# CSC 471/583 Spring 2022 Lab 3

Dr. Si Chen

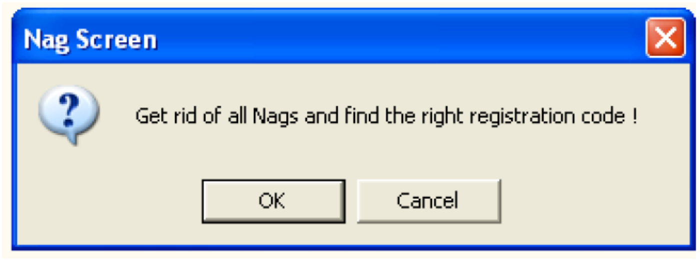
## Stack and Stack Frame

The goals of this lab:

- Understanding the concepts of Stack and Stack Frame.
- Know how to use OllyDbg to modify binary files.

### Objectives and Targets

Please download the lab3.exe into your Windows XP VM and run it. It will pop up a Nag screen (Shown in Fig. 1). Your task is to complete remove the Nag screen by modifying the binary program using OllyDbg.



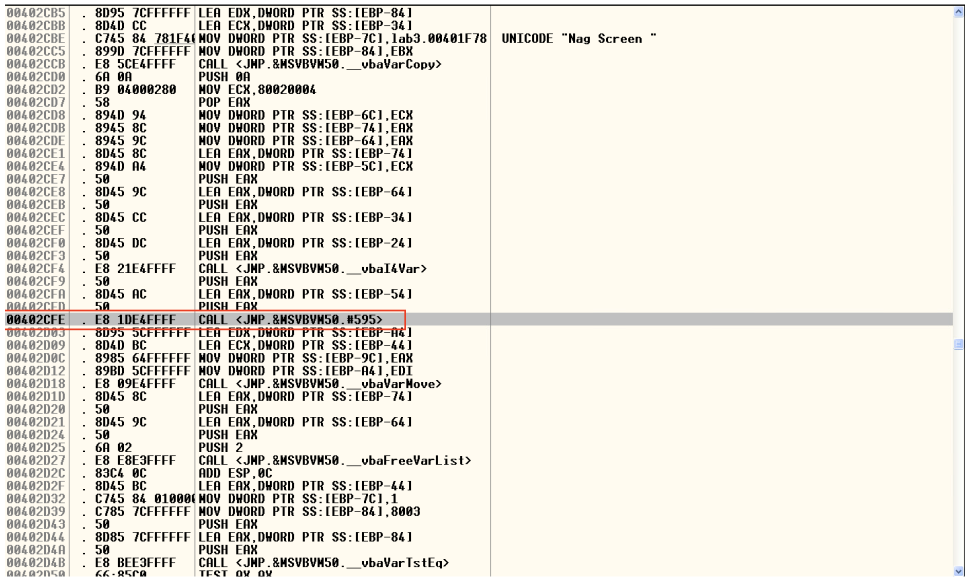
Get rid of all Nags and find the right registration code !

Figure 1: Nag Screen of lab3

#### Experiment Setup

1. Boot up windows XP inside VirtualBox.
2. Inside windows XP, download or copy the lab3.exe to a folder.
   ```
   https://www.cs.wcupa.edu/schen/malware2022/download/lab3.exe
   ```
3. Use Ollydbg to open lab3.exe.
4. Read the following questions (in the next section) and answer them in your report.

