Lab 1: DLL Injection and OllyDbg

Mayur Suresh

CSC 471-01: Modern Malware Analysis

Date: February 13th, 2023

Objectives:

One of the main objectives for this lab to get familiar with the VirtualBox software as well as using OllyDbg software to modify binary files. We will also understand the concept of DLL Injection. We are going to change the debug information in the DebugView window in VirtualBox from 'Injection!!! – CSC 497/583 – Si Chen' to 'Hello World! – Mayur Suresh'

Process:

We first download VirtualBox and download the Windows XP VirtualBox image to use for this lab. We then download and unzip the hack_dll.zip folder. We see underneath there are two files, one is the dynamic-linkage software and the other a dynamic-linkage file that we are going to inject. We then open a software like a notepad to see under the ProcessExplorer to see a process being created when a software is opened as shown in Image1. In order to inject this malware we type this follow address in the command prompt with the notepad's PID from the ProcessExplorer.

'InjectDll.exe 1016 C:\Work\myhack.dll'

After launching this task we see the confirmation of the attack in the cmd as shown below in Image2. We see after launching this attack in the DebugView that there is a message printed out as shown in Image3, although we do not see the whole message in the screenshot, we see the message 'Injection!!! – CSC 497/583 – Si Chen' when we hover over the row.

hack.dll with OllyDbg:

At this point we can go ahead and start using OllyDbg to launch the attack using the dynamic-linked file. We run the OllyDbg software and open the hack.dll file as shown in Image 4 below. We use this software to reverse engineer a binary file to see what goes on under the hood with assembly language. We see the memory address at the left column, next we see the assembly instructions in the middle column which shows how the code moves, to the right most column we see the registers for the assembly, in the bottom left column it displays real time memory addresses when the program is executed and at the bottom right column, we see the stack frame. We move on to navigate this software and we see the link from which data is downloaded in Unicode at the memory address '1000107b' as shown in Image 5. Similarly, when we move further down the middle column, we come across the string we received when the malware was injected at memory address '100010B1' as shown in Image 6.

To achieve the goal of this lab and modify that exact string through OllyDbg as we don't have access to the source code. We do this by taking note of the memory address of where the myhack.dll string is being pushed on to, in this case '10010B20' we can go to this memory address and hack this string. We do this by first going to the Hex dump column right-click, go to the 'go to' option and click Expression which brings us to a text box where we type in the memory address as shown in Image 7 which gives us the hex values along with the associated memory address as shown in Image 8.

We then highlight the corresponding strings you like to change right-click it chooses the binary option and edit which brings us to a box with the corresponding ASCII values, Unicode and the Hex decimal values as shown in Image 9. We then change the values in the corresponding ASCII value box and change it, we then see the changes in the hex dump as shown in Image 10. We then highlight the edit parts, right-click it then selects copy to executable file which brings us to a window that shows us that change as shown in Image 11. Next, we need to save the file under a different name as shown in Image

12. Now that we have the hack2.ddl we restart the whole process and get the edited message in the DebugView as shown in Image 13.

Conclusion:

Finally, with this lab we learned the vulnerability of a program even without the source code. We used the OllyDbg software to get access to the assembly code and with the capabilities to edit, used it to change the string, in other words the ability to hack a program from the inside by rewriting a string. Using the procedures above we get from Image 3 with the initial message of 'Injection!!! – CSC 497/583 – Si Chen' to the expected message of 'Hello World! – Mayur Suresh' in Image 13

