Static analysis of WannaCry

Alföldi Mátyás

May 27, 2020

Contents

Introduction

Sample source: https://github.com/ytisf/theZoo

Tools used:

- Ghidra
- CFF Explorer
- Git bash(for file, etc.)
- VirtualBox
- Exeinfo PE
- strings.exe from SysinternalSuite
- oletools2

First Steps

1. Looking at the file in Exeinfo PE I see that a .Zip is found in section 4, and below it it also shows, that section 4, which is the .rsrc is big. This already hints that it might unpack more malware while running.

2. Opening it in CFF Explorer, and checking the Resource Editor tab shows that there is a XIA resource named 2058 that starts with the zip header.

3. Saving the resource and trying to unzip it results in a password being asked, which is most likely found in the file.

4. Using strings on the file, and looking through the output, we can see multiple interesting things:

   (a) Most likely messages designed for different languages: msg/m_*language*.wnry, also other .wnry files

   (b) Various exes: diskpart.exe, taskdl.exe, taskse.exe, tasksche.exe, cmd.exe /c "%s"

   (c) Various dll-s that are most likely loaded with LoadLibrary: kernel32.dll, advapi32.dll, msvcp60.dll, msvcrt.dll, ws_32.dll, oleaut32.dll, shell32.dll, user32.dll

   (d) Using a filter for winapi calls from here: https://resources.infosecinstitute.com/windows-functions-in-malware-analysis-cheat-sheet-part-1/ gives us some findings.

Interesting things found with Ghidra

Note: All files were renamed, so that their extension is only .ex .
Starting at the WinMain function we can see that calls GetModuleFileNameA for gettings its path, and it generates afterwards a random string. Afterwards we can see that it behaves in different ways:

1. If it has /i as its argument.
2. Any other cases.

Lets start with the first one:

1. Creation of a hidden system directory(First successful of these):
   - %WinDir% \ProgramData
   - %WinDir% \Intel
   - TMP folder
2. Using CopyFileA for naming itself as tasksche.exe
3. Creates an automatic service running tasksche.exe or just a hidden process if the first fails. (Using the random string generated at the beginning.

Second case:

1. Sets the directory to where the program is running.
2. Creates the following registry key(First successful):
   - HKLM \Software \WanaCrypt0r
   - HKCU \Software \WanaCrypt0r
3. We also set in the above mentioned key the following way:
   - dw value
   - String type
   - With the value returned by GetCurrentDirectoryA
   - Size: Length of the value returned + 1 (for zero terminator, also regedit only shows values for strings till the first zero terminator, hiding things would be possible after the zero terminator)
4. unpacks the resources found at the start, here we see that the password is: WNcry@2ol7 (TODO: go more in depth in the unpacking algo)

5. Modifies the extracted c.wnry to contain one of 3 bitcoin addresses. More about c.wnry in its subsection.
6. Adds the hidden attribute to the current directory.
7. Grans everyone Full access to the current directory.
8. Uses LoadLibraryA to load advapi32.dll and kernel32.dll to gather func ptrs to the following:
    - Cryptography related functions:
        - CryptAcquireContextA
        - CryptImportKey
        - CryptDestroyKey
        - CryptEncrypt
        - CryptDecrypt
        - CryptGenKey
    - File operation functions:
        - CreateFileW
        - WriteFile
        - ReadFile
        - MoveFile
        - MoveFileExW
        - DeleteFileW
        - CloseHandle
9. Imports a cryptographic key from an in memory key BLOB (public/private key pair)
10. Starts parsing t.wnry, which contains an AES key, that is later on decrypted with the help of the above mentioned cryptographic key
11. There is also some custom rijndael implementation, which I found out with the help of FindCrypt script.
12. TODO: continue the parsing process of t.wnry

Extracted files

0.1 b.wnry

0.2 c.wnry

c.wnry by default contains the following:

- onion links
- A link to tor

0.3 r.wnry

The Ransomware note.

0.4 s.wnry

Another zip file containing:

- RTF files in different languages, which might be malicious too.
- A Basic tor (TODO: check if it is a modified version)

0.5 t.wnry

Contains the following:

- An 8 byte magic number: WANACRY!
- A 4 byte key size: 00 00 01 00 = 256 AES has this length
- Another 4 bytes TODO: what exactly it is
- Another 8 bytes
- TODO

0.6 taskdl.exe

0.7 tasksе.exe

0.8 u.wnry

0.9 msg/m_*language*.wnry files

After looking at it in CFF Explorers hex view, I have seen some strange parts in it.
Generating its hash with sigcheck and looking it up in virustotal showed me that my suspicion was correct.

yara rules

Snort rules