软件安全实验报告

姓名：郭子涵　学号：2312145
班级：信息安全、法学双学位班

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案例 (DEBUG 模式，示例 3-1). 进而，使用 OLLYDBG 进行单步调试，获取 verifyPdw 函数对应 flag==0 的汇编代码，并对这些汇编代码进行解释。

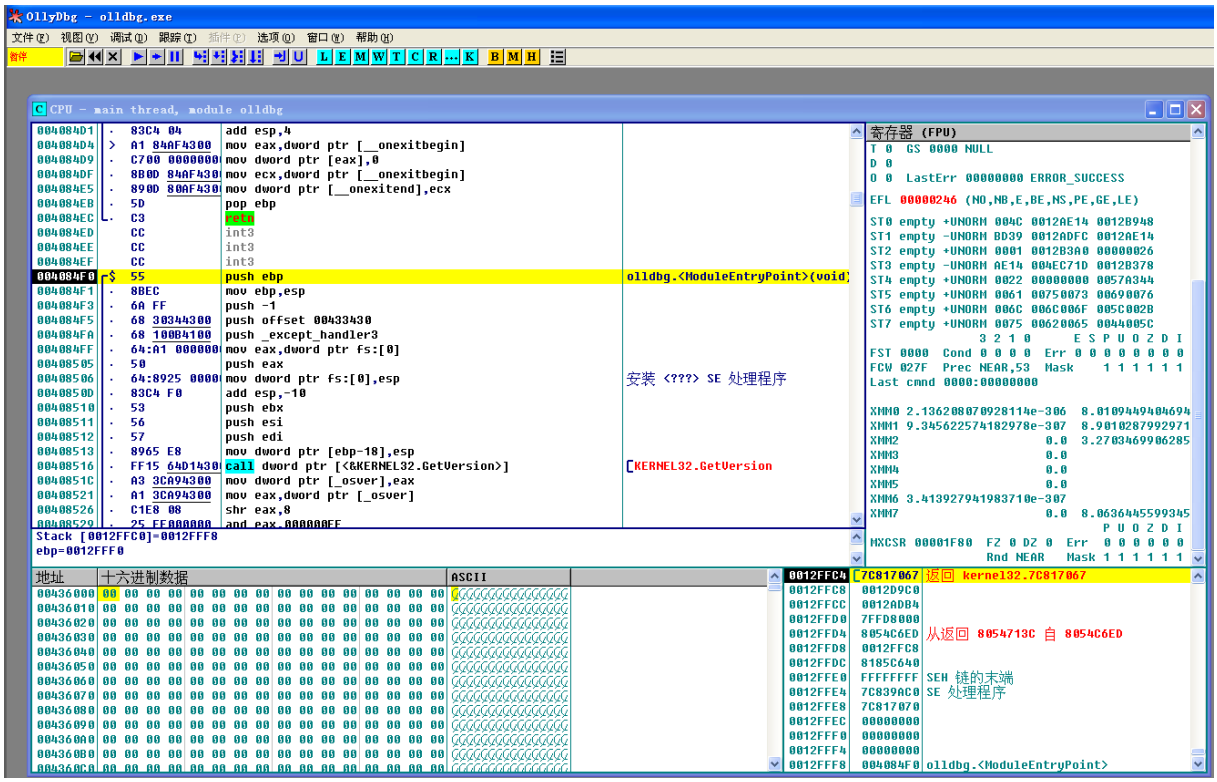
2. 对生成的 DEBUG 程序进行破解，复现课本上提供的两种破解方法。

实验过程

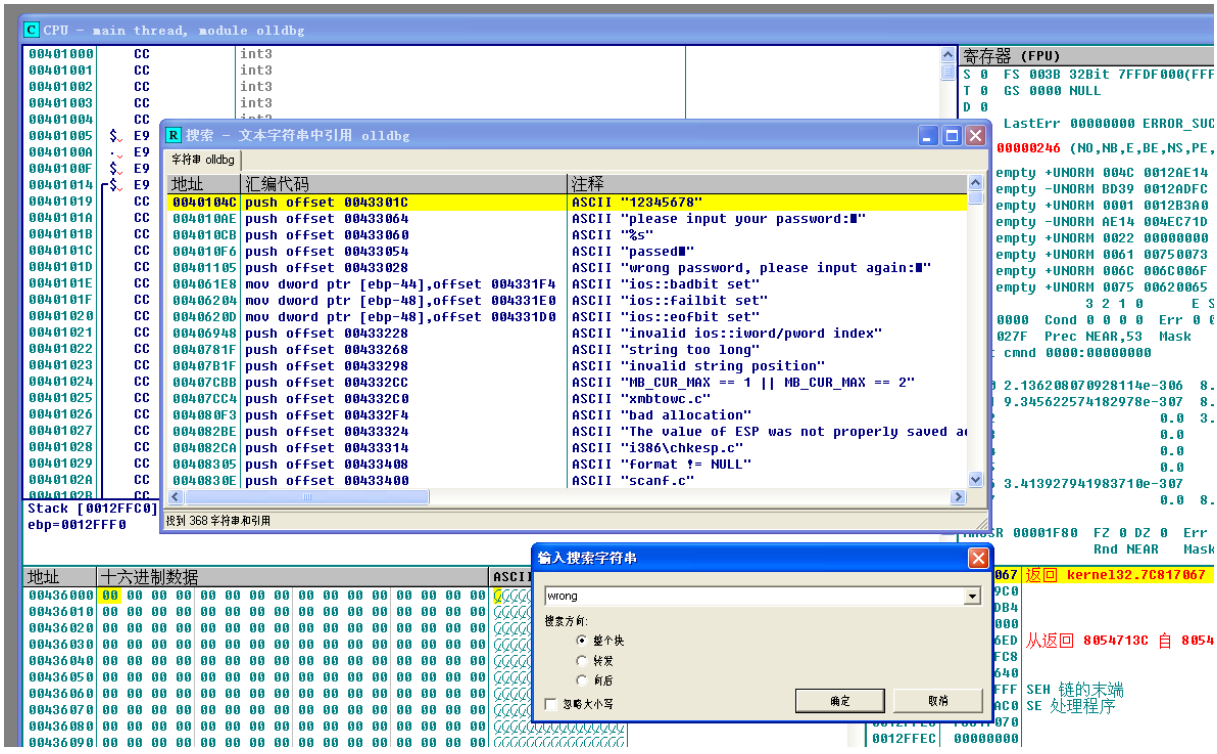
1　使用 OLLYDBG 单步调试，获取 verifyPwd 函数汇编代码

1. 在 VC6 上建立本项目，编译，运行测试，并生成 DEBUG 文件。

2. 使用 OLLYDBG，打开该 DEBUG 文件，进行调试，导入后如下图所示：



3. 通过搜索“wrong”相关字符串定位关键密码判断部分



4. 两次跟随找到 verifyPwd 函数对应的 flag==0 的汇编代码：

```
00401030  > 55          push ebp                    olldbg.verifyPwd(void)
00401031  . 8BEC        mov ebp,esp
00401033  . 83EC 44     sub esp,44
00401036  . 53          push ebx
00401037  . 56          push esi
00401038  . 57          push edi
00401039  . 8D7D BC     lea edi,[ebp-44]
0040103C  . B9 11000000 mov ecx,11
00401041  . B8 CCCCCCCC mov eax,CCCCCCCC
00401046  . F3:AB       rep stos dword ptr [edi]
00401048  . 8B45 08     mov eax,dword ptr [ebp+8]
0040104B  . 50          push eax
0040104C  . 68 1C304300 push offset 0043301C        ASCII "12345678"
00401051  . E8 CA710000 call strcmp                 [strcmp
00401056  . 83C4 08     add esp,8
00401059  . 8945 FC     mov dword ptr [ebp-4],eax
0040105C  . 33C0        xor eax,eax
0040105E    837D FC 00  cmp dword ptr [ebp-4],0
00401062    0F94C0      sete al
00401065  . 5F          pop edi
00401066  . 5E          pop esi
00401067  . 5B          pop ebx
00401068  . 83C4 44     add esp,44
0040106B  . 3BEC        cmp ebp,esp
0040106D  . E8 3E720000 call _chkesp
00401072  . 8BE5        mov esp,ebp
00401074  . 5D          pop ebp
00401075  . C3          retn
```

