Applying DLL and code injection

Lab Manual

Objectives

- Create a remote thread
- Inject code into DLL
- Evaluate file hiding techniques
- Write backdoor code
- Compile python script into windows executable

Table of Contents

Objectives.....1
Background Reading.....3
Notes common to all lab and home assignment problems.....3
Lab specific intro.....3
Problem 1.....3
Problem 2.....4
Problem 3.....4

1.

Background Reading

Textbook:

Grey Hat Python, chapter 7

Other resources:

https://www.sans.org/reading-room/whitepapers/malicious/basic-reverse-engineering-immunity-debugger-36982
https://sgros-students.blogspot.ca/2014/05/immunity-debugger-basics-part-1.html

Notes common to all lab and home assignment problems

For every lab and home assignment, all work should go into your personal repository, subdirectory named mXX, where XX stands for the module number. For each problem, carefully name the program as described. The programs are extracted from your repository by a Python script, and errors in the program name will result in the instructor never seeing your program, and your mark for it will be ZERO!

Anything to record and report in this lab is to be written in plain text file (Word document is not a plain text file). The file MUST be named mXXpYY.txt, where XX is the module number, YY is the problem number. Mis-naming these files, or not having them in the proper location will result in mark ZERO for anything to be recorded or reported.

There are always many ways how to solve a programming problem, and usually one or two ways which are fast, compact and elegant.

Make sure to push your work to the server often, and have pushed the working version of the program by the deadline specified. The script extracting your programs from your repository will be run at any time after the deadline.

Lab specific intro

In this lab we will firs create remote thread by calling windows function CreateRemoteThread() in kernel32.dll. We will utilize the ghp_inject.dll by downloading the code from greyhat python website.

The PyCommands are found in:

“/c/Program Files/Immunity Inc/Immunity Debugger/PyCommands”

To be able to edit files at that location, start your code editor (e.g., emacs) with admin privileges.

Problem 1

Practice DLL injection. Use the dll_injector.py from the book. Start a process calc.exe and find its process id. The dll to inject into the process is the ghp_inject.dll from the grey hat python website.

file ghp_inject.dll. This is a simple dll which opens a message box when loaded.

Use the LoadLibrary() function call from kernel32.dll.

Record the address kernel allocated for the DLL path.

Record the location of the LoadLibraryA() function.

What is the meaning of the values the following parameters are set to in dll_injector.py?

PAGE_READWRITE

PROCESS_ALL_ACCESS

VIRTUAL_MEM

What is the purpose of the h_loadlib variable?

Problem 2

Practice injecting code using the code_injector.py from the book.

Create one cmd.exe and one calc.exe process, use the cmd.exe as the process to inject, and the calc.exe as the process to kill.

Run the script with appropriate parameters and demonstrate the victim process being killed.

Problem 3

Exploit alternate data streams on NTFS file system. Follow chapter 7.2.1 in the Grey Hat Python book, and write the `file_hider.py` code. Improve the code so it first asks for the the name of the file to attach to, then for the name of DLL to be hidden interactively. Submit your result as **`m09p02.py`**

Problem 4

You will create a backdoor to a computer by writing a code which will masquarade as a benign windows application. We will pick calc.exe. We will move the original calc.exe to different location, as we will need to execute the real calculator (it is what the user is expecting). Use the code snippets in the Gray Hat Python book, chapter 7.2.2, and create complete exploit. Submit the complete exploit as `m09p04.py`

Problem 5

Convert the exploit written in Python to windows executable. Follow the process as described in chapter 7.2.3 of the textbook. Install the py2exe python library on your Windows machine, write the setup.py script to control the build, and execute it. The executable shall be named **`m09p05.exe`**

Problem 6

Code the **`backdoor_shell.py`** server, which will listen on port 4444. Run it in separate virtual machine, and execute the exploit of problem 5 on the victim machine. Cut and paste your interaction with the victim machine into your lab report **`m09p06.txt.`**