Source Images

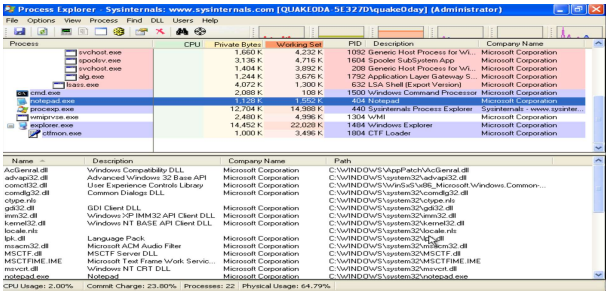

Image 1

Image 2

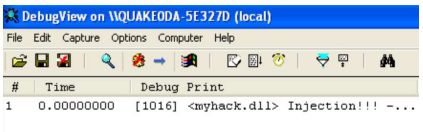

Image 3

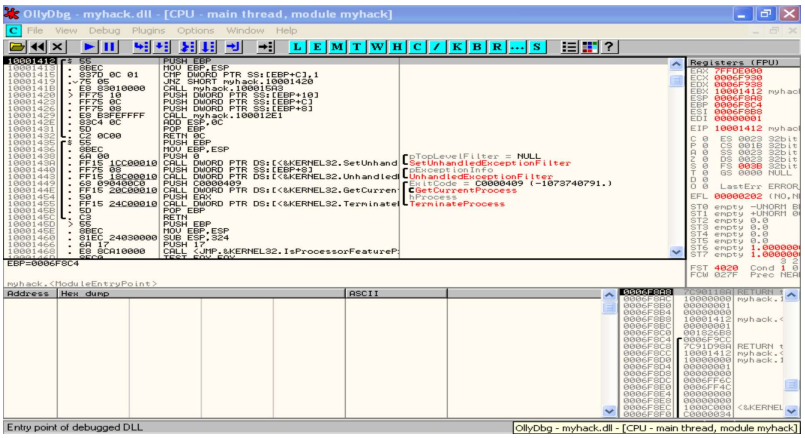

Image 4

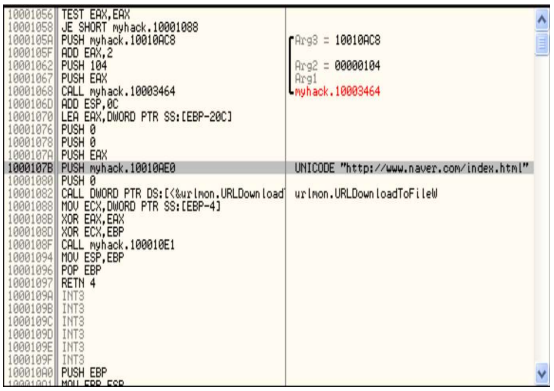

Image 5

Image 6

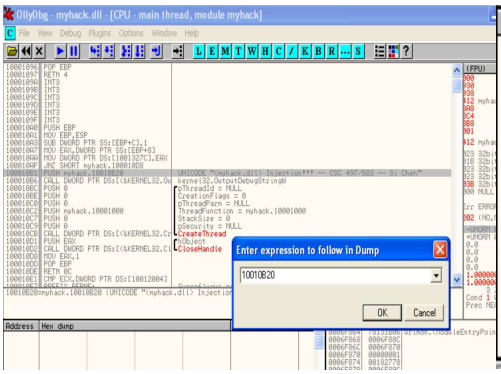

Image 7