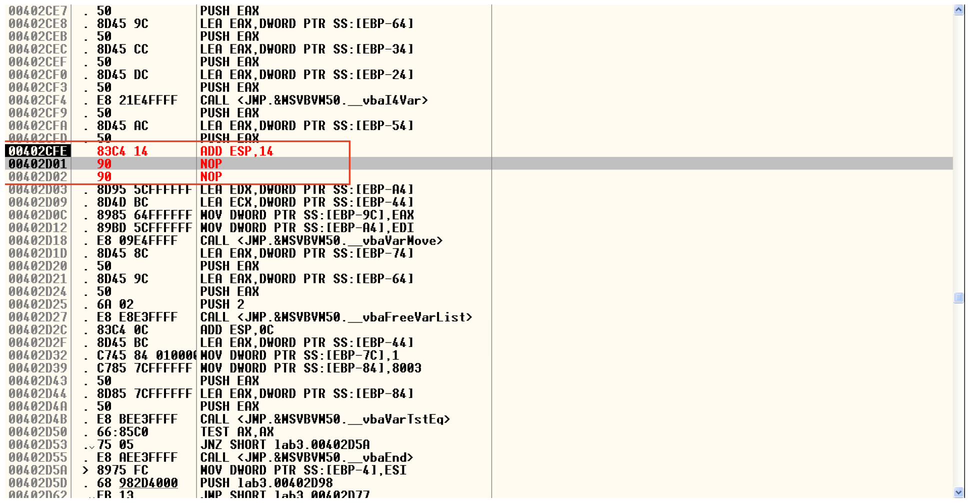
## Lab Questions



```
00402CB5  . 8D95 7CFFFFFF LEA EDX,DWORD PTR SS:[EBP-84]
00402CBB  . 8D4D CC       LEA ECX,DWORD PTR SS:[EBP-34]
00402CBE  . C745 84 781F4(MOV DWORD PTR SS:[EBP-7C],lab3.00401F78  UNICODE "Nag Screen "
00402CC5  . 899D 7CFFFFFF MOV DWORD PTR SS:[EBP-84],EBX
00402CCB  . E8 5CE4FFFF   CALL <JMP.&MSVBVM50.__vbaVarCopy>
00402CD0  . 6A 0A         PUSH 0A
00402CD2  . B9 04000280   MOV ECX,80020004
00402CD7  . 58            POP EAX
00402CD8  . 894D 94       MOV DWORD PTR SS:[EBP-6C],ECX
00402CDB  . 8945 8C       MOV DWORD PTR SS:[EBP-74],EAX
00402CDE  . 8945 9C       MOV DWORD PTR SS:[EBP-64],EAX
00402CE1  . 8D45 8C       LEA EAX,DWORD PTR SS:[EBP-74]
00402CE4  . 894D A4       MOV DWORD PTR SS:[EBP-5C],ECX
00402CE7  . 50            PUSH EAX
00402CE8  . 8D45 9C       LEA EAX,DWORD PTR SS:[EBP-64]
00402CEB  . 50            PUSH EAX
00402CEC  . 8D45 CC       LEA EAX,DWORD PTR SS:[EBP-34]
00402CEF  . 50            PUSH EAX
00402CF0  . 8D45 DC       LEA EAX,DWORD PTR SS:[EBP-24]
00402CF3  . 50            PUSH EAX
00402CF4  . E8 21E4FFFF   CALL <JMP.&MSVBVM50.__vbaI4Var>
00402CF9  . 50            PUSH EAX
00402CFA  . 8D45 AC       LEA EAX,DWORD PTR SS:[EBP-54]
00402CFD  . 50            PUSH EAX
00402CFE  . E8 1DE4FFFF   CALL <JMP.&MSVBVM50.#595>
00402D03  . 8D95 5CFFFFFF LEA EDX,DWORD PTR SS:[EBP-A4]
00402D09  . 8D4D BC       LEA ECX,DWORD PTR SS:[EBP-44]
00402D0C  . 8985 64FFFFFF MOV DWORD PTR SS:[EBP-9C],EAX
00402D12  . 89BD 5CFFFFFF MOV DWORD PTR SS:[EBP-A4],EDI
00402D18  . E8 09E4FFFF   CALL <JMP.&MSVBVM50.__vbaVarMove>
00402D1D  . 8D45 8C       LEA EAX,DWORD PTR SS:[EBP-74]
00402D20  . 50            PUSH EAX
00402D21  . 8D45 9C       LEA EAX,DWORD PTR SS:[EBP-64]
00402D24  . 50            PUSH EAX
00402D25  . 6A 02         PUSH 2
00402D27  . E8 E8E3FFFF   CALL <JMP.&MSVBVM50.__vbaFreeVarList>
00402D2C  . 83C4 0C       ADD ESP,0C
00402D2F  . 8D45 BC       LEA EAX,DWORD PTR SS:[EBP-44]
00402D32  . C745 84 01000(MOV DWORD PTR SS:[EBP-7C],1
00402D39  . C785 7CFFFFFF MOV DWORD PTR SS:[EBP-84],8003
00402D43  . 50            PUSH EAX
00402D44  . 8D85 7CFFFFFF LEA EAX,DWORD PTR SS:[EBP-84]
00402D4A  . 50            PUSH EAX
00402D4B  . E8 BEE3FFFF   CALL <JMP.&MSVBVM50.__vbaVarTstEq>
00402D50  . 66:85C0       TEST AX,AX
```

Figure 2: Dissembled code of lab3 - 0x402CFE

Dr. Chen figures out that this program is developed by the ancient Visual Basic (VB) language. And he finds that in 0x402CFE, the program is calling a VB function named rtcMsgBox(), which is the root cause of the annoying Nag screen (shown in Figure.3). So he plans to delete this function call to get rid of the Nag screen. He modify the instruction on 0x402CFE, changing it from "CALL XXXX" to "ADD ESP,14" (shown in Figure.4). Note that the length of CALL instruction is 5 bytes, and ADD instruction is 3 bytes, so OllyDbg will automatically add two "NOP" instructions to fill the remaining two bytes.



