《漏洞利用及渗透测试基础》实验报告

姓名：许健 学号：2013018 班级：信安班

**实验名称：**

API函数自搜索实验

**实验要求：**

复现第五章实验七，基于示例5-11，完成API函数自搜索的实验，将生成的exe程序，复制到windows 10操作系统里验证是否成功。

**实验过程：**

1. **通用型API编写逻辑**

通用型shellcode编写的步骤：

（1）第一步：定位kernel32.dll。

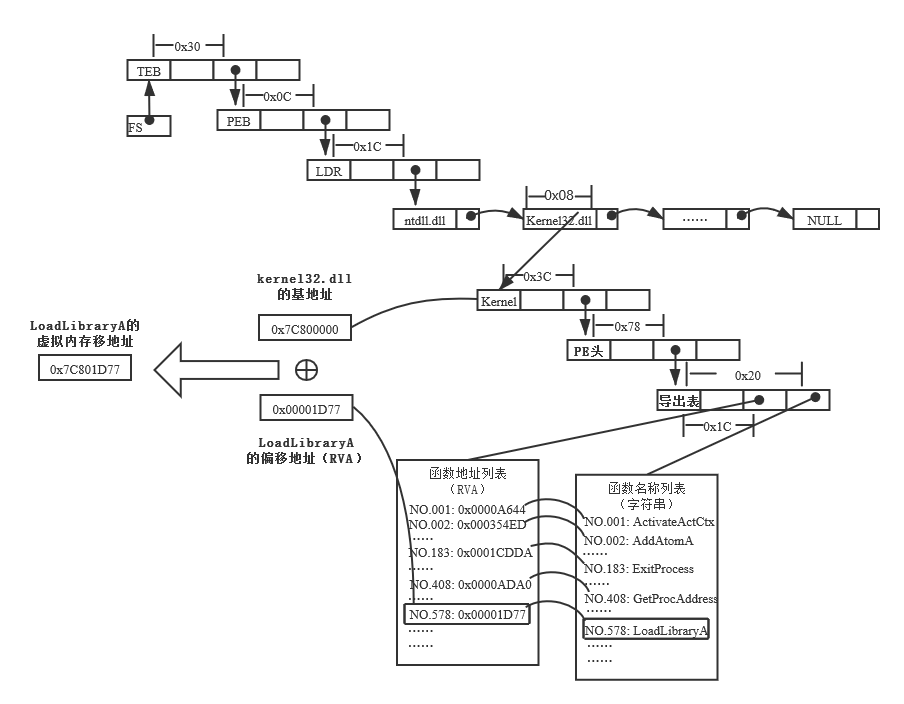
（2）第二步：定位kernel32.dll的导出表。

（3）第三步：搜索定位LoadLibrary等目标函数。

（4）第四步：基于找到的函数地址，完成Shellcode的编写。

难点在于第一步到第三步，即如何实现API函数自搜索。

大致流程如图所示：



1. **定位kernel32.dll**

|  |
| --- |
| int main()  {  \_asm  {  mov eax, fs:[0x30] ;PEB的地址  mov eax, [eax + 0x0c] ; PEB\_LDR\_DATA结构体的地址  mov esi, [eax + 0x1c] ; 指针InInitializationOrderModuleList  lodsd  mov eax, [eax + 0x08] ;eax就是kernel32.dll的地址  }  return 0;  } |

1. **定位kernel32.dll的导出表**

kernel32.dll，属于PE文件，我们可以根据PE文件的结构特征定位其导出表，进而定位导出函数列表信息，然后进行解析、遍历搜索，找到我们所需要的API函数。

定位导出表及函数名列表的步骤如下：

（1）从kernel32.dll加载基址算起，偏移0x3c的地方就是其PE头的指针。

（2）PE头偏移0x78的地方存放着指向函数导出表的指针。

（3）获得导出函数偏移地址（RVA）列表、导出函数名列表：

①导出表偏移0x1c处的指针指向存储导出函数偏移地址（RVA）的列表。

②导出表偏移0x20处的指针指向存储导出函数函数名的列表。

定位kernel32.dll导出表及其导出函数名列表的代码如下：

|  |
| --- |
| mov ebp, eax //将kernel32.dll基地址赋值给ebp  mov eax,[ebp+0x3C] //dll的PE头的指针（相对地址）  mov ecx,[ebp+eax+0x78] //导出表的指针（相对地址）  add ecx,ebp //ecx=0x78C00000+0x262c 得到导出表的内存地址  mov ebx,[ecx+0x20] //导出函数名列表指针  add ebx,ebp //导出函数名列表指针的基地址 |

