Malware Analysis Report

Ransomware.Wannacry Malware

June 2023 | Theodoros Vergos “cde” | v1.0

Table of Contents

Executive Report

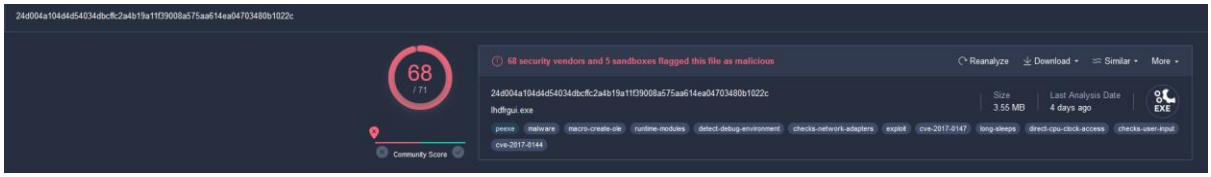
Name	Ransomware.wannacry.exe
MD5:	db349b97c37d22f5ea1d1841e3c89eb4
SHA1:	e889544aff85ffaf8b0d0da705105dee7c97fe26
SHA256:	24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614ea04703480b1022c
Architecture:	x86
Signature:	Microsoft Visual C++ v6.0

The file in question has been identified as an encryptor with worm capabilities. The malware has two main components, the propagator and the encryptor. Symptoms of malware presence if the network include DNS requests for the URL *hxxp://www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com/*, systems performing ARP requests to the network in order to discover other systems followed by connection attempts on TCP port 445 (SMB). For a successful infection the malware needs to be executed with administrative rights. Then a DNS request and an http request are performed, if the HTTP request returns an HTTP -200 response the malware does not continue with the execution.

In order to mitigate this particular strain, it is recommended to add the following URL *hxxp://www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com/* to systems' hosts file and DNS server records, redirecting any requests to a sinkhole server.

Additionally, a YARA rule has been included in this report to aid in identifying the malware in the wild.

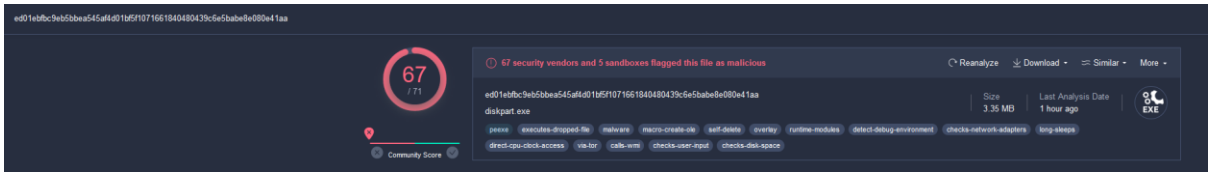
Currently, this malware has 68/71 vendors detections in *virustotal.com*



The main encryption mechanism is considered the following executable that is unpacked from the initial malware and can be used with other delivery methods as well, without relying on the *Ransomware.wannacry.exe* executable:

Name	Tasksche.exe
MD5	84c82835a5d21bbcf75a61706d8ab549
Sha1	5ff465afaabcbf0150d1a3ab2c2e74f3a4426467
SHA256	ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa
Architecture	x86
Signature	Microsoft Visual C++ v6.0

This malware has 67/71 vendor detection in *virustotal.com*



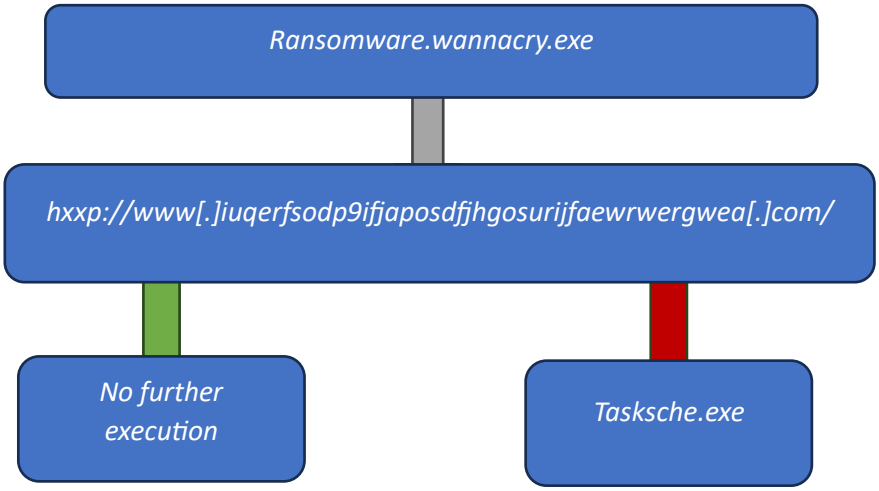
For this executable no workaround was detected in order to stop the execution.

YARA Rule

```
rule wannacry {

    meta:
        last_updated = "2023-06-18"
        author = "cde"
        description = "A Yara rule for detecting wannacry ransomware. This is
a part of the final lab for PMAT"

    strings:
        // Fill out identifying strings and other criteria
        $string1 = "http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com"
ascii
        $string2 = "C:\\%s\\qeriuwjhrf" ascii
        $string3 = "WANACRY!" ascii
        $string4 = "cmd.exe /c \"%s\""
        $string5 = "icacls . /grant Everyone:"
        $string6 = ".wnry"
        $string7 = "13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94"
        $string8 = "12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw"
        $string9 = "115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn"
        $PE_byte = "MZ"

    condition:
        // Fill out the conditions that must be met to identify the binary
        $PE_byte at 0 and
        ($string1 and $string2 and $string3 and $string4 and $string5 and
$string6 and $string7 and $string8 and $string9) or
        $PE_byte at 0 and
        ($string5 and $string6 and $string7 and $string8 and $string9)
}
```

High Level Technical Summary

When *Ransomware.wannacry.exe* is executed with administrative privileges the malware checks if it can make a successful callback to the URL *hxxp://www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com/* and if the callback is successful the malware is not executed. If on the other hand there is no successful response the malware proceeds to unpack its components.



Ransomware.wannacry.exe will try to identify other hosts on the network and infect them as well by connecting to systems that allow connections on the TCP port 445, which is typically used for SMB. If no hosts are identified the rest of the malware is unpacked inside C:\ProgramData\bqtztzryebik717. From there Tasksche.exe is executed which enumerates the system and encrypts user files that are related to business, productivity or entertainment.

In the following sections of Static and Dynamic analysis all results are documented and reported.

Static Analysis

The following interesting strings have been identified using floss -n 7:

First batch of suspicious strings include the callback URL and the first executable to be unpacked.

hxxp://www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com

```
427 C:\%s\qeriuwjhrf
428 C:\%s\%s
429 WINDOWS
430 tasksche.exe
431 CloseHandle
432 WriteFile
433 CreateFileA
434 CreateProcessA
435 http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com
436 !This program cannot be run in DOS mode.
437 `.rdata
```

Some of the libraries used by the malware, strings that are used in the YARA rule and icacls granting everyone the full access to all files.

```
571 MSVCP60.dll
572 GetStartupInfoA
573 advapi32.dll
574 WANACRY!
575 CloseHandle
576 DeleteFileW
577 MoveFileExW
578 MoveFileW
579 ReadFile
580 WriteFile
581 CreateFileW
582 kernel32.dll
583 O|x8+^_
584 2/O-_.X8w.+
585 |~}%.15
586 Microsoft Enhanced RSA and AES Cryptographic Provider
587 CryptGenKey
588 CryptDecrypt
589 CryptEncrypt
590 CryptDestroyKey
591 CryptImportKey
592 CryptAcquireContextA
593 cmd.exe /c "%s"
594 115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn
595 12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
596 13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94
597 Global\MsWinZonesCacheCounterMutexA
598 tasksche.exe
599 TaskStart
600 icacls . /grant Everyone:F /T /C /Q
601 attrib +h .
602 WNcry@2ol7
603 GetNativeSystemInfo
604 .?AVexception@@
605 incompatible version
```

