

**课 程 实 验 报 告**

**课程名称： 计算机系统基础**

**实验名称： ELF文件与程序链接**

**院 系 ：计算机科学与技术**

**专业班级 ： CS2201**

**学 号 ： U202215365**

**姓 名 ： 叶俊江**

**指导教师 ： 朱虹**

**2024 年 6 月 9 日**

**一、实验目的与要求**

通过修改给定的可重定位的目标文件（链接炸弹），加深对可重定位目标文件格式、目标文件的生成、以及链接的理论知识的理解。

实验环境：Ubuntu

工具：GCC、GDB、readelf、hexdump、hexedit、od等。

**二、实验内容**

**任务** 链接炸弹的拆除

**在二进制层面，逐步修改构成目标程序“linkbomb”的多个二进制模块（“.o文件”），然后链接生成可执行程序，要求可执行程序运行能得到指定的效果。修改目标包括可重定位目标文件中的数据、机器指令、重定位记录等。**

**1、第1关 数据节的修改**

修改二进制可重定位目标文件 phase1.o 的数据节中的内容（不允许修改其他节），使其与main.o链接后，生成的执行程序，可以输出自己的学号。

**2、第2关 简单的机器指令修改**

修改二进制可重定位目标文件 phase2.o 的代码节中的内容（不允许修改其他节），使其与main.o链接后，生成的执行程序。在phase\_2.c 中，有一个静态函数 static void myfunc( ) ，要求在 do\_phase 函数中调用myfunc( )，显示信息myfunc is called. Good!。

**3、第3关 有参数的函数调用的机器指令修改**

修改二进制可重定位目标文件 phase3.o 的代码节中的内容（不允许修改其他节），使其与main.o链接后，生成的执行程序。在phase\_3.c 中，有一个静态函数 static void myfunc(int offset) ，要求在 do\_phase函数中调用myfunc(pos )，将do\_phase的参数pos直接传递myfunc，显示相应的信息。

**4、第4关 有局部变量的机器指令修改**

修改二进制可重定位目标文件 phase4.o 的代码节中的内容（不允许修改其他节），使其与main.o链接后，生成的执行程序。在phase\_4.c 中，有一个静态函数 static void myfunc(char \*s) ，要求在 do\_phase 函数中调用myfunc(s )，显示出自己的学号。

**5、第5关 重定位表的修改**

修改二进制可重定位目标文件 phase5.o 的重定位节中的内容（不允许修改代码节和数据节），使其与main.o链接后，生成的执行程序运行时，显示Class Name : Computer Foundation. Teacher Name : Zhu Hong。