代码分析如下：

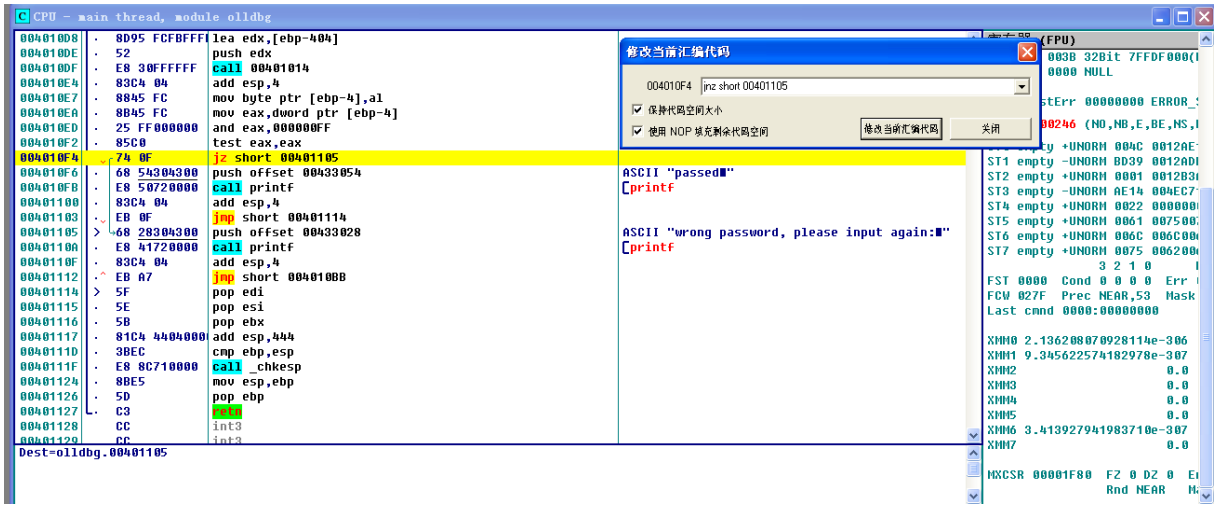
```
01  push ebp  //olldbg.verifyPwd(void)
02  mov ebp,esp //保存旧的ebp,建立新的栈帧
03  sub esp,44 //在栈上分配0x44字节的局部变量空间
04  push ebx
05  push esi
06  push edi  //保存ebx\esi\esi寄存器
07  lea edi,[ebp-44] //将edi指向ebp-44位置
08  mov ecx,11 //ecx赋值0x11
09  mov eax,CCCCCCCC //复制eax
10  rep stos dword ptr [edi] //eax填充的值填充edi指向的内存区域,清空局部变量
11  mov eax,dword ptr [ebp+8] //获取函数参数，及用户输入的字符串地址
12  push eax //用户输入的字符串地址压栈
13  push offset 0043301C //ASCII "12345678"压入地址
14  call strcmp //调用strcmp进行字符串比较，如果用户输入的字符串是“12345678”返回0，否则返回一非零值
15  add esp,8 //清理strcmp调用时压入的参数
16  ov dword ptr [ebp-4],eax //将strcmp返回值存入局部变量flag中
17  xor eax,eax //eax置零
18  cmp dword ptr [ebp-4],0 //将flag值与0进行比较，如果相等的话设置ZF=1，否则ZF=0
19  sete al //如果ZF=1，al=1,否则al=0
20  pop edi
21  pop esi
22  pop ebx //恢复edi、esi、ebx
23  add esp,44 //释放局部变量栈空间
24  cmp ebp,esp
25  call _chkesp //检查栈是否平很
26  mov esp,ebp
27  pop ebp
28  retn //返回
```

