchgate.net/publication/328760930_A_Survey_on_Malware_Analysis_Techniques_Static_Dynamic_Hybrid_and_Memory_Analysis

Online Resources:

- TCM Security Academy. (n.d.). "Practical Malware Analysis & Triage." Retrieved from https://academy.tcm-sec.com/p/practical-malware-analysis-triage
- HuskyHacks. (n.d.). "Notes on Malware Analysis and Cybersecurity." Retrieved from https://notes.huskyhacks.dev/
- https://github.com/HuskyHacks/PMAT-labs