逆向分析实验一

[1. 实验内容 1](#_Toc31684)

[2. 实验准备 1](#_Toc31018)

[3. 实验过程 1](#_Toc11162)

[3.1. 代码编写 1](#_Toc18643)

[3.2. 逆向实例程序 1](#_Toc30268)

[3.3. “Flag”分析和跟踪 1](#_Toc14002)

[4. 扩展 2](#_Toc21360)

[5. 实验成绩评定 2](#_Toc31302)

# 实验内容

1. 编程（Java、C/C++或Python）分析PE文件（experient-1.exe）的结构，提取文件相关信息，描述文件的主要参数和构成；
2. 结合程序实例，分析函数“sub\_4A19B0”出现“栈平衡”问题的原因，并根据程序运行流程，借助IDA工具调整使其达到栈平衡；
3. 在上述工作基础上，尝试分析实例代码中的“Flag”。

# 实验准备

1. 深入理解“第2讲 PE和ELF文件格式”中的内容。
2. 下载IDA Pro7.0软件。

# 实验过程

## 代码编写

参考以下文章中的代码：

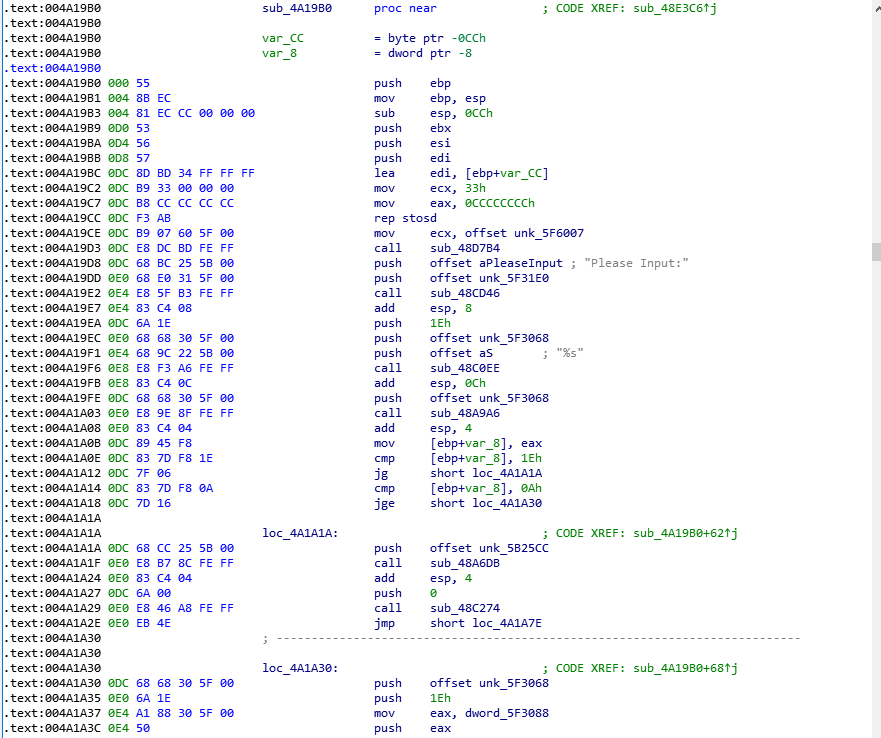
1. https://blog.csdn.net/as14569852/article/details/78120335
2. https://blog.csdn.net/KoalaZB/article/details/53590404

调试程序，以给定的实例文件作为输入，输出相关信息。

## 逆向实例程序

1. 借助于IDA逆向实例程序，比较该工具显示的信息与自己代码提取的信息是否一致；
2. 在反编译程序代码中定位函数sub\_4A19B0；

定位函数如下图



**尝试将该函数还原为C代码，提示该函数return的时候栈指针并未归0**

**因此无法逆向为C语言代码**

1. 根据代码中“push”和“pop”指令以及“call”指令的特性，分析产生栈平衡问题的原因。

**分析如下：**

**当我们在该函数中通过call指令调用其他函数时，首先需要将调用函数的参数压栈，即使用push指令，而每当使用push指令之后，esp会-4（32位下-4，16位下-2）.**

**而这些函数根据调用方式的不同，有些需要在ret后还原栈顶，比如add esp,4,具体数字依据调用时的参数个数。而有些则会由调用函数还原。**

**例如：\_stdcall会在被调用函数中还原栈顶，cdecl则需要调用者手动还原。**

**甚至可能由于编写者的失误也会破坏栈平衡，往往平衡的破坏并不影响程序的执行，但是对逆向分析是致命的**

**因此，当我们进行逆向分析时，为了弥补这些因为入栈后未还原而被破坏的栈平衡，需要手动的添加上还原的代码。但这只是一种理想的情况，实际上我们很难插入一段指令，这样做的话会破坏原本的代码结构。更好的选择是对某些代码进行替换。常用的替换是90（空指令）。**