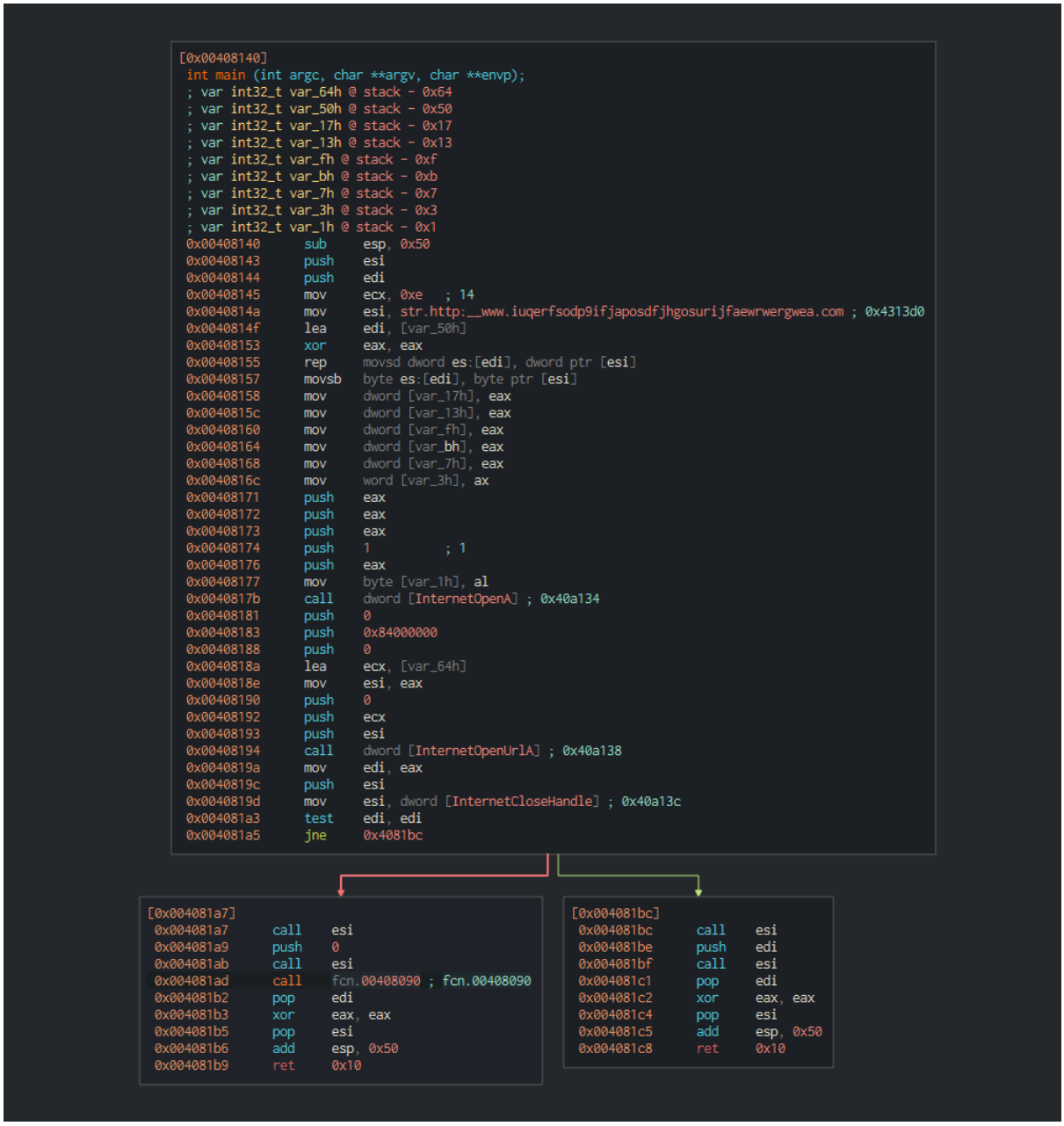
This is the list of files to be unpacked by the encryptor.

```
2885    %l?E)!o
2886    msg/m_bulgarian.wnry
2887    msg/m_chinese (simplified).wnry
2888    "t=.|Vbq-
2889    msg/m_chinese (traditional).wnry
2890    msg/m_croatian.wnry
2891    msg/m_czech.wnry
2892    msg/m_danish.wnry
2893    msg/m_dutch.wnry
2894    msg/m_english.wnry
2895    msg/m_filipino.wnry
2896    msg/m_finnish.wnry
2897    msg/m_french.wnry
2898    msg/m_german.wnry
2899    msg/m_greek.wnry
2900    msg/m_indonesian.wnry
2901    msg/m_italian.wnry
2902    msg/m_japanese.wnry
2903    msg/m_korean.wnry
2904    msg/m_latvian.wnry
2905    msg/m_norwegian.wnry
2906    msg/m_polish.wnry
2907    msg/m_portuguese.wnry
2908    msg/m_romanian.wnry
2909    msg/m_russian.wnry
2910    msg/m_slovak.wnry
2911    msg/m_spanish.wnry
2912    msg/m_swedish.wnry
2913    msg/m_turkish.wnry
2914    msg/m_vietnamese.wnry
2915    taskdl.exe
2916    taskse.exe
```

Advanced Static Analysis

Adding the malware in cutter, we were able to identify the main functions and logic of the initial program.

Main:

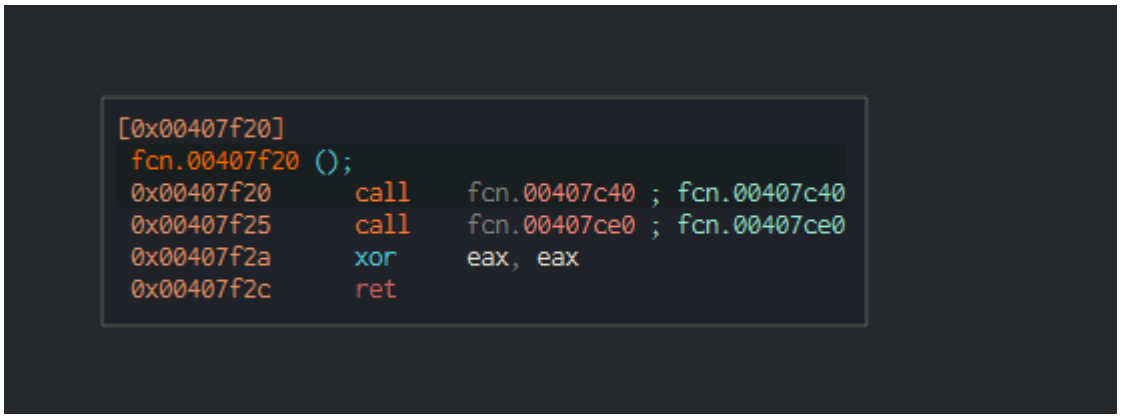


Fnc00408090 - No available callback:

Calls GetModuleFineNameA, then OpenSCManagerA then OpenServiceA and after that fcn.00407fa0