2 破解 DEBUG 程序

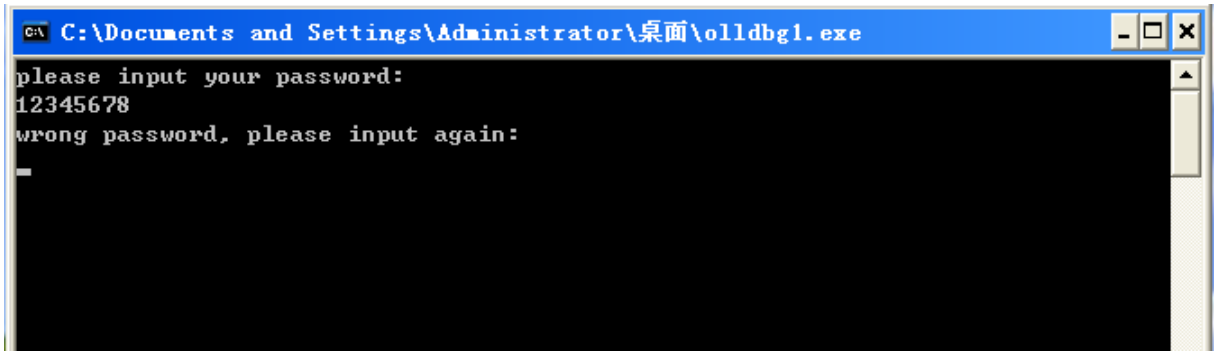
2.1 修改主程序中的条件跳转指令逻辑

我们通过搜索关键词 wrong, 找到了在 verifyPwd 函数部分中判断密码是否正确的关键语句：jz short 00401105, 程序运行至此会跳转到输出错误。

因此将判断语句修改为：jnz short 00401105, 修改代码的逻辑，输入正确密码会输出错误，反之，输入错误密码会挑战正确。下图分别展示修改过程及修改后的运行结果：



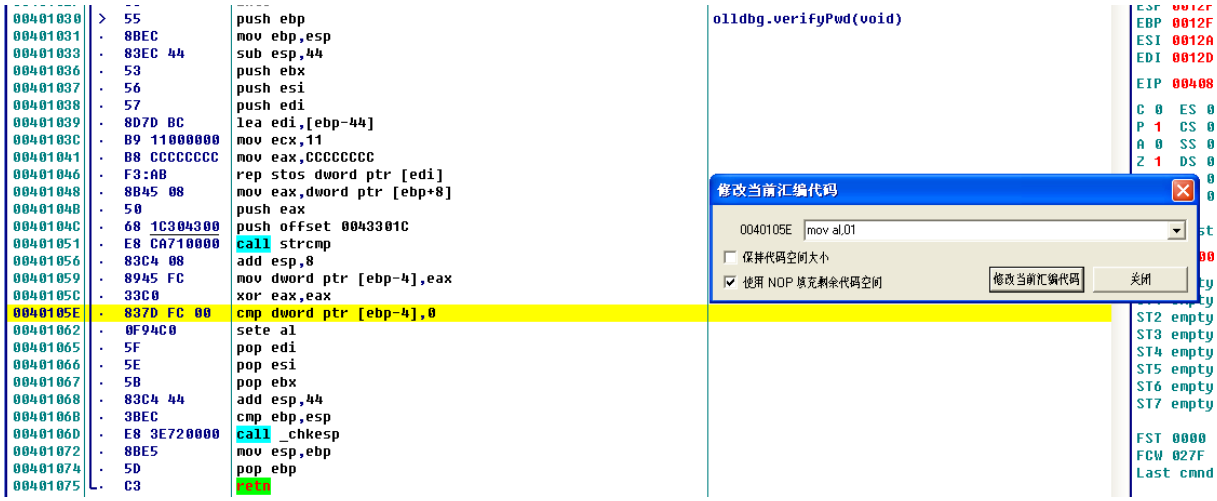
双击该语句进行修改右键选择“编辑”，选择“复制当前修改到所有可执行文件”，保存为 ollydbg1.exe。运行该可执行文件，由结果可看出输入正确密码 12345678，会输出 wrong, 输入错误的答案，窗口一闪而过，转达 passed 界面，证明我们破解成功。