**基于上述分析，定位到push 参数后未还原的调用如下：**

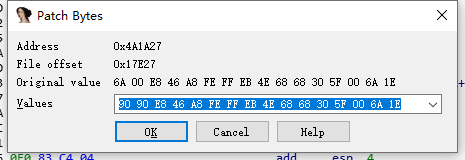




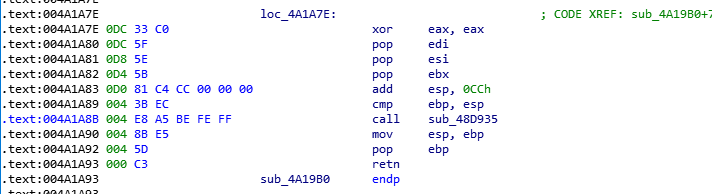
**正因如此，最终ret时栈不平衡，无法逆向。**

1. 借助于指令修改工具，消除栈平衡问题。

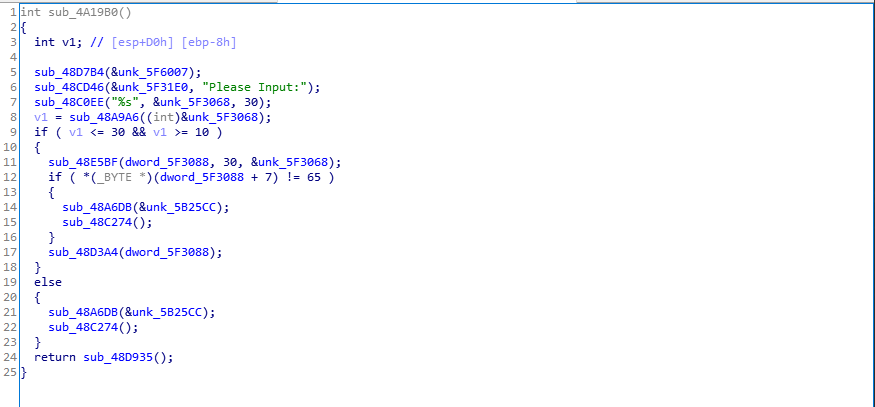
**基于在（3）中的分析，将两个Push指令修改为90**



**修改完毕后，注意到在ret时栈指针为0，完成了栈平衡。**



**最终该函数可以逆向**



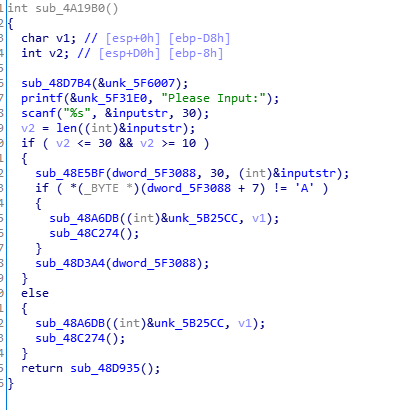
## “Flag”分析和跟踪

1. 在上述工作的将函数转换为伪C代码；

发现%s为参数，根据”please Input”字符串，猜测sub\_48C0EE是scanf函数。

由此可进一步推测v2为len(inputstr)

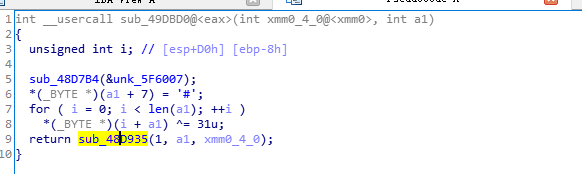
于是将伪代码标注如下



1. 以变量“dword\_5F3088”为线索，跟踪“Flag”相关的信息；

注意到inputstr被拷贝到dword\_5F3088中

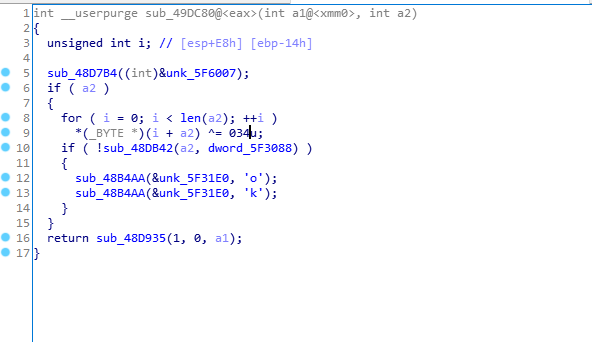
随后判断第8位是否为A，并进入sub\_48D3A4.

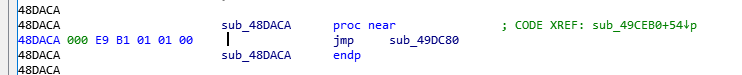


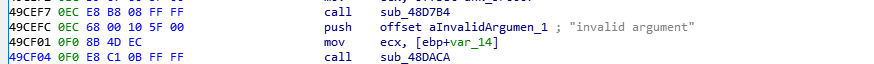
在该函数中，第八位被修改为A，并按位异或31.

随后追溯dword\_5F3088的交叉引用，定位到dword\_49DC80

至此，发现字符串ok，无限接近答案，我们发现在这里对dword\_5F3088和a2做了一次操作，疑似判断是否相等，于是我们追溯a2的来源







我们最终发现a2的值为”invalid argument”

1. 尝试恢复“Flag”。
2. 得到计算公式**a2^28=inputstr^31**
3. **使用python脚本运算得到**
4. **Inputstr=jmubojg#bqdvnfmw**
5. **最后一步，将此前修改的第8位修改为A,得到正确结果**

# 扩展

IDA Pro是一个非常重要的逆向分析工具，建议同学们多花些时间熟悉该工具，此外结合本实验中的实例，尝试一下动态调试方法。

# 实验成绩评定

实验成绩的评定主要依据包括两个方面：

1. 文档的规范性和细节的描述。规范性体现的是态度；细节则是分析工作的深入程度。
2. 分析的结果。结果包括是否达到了实验得一些基本要求（分析的方法和逆向分析工具的使用），另外就是分析的深度（例如栈平衡问题的原因等）。