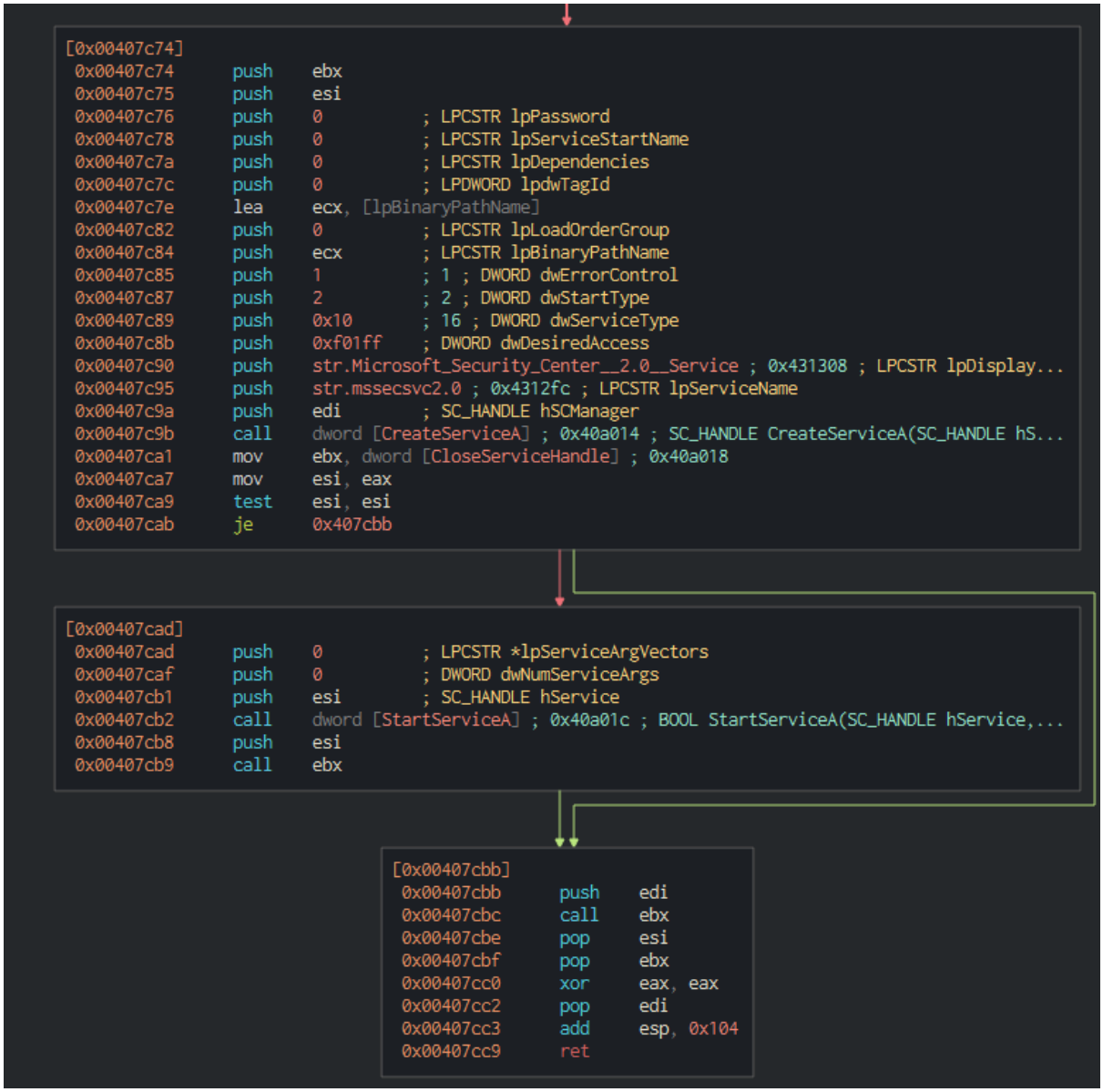
And finally, StarServiceCtrlDispacherA and returns to the main function.

