Web Security and Malware Analysis

Assignment 6 – 25/03/2021

Goal: Solve each task by providing a brief explanation of the solution that you adopted. Answers should mainly include screenshots and some brief comments. Use the provided template to write your assignment. It is recommended to use IDA FREE or IDA PRO to solve Task 2.

What to do: Save the Assignment as a PDF and submit it to the E-Learning platform (**Section: Assignment 6**). The file should be named as follows: **"websec_report_6_name_surname.pdf".** Remember to include your name, surname, and matriculation number in your report.

Starting Notes: Please remember that the files you are going to deal with are real malicious samples, which could potentially destroy your machine. DO NOT EXECUTE OR OPEN THEM in your own system. ALWAYS USE A VIRTUAL MACHINE (VMware/VirtualBox) FOR EVERY OPERATION. Save a snapshot. Additionally, you could execute the tools in a virtualized environment with the Sandboxie tool (https://www.sandboxie.com/). You find the executables for this assignment on the e-learning platform.

(password: **malwanalysis**)

Additional notes: if you have problems with Windows Defender, please follow these instructions to disable it:

https://www.windowscentral.com/how-permanently-disable-windows-defender-antivirus-windows-10

If you have problems at running IDA FREE/PRO, please install Microsoft Visual C++ Redistributable Package:

https://www.microsoft.com/en-us/download/details.aspx?id=5555

Task 1 – PE Structure

Consider the files Lab_01.exe and Lab_01.dll. What can you say about the structures of the two files (**use PEid and CFF Explorer**)? In particular, for each file, you are required to analyze the following elements:

1) The DOS Header and Stub. Which differences can you find, if any?
2) The File and the Optional headers of both files. Which parts of these headers you think are critical for the execution of these programs? In particular, discuss the differences that you see between:
   a. The bases of code and data.
   b. The image bases. What do they represent and what can you say about them?
3) The Section header, along with the role of relocation.

Now, consider the file Lab_03.exe. What about its Sections? Do you see anything strange? Why? Comment on it.

BONUS: Try using VirusTotal (www.virustotal.com) and IDA to have a quick look at Lab_01.exe and Lab_01.dll. What do you think these programs do?

Task 2 – IDA PRO Practice

Consider the file Lab_05.dll. Answer the following questions by using IDA PRO (don't be scared by the number, each of them requires a fast answer). Hint: if you need help on how IDA works, check chapter 5 of the "Practical Malware Analysis" book.

1. What is the address of **DllMain (Note: if you use IDA Freeware, find the address of DllEntryPoint)**?

2. Use the Imports window to browse to **gethostbyname**. Where is the import located?

3. How many functions call **gethostbyname**?

4. Focusing on the call to **gethostbyname** located at **0x10001757**, can you figure out which DNS request will be made?

5. How many local variables has IDA Pro recognized for the subroutine at **0x10001656**?

6. How many parameters has IDA Pro recognized for the subroutine at **0x10001656**?

7. Use the Strings window to locate the string **\cmd.exe /c** in the disassembly. Where is it located?

8. What is happening in the area of code that references **\cmd.exe /c**?

9. In the same area, at **0x100101C8**, it looks like **dword_1008E5C4** is a global variable that helps decide which path to take. How does the malware set **dword_1008E5C4**? (Hint: Use **dword_1008E5C4'**s cross-references.)

10. A few hundred lines into the subroutine at **0x1000FF58**, a series of comparisons use **memcmp** to compare strings. What happens if the string comparison to **robotwork** is successful (when memcmp returns 0)?

11. What does the **export PSLIST** do?

12. Use the graph mode to graph the cross-references from sub_10004E79. Which API functions could be called by entering this function? Based on the API functions alone, what could you rename this function?

13. How many Windows API functions does **DllMain** call directly? How many at a depth of 2?

14. At 0x10001358, there is a call to **Sleep** (an API function that takes one parameter containing the number of milliseconds to sleep). Looking backward through the code, how long will the program sleep if this code executes?

15. At 0x10001701 is a call to socket. What are the three parameters?