Practical Malware Analysis & Triage

Malware Analysis Report

WannaCry Malware

Feb 2023 | Gian Paris C. Agsam | v1.0

Table of Contents

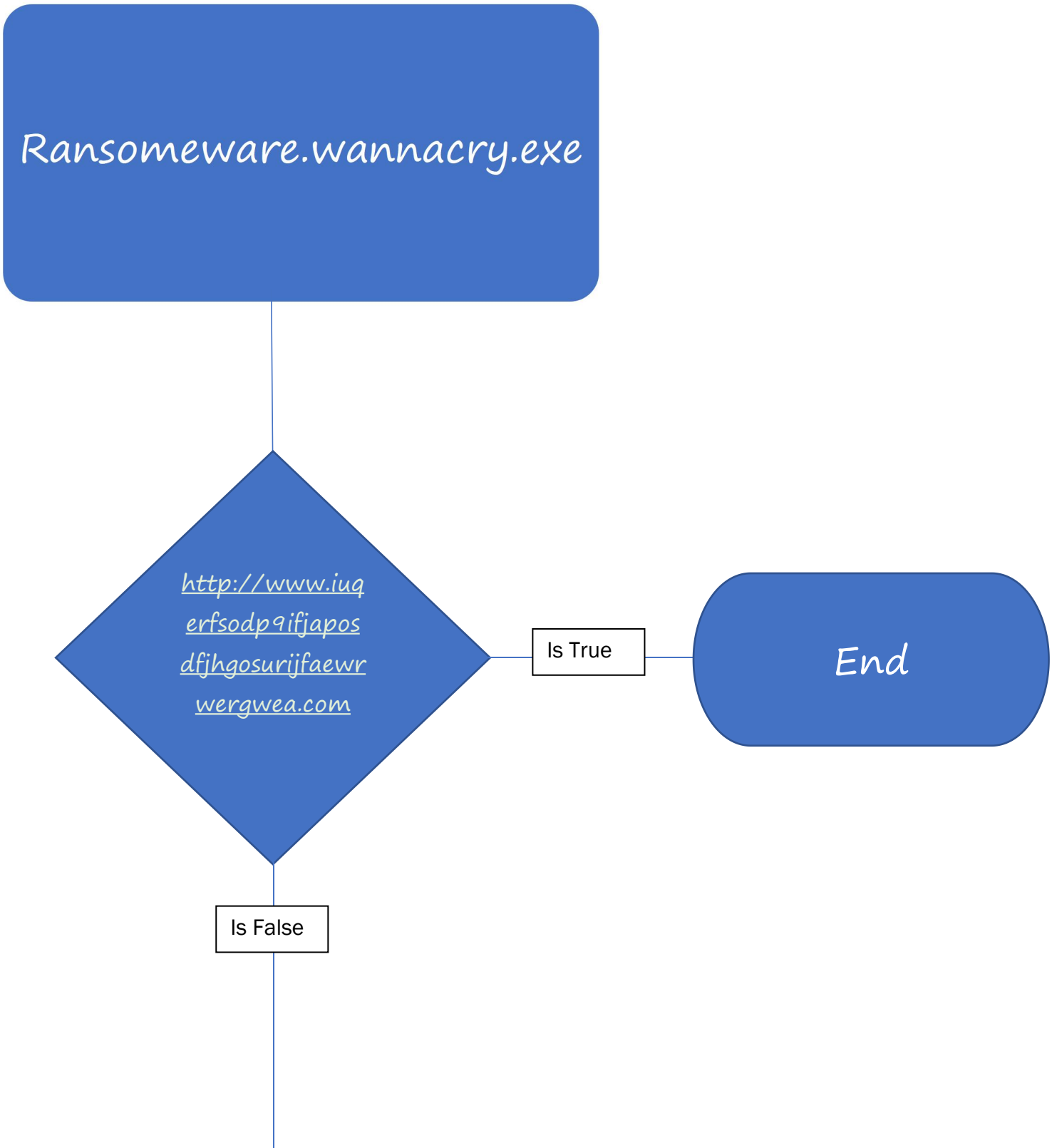
Executive Summary

SHA256 hash	24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614ea04703480b1022c

Wannacry is a ransomware malware sample first identified on May 12 2017. It is a Microsoft Visual C++ compiled that runs on the x32 Windows operating system. When the malware is run it encrypts your files and asks for ransom in crypto. When run malware spawns a process called tasksche.exe. However, wannacry malware only runs unless the return status to the url: http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com is not successful.

High-Level Technical Summary

WannaCry




Execute Malware
tasksche.exe
(Creates a files, folder to dump it's resources)
tasksvc.exe
(Opens port 9050 to a LISTENING state and attempts to connect to non-private remote addresses over HTTPS.)
End

Malware Composition

DemoWare consists of the following components:

File Name	SHA256 Hash
tasksche.exe	ED01EBFBC9EB5BBEA545AF4D01BF5F1071661840480439C6E5BABE8E080E41AA
taskdl.exe	4A468603FDCB7A2EB5770705898CF9EF37AADE532A7964642ECD705A74794B79
taskse.exe	2CA2D550E603D74DEDDA03156023135B38DA3630CB014E3D00B1263358C5F00DED01EB
taskhsvc.exe	E48673680746FBE027E8982F62A83C298D6FB46AD9243DE8E79B7E5A24DCD4EB

tasksche.exe

Creates directory and dumps it's resources to that directory. The generated directory has a random name.

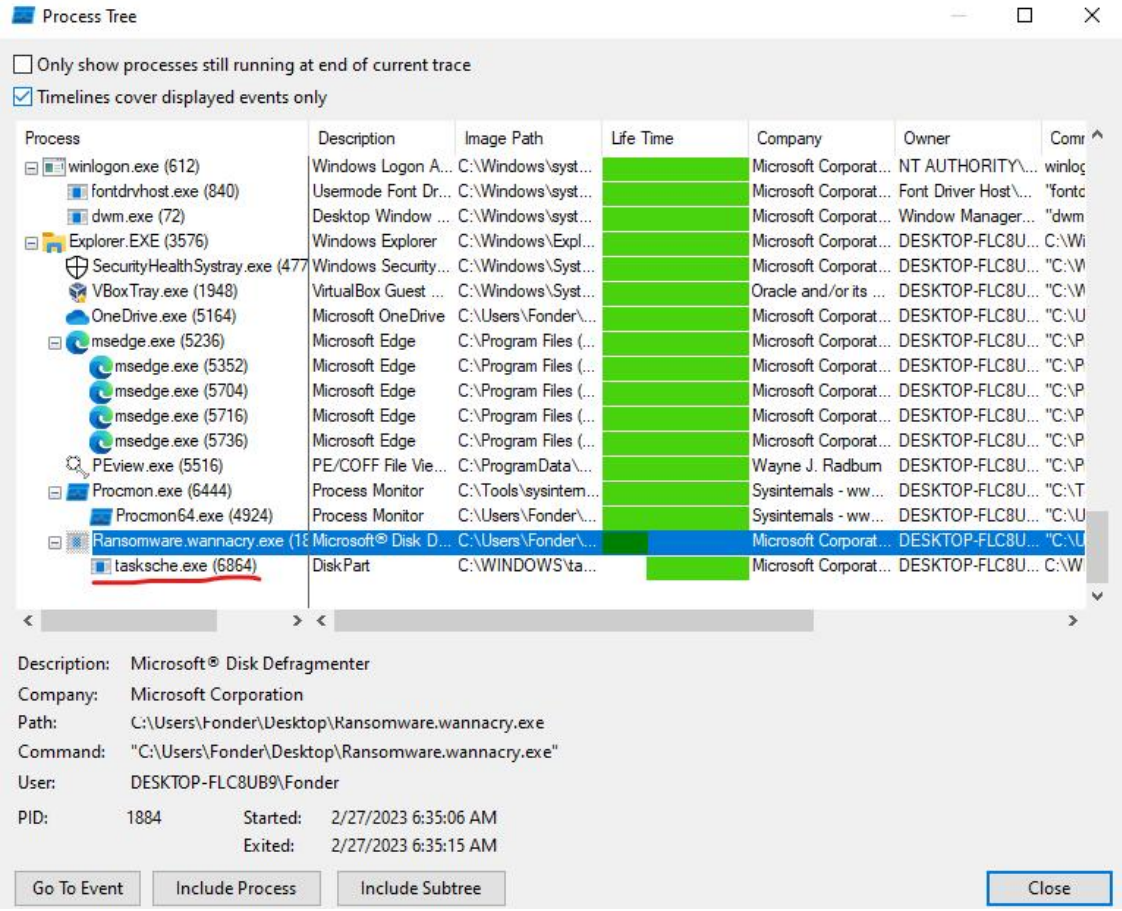


Fig 1:taskhsche.exe in processs tree

taskhsvc.exe:

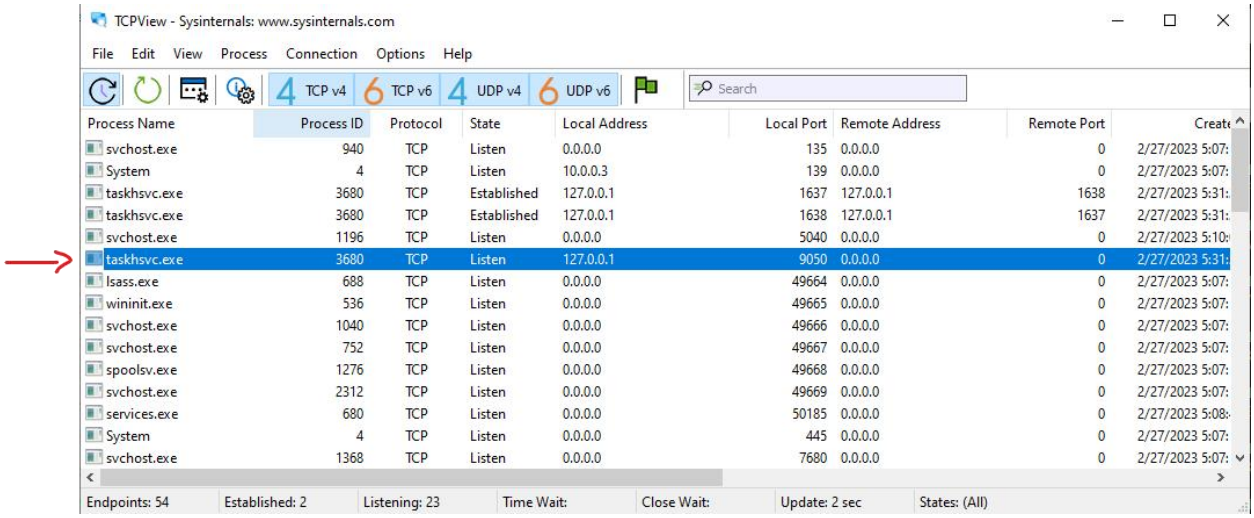
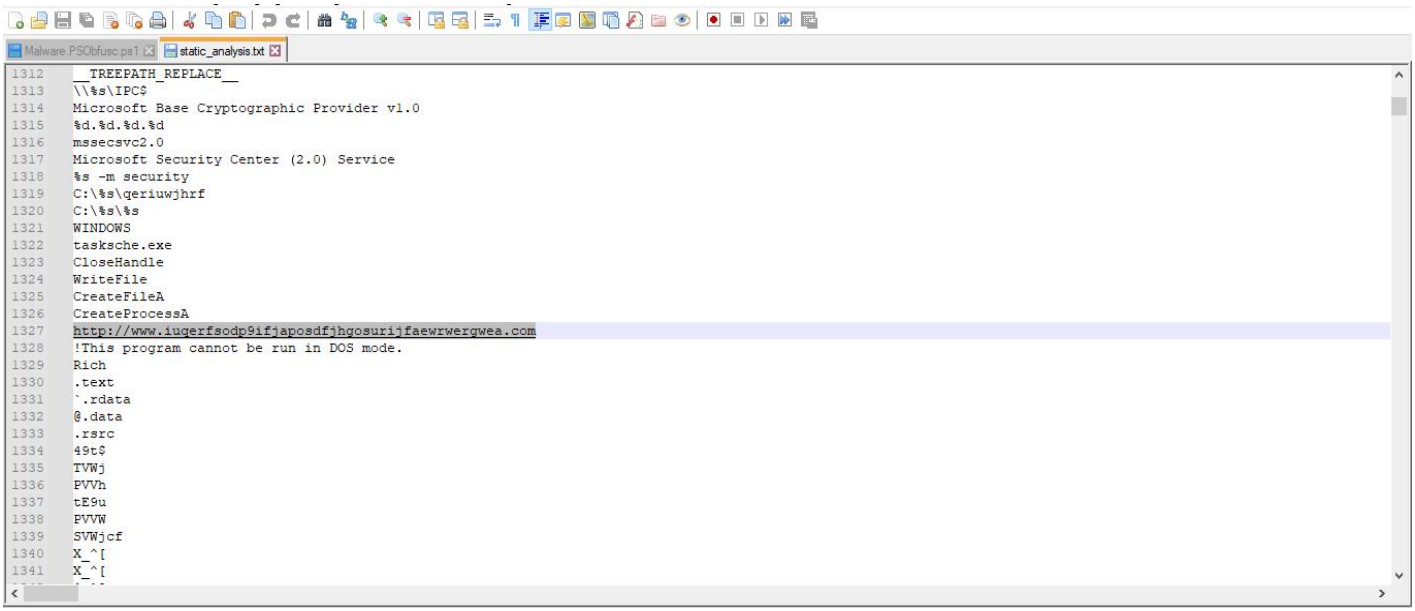


Fig 2:taskhsvc.exe listening on port 9050.

Basic Static Analysis

{Screenshots and description about basic static artifacts and methods}

Floss

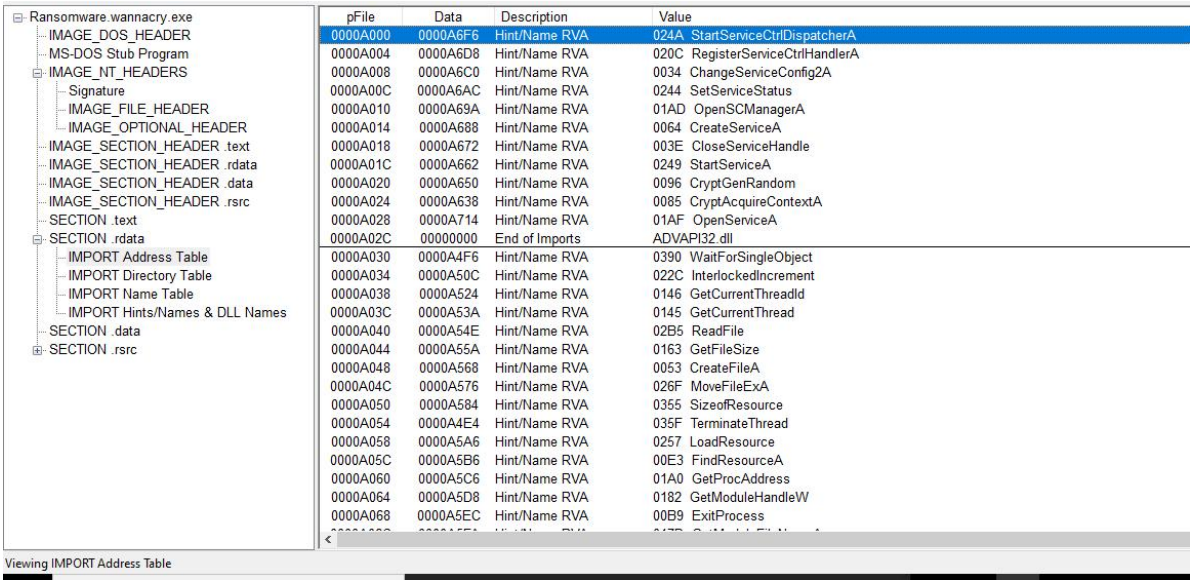


```
__TREEPATH_REPLACE__
\\%s\IPC$
Microsoft Base Cryptographic Provider v1.0
%d.%d.%d.%d
mssecsvc2.0
Microsoft Security Center (2.0) Service
%s -m security
C:\%s\qeriuwjhrf
C:\%s\%s
WINDOWS
tasksche.exe
CloseHandle
WriteFile
CreateFileA
CreateProcessA
http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com
!This program cannot be run in DOS mode.
Rich
.text
`.rdata
@.data
.rsrc
49t$
TVWj
PVVh
tE9u
PVVW
SVWjcf
X_^[
X_^[
```

Analyzed the binary with floss and got interesting strings but most importantly the url:

`http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com`

PEView



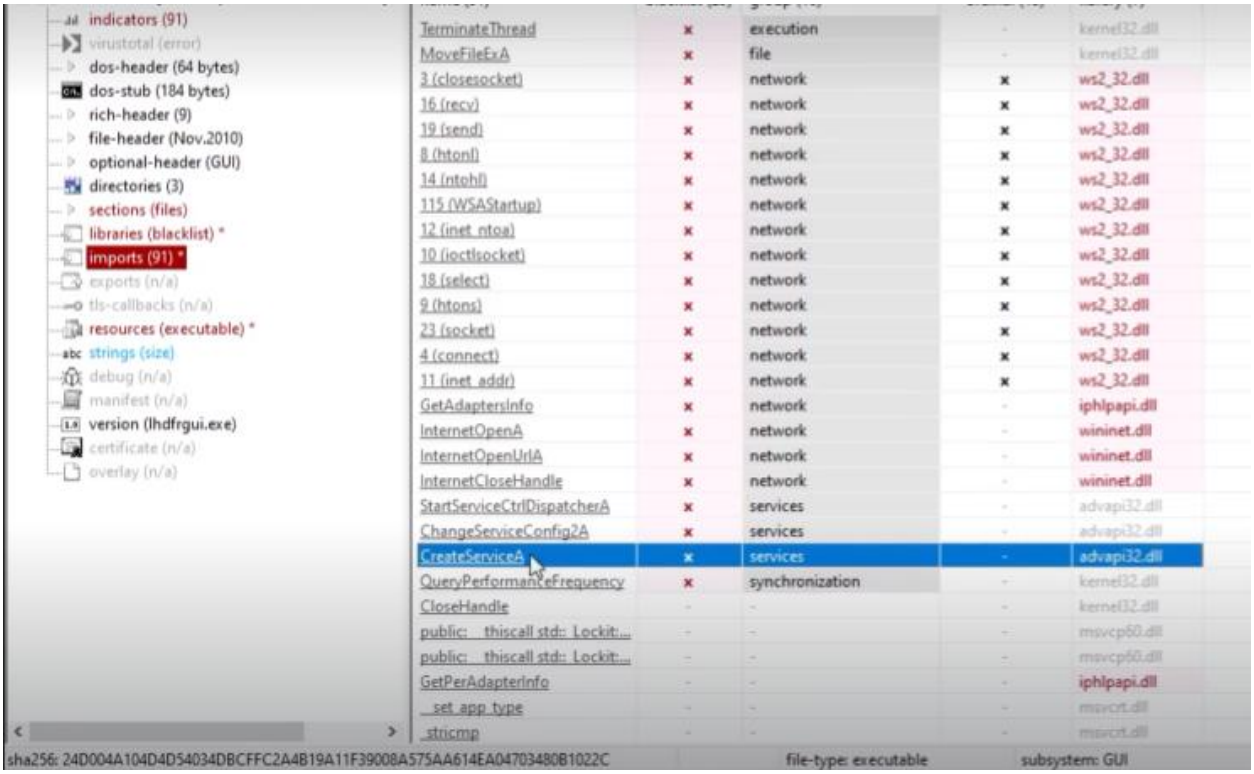
- Ransomware.wannacry.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
    - Signature
    - IMAGE_FILE_HEADER
    - IMAGE_OPTIONAL_HEADER
  - IMAGE_SECTION_HEADER .text
  - IMAGE_SECTION_HEADER .rdata
  - IMAGE_SECTION_HEADER .data
  - IMAGE_SECTION_HEADER .rsrc
  - SECTION .text
  - SECTION .rdata
    - IMPORT Address Table
    - IMPORT Directory Table
    - IMPORT Name Table
    - IMPORT Hints/Names & DLL Names
  - SECTION .data
  - SECTION .rsrc