Otherwise calls function 00407f20

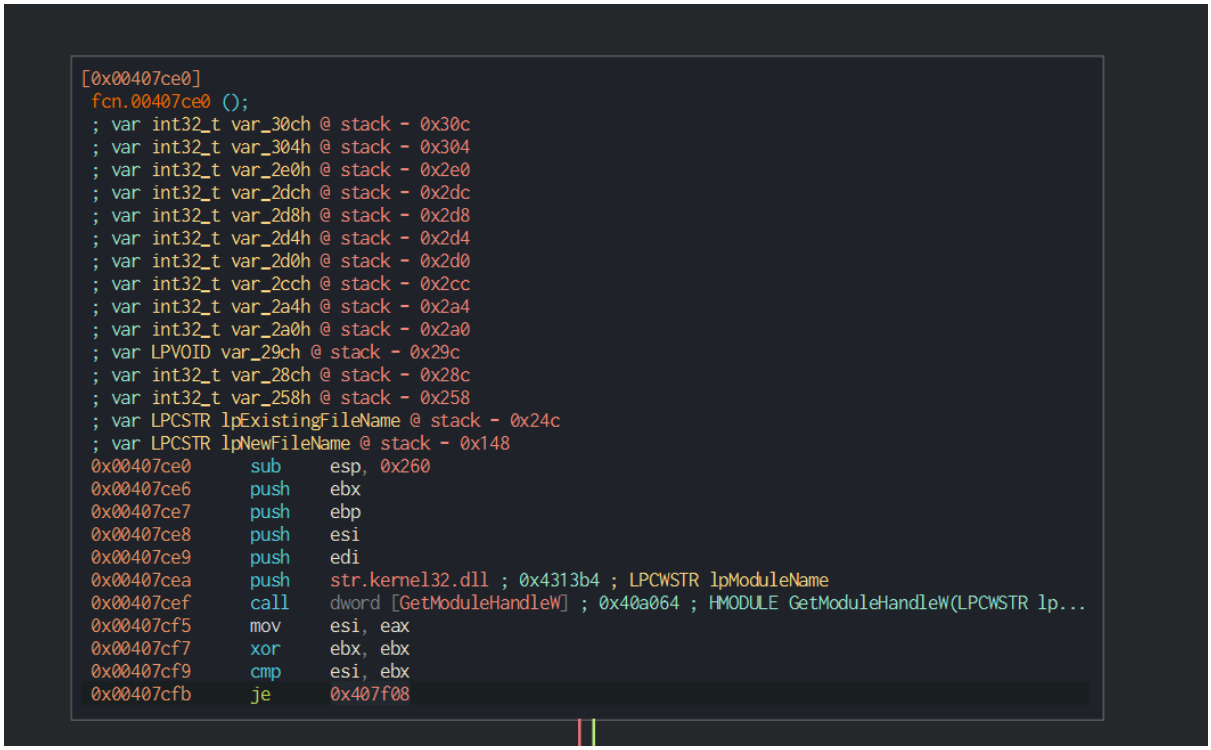


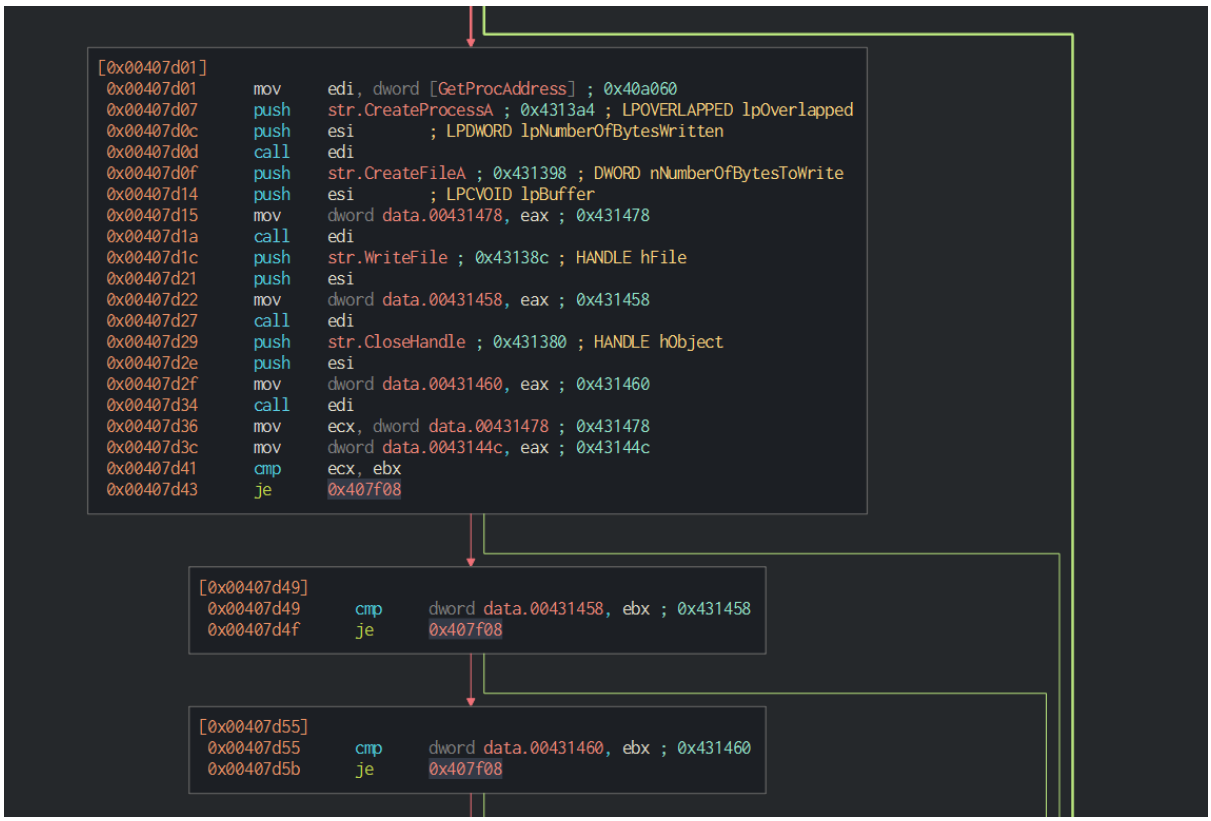
Fcn.00407c40

Calls OpenSCManagerA and on successful call jumps to create a service with the following characteristics on the system

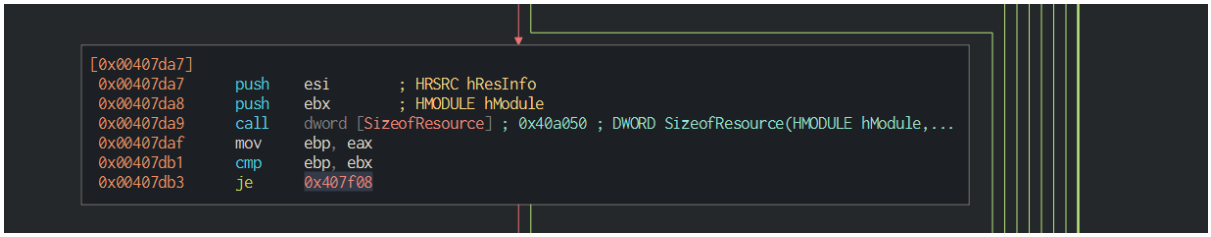

[0x00407c74]
0x00407c74 push ebx
0x00407c75 push esi
0x00407c76 push 0 ; LPCSTR lpPassword
0x00407c78 push 0 ; LPCSTR lpServiceStartName
0x00407c7a push 0 ; LPCSTR lpDependencies
0x00407c7c push 0 ; LPDWORD lpdwTagId
0x00407c7e lea ecx, [lpBinaryPathName]
0x00407c82 push 0 ; LPCSTR lpLoadOrderGroup
0x00407c84 push ecx ; LPCSTR lpBinaryPathName
0x00407c85 push 1 ; 1 ; DWORD dwErrorControl
0x00407c87 push 2 ; 2 ; DWORD dwStartType
0x00407c89 push 0x10 ; 16 ; DWORD dwServiceType
0x00407c8b push 0xf01ff ; DWORD dwDesiredAccess
0x00407c90 push str.Microsoft_Security_Center__2.0__Service ; 0x431308 ; LPCSTR lpDisplay...
0x00407c95 push str.mssecsvc2.0 ; 0x4312fc ; LPCSTR lpServiceName
0x00407c9a push edi ; SC_HANDLE hSCManager
0x00407c9b call dword [CreateServiceA] ; 0x40a014 ; SC_HANDLE CreateServiceA(SC_HANDLE hS...
0x00407ca1 mov ebx, dword [CloseServiceHandle] ; 0x40a018
0x00407ca7 mov esi, eax
0x00407ca9 test esi, esi
0x00407cab je 0x407cbb
[0x00407cad]
0x00407cad push 0 ; LPCSTR *lpServiceArgVectors
0x00407caf push 0 ; DWORD dwNumServiceArgs
0x00407cb1 push esi ; SC_HANDLE hService
0x00407cb2 call dword [StartServiceA] ; 0x40a01c ; BOOL StartServiceA(SC_HANDLE hService,...
0x00407cb8 push esi
0x00407cb9 call ebx
[0x00407cbb]
0x00407cbb push edi
0x00407cbc call ebx
0x00407cbe pop esi
0x00407cbf pop ebx
0x00407cc0 xor eax, eax
0x00407cc2 pop edi
0x00407cc3 add esp, 0x104
0x00407cc9 ret

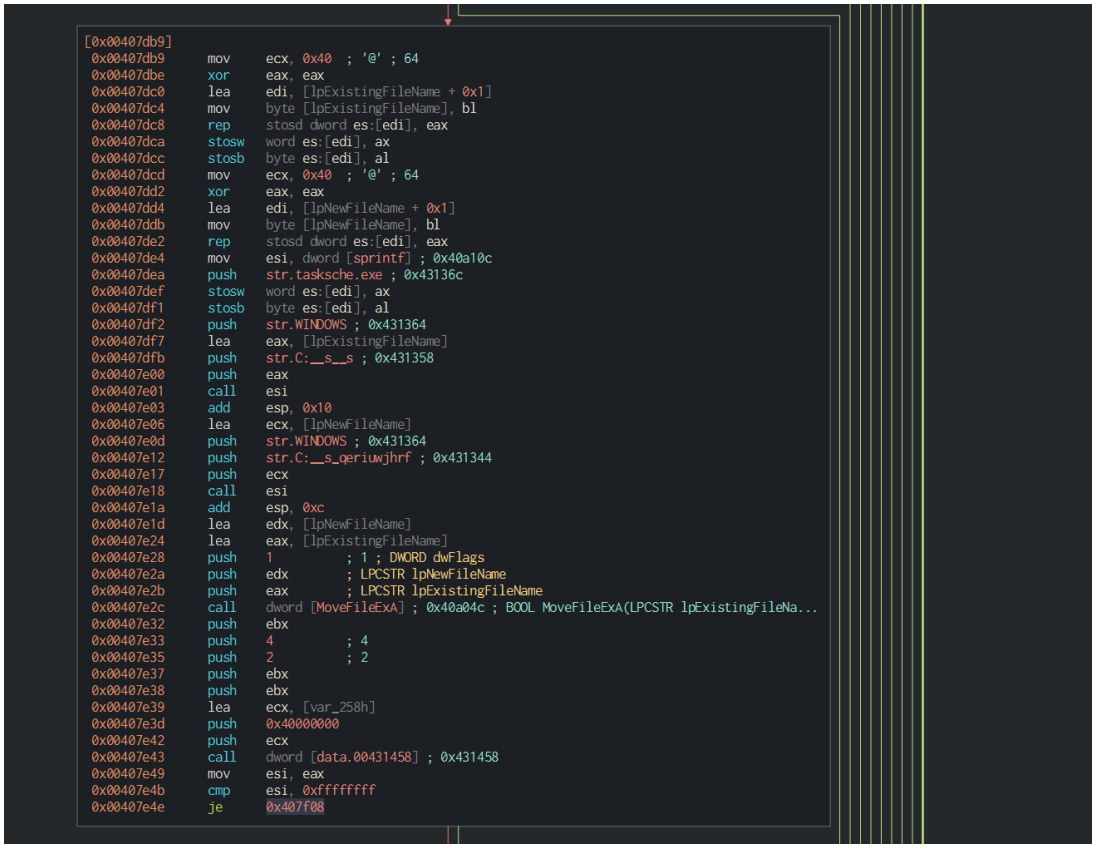
Fcn.00407ce0 - the encryptor:

