

《软件安全》实验报告

姓名：李雅帆

学号：2213041

班级：信安班

一、实验名称：

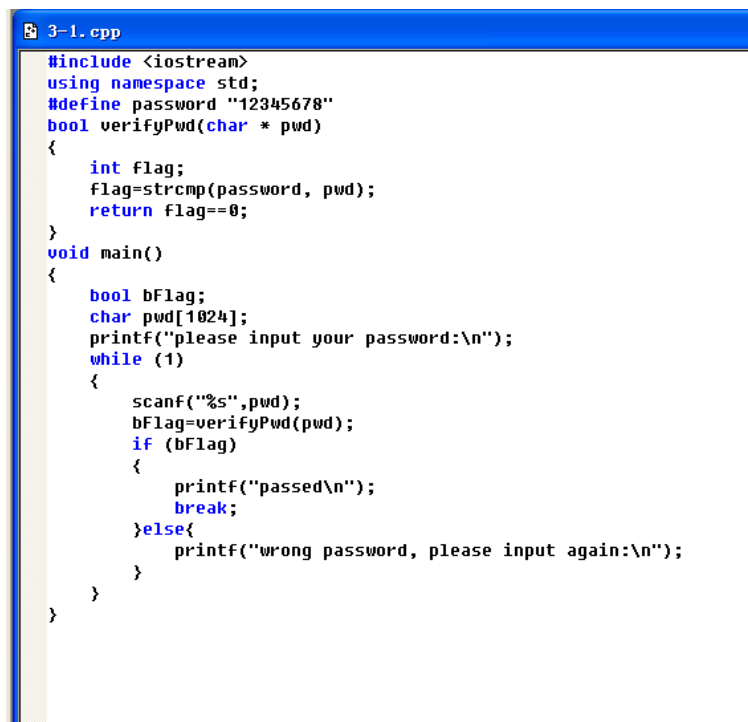
OLLYDBG 软件破解

二、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 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。

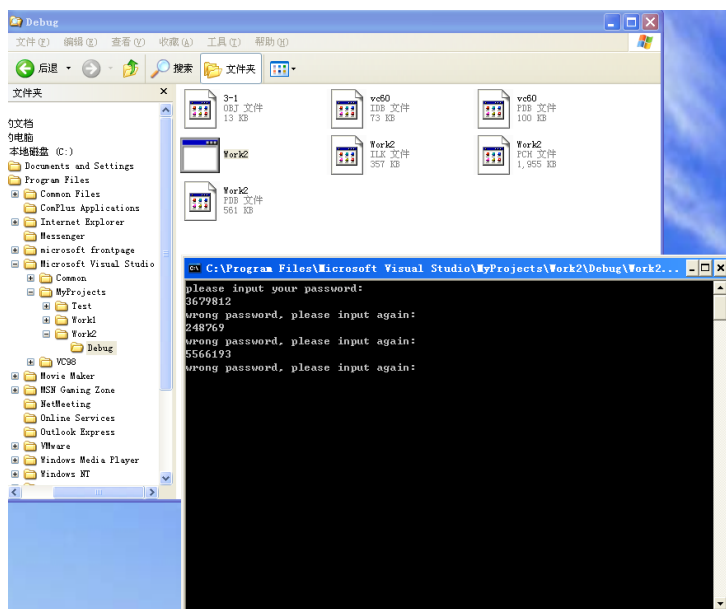
三、实验过程：

1. 在 XP VC6 中生成课本第三章软件破解案例（DEBUG 模式，示例 3-1）。



```
3-1.cpp
#include <iostream>
using namespace std;
#define password "12345678"
bool verifyPwd(char * pwd)
{
    int flag;
    flag=strcmp(password, pwd);
    return flag==0;
}
void main()
{
    bool bFlag;
    char pwd[1024];
    printf("please input your password:\n");
    while (1)
    {
        scanf("%s",pwd);
        bFlag=verifyPwd(pwd);
        if (bFlag)
        {
            printf("passed\n");
            break;
        }else{
            printf("wrong password, please input again:\n");
        }
    }
}
```

2. 用 Debug 模式生成可执行文件。打开可执行文件进行测试，输入正确密码“12345678”程序进入核心逻辑，输入错误的代码，则会显示“wrong password,please input again :”。



3. 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。

(1) 方法一

①定位到原代码：

这是实现判断跳转的关键逻辑，如果 eax 中的值为 0，test 过后会将 ZF 置为 1，触发 je 跳转大密码错误的提示输出处；如果 eax 中的值为 1，test 过后会将 ZF 置 0，不触发 je 跳转，进入到核心逻辑。

004010F2	85C0	test eax,eax	
004010F4	74 0F	jz short 00401105	
004010F6	68 54304300	push offset 00433054	ASCII "passed"
004010F8	E8 50720000	call printf	[printf]
00401100	83C4 04	add esp,4	
00401103	EB 0F	jnp short 00401114	
00401105	68 28304300	push offset 00433028	ASCII "wrong password, please input again:"

