

《软件安全》实验报告

姓名：禹相祐

学号: 2312900

班级： 计科

实验名称:

OLLYDBG 软件破解

实验要求:

1. 请在 XP VC6 生成课本第三章软件破解的案例 (DEBUG 模式, 示例 3-1) 。进而, 使用 OllyDBG 进行单步调试, 获取 verifyPWD 函数对应 flag==0 的汇编代码, 并对这些汇编代码进行解释。
2. 对生成的 DEBUG 程序进行破解, 复现课本上提供的两种破解方法。

实验过程:

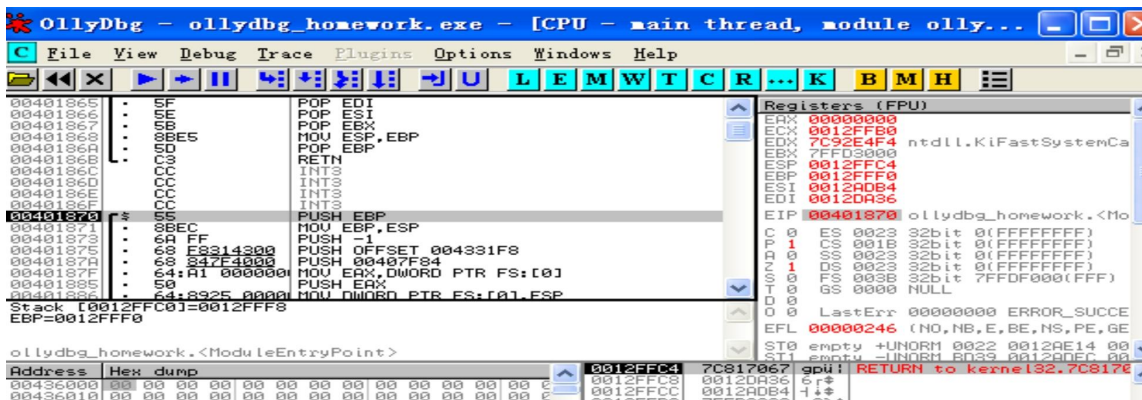
一. 011yDBG 的准备工作:

1. 先在 VC6 下生成实验所用的 exe 文件, 如图:



可以从此处看出：当输入的密码错误时，会再一次执行输入密码，并弹出“wrong password, please input again”并且再一次输入密码。直到输入正确密码“12345678”时，才会弹出“passed”。

2. 然后将此程序生成的 oolldbg_homework.exe 拖入 oolldbg 进行测试, 如图:



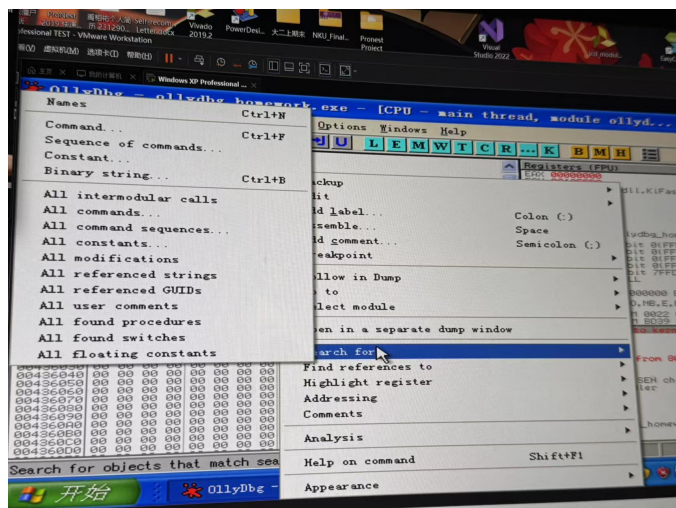
至此，已经完成所有准备工作，现在开始采用两种不同方式进行破解。

二. 两种破解方式:

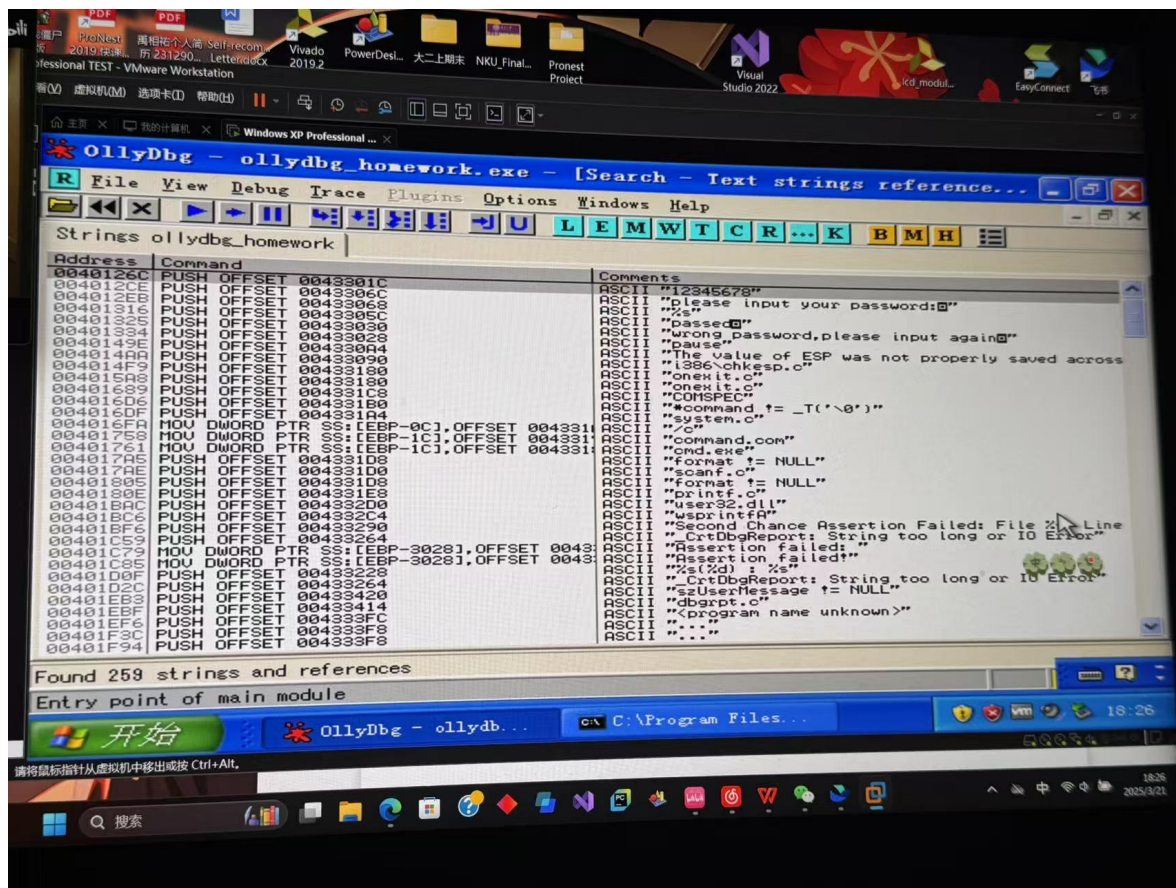
当我们想要破解时。我们首先要查看一下 verifyPwd 函数的汇编代码:

首先我们知道，在判断密码是否正确的时候，会有输出“passed”或“wrong password”的提示，那么我们可以根据输出的“wrong password”大致定位程序:

I) 先右键选择“search for”再选择“all referenced strings”:



II) 我们可以看到“wrong password”(若看不到,可以 ctrl+f 直接查询“wrong”)



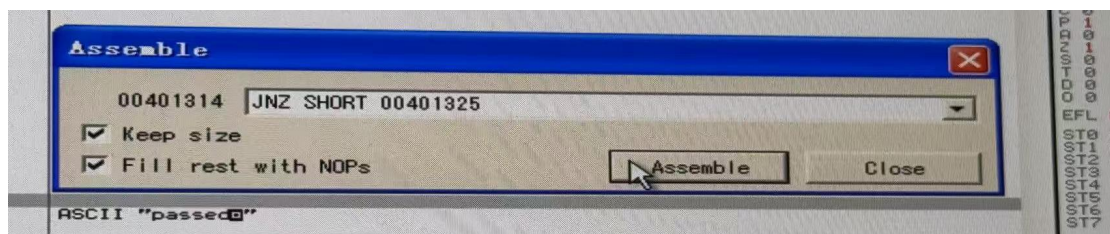
III) 然后我们双击“wrong password”那一行就可以看到：



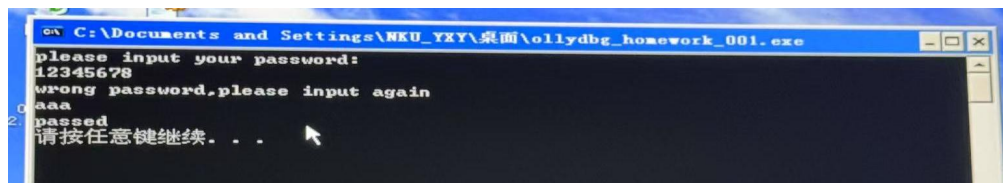
此处 OllyDbg 替我们标识了从何处跳转到了输出“wrong password”，也就意味着最上方的“JZ SHORT 00401326”承担了判断条件的作用。

因此，我们有了第一种破解方法：就是让判断条件取反，即输入错误的时候才通过，输入正确反而显示“wrong password”：

这个实现也很简单，就是将此处的“JZ SHORT 00401326”改为“JNZ SHORT 00401326”，这个 N 指的是“not”，这样就轻松实现了条件的取反：