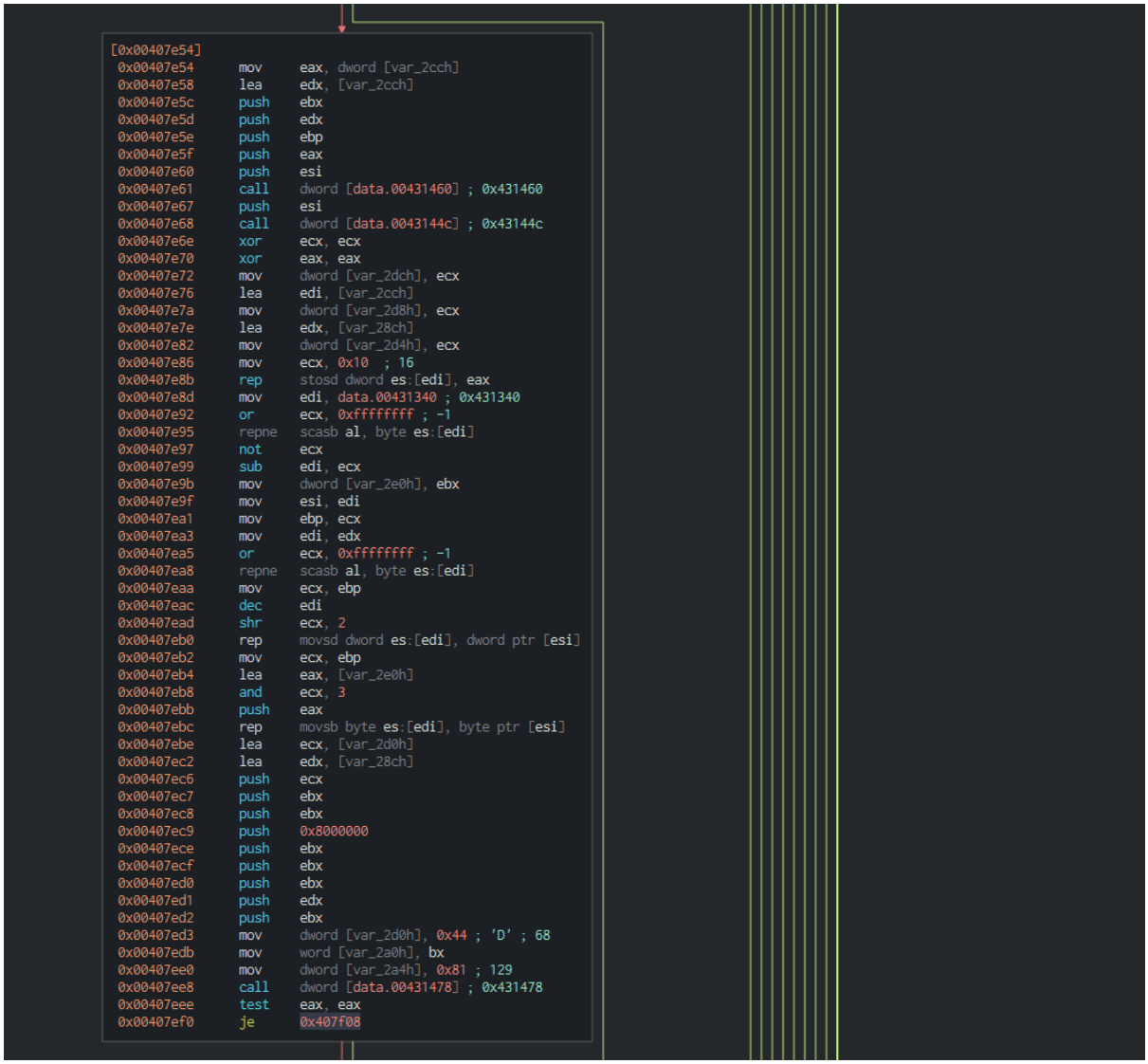


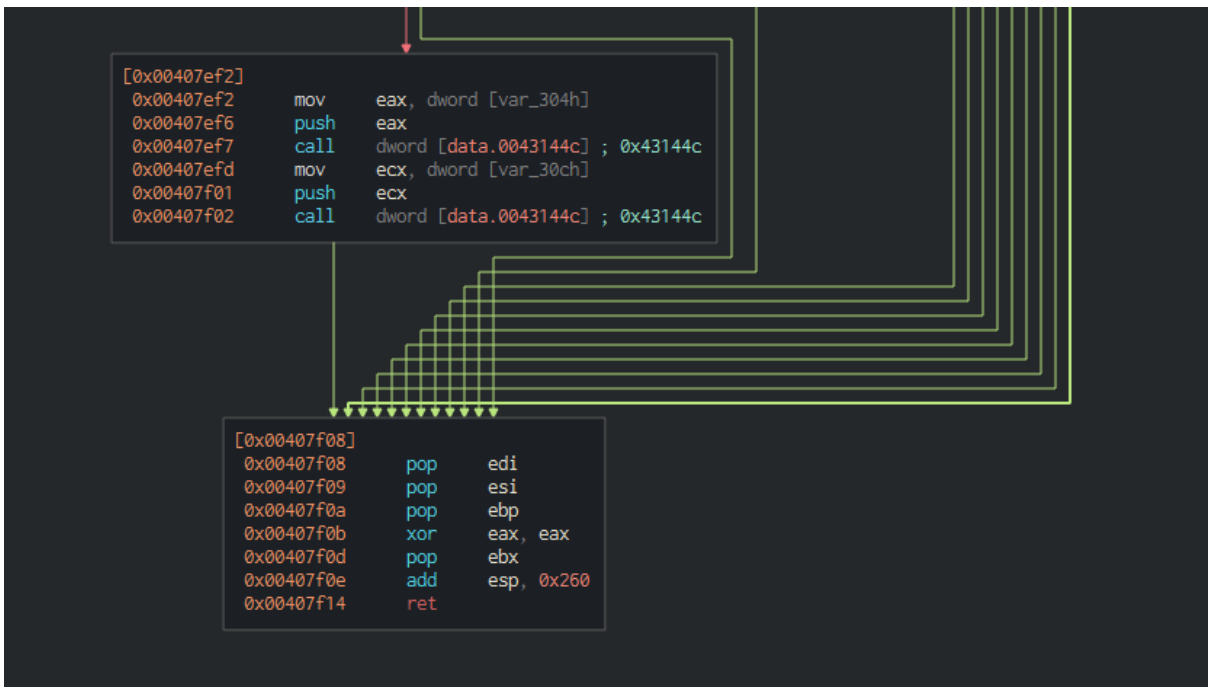



[0x00407d61]
0x00407d61 cmp eax, ebx
0x00407d63 je 0x407f08
[0x00407d69]
0x00407d69 push data.0043137c ; 0x43137c ; LPCSTR lpType
0x00407d6e push 0x727 ; 1831 ; LPCSTR lpName
0x00407d73 push ebx ; HMODULE hModule
0x00407d74 call dword [FindResourceA] ; 0x40a05c ; HRSRC FindResourceA(HMODULE hModule, L...
0x00407d7a mov esi, eax
0x00407d7c cmp esi, ebx
0x00407d7e je 0x407f08
[0x00407d84]
0x00407d84 push esi ; HRSRC hResInfo
0x00407d85 push ebx ; HMODULE hModule
0x00407d86 call dword [LoadResource] ; 0x40a058 ; HGLOBAL LoadResource(HMODULE hModule, H...
0x00407d8c cmp eax, ebx
0x00407d8e je 0x407f08
[0x00407d94]
0x00407d94 push eax ; HGLOBAL hResData
0x00407d95 call dword [LockResource] ; 0x40a0a0 ; LPVOID LockResource(HGLOBAL hResData)
0x00407d9b cmp eax, ebx
0x00407d9d mov dword [var_29ch], eax
0x00407da1 je 0x407f08