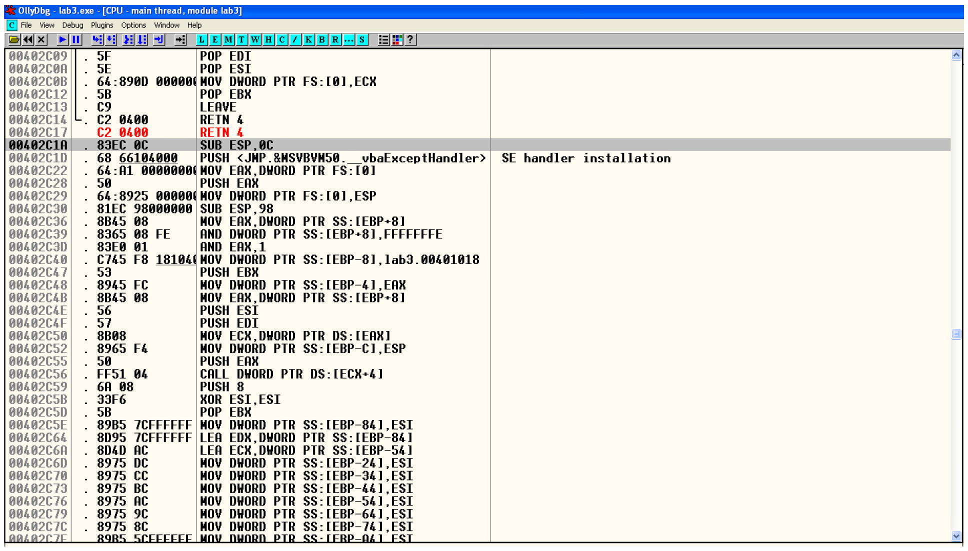
```
00402CE7  . 50            PUSH EAX
00402CE8  . 8D45 9C       LEA EAX,DWORD PTR SS:[EBP-64]
00402CEB  . 50            PUSH EAX
00402CEC  . 8D45 CC       LEA EAX,DWORD PTR SS:[EBP-34]
00402CEF  . 50            PUSH EAX
00402CF0  . 8D45 DC       LEA EAX,DWORD PTR SS:[EBP-24]
00402CF3  . 50            PUSH EAX
00402CF4  . E8 21E4FFFF   CALL <JMP.&MSVBVM50.__vbaI4Var>
00402CF9  . 50            PUSH EAX
00402CFA  . 8D45 AC       LEA EAX,DWORD PTR SS:[EBP-54]
00402CFD  . 50            PUSH EAX
00402CFE    83C4 14       ADD ESP,14
00402D01    90            NOP
00402D02    90            NOP
00402D03  . 8D95 5CFFFFFF LEA EDX,DWORD PTR SS:[EBP-A4]
00402D09  . 8D4D BC       LEA ECX,DWORD PTR SS:[EBP-44]
00402D0C  . 8985 64FFFFFF MOV DWORD PTR SS:[EBP-9C],EAX
00402D12  . 89BD 5CFFFFFF MOV DWORD PTR SS:[EBP-A4],EDI
00402D18  . E8 09E4FFFF   CALL <JMP.&MSVBVM50.__vbaVarMove>
00402D1D  . 8D45 8C       LEA EAX,DWORD PTR SS:[EBP-74]
00402D20  . 50            PUSH EAX
00402D21  . 8D45 9C       LEA EAX,DWORD PTR SS:[EBP-64]
00402D24  . 50            PUSH EAX
00402D25  . 6A 02         PUSH 2
00402D27  . E8 E8E3FFFF   CALL <JMP.&MSVBVM50.__vbaFreeVarList>
00402D2C  . 83C4 0C       ADD ESP,0C
00402D2F  . 8D45 BC       LEA EAX,DWORD PTR SS:[EBP-44]
00402D32  . C745 84 01000(MOV DWORD PTR SS:[EBP-7C],1
00402D39  . C785 7CFFFFFF MOV DWORD PTR SS:[EBP-84],8003
00402D43  . 50            PUSH EAX
00402D44  . 8D85 7CFFFFFF LEA EAX,DWORD PTR SS:[EBP-84]
00402D4A  . 50            PUSH EAX
00402D4B  . E8 BEE3FFFF   CALL <JMP.&MSVBVM50.__vbaVarTstEq>
00402D50  . 66:85C0       TEST AX,AX
00402D53  .v75 05         JNZ SHORT lab3.00402D5A
00402D55  . E8 AEE3FFFF   CALL <JMP.&MSVBVM50.__vbaEnd>
00402D5A  > 8975 FC       MOV DWORD PTR SS:[EBP-4],ESI
00402D5D  . 68 982D4000   PUSH lab3.00402D98
00402D62  .vEB 13         JMP SHORT lab3.00402D77
```