pFile	Data	Description	Value
0000A000	0000A6F6	Hint/Name RVA	024A StartServiceCtrlDispatcherA
0000A004	0000A6D8	Hint/Name RVA	020C RegisterServiceCtrlHandlerA
0000A008	0000A6C0	Hint/Name RVA	0034 ChangeServiceConfig2A
0000A00C	0000A6AC	Hint/Name RVA	0244 SetServiceStatus
0000A010	0000A69A	Hint/Name RVA	01AD OpenSCManagerA
0000A014	0000A688	Hint/Name RVA	0064 CreateServiceA
0000A018	0000A672	Hint/Name RVA	003E CloseServiceHandle
0000A01C	0000A662	Hint/Name RVA	0249 StartServiceA
0000A020	0000A650	Hint/Name RVA	0096 CryptGenRandom
0000A024	0000A638	Hint/Name RVA	0085 CryptAcquireContextA
0000A028	0000A714	Hint/Name RVA	01AF OpenServiceA
0000A02C	00000000	End of Imports	ADVAPI32.dll
0000A030	0000A4F6	Hint/Name RVA	0390 WaitForSingleObject
0000A034	0000A50C	Hint/Name RVA	022C InterlockedIncrement
0000A038	0000A524	Hint/Name RVA	0146 GetCurrentThreadId
0000A03C	0000A53A	Hint/Name RVA	0145 GetCurrentThread
0000A040	0000A54E	Hint/Name RVA	02B5 ReadFile
0000A044	0000A55A	Hint/Name RVA	0163 GetFileSize
0000A048	0000A568	Hint/Name RVA	0053 CreateFileA
0000A04C	0000A576	Hint/Name RVA	026F MoveFileExA
0000A050	0000A584	Hint/Name RVA	0355 SizeofResource
0000A054	0000A4E4	Hint/Name RVA	035F TerminateThread
0000A058	0000A5A6	Hint/Name RVA	0257 LoadResource
0000A05C	0000A5B6	Hint/Name RVA	00E3 FindResourceA
0000A060	0000A5C6	Hint/Name RVA	01A0 GetProcAddress
0000A064	0000A5D8	Hint/Name RVA	0182 GetModuleHandleW
0000A068	0000A5EC	Hint/Name RVA	00B9 ExitProcess

Viewing IMPORT Address Table

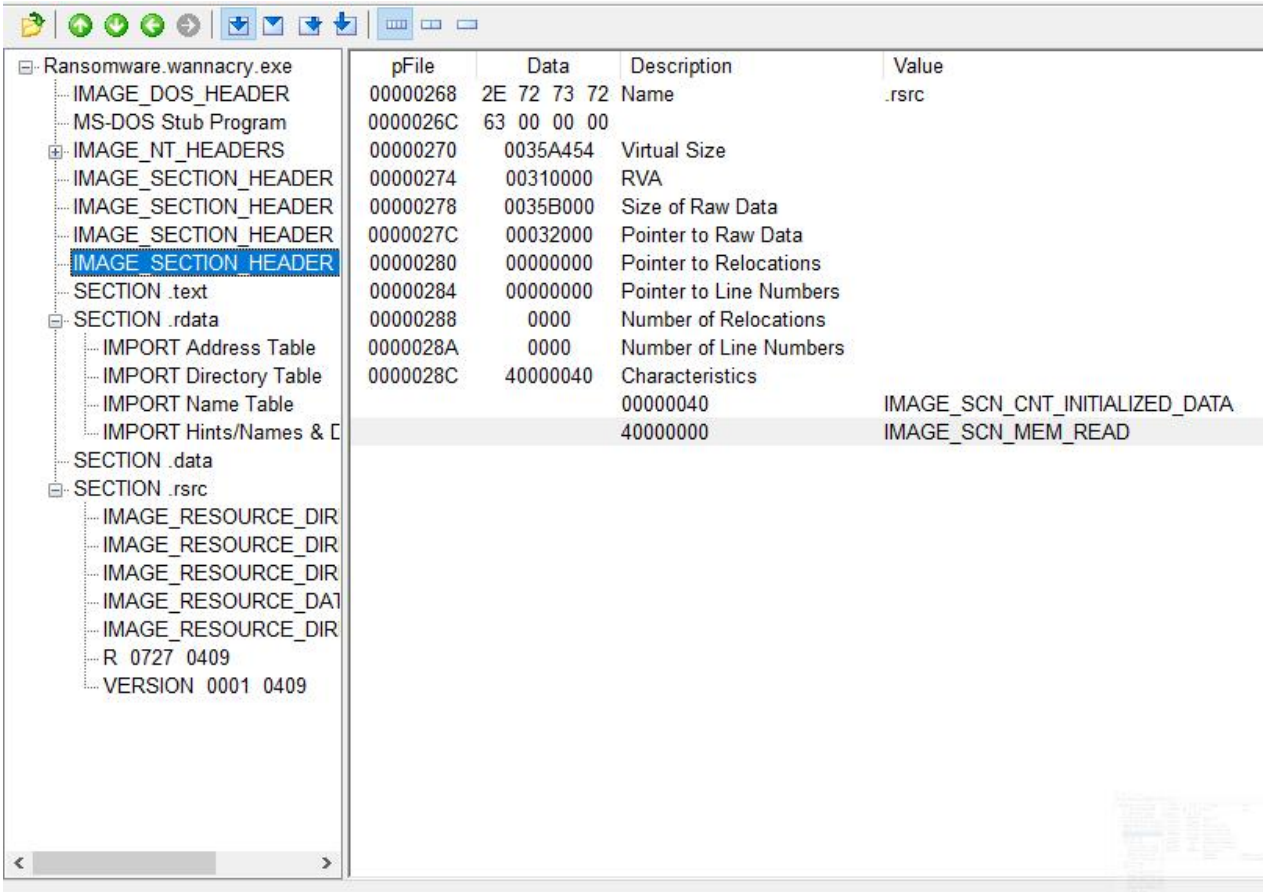
Import Address Table(IAT): API functions that the binary used.

PEStudio



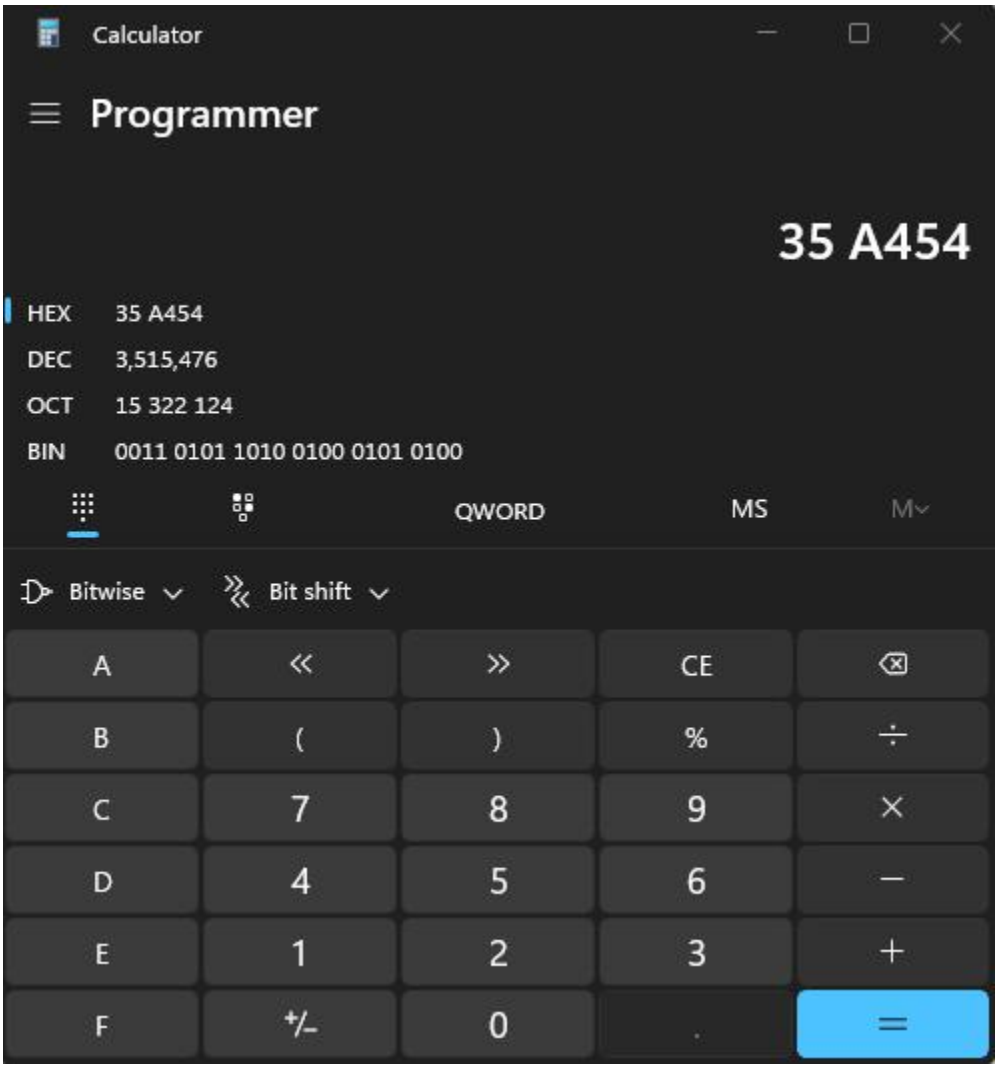
Imported functions flagged as malicious(Most likely).