[0x00407da7]
0x00407da7 push esi ; HRSRC hResInfo
0x00407da8 push ebx ; HMODULE hModule
0x00407da9 call dword [SizeofResource] ; 0x40a050 ; DWORD SizeofResource(HMODULE hModule,...
0x00407daf mov ebp, eax
0x00407db1 cmp ebp, ebx
0x00407db3 je 0x407f08

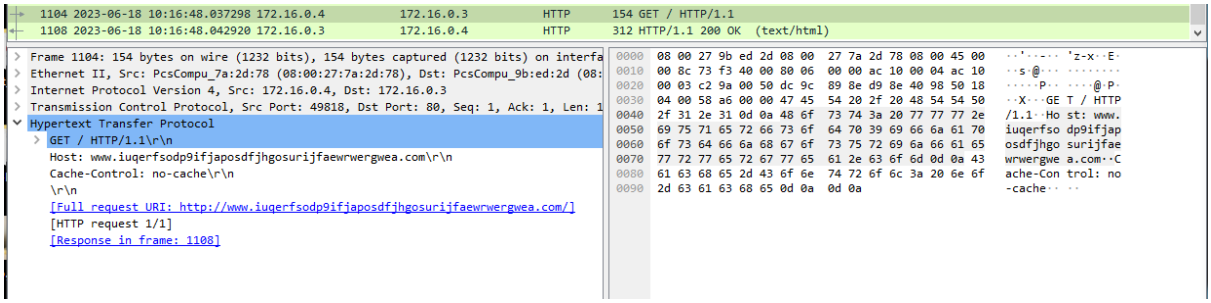

[0x00407db9]
0x00407db9 mov ecx, 0x40 ; '@' ; 64
0x00407dbe xor eax, eax
0x00407dc0 lea edi, [lpExistingFileName + 0x1]
0x00407dc4 mov byte [lpExistingFileName], bl
0x00407dc8 rep stosd dword es:[edi], eax
0x00407dca stosw word es:[edi], ax
0x00407dcc stosb byte es:[edi], al
0x00407dcd mov ecx, 0x40 ; '@' ; 64
0x00407dd2 xor eax, eax
0x00407dd4 lea edi, [lpNewFileName + 0x1]
0x00407ddb mov byte [lpNewFileName], bl
0x00407de2 rep stosd dword es:[edi], eax
0x00407de4 mov esi, dword [sprintf] ; 0x40a10c
0x00407dea push str.tasksche.exe ; 0x43136c
0x00407def stosw word es:[edi], ax
0x00407df1 stosb byte es:[edi], al
0x00407df2 push str.WINDOWS ; 0x431364
0x00407df7 lea eax, [lpExistingFileName]
0x00407dfb push str.C:__s__s ; 0x431358
0x00407e00 push eax
0x00407e01 call esi
0x00407e03 add esp, 0x10
0x00407e06 lea ecx, [lpNewFileName]
0x00407e0d push str.WINDOWS ; 0x431364
0x00407e12 push str.C:__s_qeriuwjhrf ; 0x431344
0x00407e17 push ecx
0x00407e18 call esi
0x00407e1a add esp, 0xc
0x00407e1d lea edx, [lpNewFileName]
0x00407e24 lea eax, [lpExistingFileName]
0x00407e28 push 1 ; 1 ; DWORD dwFlags
0x00407e2a push edx ; LPCSTR lpNewFileName
0x00407e2b push eax ; LPCSTR lpExistingFileName
0x00407e2c call dword [MoveFileExA] ; 0x40a04c ; BOOL MoveFileExA(LPCSTR lpExistingFileNa...
0x00407e32 push ebx
0x00407e33 push 4 ; 4
0x00407e35 push 2 ; 2
0x00407e37 push ebx
0x00407e38 push ebx
0x00407e39 lea ecx, [var_258h]
0x00407e3d push 0x40000000
0x00407e42 push ecx
0x00407e43 call dword [data.00431458] ; 0x431458
0x00407e49 mov esi, eax
0x00407e4b cmp esi, 0xffffffff
0x00407e4e je 0x407f08


[0x00407e54]
0x00407e54 mov eax, dword [var_2cch]
0x00407e58 lea edx, [var_2cch]
0x00407e5c push ebx
0x00407e5d push edx
0x00407e5e push ebp
0x00407e5f push eax
0x00407e60 push esi
0x00407e61 call dword [data.00431460] ; 0x431460
0x00407e67 push esi
0x00407e68 call dword [data.0043144c] ; 0x43144c
0x00407e6e xor ecx, ecx
0x00407e70 xor eax, eax
0x00407e72 mov dword [var_2dch], ecx
0x00407e76 lea edi, [var_2cch]
0x00407e7a mov dword [var_2d8h], ecx
0x00407e7e lea edx, [var_28ch]
0x00407e82 mov dword [var_2d4h], ecx
0x00407e86 mov ecx, 0x10 ; 16
0x00407e8b rep stosd dword es:[edi], eax
0x00407e8d mov edi, data.00431340 ; 0x431340
0x00407e92 or ecx, 0xffffffff ; -1
0x00407e95 repne scasb al, byte es:[edi]
0x00407e97 not ecx
0x00407e99 sub edi, ecx
0x00407e9b mov dword [var_2e0h], ebx
0x00407e9f mov esi, edi
0x00407ea1 mov ebp, ecx
0x00407ea3 mov edi, edx
0x00407ea5 or ecx, 0xffffffff ; -1
0x00407ea8 repne scasb al, byte es:[edi]
0x00407eaa mov ecx, ebp
0x00407eac dec edi
0x00407ead shr ecx, 2
0x00407eb0 rep movsd dword es:[edi], dword ptr [esi]
0x00407eb2 mov ecx, ebp
0x00407eb4 lea eax, [var_2e0h]
0x00407eb8 and ecx, 3
0x00407ebb push eax
0x00407ebc rep movsb byte es:[edi], byte ptr [esi]
0x00407ebe lea ecx, [var_2d0h]
0x00407ec2 lea edx, [var_28ch]
0x00407ec6 push ecx
0x00407ec7 push ebx
0x00407ec8 push ebx
0x00407ec9 push 0x8000000
0x00407ece push ebx
0x00407ecf push ebx
0x00407ed0 push ebx
0x00407ed1 push edx
0x00407ed2 push ebx
0x00407ed3 mov dword [var_2d0h], 0x44 ; 'D' ; 68
0x00407edb mov word [var_2a0h], bx
0x00407ee0 mov dword [var_2a4h], 0x81 ; 129
0x00407ee8 call dword [data.00431478] ; 0x431478
0x00407eee test eax, eax
0x00407ef0 je 0x407f08


[0x00407ef2]
0x00407ef2 mov eax, dword [var_304h]
0x00407ef6 push eax
0x00407ef7 call dword [data.0043144c] ; 0x43144c
0x00407efd mov ecx, dword [var_30ch]
0x00407f01 push ecx
0x00407f02 call dword [data.0043144c] ; 0x43144c
[0x00407f08]
0x00407f08 pop edi
0x00407f09 pop esi
0x00407f0a pop ebp
0x00407f0b xor eax, eax
0x00407f0d pop ebx
0x00407f0e add esp, 0x260
0x00407f14 ret