Figure 3: Modified dissembled code of lab3

However, this modification does not work – it keeps showing errors. After reading the code, he found that the function rtcMsgBox() should return a value 1 to indicate the message box is successfully displayed, but his modification does not provide the correct return value.

Please answer the following question:

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


OllyDbg - lab3.exe - [CPU - main thread, module lab3]
File View Debug Plugins Options Window Help

```
00402C09  . 5F              POP EDI
00402C0A  . 5E              POP ESI
00402C0B  . 64:890D 000000( MOV DWORD PTR FS:[0],ECX
00402C12  . 5B              POP EBX
00402C13  . C9              LEAVE
00402C14  . C2 0400         RETN 4
00402C17    C2 0400         RETN 4
00402C1A  . 83EC 0C         SUB ESP,0C
00402C1D  . 68 66104000     PUSH <JMP.&MSVBVM50.__vbaExceptHandler>   SE handler installation
00402C22  . 64:A1 00000000( MOV EAX,DWORD PTR FS:[0]
00402C28  . 50              PUSH EAX
00402C29  . 64:8925 000000( MOV DWORD PTR FS:[0],ESP
00402C30  . 81EC 98000000   SUB ESP,98
00402C36  . 8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
00402C39  . 8365 08 FE      AND DWORD PTR SS:[EBP+8],FFFFFFFE
00402C3D  . 83E0 01         AND EAX,1
00402C40  . C745 F8 18104(  MOV DWORD PTR SS:[EBP-8],lab3.00401018
00402C47  . 53              PUSH EBX
00402C48  . 8945 FC         MOV DWORD PTR SS:[EBP-4],EAX
00402C4B  . 8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
00402C4E  . 56              PUSH ESI
00402C4F  . 57              PUSH EDI
00402C50  . 8B08            MOV ECX,DWORD PTR DS:[EAX]
00402C52  . 8965 F4         MOV DWORD PTR SS:[EBP-C],ESP
00402C55  . 50              PUSH EAX
00402C56  . FF51 04         CALL DWORD PTR DS:[ECX+4]
00402C59  . 6A 08           PUSH 8
00402C5B  . 33F6            XOR ESI,ESI
00402C5D  . 5B              POP EBX
00402C5E  . 89B5 7CFFFFFF   MOV DWORD PTR SS:[EBP-84],ESI
00402C64  . 8D95 7CFFFFFF   LEA EDX,DWORD PTR SS:[EBP-84]
00402C6A  . 8D4D AC         LEA ECX,DWORD PTR SS:[EBP-54]
00402C6D  . 8975 DC         MOV DWORD PTR SS:[EBP-24],ESI
00402C70  . 8975 CC         MOV DWORD PTR SS:[EBP-34],ESI
00402C73  . 8975 BC         MOV DWORD PTR SS:[EBP-44],ESI
00402C76  . 8975 AC         MOV DWORD PTR SS:[EBP-54],ESI
00402C79  . 8975 9C         MOV DWORD PTR SS:[EBP-64],ESI
00402C7C  . 8975 8C         MOV DWORD PTR SS:[EBP-74],ESI
00402C7F  . 89B5 5CFFFFFF   MOV DWORD PTR SS:[EBP-A4],ESI
```

Figure 4: Another way to modify the dissembled code of lab3

Dr. Chen find another way to "hack" this program. He changes the instruction on 0x402C17 from "PUSH EBP" to "RETN 4" (shown in Figure.4). And he successfully remove the Nag screen.

Please answer the following questions:

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

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

## Hint

Please check the lecture slides and video – Class 7 Stack and Stack Frame. Check Class 6 for IA32 CPU register and X86 ASM basics.

## Submission

- The lab due date is available on our course website. Late submission will not be accepted;
- The assignment should be submitted to D2L directly.

- Your submission should include: A **detailed project report in PDF format** to describe what you have done, including screenshots of the final result
- **No copy or cheating is tolerated**. If your work is based on others', please give clear attribution. Otherwise, you **WILL FAIL** this course.