Binary Packed or Not Packed

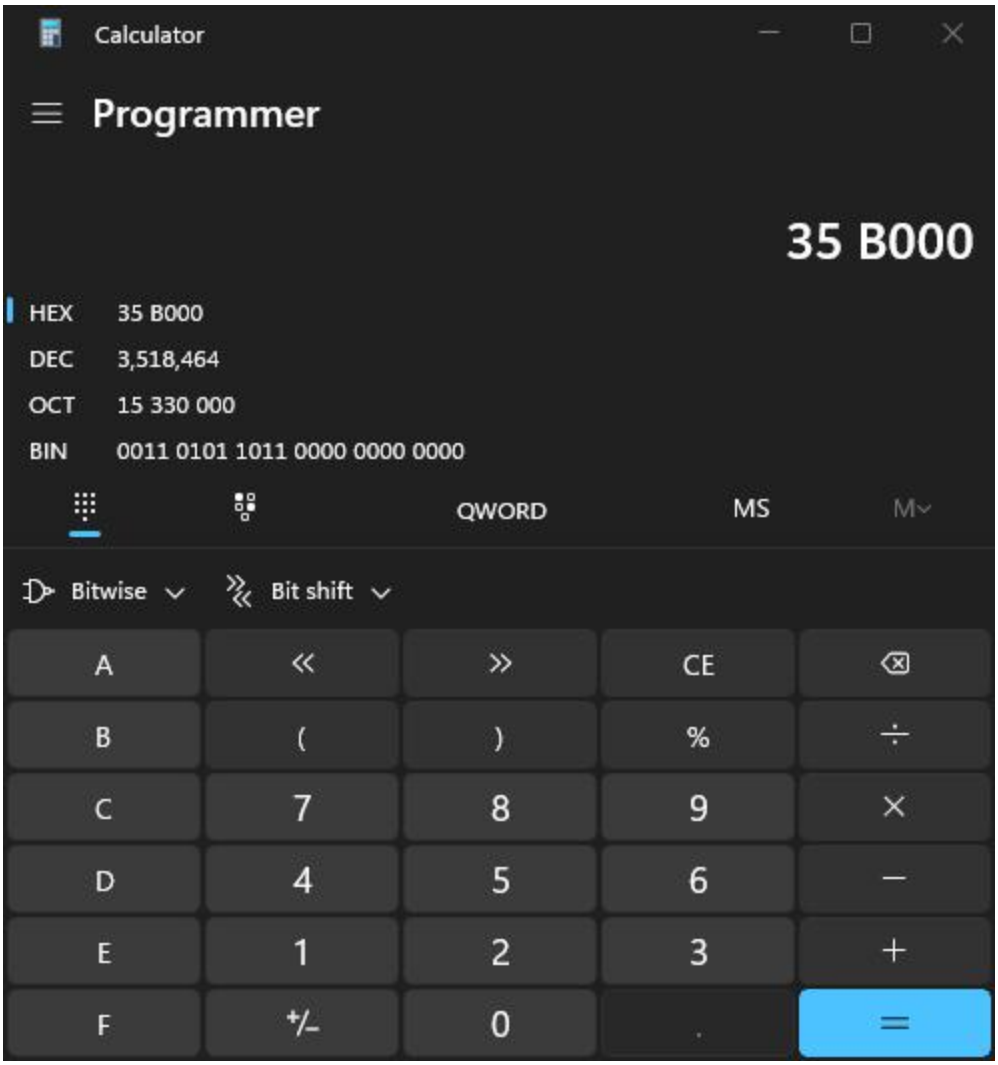


The virtual size and size of raw data are used as reference for the size of the binary.

If the sizes are far apart from each other then the binary is most likely packed.



Here the value of the virtual size is 3,515,476 bytes.

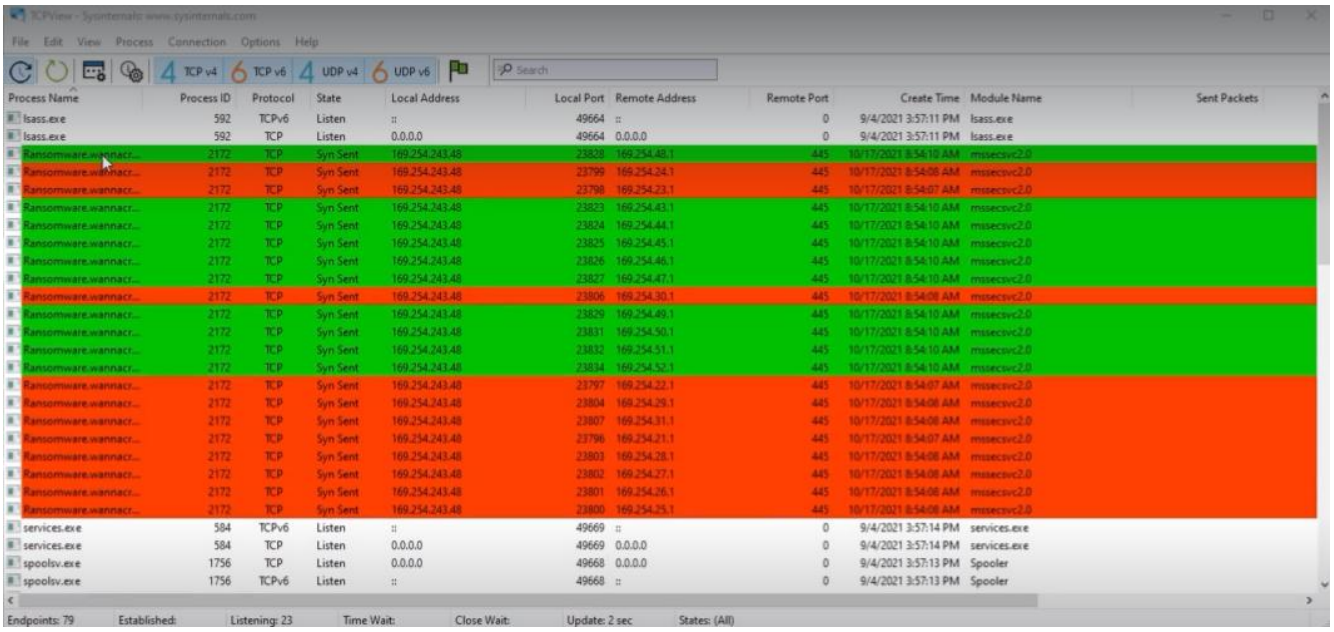


Here the value of the size of raw data is 3,518,464 bytes.

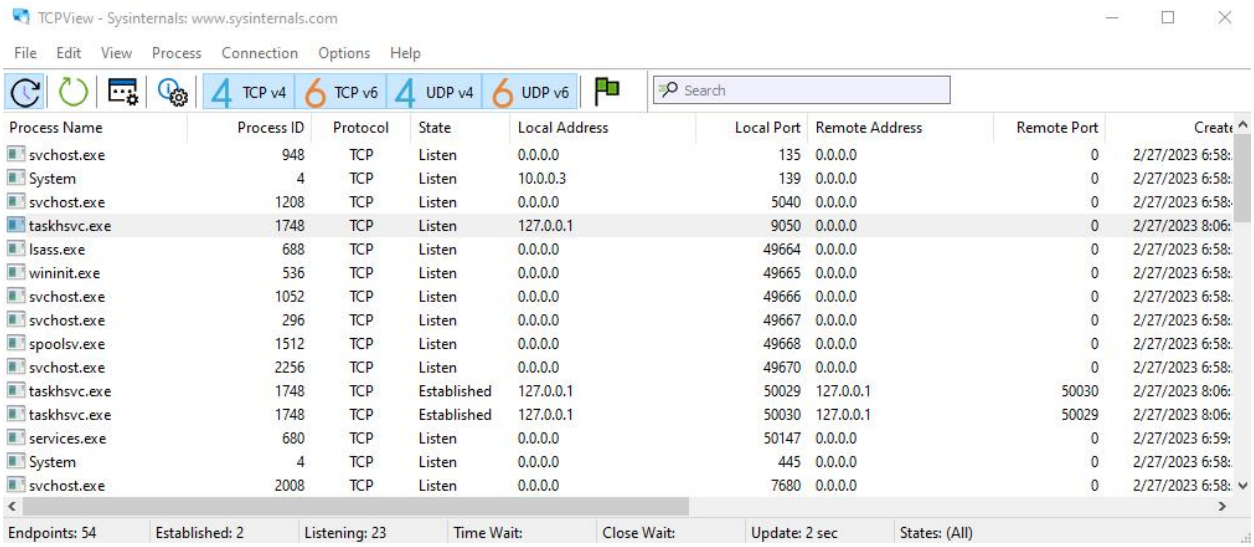
So in comparison the virtual size is 3,515,476 bytes and size of raw data is 3,518,464 bytes which is not too far apart from each other. So most likely this binary is not packed.

Basic Dynamic Analysis

TCPView



A flood of numerous SMB connection requests to non-private remote addresses.

