Lab 2: Stack and Stack Frame

Mayur Suresh

CSC 471-01: Modern Malware Analysis

Date: March 3rd, 2023

Objectives:

The main objective of this lab is to remove the 'Nag Screen' that pops up when we run the lab.

Process:

We first launch Virtual Box and Boot up windows XP. We then download the files necessary to operate this lab with the link `https://www.cs.wcupa.edu/schen/malware23/download/lab2.exe`. We use the Ollydbg to run Lab2 as shown in Image1. We also see the 'Nag Tab' in Image 2 below.

Prompt #1:

Which CPU register is used to store the return value (1) of the function rtcMsgBox()? Why?

The return value (1) is stored in the register EAX (as shown in Image 3) because the address 0x402CFD returns the EAX register, which then goes to the next line where the address 0x402CFE calls the function rtcMsgBox() with the `jmp` command. This is because it is more effective to store the return address in this CPU register. Another reason why it is stored this way is that arguments are passed on the stack in a Right-to-Left pattern, so return values are passed in EAX.

Prompt #2:

What's the meaning of "PUSH EBP, MOV EBP, ESP"?

In most assembly language programs, the combination of PUSH EBP with MOV EBP, ESP is typically the sequence that is used to start a stack frame. Since EBP is the base pointer, the PUSH instruction will push the base pointer onto the stack. ESP is the stack pointer, so the MOV instruction will move the value of the stack pointer into EBP, which will start the stack frame of the function by having the base pointer point to the beginning of the stack. We can see an example of this command in Image 4, which starts the stack frame for a function.

Prompt #3:

Please explain why changing the instruction on 0x402C17 from "PUSH EBP" to "RETN 4" removes the Nag screen.

We first locate the address 0x402C17 on the stack frame and switch the instruction from 'PUSH EBP' to 'RETN 4', as shown in Image 5. We now see that the nag tab is no longer produced, and instead we see the actual tab, as shown in Image 6. One reason why switching the commands from PUSH EBP to RETN 4 terminated the nag tab is because the commands that initially followed the address 0x0402C17 are 'MOV EBP, ESP', which as discussed above triggers the start of a new stack frame. Hence, switching the base pointer with RETN 4 literally skipped

the function that gave us the nag tab. By switching the commands, we get the new tab shown in Image 6.

Conclusion:

In this lab, we discussed the stack frame using OllyDbg, the various assembly instructions such as the base pointer, the stack pointer, and the return address instructions. We also modified the instructions to bypass a function by slightly tweaking its instructions to gain access to the next function. By doing this, we achieved the goal of this lab, which was to eliminate the nag tab and gain access to the next tab.

Source Images:

