实验4

班级：191211

姓名：孙鹤轩

学号：20211000156

实验内容：

1 二进制炸弹实验的内容、方法和基本步骤；

2 程序机器级表示、分析与调试基本知识和方法的应用。

实验目标：

1 加深对程序的机器级表示、汇编与反汇编、二进制程序分析与调试、逆向工程等方面知识

的理解和掌握；

2 从程序员角度认识计算机系统，分析高级语言对应的机器行为及其对程序执行结果和性能

的影响，解决计算机系统设计、程序开发过程中的关键问题。

3 掌握各种开源的编译调试工具。

实验任务：

1 学习 MOOC 内容

https://www.icourse163.org/learn/NJU-1449521162

第五周 二进制程序逆向工程

第 1 讲 二进制炸弹实验：概述

第 2 讲 二进制炸弹实验：字符串比较

第 3 讲 二进制炸弹实验：浮点数表示

第 4 讲 二进制炸弹实验：课后实验

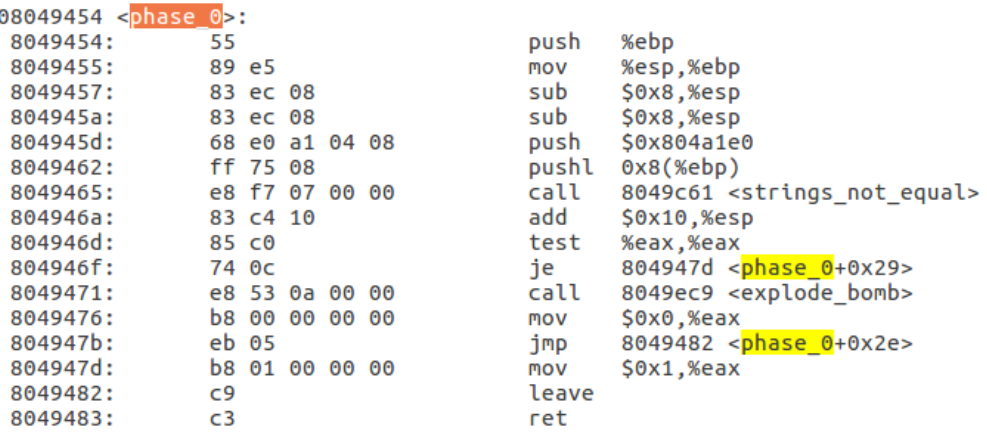
2 完成实验

详见二进制逆向工程实验文档

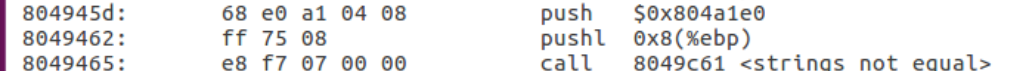
2.1 字符串 phase0

2.1.1实验结果与分析

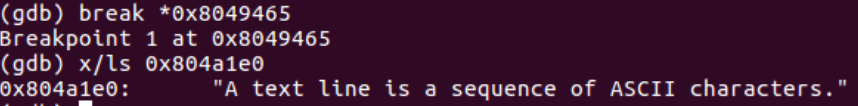
首先通过反汇编找到phase\_0函数对应的反汇编结果，如下：



我们首先来看如何会调用bomb函数，在8049471处call指令调用了explode\_bob函数，这之前是一个je跳转指令，当eax寄存器的值为0时跳转到804947d处，程序正确退出，所以我们需要让eax的寄存器的值为0，eax的值由test指令决定，test指令是与自身的值进行and操作，所以eax的值应为0，此时eax的值根据strings\_not\_equal函数的返回值决定，所以strings\_not\_equal函数的返回值应该为0，即他的两个输入参数应该是相同的；其输入参数汇编代码位置如下：

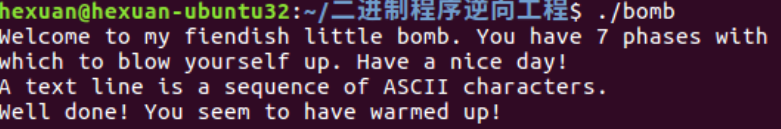


其中0x804a1e0为第二个参数应是内部定义的常量，0x8(%ebp)为第一个参数应是我们输入的字符串，所以我们只需要找出0x804a1e0地址处存放的字符串是多少即可，在call指令前设置断点，通过gdb调试结果如下：



此时我们直到拆弹的密码应为字符串：A text line is a sequence of ASCII characters.