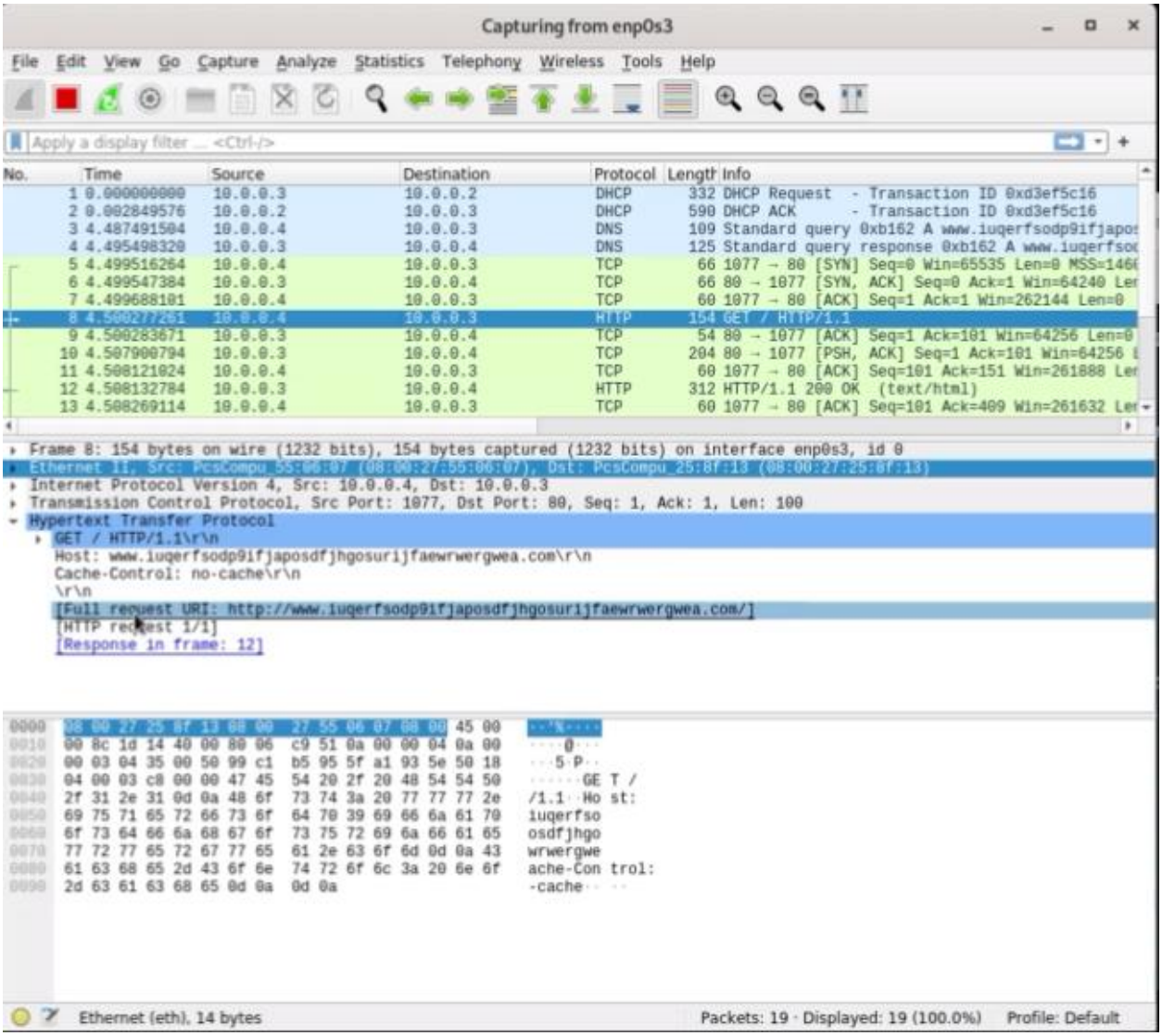


Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create
svchost.exe	948	TCP	Listen	0.0.0.0	135	0.0.0.0	0	2/27/2023 6:58:
System	4	TCP	Listen	10.0.0.3	139	0.0.0.0	0	2/27/2023 6:58:
svchost.exe	1208	TCP	Listen	0.0.0.0	5040	0.0.0.0	0	2/27/2023 6:58:
taskhsvc.exe	1748	TCP	Listen	127.0.0.1	9050	0.0.0.0	0	2/27/2023 8:06:
lsass.exe	688	TCP	Listen	0.0.0.0	49664	0.0.0.0	0	2/27/2023 6:58:
wininit.exe	536	TCP	Listen	0.0.0.0	49665	0.0.0.0	0	2/27/2023 6:58:
svchost.exe	1052	TCP	Listen	0.0.0.0	49666	0.0.0.0	0	2/27/2023 6:58:
svchost.exe	296	TCP	Listen	0.0.0.0	49667	0.0.0.0	0	2/27/2023 6:58:
spoolsv.exe	1512	TCP	Listen	0.0.0.0	49668	0.0.0.0	0	2/27/2023 6:58:
svchost.exe	2256	TCP	Listen	0.0.0.0	49670	0.0.0.0	0	2/27/2023 6:58:
taskhsvc.exe	1748	TCP	Established	127.0.0.1	50029	127.0.0.1	50030	2/27/2023 8:06:
taskhsvc.exe	1748	TCP	Established	127.0.0.1	50030	127.0.0.1	50029	2/27/2023 8:06:
services.exe	680	TCP	Listen	0.0.0.0	50147	0.0.0.0	0	2/27/2023 6:59:
System	4	TCP	Listen	0.0.0.0	445	0.0.0.0	0	2/27/2023 6:58:
svchost.exe	2008	TCP	Listen	0.0.0.0	7680	0.0.0.0	0	2/27/2023 6:58:

Endpoints: 54 | Established: 2 | Listening: 23 | Time Wait: | Close Wait: | Update: 2 sec | States: (All)

taskhsvc.exe listense on port 9050

Wireshark



The binary attempts to initiate a connection with the weird URL

http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com

Inetsim

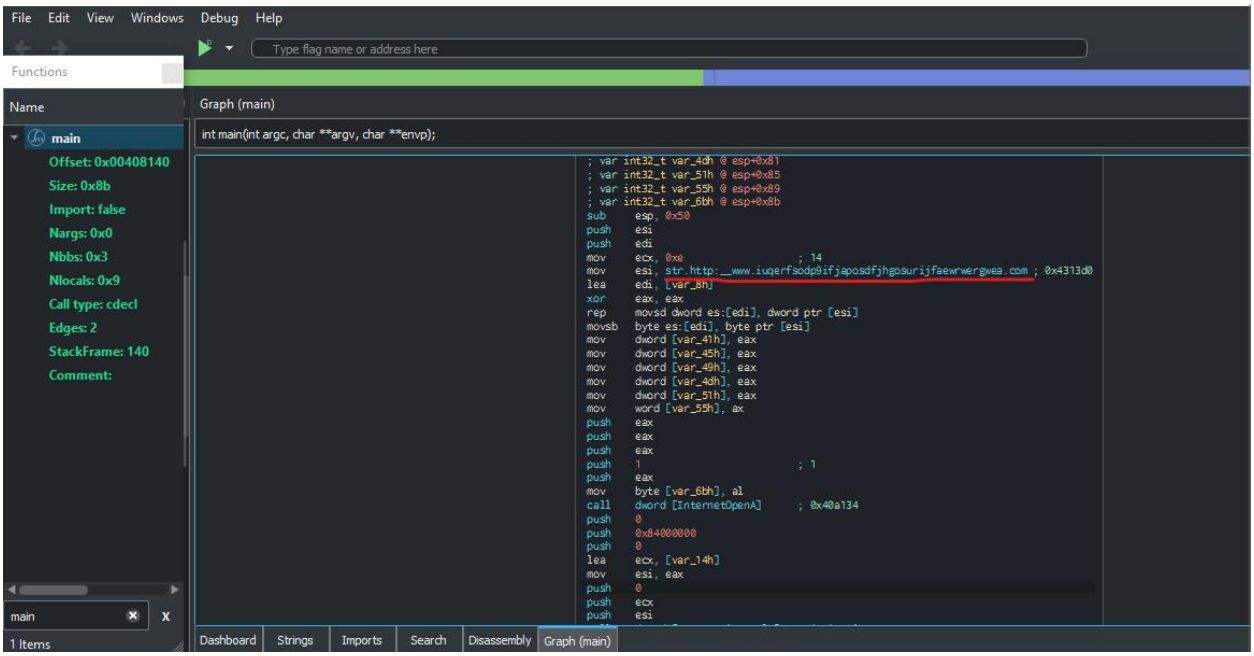
```
remnux@remnux:~$ inetsim
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg
Using log directory:      /var/log/inetsim/
Using data directory:     /var/lib/inetsim/
Using report directory:   /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
=== INetSim main process started (PID 5927) ===
Session ID:     5927
Listening on:   10.0.0.4
Real Date/Time: 2023-02-26 00:52:45
Fake Date/Time: 2023-02-26 00:52:45 (Delta: 0 seconds)
 Forking services...
  * dns_53_tcp_udp - started (PID 5934)
  * http_80_tcp - started (PID 5935)
  * pop3s_995_tcp - started (PID 5940)
  * https_443_tcp - started (PID 5936)
  * pop3_110_tcp - started (PID 5939)
  * ftps_990_tcp - started (PID 5942)
  * ftp_21_tcp - started (PID 5941)
  * smtps_465_tcp - started (PID 5938)
  * smtp_25_tcp - started (PID 5937)
 done.
Simulation running.
```

When inetsim is off the malware executes, but when it is on it doesn't. The reason for this is because of how to malware is made, inetsim simulates a fake dns server which when the malware connects to it will think it is connecting to the url: http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com

when in reality it's connected to inetsim's simulated server.

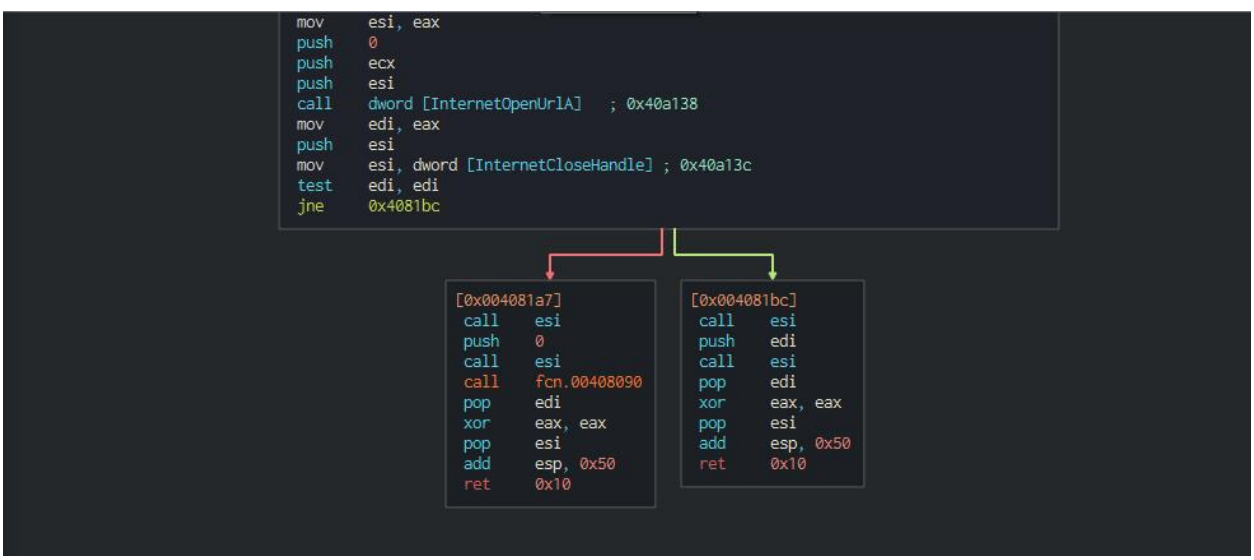
Advanced Static Analysis

Cutter



In the main function we can see the url and if we look below there is a call to InternetOpenA which is an api of windows.

The asm code basically moves the content of the string to esi then a bunch of parameters are filled to do an api call.



There is also a conditional jump of ne(not equal).