1. **搜索定位目标函数**

可以通过遍历两个函数相关列表，算出所需函数的入口地址：

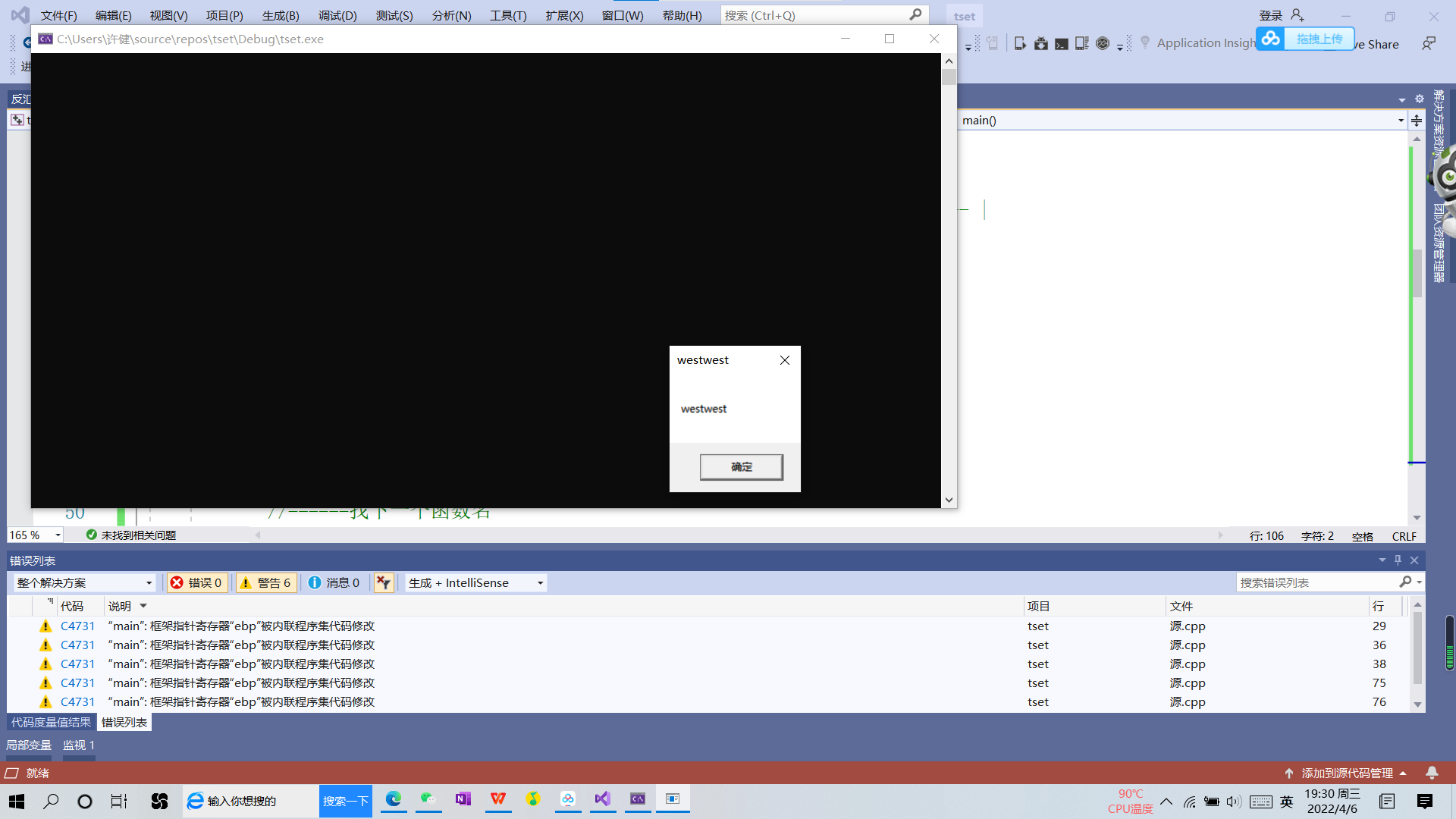
（1）函数的RVA地址和名字按照顺序存放在上述两个列表中，我们可以在名称列表中定位到所需的函数是第几个，然后在地址列表中找到对应的RVA。

（2）获得RVA后，再加上前边已经得到的动态链接库的加载地址，就获得了所需API此刻在内存中的虚拟地址，这个地址就是最终在ShellCode中调用时需要的地址。

可以对所需的API函数名进行hash运算，这样只要比较hash所得的摘要就能判定是不是我们所需的API了；在shellcode中通过压栈的方式将这个hash值压入栈中，再通过比较得到动态链接库中的API地址。

1. **编写shellcode运行**

程序运行在win10上也可以正常执行：



**心得体会：**

通过实验，掌握了通用型Shellcode的编写逻辑；以及定位导出表、定位搜索函数地址的一般步骤，理解PE文件结构。还可以通过hash运算简化字符串的比较操作，增加shellcode的通用性。