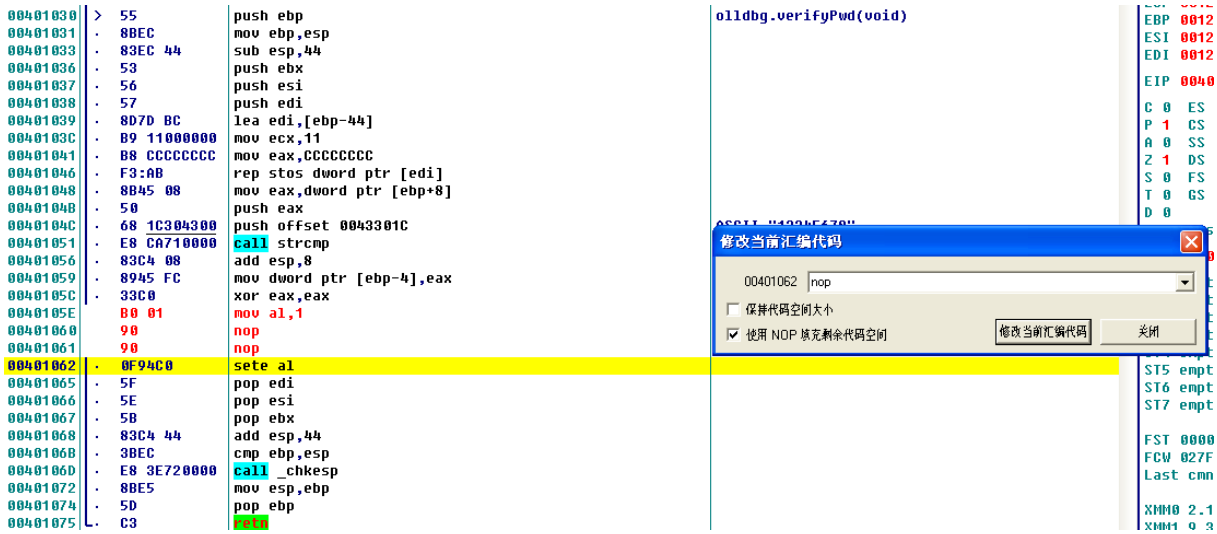
2.2 修改 flag==0 的返回值指令，返回值恒 1

根据前述代码分析，令 flag==1，可以实现无论输入什么密码，都会成功。

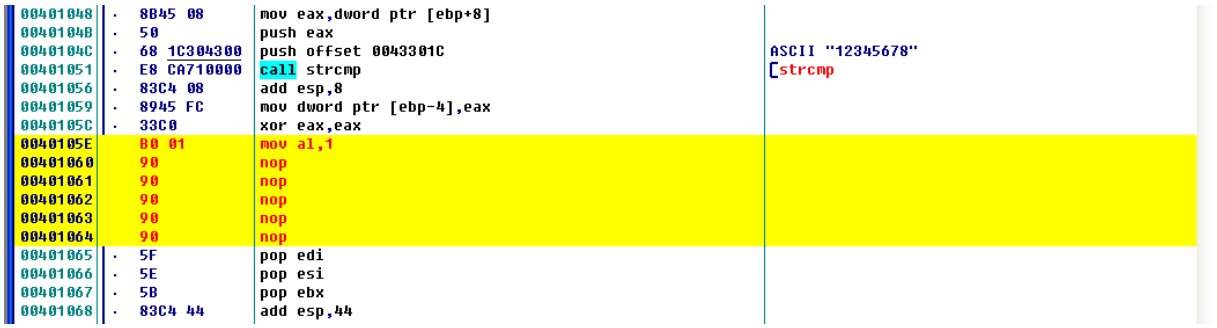
1. 将 cmp dword ptr [ebp-4],0 修改为直接对 al 寄存器赋值为 1



2. 修改 sete al。上述将 al 寄存器恒为 1，此代码不需要，用 nop 指令填充即可



代码修改结果如下，此时我们同样右键选择编辑，复制当前修改到可执行文件，然后再保存运行，发现无论输入什么样的密码，窗口都会一闪而过，证明我们破解成功！



心得体会

本次软件破解实验通过 OllyDbg 的实战操作，学会了如何使用 ollydbg 进行动态调试等简单操作，以及学会了逆向分析的基础方法论：在栈空间追踪时，通过 ESP 寄存

器回溯定位到关键函数调用链；利用字符串参考，快速锚定关键代码段和程序逻辑入口，结合反汇编窗口的条件断点设置，精准捕捉到验证分支的 jz/jnz 指令。通过修改核心跳转指令（将 je 改为 jne）或者修改关键指令逻辑等方法实现程序流程劫持。

本次实验这与上学期在《汇编语言与逆向技术》课程中所学过的实验非常相似，让我更加深入理解了破解程序中如何利用搜索字符串，cmp 后条件跳转指令来精准定位等破解思路，收获良多。