Decompiler

```
    int32_t var_49h;
    int32_t var_4dh;
    int32_t var_51h;
    int32_t var_55h;
    int32_t var_6bh;
    ecx = 0xe;
    esi = "http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com";
    edi = &var_8h;
    eax = 0;
```

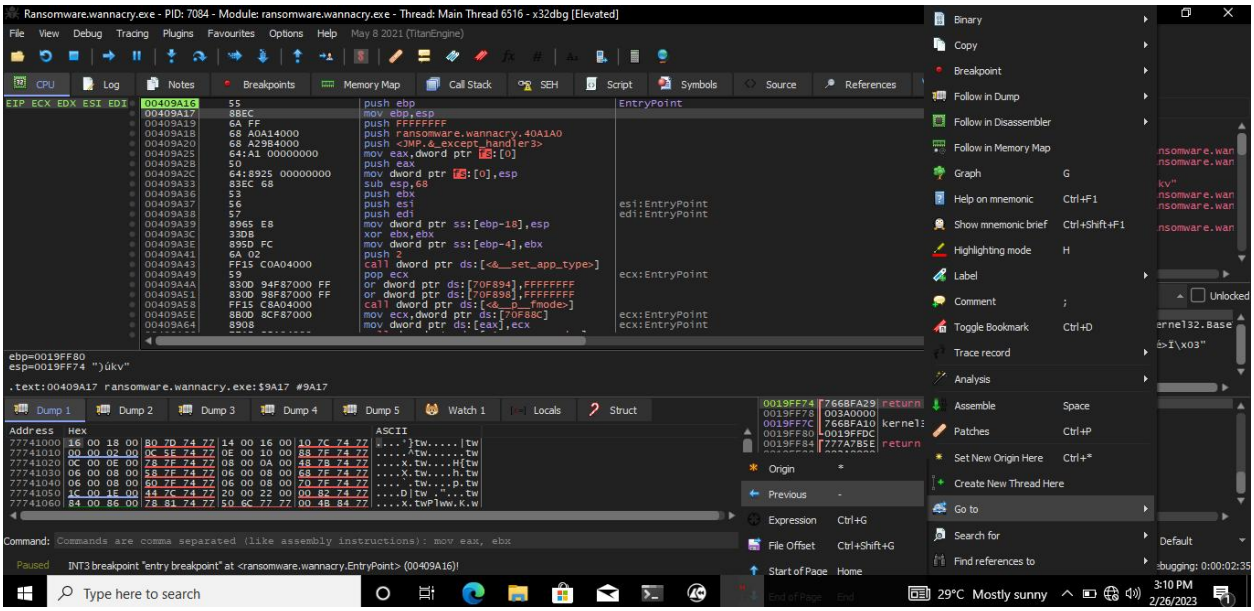
```
    do {
        *(es:edi) = *(esi);
        ecx--;
        esi += 4;
        es:edi += 4;
    } while (ecx != 0);
    *(es:edi) = *(esi);
    esi++;
    es:edi++;
    eax = InternetOpenA (eax, 1, eax, eax, eax, eax, eax, eax, ax, al);
    ecx = &var_14h;
    esi = eax;
    eax = InternetOpenUrlA (esi, ecx, 0, 0, 0x84000000, 0);
    edi = eax;
    esi = imp.InternetCloseHandle;
    if (edi == 0) {
        void (*esi)() ();
        void (*esi)(uint32_t) (0);
        eax = fcn_00408090 ();
        eax = 0;
        return eax;
    }
    void (*esi)() ();
    eax = void (*esi)(uint32_t) (edi);
    eax = 0;
    return eax;
}
```

Here we can better understand how the condition works. API is placed in eax then parameters are filled. A check is performed to see if edi is equal to zero. If it is, then the code declares a function pointer, assigns a function pointer of type void (*)(uint32_t) to it with a parameter value of zero, calls the fcn_00408090 function, sets eax to 0, and returns eax.

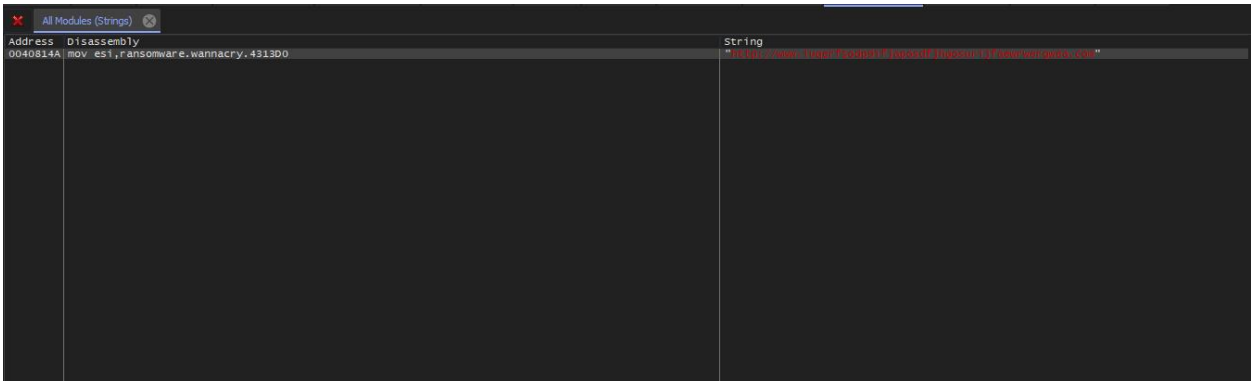
If edi is not equal to zero, the code declares a function pointer, assigns a function pointer of type void (*)(uint32_t) to it with a parameter value of edi, sets eax to 0, and returns eax which calls InternetOpenUrlA.

Advanced Dynamic Analysis

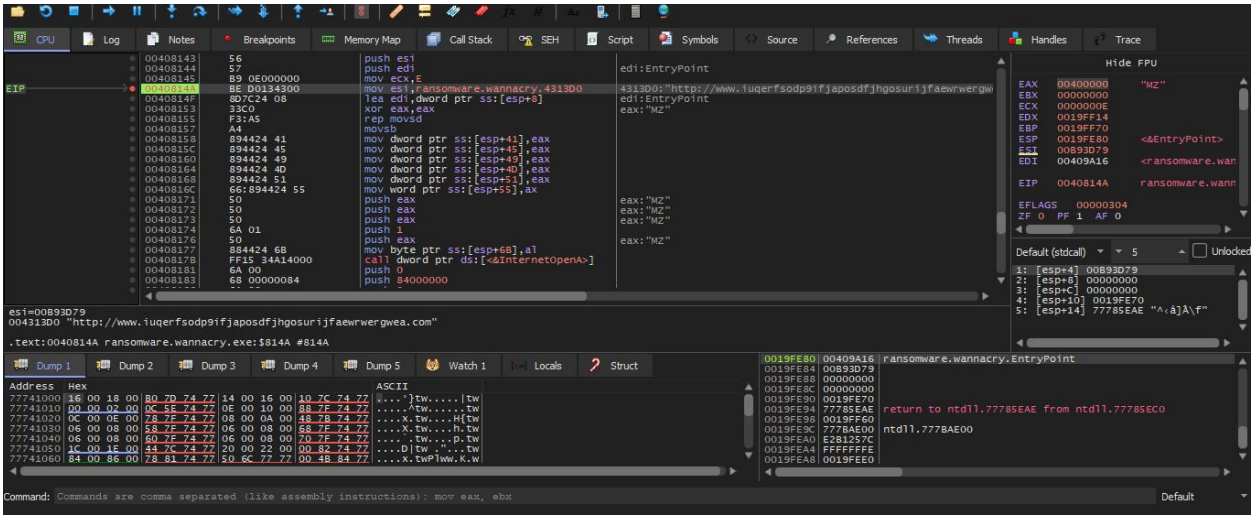
X32 dbg



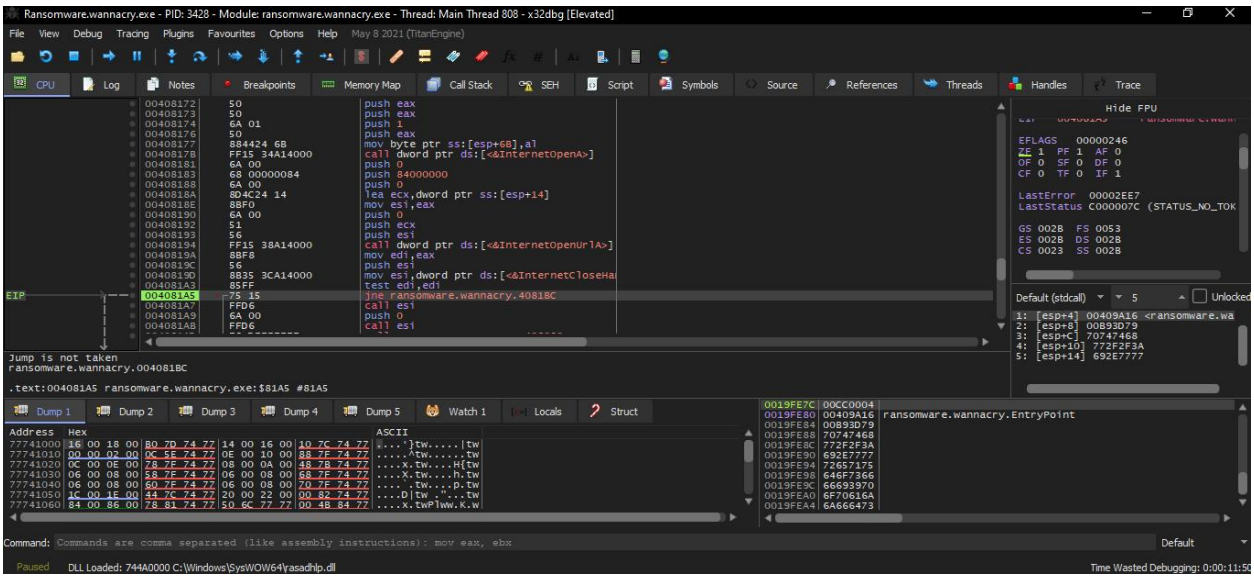
Here we will try to find the string with the url and set our breakpoint there.



We have successfully found our string, now we click the instruction to go there immediately.



Now we are in the instruction, we now set our break point.



Here we are at the conditional jump.

```
004081A3   85FF          test edi,edi
004081A5   75 15         jne ransomware.wannacry.4081BC
004081A7   FFD6          call esi
004081A9   6A 00         push 0
004081AB   FFD6          call esi
004081AD   E8 DEFEFFFF   call ransomware.wannacry.408090
004081B2   5F            pop edi
004081B3   33C0          xor eax,eax
004081B5   5E            pop esi
004081B6   83C4 50       add esp,50
004081B9   C2 1000       ret 10
004081BC   FFD6          call esi
004081BE   57            push edi
004081BF   FFD6          call esi
```

Now we have our self a conditional jump of not equal similar to the one we saw in cutter. If ZF(zero flag) is set to 0 it will jump to the end and continue the rest instructions.