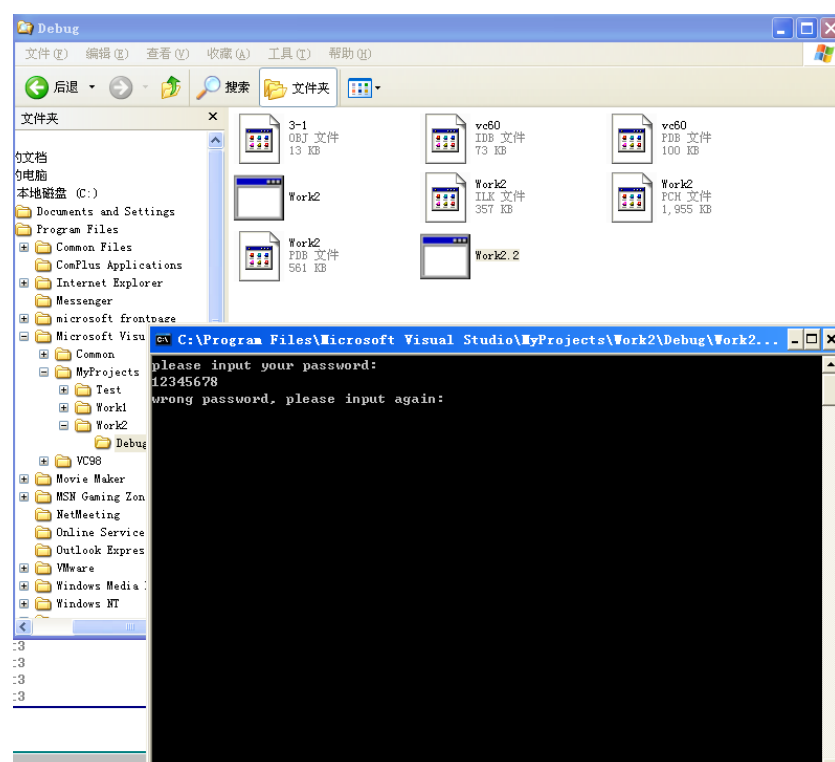
②破解：

按照实验指导中将逻辑倒转，即密码错误事件将会正常跳转到验证正确的部分，进入核心逻辑；而密码正确时则会显示“wrong password,please input again :”。

004010F2	85C0	test eax,eax	
004010F4	75 0F	jnz short 00401105	
004010F6	68 54304300	push offset 00433054	ASCII "passed"
004010F8	E8 50720000	call printf	[printf]
00401100	83C4 04	add esp,4	
00401103	EB 0F	jnp short 00401114	
00401105	68 28304300	push offset 00433028	ASCII "wrong password, please input again:"

③验证破解是否成功：

保存新的可执行文件并运行。发现输入错误的密码会进入核心逻辑；而输入正确密码正确时则会显示“wrong password,please input again :”，破解成功。



(2) 方法二

第二种思路是取修改 verify—password 函数，想办法使其永远返回的都是真值，即 eax 中的值不为零。

① 定位到原代码：

call 指令调用了 strcmp 函数，比较了输入的字符串和给定的密码是否相等，将返回到 eax 的结果保存到局部变量中，如果相同返回的就是 0，不相同返回的是 1，因此我们希望能够在整个函数返回的时候，保证 eax 中的值为 1。

00401051	·	E8 CA710000	call 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

②破解:

将 cmp 指令修改为 mov 指令, 强行将 eax 中的值设置为 1, 但是由于指令的空间大小是固定的, 同时考虑到 sete 指令会根据 ZF 的状态设置 eax 的低 8 位, 为了能够消除 sete 的意外影响, 需要将其 nop 掉。

00401051	. E8 CA710000	call strcmp
00401056	. 83C4 08	add esp,8
00401059	. 8945 FC	mov dword ptr [ebp-4],eax
0040105C	. 33C0	xor eax,eax
0040105E	. B0 01	mov al,1
00401060	. 90	nop
00401061	. 90	nop
00401062	. 90	nop
00401063	. 90	nop
00401064	. 90	nop

③验证破解是否成功: 保存新的可执行文件并运行。发现无论输入的密码正确还是错误, 都会跳转到验证正确的部分, 进入核心逻辑, 破解成功。

四、心得体会:

1. 认识并熟悉了 OlllyDBG 的调试方法, 能够对简单的可执行文件进行动态逆向分析, 通过合理利用调试工具, 修改指令实现软件的破解。
2. 我对汇编代码指令的底层的逻辑有了更深的了解, 能够通过修改指令的逻辑和内容达到软件破解的目的。