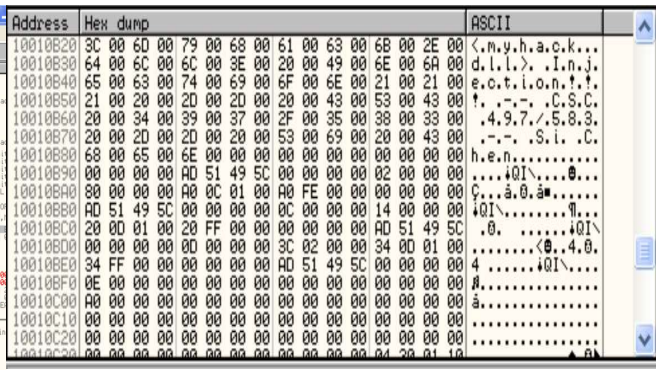

Image 8

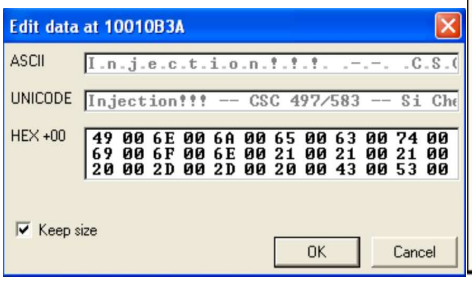

Image 9

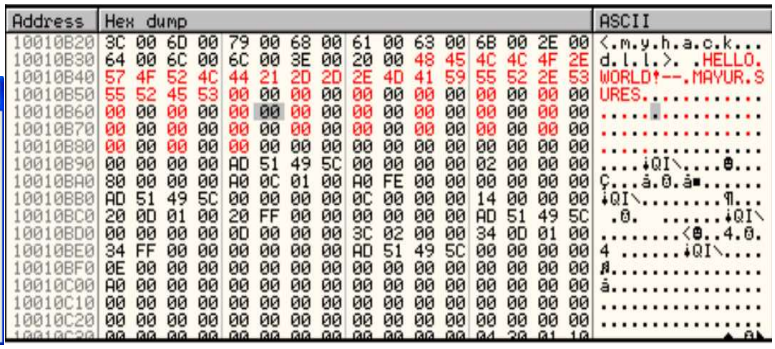

Image 10

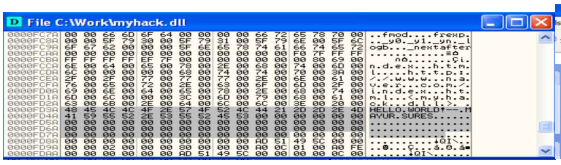

Image 11

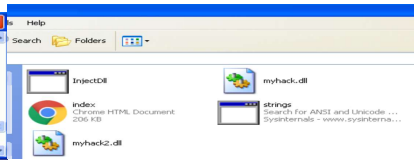

Image 12

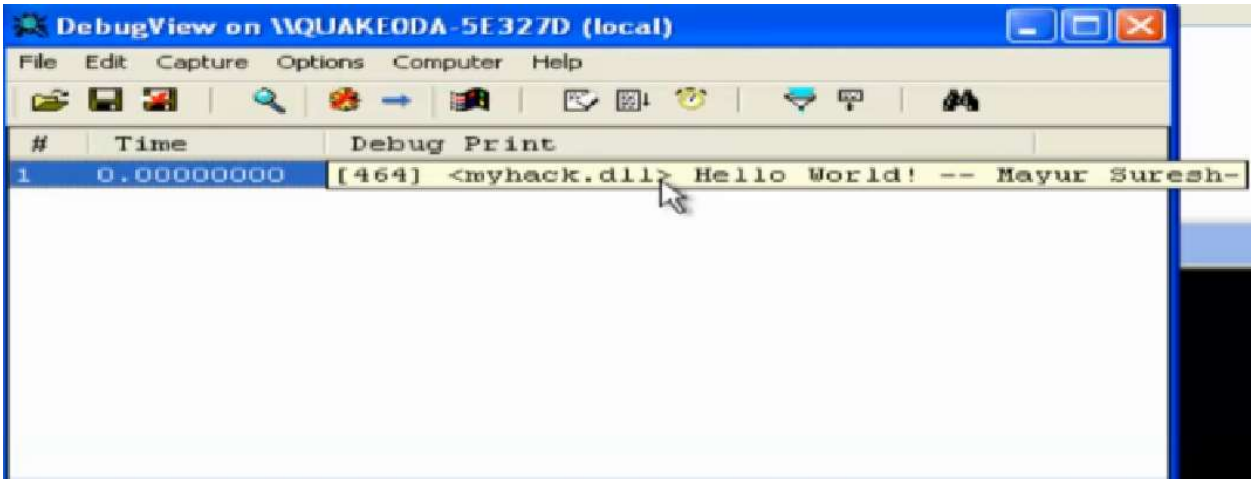

Image 13