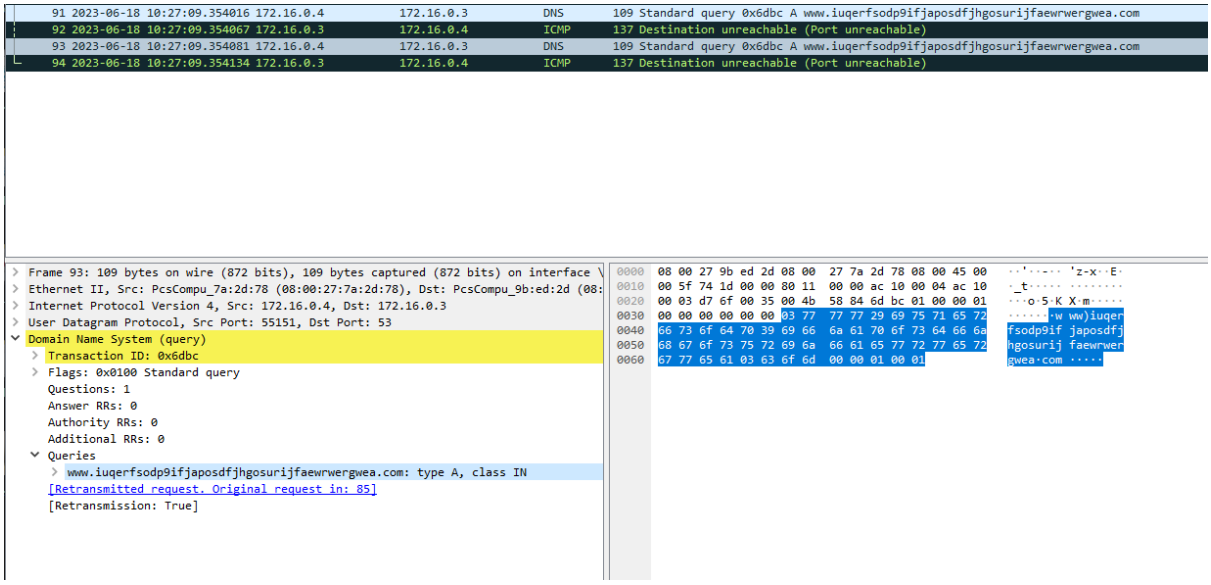
Dynamic Analysis

When executed with user privileges and inetsim the program makes a call to the URL hxxp://www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com/ we found earlier in static analysis and closes:

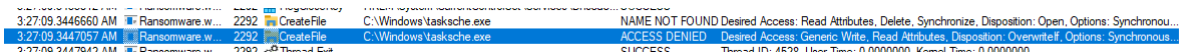


User privileges without Inetsim:

DNS request to the same url:

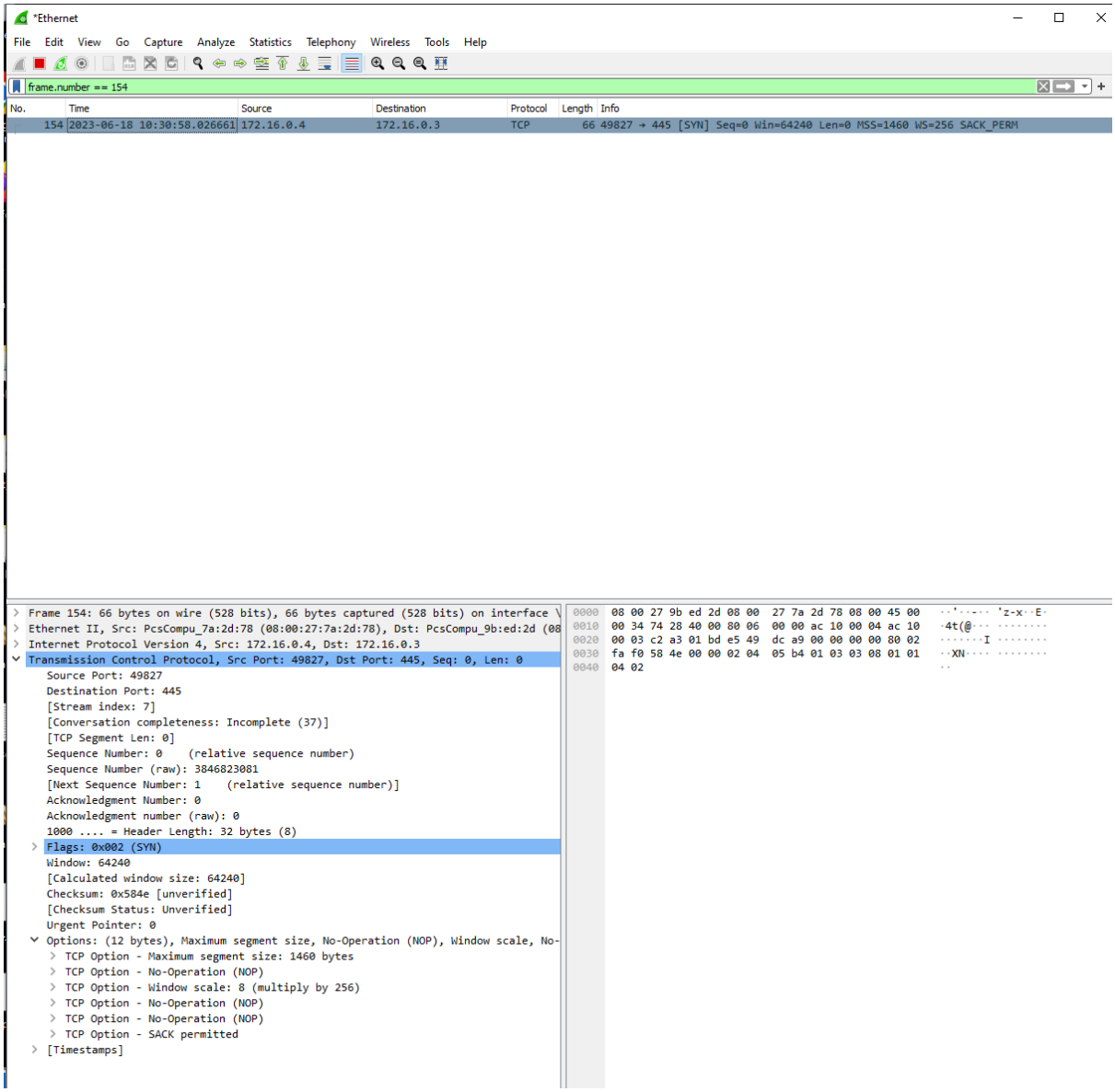


Attempts to create a file in C:\Windows named taskche.exe, fails and exits.