然后将文件保存为“ollydbg_homework_001.exe”，再试一次：

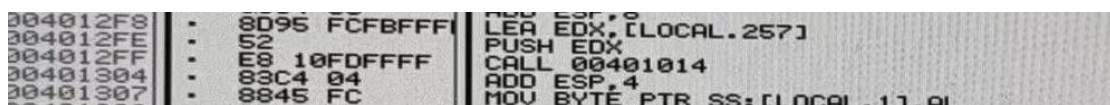


此处就能看出，输入正确答案“12345678”输出了“wrong password”，输入错误答案“aaa”反而输出了“passed”，第一种方法实现。

接下来是第二种方法：

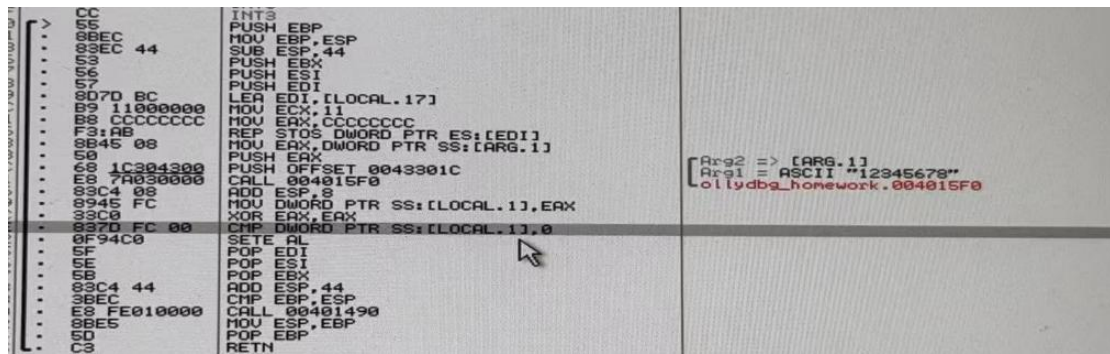
既然存在 verify_pwd 函数，那我们也可以试着让这个函数始终返回 true，这样无论输入什么都会显示“passed”。

I) 那我们先实现对于 verify_pwd 函数的定位：



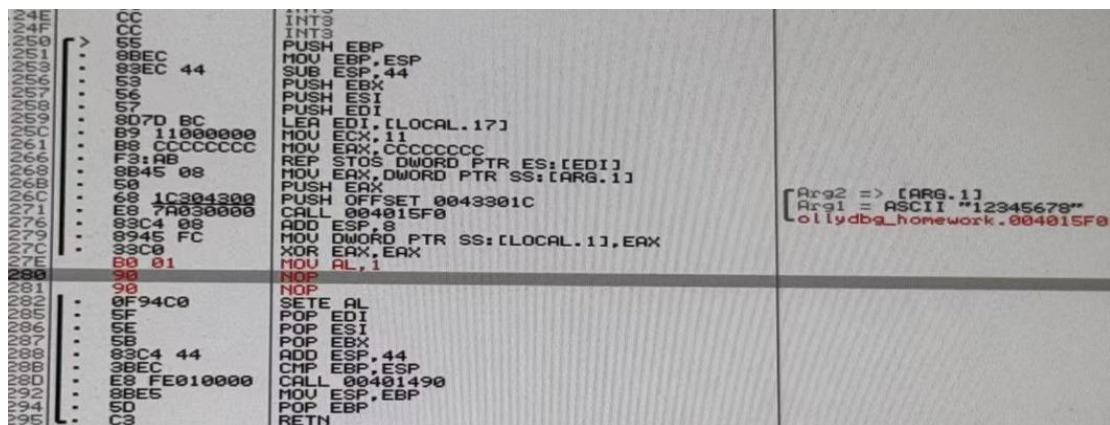
即此处的“call 00401014”