Since we did a simple dynamic analysis with inetsim we can conclude that this has to do with the strange url. If the malware connects to it(ZF = 1) it exits otherwise it just runs completely(ZF = 0).

```
00409837   6A 0A          push A
00409839   58            pop eax
0040983A   50            push eax
0040983B   56            push esi
0040983C   53            push ebx
0040983D   53            push ebx
0040983E   FF15 ACA04000 call dword ptr ds:[<&GetModuleHandleA>]
00409844   50            push eax
00409845   E8 F6E5FFFF   call ransomware.wannacry.408140
0040984A   8945 98       mov dword ptr ss:[ebp-68],eax
0040984D   50            push eax
0040984E   FF15 ECA04000 call dword ptr ds:[<&exit>]
00409854   8B45 EC       mov eax,dword ptr ss:[ebp-14]
00409857   8B08          mov ecx,dword ptr ds:[eax]
00409859   8B09          mov ecx,dword ptr ds:[ecx]
0040985B   894D 88       mov dword ptr ss:[ebp-78],ecx
0040985E   50            push eax
0040985F   51            push ecx
00409860   E8 1B000000   call <JMP.&_XcptFilter>
00409865   59            pop ecx
00409866   59            pop ecx
00409867   C3            ret
00409868   8B65 E8       mov esp,dword ptr ss:[ebp-18]
```

Since we set the ZF value to 0 even if we turn on inetsim it will still run and execute the rest of it's instructions and exit out completely.

Indicators of Compromise

The full list of IOCs can be found in the Appendices.

Network Indicators

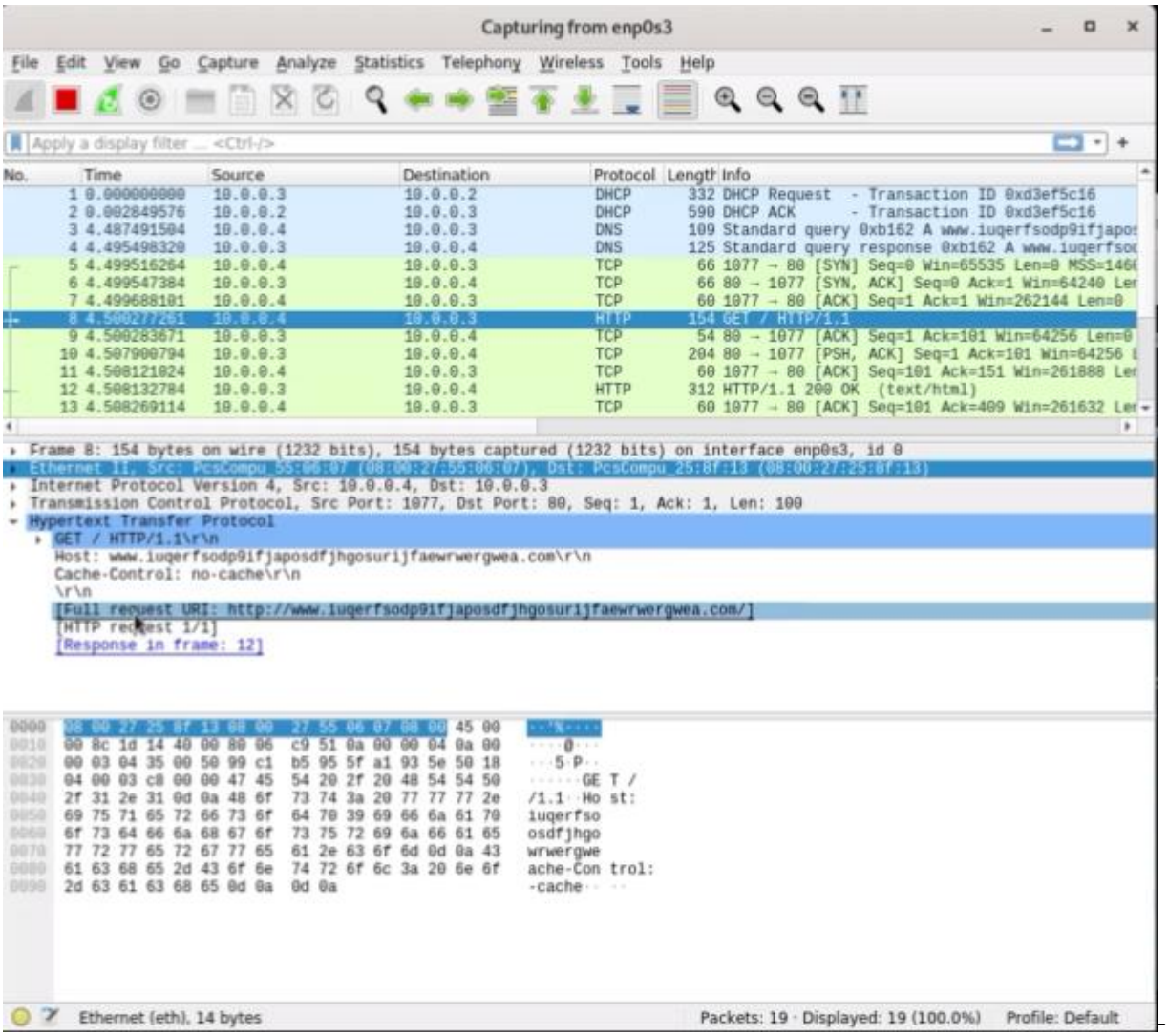


Fig 3: WireShark Packet Capture of HTTP GET Request

The binary tries to access the weird url.

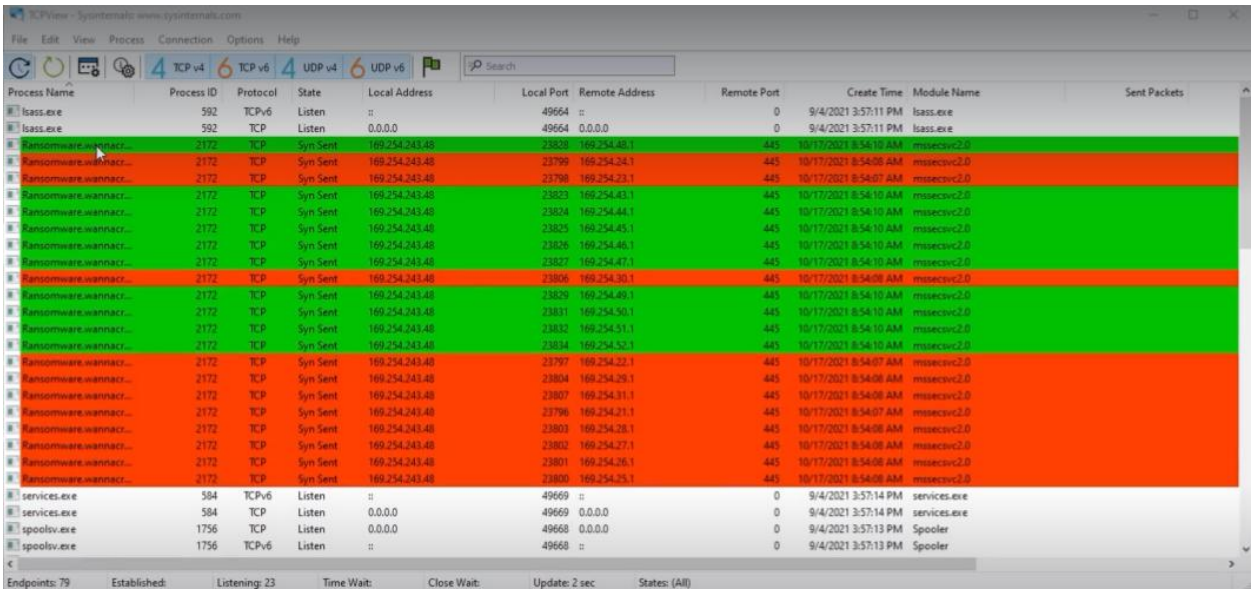


Fig 4: TCPView A flood of numerous SMB connection requests to non-private remote addresses.

Host-based Indicators

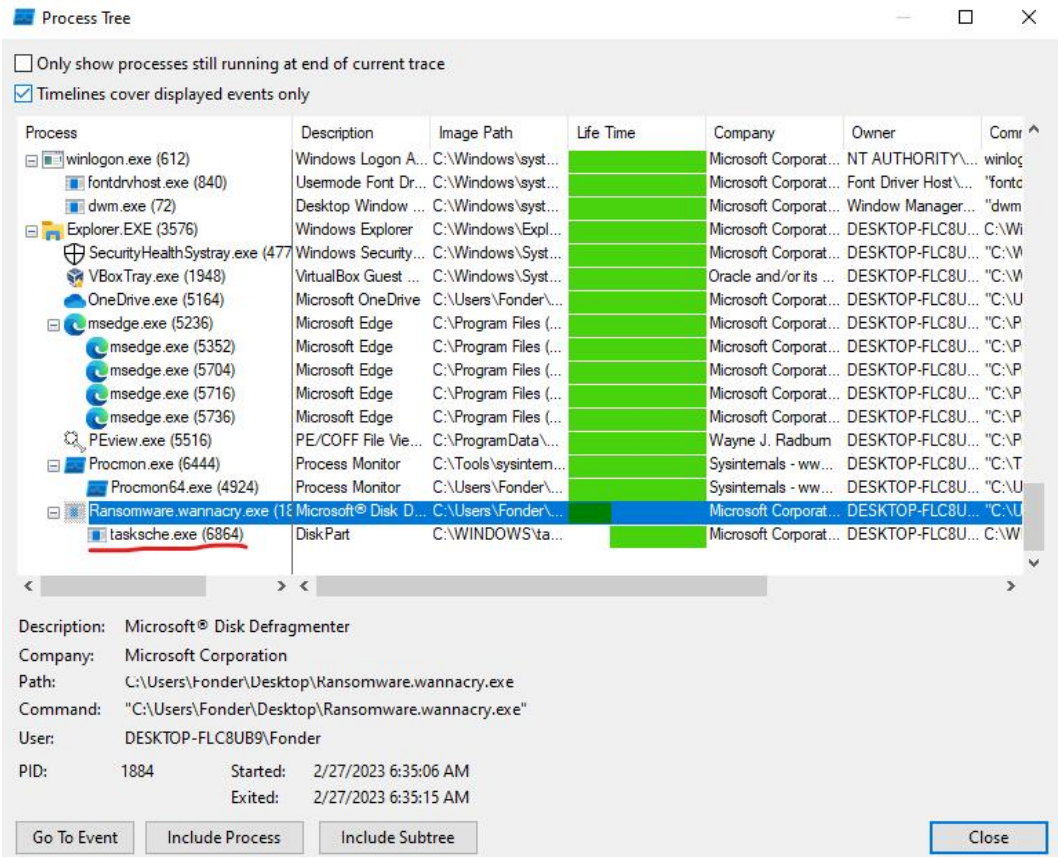


Fig 5: Process Tree Ransomware.wannacry.exe is present along with another stage 2 payload.