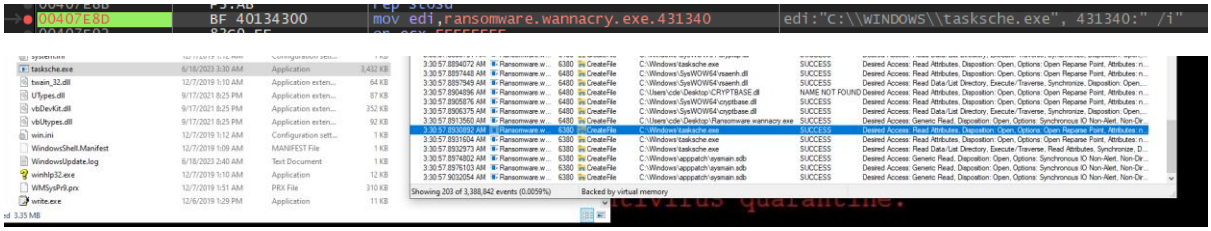


With admin Privileges:

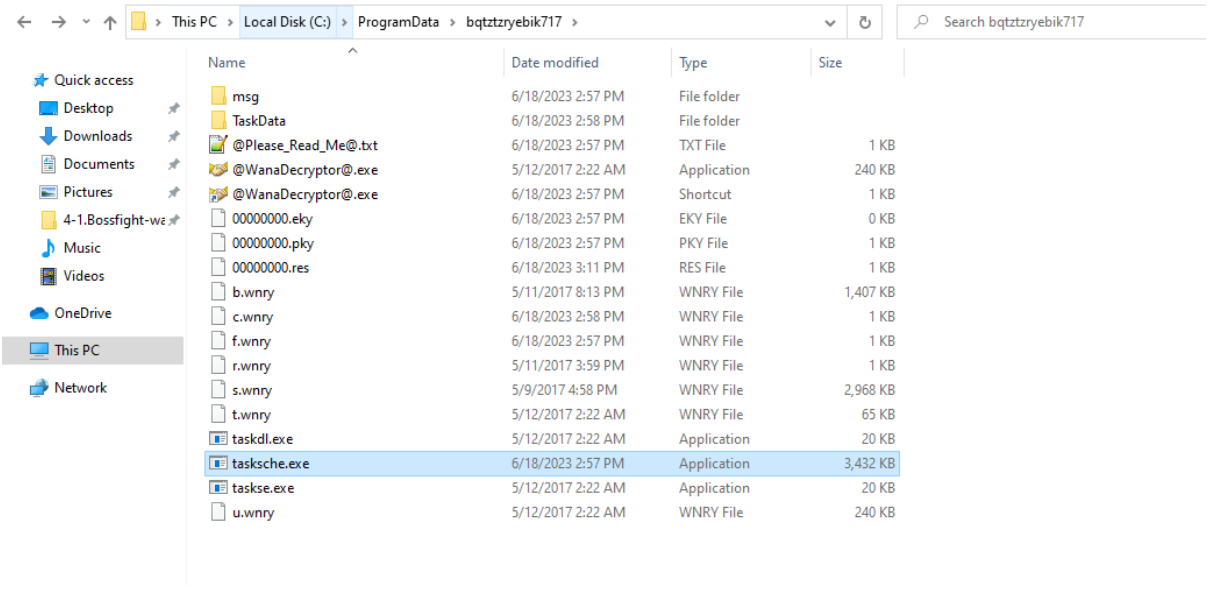
The program executes and encrypts user files. Performs ARP request to the network to find live hosts. Against the only other live host on the network, it attempts to connect to TCP port 445



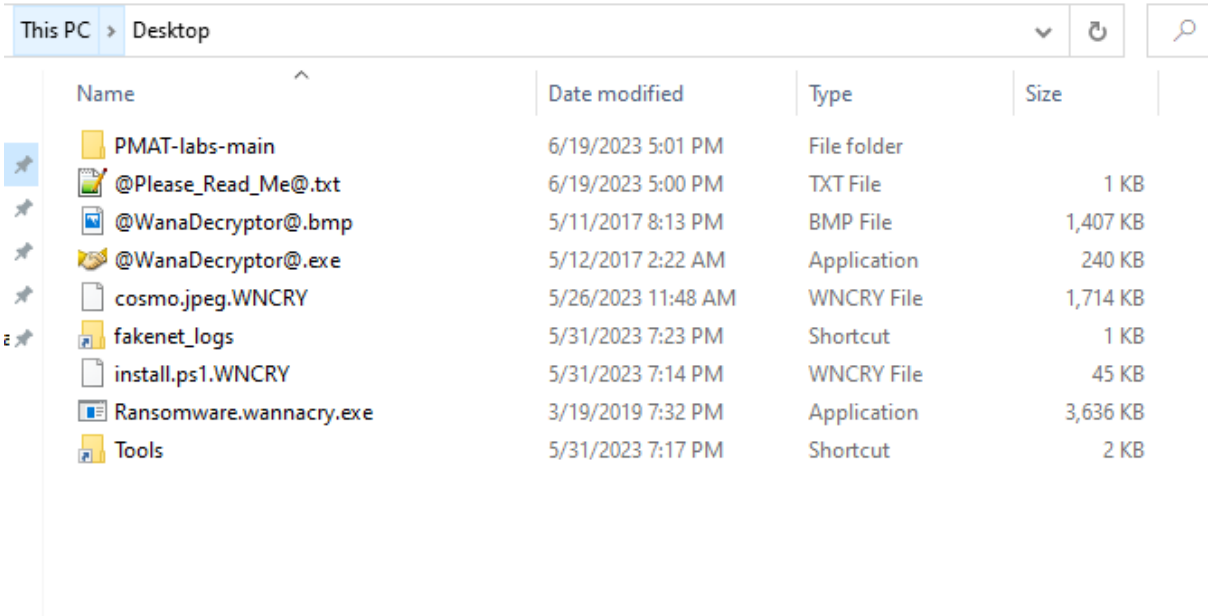
Creates an executable named taskche.exe in C:\ProgramData\bqtztzryebik717 along with the rest of the files and executes the encryptor with the command C:\Windows\Taskche.exe /i.



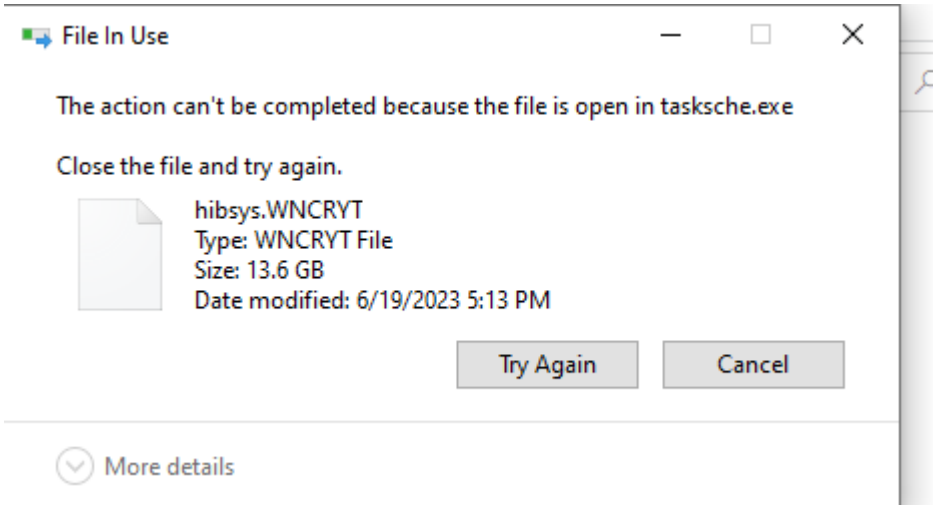
In this part we confirm that the strings we discovered earlier during the part of static analysis were actually file names.



Shortly after the execution, user files are encrypted and the suffix ".WNCRY" is appended to them, the background changes to the ransomware note and the files seen in the picture below are created on the Desktop.



Another finding that after the system encryption the file hibsys.WNCRYT is generated under the C:\Windows\Temp folder that is constantly used by tasksche.exe and grows in size. No further analysis was performed on that file.



During the analysis process it was confirmed that the malware encrypts the following type of files:

1. 7zip
2. Txt
3. Jpeg
4. Ps1

It is confirmed that the malware does not encrypt files with the following suffixes:

1. Exe
2. Md
3. Dat
4. Dll
5. Tmp
6. config
7. No suffix
8. Files under C:\Windows
9. Files under C:\Program Files
10. Files under C:\Program Files (x86)