《软件安全》实验报告

姓名：梁婧涵 学号：2112155 班级：1120

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。

#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;

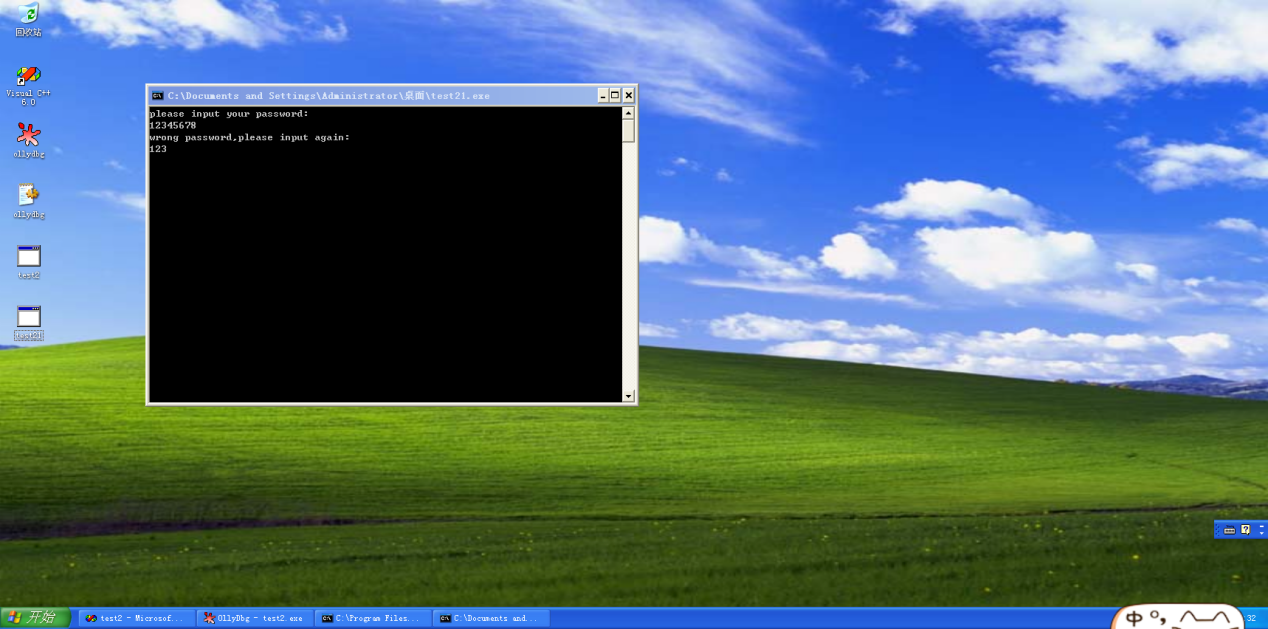
}else{

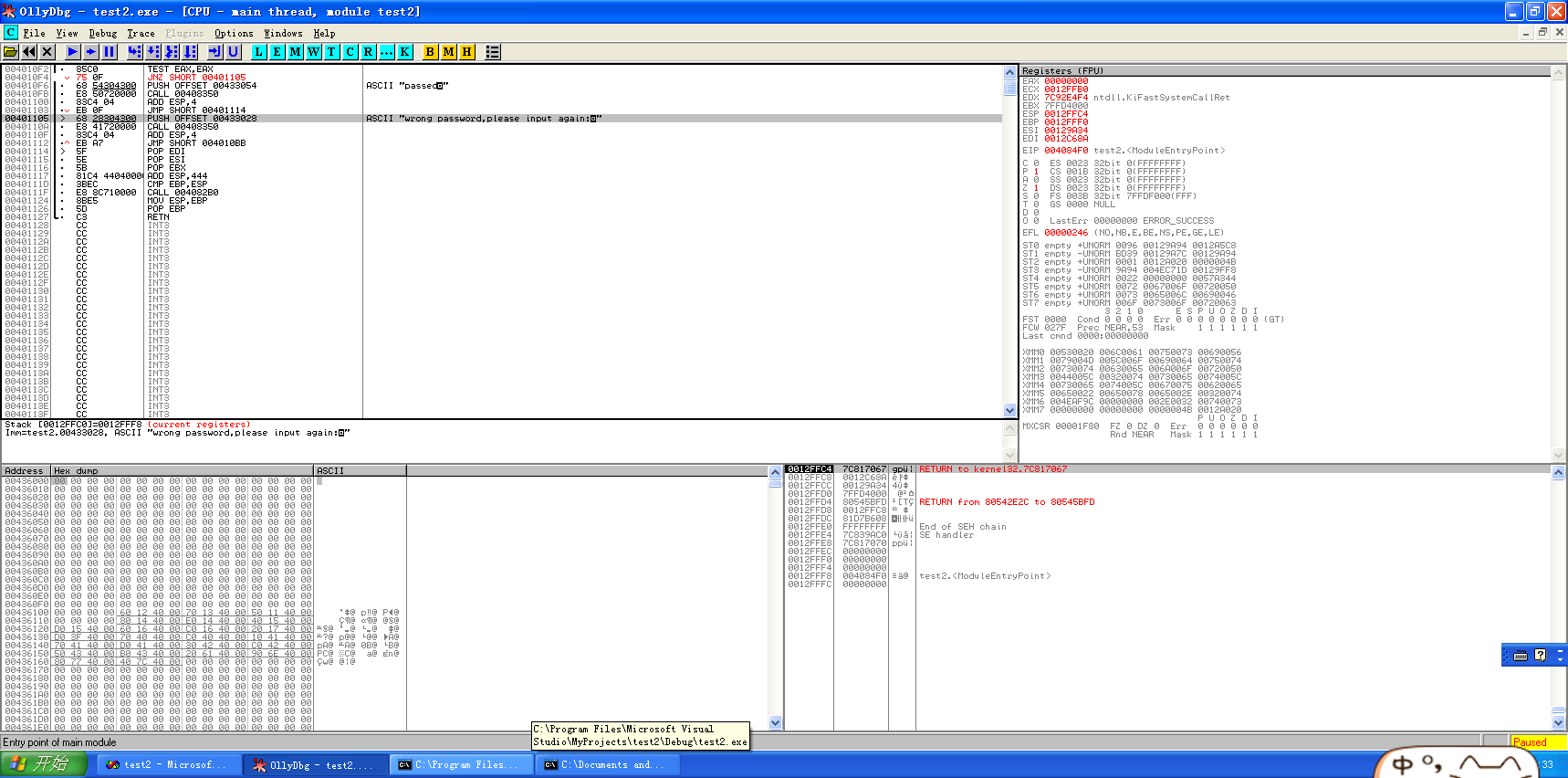
printf("wrong password,please input again:\n");