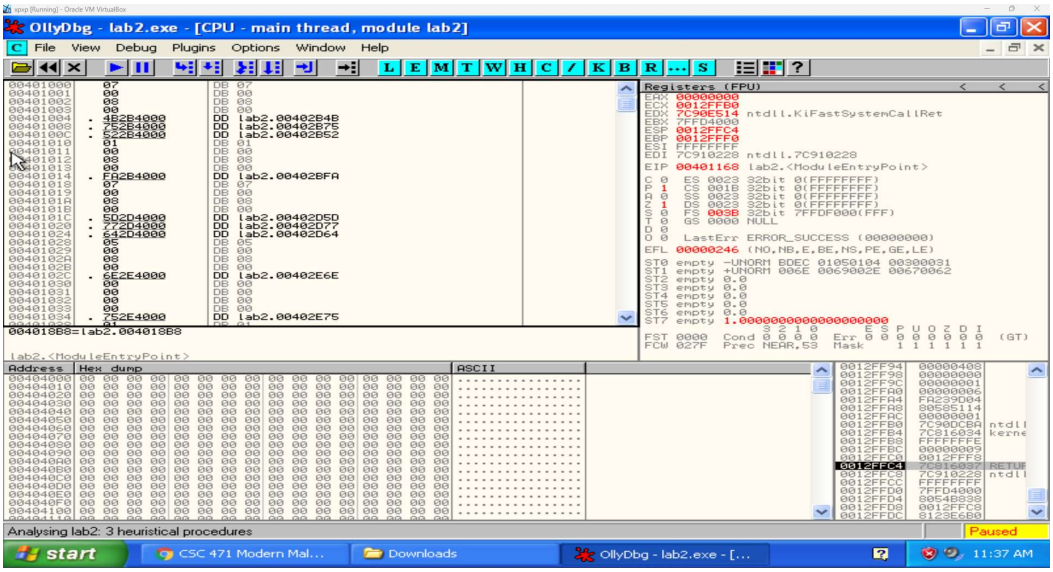

Image 1

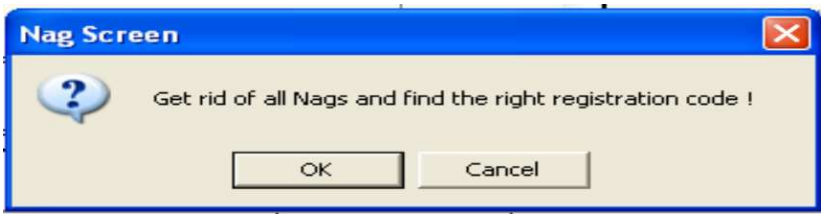

Image 2

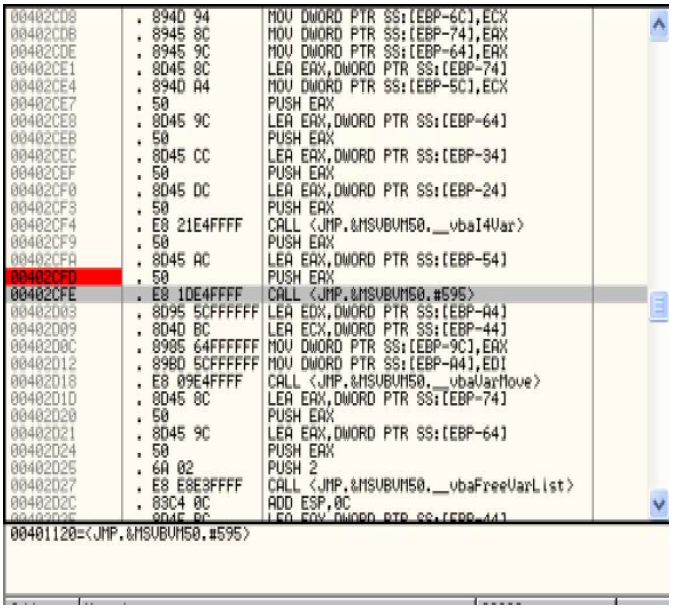