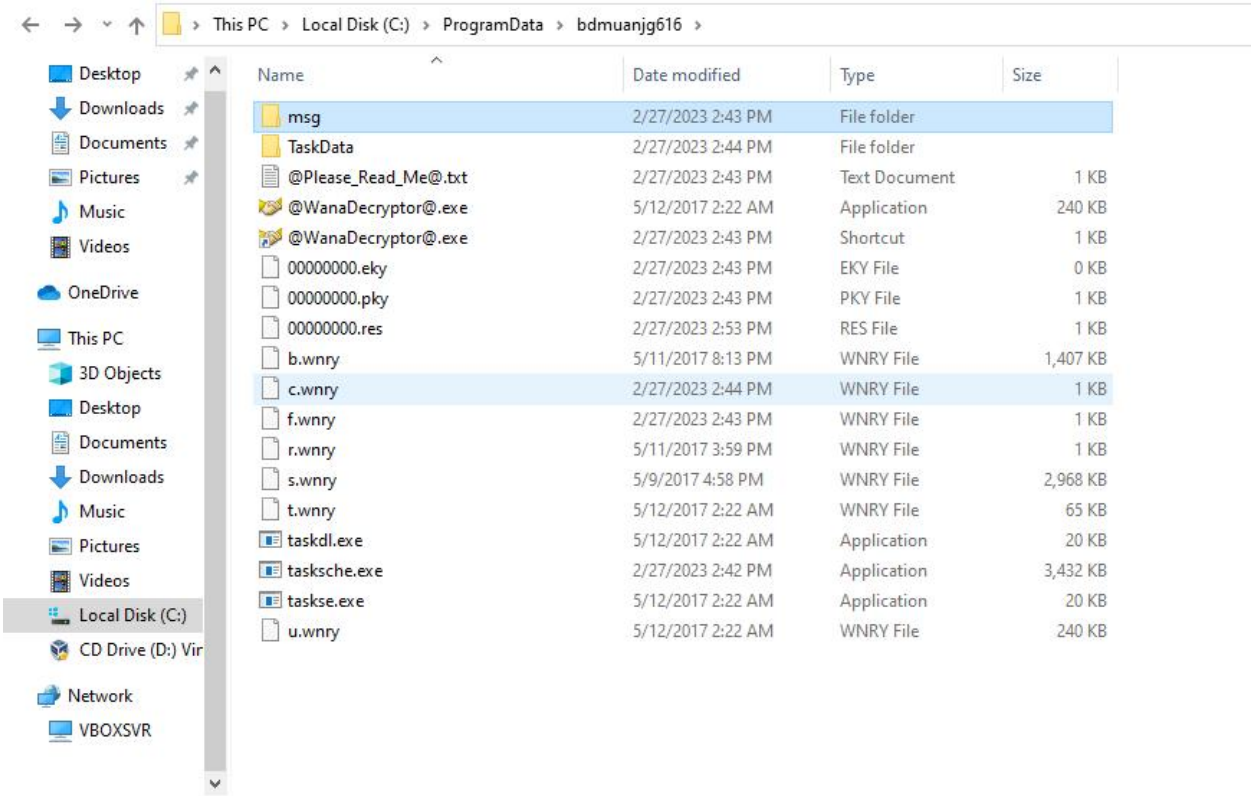


Fig 6: New folder created, seen in Procmon with a random generated chars.

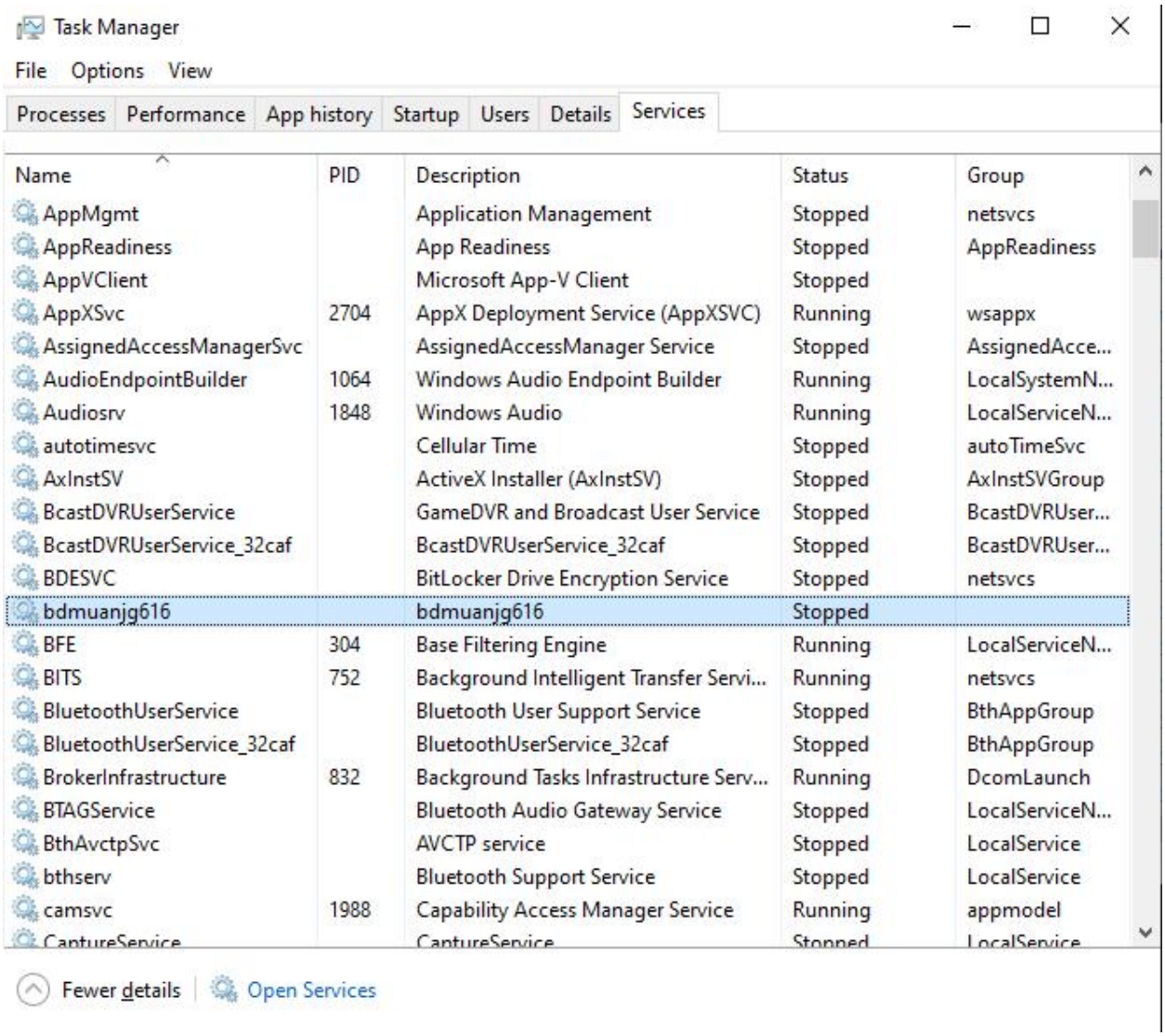


Fig 7: Persistent service, has the same name with the folder created.

Rules & Signatures

Appendices

A. Yara Rules

Full Yara repository located at:

```
rule WannaCry_Detection {
    meta:
        description = "Matches a collection of strings from Wannacry"
        author = "FonderElite"
        date = "2023-02-28"
        reference = "Ransomware.wannacry.exe"
        category = "security"
    strings:
        $str1 = "%s -m security"
        $str2 = "C:\\%s\\qeriuwjhrf"
        $str3 = "tasksche.exe"
        $str4 = "icacls . /grant Everyone:F /T /C /Q"
        $str5 = "WNcry@2ol7"
        $str6 = "http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com"
    condition:
        any of ($str*) or all of them
}
```

```
C:\Users\Fonder\Desktop
λ yara64 yaratest.yara -w -p 32   Ransomware.wannacry.exe
string_rule Ransomware.wannacry.exe
```

B. Callback URLs

Domain	Port
http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com	80

C. Decompiled Code Snippets

```
File  Edit  View  Windows  Debug  Help
Type flag name or address here

Functions
Name
main
Offset: 0x00408140
Size: 0x8b
Import: false
Nargs: 0x0
Nbbs: 0x3
Nlocals: 0x9
Call type: cdecl
Edges: 2
StackFrame: 140
Comment:

Graph (main)
int main(int argc, char **argv, char **envp);

; var int32_t var_4dh @ esp+0x81
; var int32_t var_51h @ esp+0x85
; var int32_t var_55h @ esp+0x89
; var int32_t var_6bh @ esp+0x8b
sub     esp, 0x50
push    esi
push    edi
mov     ecx, 0xe                ; 14
mov     esi, str.http:__www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com ; 0x4313d0
lea     edi, [var_8h]
xor     eax, eax
rep     movsd dword es:[edi], dword ptr [esi]
movsb   byte es:[edi], byte ptr [esi]
mov     dword [var_41h], eax
mov     dword [var_45h], eax
mov     dword [var_49h], eax
mov     dword [var_4dh], eax
mov     dword [var_51h], eax
mov     word [var_55h], ax
push    eax
push    eax
push    eax
push    1                       ; 1
push    eax
mov     byte [var_6bh], al
call    dword [InternetOpenA]   ; 0x40a134
push    0
push    0x84000000
push    0
lea     ecx, [var_14h]
mov     esi, eax
push    0
push    ecx
push    esi

Dashboard  Strings  Imports  Search  Disassembly  Graph (main)
```

```
push    eax
push    eax
push    1                       ; 1
push    eax
mov     byte [var_6bh], al
call    dword [InternetOpenA]   ; 0x40a134
push    0
push    0x84000000
push    0
lea     ecx, [var_14h]
mov     esi, eax
push    0
push    ecx
push    esi
call    dword [InternetOpenUrlA]   ; 0x40a138
mov     edi, eax
push    esi
mov     esi, dword [InternetCloseHandle] ; 0x40a13c
test    edi, edi
jne     0x4081bc

[0x004081a7]
call    esi
push    0
call    esi
call    fcn.00408090
pop     edi
xor     eax, eax
pop     esi
add     esp, 0x50
ret     0x10

[0x004081bc]
call    esi
push    edi
call    esi
pop     edi
xor     eax, eax
pop     esi
add     esp, 0x50
ret     0x10
```

Fig 6: Opens a URL and closes itself upon a condition.