CMSC 426 – Principles of Computer Security

Spring 2019

Lab 2 – Malware Analysis

Full name: Daniel Roh
UMBC username: Droh1

Part 1: Basic Static Analysis

1A. How many antivirus engines detect Lab2.exe as malicious? In the antivirus detections, which category of malware do they most frequently suggest Lab2.exe belongs to?
[4 points]

- 44 engines were detected for Lab2.exe.
- Lab2.exe was catorgraied mostly as a Trojan made for ransomware.

1B. What is the MD5 hash of Lab2.exe?
[2 points]

- MD5 = ef7ade5f3b8e395fcd03ae6da0d56fa9

1C. List three different Windows API functions imported by Lab2.exe that suggest that the malware belongs to the category you answered for question 1A. You can find official documentation by searching the Microsoft API documentation (we recommend googling “Microsoft documentation [API call name]”). Justify why you listed each function.
[9 points]

- CrpytEncrypt (ADVAPI32.dll) This function is used to encrypt data, This function can be called by an attacker to encrypt data in a ransom attack.
- CryptDestroyKey (ADVAPI32.dll) This function is used to invalidate a key used to encrypt data. This can be used to delete the key off the victim's computer to force the victim to get the key from the attacker.

- SHGetFolderPathW (SHELL32.dll) This function allows a user or an attacker to gain the information of the current location they are at in. By using this information, a ransomware can then make the necessary calls to get to a upper directory to launch an attack.

1D. What is the file type of the resource contained within Lab2.exe?
[2 points]

- Win32 EXE

1E. List two strings that appear in Lab2.exe that you believe to be significant. (The Windows API functions and their corresponding .dll files *do not* count for this question.) Justify why you chose each string.
[6 points]

- %&'()*456789:CDEFGHIJSTUVWXYZcdefhijstuvwxyz
- %'()*456789:CDEFGHIJSTUVWXYZcdefhijstuvwxyz
    - I chose these two strings because these seems to be two hashes that were created when the program ran the program. It could be the potential public and private keys that are being generated.

1F. What does this resource seem to be? Which indicators of compromise are in the resource?
[4 points]

- This resource seems to be the image that is shown to the victim after the attack has happened giving the victim information on how to send money.
- Indicators of compromise are the Dogecoin wallet address and a email address left on the image.
    - side note: dog picture is not a doge :(

Part 2: Basic Dynamic Analysis

2A. What debug message does Lab2.exe print?
[2 points]

- “Cannot find Documents folder on desktop…”

2B. What does the second debug message say?
[2 points]

- “Bad volume serial num”

Part 3: Advanced Dynamic Analysis

3A. What is the address of the instruction
`CALL GetVolumeInformationA`?
[2 points]

- 0040105D

3B. What is the address at `EBP-4`? What is the value currently at this address?
[5 points]

- 0012FB68

3C. What is the value of the VM’s volume serial number? (Leave your answer in hexadecimal.)
[3 points]

- 54B1775E

3D. What is the value of the volume serial number that the malware is targeting? (Leave your answer in hexadecimal.)
[4 points]

- 7DAB1D35

Part 4: Advanced Static Analysis (and more Advanced Dynamic)

4A. In the xrefs window, you'll see an address. The first part of it is the calling subroutine's name, and the second part is the offset. (So, for example, an address of **`sub_A87E03+200`** would be called **`sub_A87E03`**, and its address would be **`0xA88003`**, the sum of the two. Remember, the subroutine and the offset are both in hexadecimal.) Which subroutine calls **`sub_401940`**? What is the address of the call **`sub_401940`** instruction?
[8 points]

- sub_4019A0
- 401AC2

4B. What is the address of the byte sequence?
[2 points]

- 00403000

4C. What is the address of the empty buffer?
[2 points]

- 001443B8

4D. What is the length of the byte sequence? (Leave your answer in hexadecimal.)
[2 points]

- 00000025

4E. What is the de-obfuscated string that the malware uses to seed its cryptographic key?
[3 points]

- "THIS_IS_THE_SUPER_SECRET_AES_256_SEED"

4F. Which subroutine calls **CryptDeriveKey**? What is the address of the **call CryptDeriveKey** instruction?
[3 points]

- sub_401180
- 4011FF

4G. What is the value of **Algid**? (You will probably need to cross-reference information in IDA Pro about where **Algid** is being stored, and the value OllyDbg shows is stored at that spot.)
[6 points]

- Algid = 00006610

4H. Using this website: https://docs.microsoft.com/en-us/windows/desktop/seccrypto/alg-id, search for the algorithm ID. Which cryptographic algorithm corresponds to the algorithm ID you found in question 4G?
[4 points]

- 256 bit AES

Part 5: Wrapping Up

5A. What is the offset of **CryptEncrypt** in HxD?
[4 points]

- 12 bytes
- 000014FC

5B. Search the Microsoft API documentation for the **CryptEncrypt** function. You should be able to find another function in the documentation that is its "inverse." What is the name of this function that is the "inverse" of **CryptEncrypt**?
[6 points]

- CryptDecrypt

5C. Follow the instructions in flag1.txt. Put your answer here.
[15 points]

- The Jerusalem Virus. This virus that ran on DOS machines and hides inside the memory of its host computer and infects every executable file that is run on the computer. Then on every Friday the 13th, the virus deletes every executable file on the infected computer. I think it's interesting that someone created a virus that has ties to Friday the 13th.