Image 3

Image 4

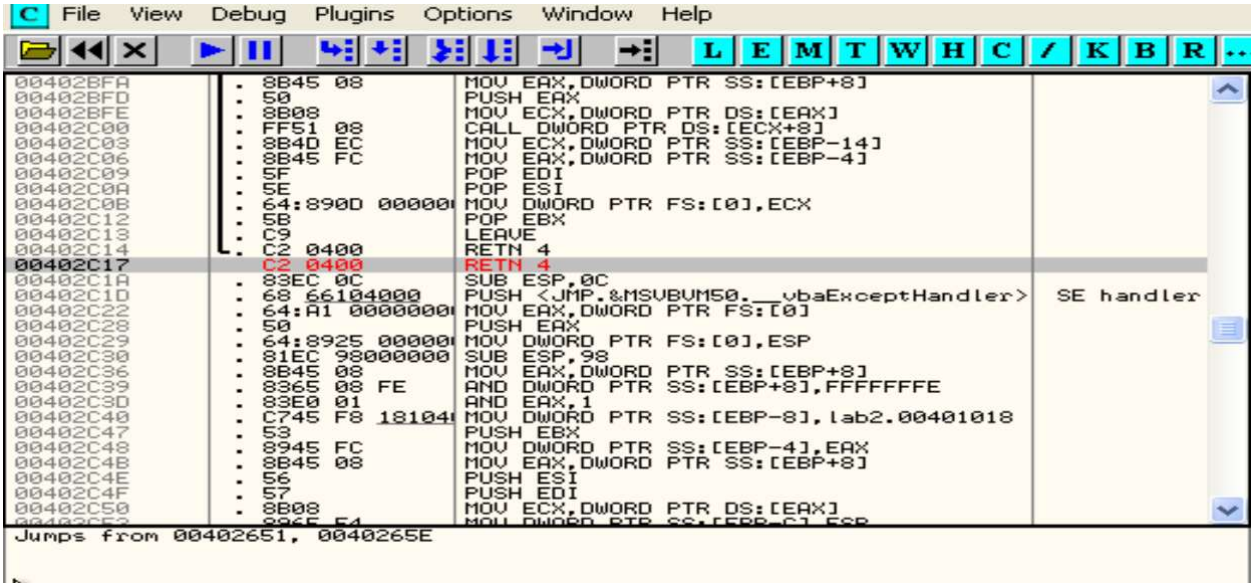

Image 5

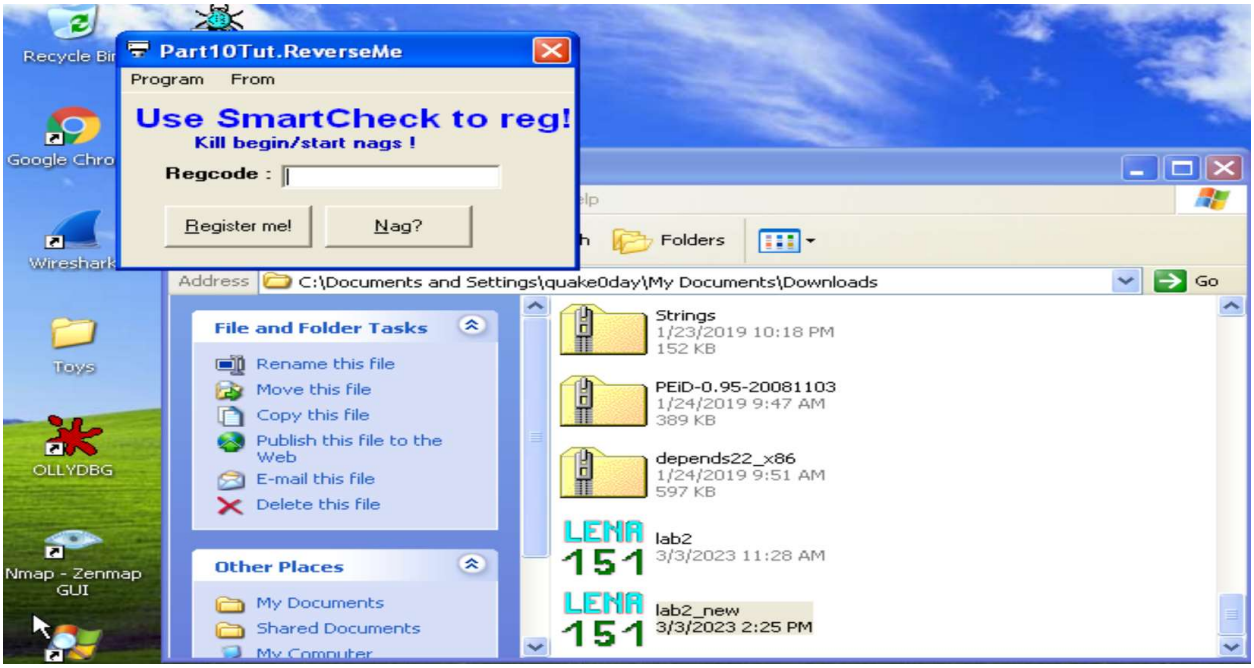

Image 6