运行试验结果如下：



拆弹成功

2.2 浮点数表示 phase1

2.2.1实验结果及分析

首先通过反汇编找到phase\_1函数对应的反汇编结果，如下：

08049484 <phase\_1>:

8049484: 55 push %ebp

8049485: 89 e5 mov %esp,%ebp

8049487: 83 ec 28 sub $0x28,%esp

804948a: c7 45 f4 a1 84 76 09 movl $0x97684a1,-0xc(%ebp)

8049491: db 45 f4 fildl -0xc(%ebp)

8049494: dd 5d e8 fstpl -0x18(%ebp)

8049497: 8d 45 e0 lea -0x20(%ebp),%eax

804949a: 50 push %eax

804949b: 8d 45 e4 lea -0x1c(%ebp),%eax

804949e: 50 push %eax

804949f: 68 0f a2 04 08 push $0x804a20f

80494a4: ff 75 08 pushl 0x8(%ebp)

80494a7: e8 24 fc ff ff call 80490d0 <\_\_isoc99\_sscanf@plt>

80494ac: 83 c4 10 add $0x10,%esp

80494af: 83 f8 02 cmp $0x2,%eax

80494b2: 74 0c je 80494c0 <phase\_1+0x3c>

80494b4: e8 10 0a 00 00 call 8049ec9 <explode\_bomb>

80494b9: b8 00 00 00 00 mov $0x0,%eax

80494be: eb 2c jmp 80494ec <phase\_1+0x68>

80494c0: 8d 45 e8 lea -0x18(%ebp),%eax

80494c3: 8b 10 mov (%eax),%edx

80494c5: 8b 45 e4 mov -0x1c(%ebp),%eax

80494c8: 39 c2 cmp %eax,%edx

80494ca: 75 0f jne 80494db <phase\_1+0x57>

80494cc: 8d 45 e8 lea -0x18(%ebp),%eax

80494cf: 83 c0 04 add $0x4,%eax

80494d2: 8b 10 mov (%eax),%edx

80494d4: 8b 45 e0 mov -0x20(%ebp),%eax

80494d7: 39 c2 cmp %eax,%edx

80494d9: 74 0c je 80494e7 <phase\_1+0x63>

80494db: e8 e9 09 00 00 call 8049ec9 <explode\_bomb>

80494e0: b8 00 00 00 00 mov $0x0,%eax

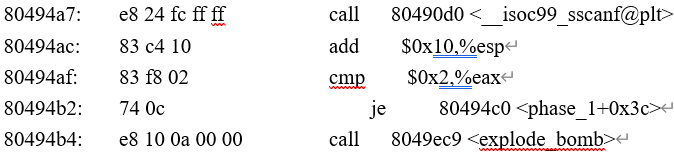
80494e5: eb 05 jmp 80494ec <phase\_1+0x68>

80494e7: b8 01 00 00 00 mov $0x1,%eax

80494ec: c9 leave

80494ed: c3 ret

我们还是先来看如何会调用bomb函数



在这段代码中，几条指令对sscanf的返回值的个数进行了检查，如果不为2则爆炸，所以我们推断我们应该输入的是整数数量为两个

80494c0: 8d 45 e8 lea -0x18(%ebp),%eax

80494c3: 8b 10 mov (%eax),%edx

80494c5: 8b 45 e4 mov -0x1c(%ebp),%eax

80494c8: 39 c2 cmp %eax,%edx

80494ca: 75 0f jne 80494db <phase\_1+0x57>

80494cc: 8d 45 e8 lea -0x18(%ebp),%eax

80494cf: 83 c0 04 add $0x4,%eax

80494d2: 8b 10 mov (%eax),%edx

80494d4: 8b 45 e0 mov -0x20(%ebp),%eax

80494d7: 39 c2 cmp %eax,%edx

80494d9: 74 0c je 80494e7 <phase\_1+0x63>

80494db: e8 e9 09 00 00 call 8049ec9 <explode\_bomb>

当我们确实输入两个整数时，代码执行如上图所示，程序继续对我们输入的两对参数与内置浮点数常熟进行比较，不相等也会爆炸，所以我们需要直到浮点数常数为多少

08049484 <phase\_1>:

8049484: 55 push %ebp

8049485: 89 e5 mov %esp,%ebp

8049487: 83 ec 28 sub $0x28,%esp

804948a: c7 45 f4 a1 84 76 09 movl $0x97684a1,-0xc(%ebp)

8049491: db 45 f4 fildl -0xc(%ebp)

8049494: dd 5d e8 fstpl -0x18(%ebp)

从上述代码中我们可以看到程序将0x97684a1常数压入栈中，并通过浮点栈转为双精度表示存放到-0x18(%ebp)中占8个字节，其IEE754双精度16进制表示为：