II) 然后右键并连着两次“follow”，就会到如下页面：



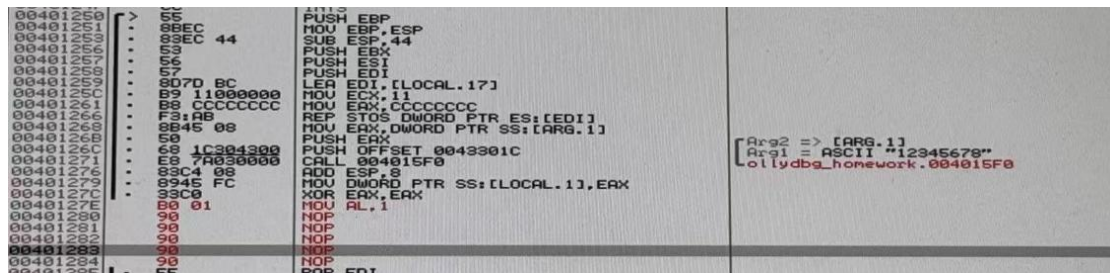
```
CC      INT3
55      PUSH EBP
56      MOV EBP,ESP
57      SUB ESP,44
58      PUSH EBX
59      PUSH ESI
60      PUSH EDI
61      LEA EDI,[LOCAL.17]
62      MOV ECX,11
63      MOV EAX,CCCCCCCC
64      REP STOS DWORD PTR ES:[EDI]
65      MOV EAX,DWORD PTR SS:[ARG.1]
66      PUSH EAX
67      PUSH OFFSET 0043301C
68      CALL 004015F0
69      ADD ESP,8
70      MOV DWORD PTR SS:[LOCAL.1],EAX
71      XOR EAX,EAX
72      CMP DWORD PTR SS:[LOCAL.1],0
73      SETB AL
74      POP EDI
75      POP ESI
76      POP EBX
77      ADD ESP,44
78      CMP EBP,ESP
79      CALL 00401490
80      MOV ESP,EBP
81      POP EBP
82      RETN
```

可以从代码得知，当返回的时候会有个比较产生的 bool 值，可以定位到 SETB AL，接着考虑相关的“CMP DWORD PTR SS:[LOCAL.1].0”。易知此处是产生 bool 值的指令，考虑将其始终设置为 1，修改为“mov al,1”：



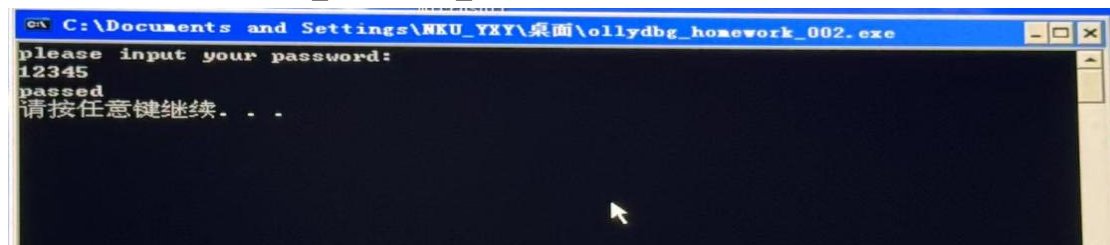
```
CC      INT3
55      PUSH EBP
56      MOV EBP,ESP
57      SUB ESP,44
58      PUSH EBX
59      PUSH ESI
60      PUSH EDI
61      LEA EDI,[LOCAL.17]
62      MOV ECX,11
63      MOV EAX,CCCCCCCC
64      REP STOS DWORD PTR ES:[EDI]
65      MOV EAX,DWORD PTR SS:[ARG.1]
66      PUSH EAX
67      PUSH OFFSET 0043301C
68      CALL 004015F0
69      ADD ESP,8
70      MOV DWORD PTR SS:[LOCAL.1],EAX
71      XOR EAX,EAX
72      MOV AL,1
73      SETB AL
74      POP EDI
75      POP ESI
76      POP EBX
77      ADD ESP,44
78      CMP EBP,ESP
79      CALL 00401490
80      MOV ESP,EBP
81      POP EBP
82      RETN
```

然后接着将“SETB AL”修改为空指令“NOP”：



```
CC      INT3
55      PUSH EBP
56      MOV EBP,ESP
57      SUB ESP,44
58      PUSH EBX
59      PUSH ESI
60      PUSH EDI
61      LEA EDI,[LOCAL.17]
62      MOV ECX,11
63      MOV EAX,CCCCCCCC
64      REP STOS DWORD PTR ES:[EDI]
65      MOV EAX,DWORD PTR SS:[ARG.1]
66      PUSH EAX
67      PUSH OFFSET 0043301C
68      CALL 004015F0
69      ADD ESP,8
70      MOV DWORD PTR SS:[LOCAL.1],EAX
71      XOR EAX,EAX
72      MOV AL,1
73      NOP
74      POP EDI
75      POP ESI
76      POP EBX
77      ADD ESP,44
78      CMP EBP,ESP
79      CALL 00401490
80      MOV ESP,EBP
81      POP EBP
82      RETN
```

然后保存为“ollydbg_homework_002.exe”，执行如下：



至此两种方法均得到实现。

心得体会：

通过这次实验，我学会了使用 ollyDBG 进行基本调试的方法，也掌握了几个基本操作，比如：通过“follow”操作实现程序的跟随，通过“searching string”查找对应字符串，实现定位，也学会了利用基本的汇编语句的更改实现简单程序的破解。

通过这次实验，我也更进一步熟悉了汇编语言中的寻找、判定，也对汇编语言的语法有了更深的了解，这次实验很有收获！