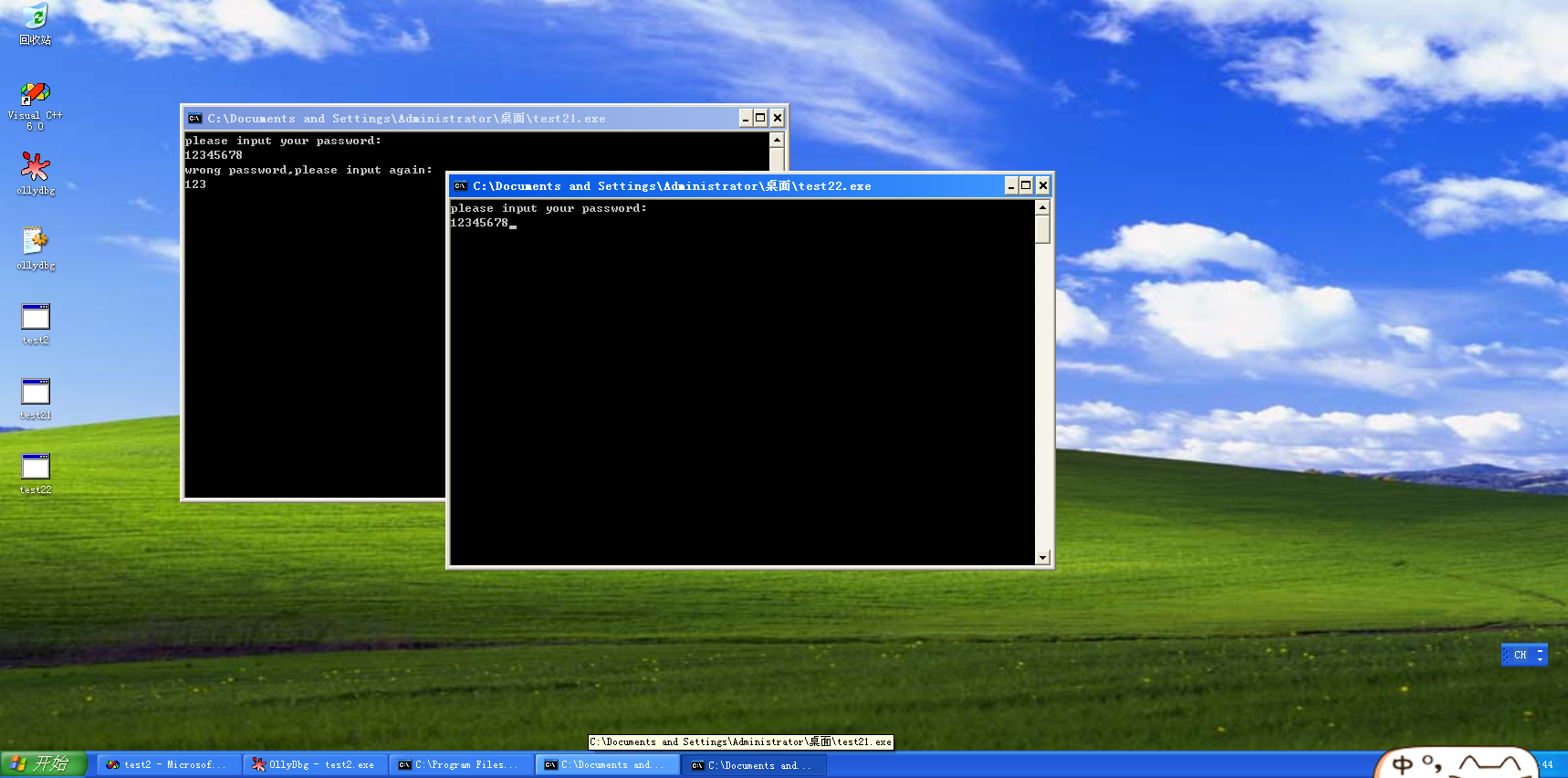
}}}

实验过程：

1. 复制源代码到VC6中，在DEBUG模式下运行，找到文件夹中可执行文件，输入口令111111，输出wrong password，please input again:,一直输入，直到输入对了才会进入核心逻辑，想法：破解这个代码，不输入口令或者输入错口令或输入任意口令都能通过完成破解目的。
2. 方法一：输入一个口令，让其发生一个跳转，跳到正确的逻辑，强制控制返回值，使程序总是转入正确的逻辑值。打开OllyDBG，打开可执行文件，找到最重要的信息，错误的会显示错误的信息，即wrong password，查找所有引用的字符串，会体现所有的字符串常量信息，找到wrong那行，双击找到引用的位置。可以看到从74到68，出现wrong，如果jz short 00401325这个条件成立，会跳到wrong，如果破坏其逻辑，逻辑变反。双击jz行修改代码，取反JZ变成JNZ，这个修改只是在内存中进行修改，还要编辑复制所有修改到可执行文件，保存文件。
3. 输入12345678，输出wrong，这是正确指令，任意输入指令，会退出程序，即得到了正确的逻辑。
4. 方法二：再修改回以前的状态JZ。在原来状态之前还调用了call 00401014这个函数，跟随这个函数，转到另一个函数继续跟随，转到push ebp。这个函数retn返回之前一定有一个返回值的出现sete al（bool值返回），上面成功不成功比较了一个状态位，相等设al，不相等al就为0。想实现比较不比较al的值都设为01，强制更改逻辑mov al.01,空间用nop填充，sete al改成nop。这回不管是输入12345678正确指令还是任意输入指令都会得到正确的逻辑。







心得体会：

通过实验，掌握了OllyDBG破解程序的两种方法，修改汇编代码或修改函数使其得到正确的结果。

OllyDbg是一款非常强大的反汇编和调试工具，对于软件逆向分析和调试非常有用。以下是我使用OllyDbg的心得体会：OllyDbg主要是以汇编语言的形式显示和调试程序，因此熟悉汇编语言和调试技巧对于使用OllyDbg非常重要。在使用OllyDbg进行调试时，需要对程序的结构和功能有一定的了解，包括函数的用途、变量的作用等等。这将有助于我们更好地理解程序的行为和调试过程。

OllyDbg提供了许多有用的调试命令和功能，包括单步执行、断点调试、寄存器查看和修改等等。熟练掌握这些命令和功能将有助于我们更好地进行调试和分析。