**6、第6关 强弱符号**

不准修改 main.c 和phase6.o，通过增补一个文件，使得程序链接后，能够输出自己的学号。

#gcc -no-pie -o linkbomb6 main.o phase6.o phase6\_patch.o

**7、第7关 只读数据节的修改**

修改 phase7.o 中只读数据节（不准修改代码节），使其与main.o链接后，能够输出自己的学号。

**三、实验记录及问题回答**

**（1）实验结果及操作过程记录**

实验完成结果截图：

1. **第1关 数据节的修改**

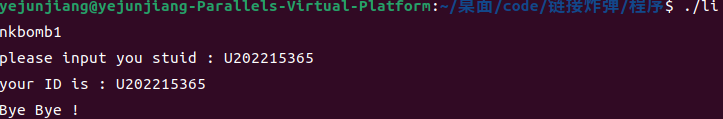


图 1-1 第1关完成截图

**2、第2关 简单的机器指令修改**

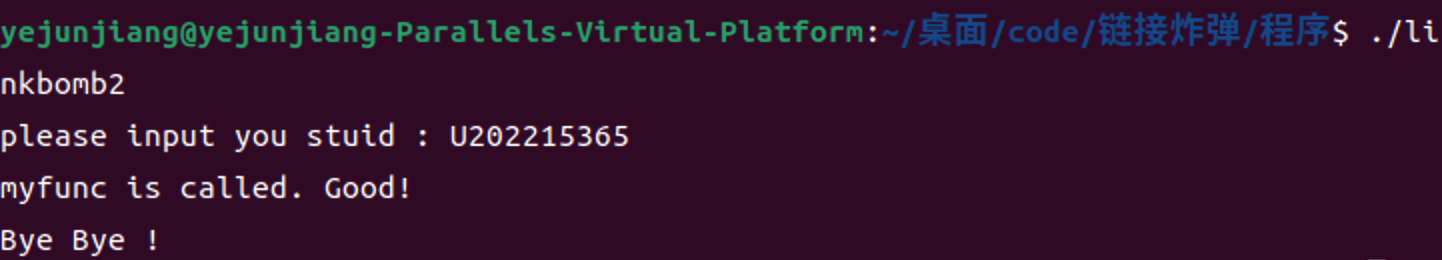


图 1-2 第2关linkbomb2运行截图

**3、第3关 有参数的函数调用的机器指令修改**

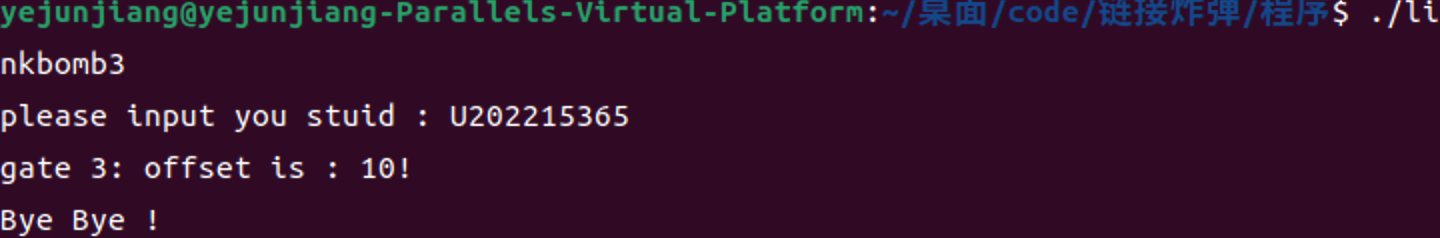


图 1-3 第3关linkbomb3运行截图

**4、第4关 有局部变量的机器指令修改**

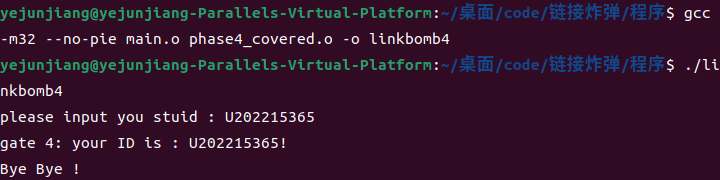


图 1-4 第4关linkbomb4运行截图

**5、第5关 重定位表的修改**



图 1-5 第5关linkbomb5运行截图

**6、第6关 强弱符号**

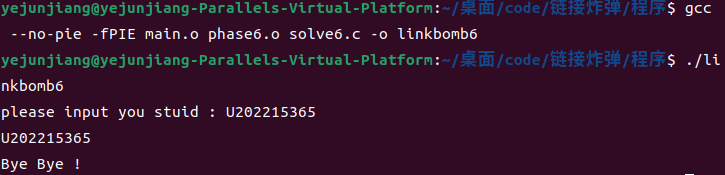


图 2-27 第6关linkbomb6运行截图

**7、第7关 只读数据节的修改**

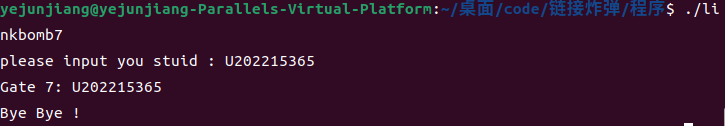


图 2-29 第7关linkbomb7运行截图

下面是每个实验关键操作的截图：

**1、第1关 数据节的修改**

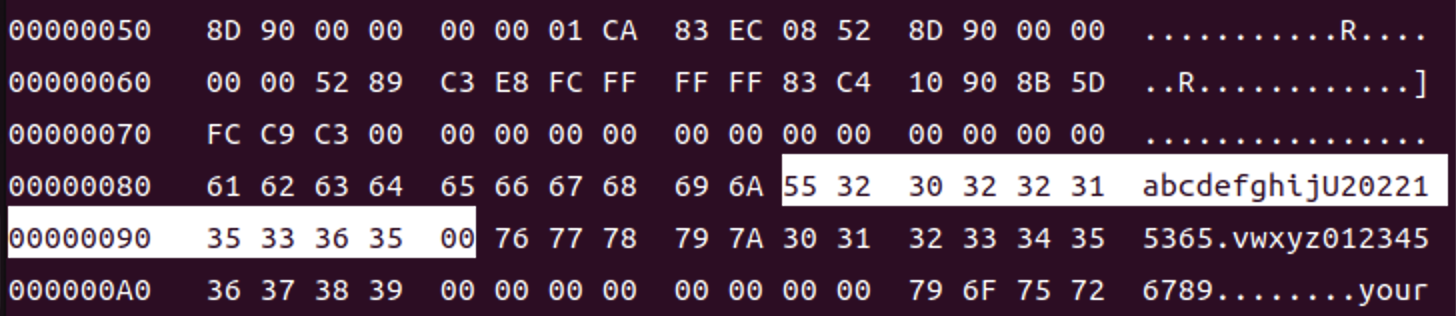


图 1-8 第1关修改phase1.o截图

**2、第2关 简单的机器指令修改**

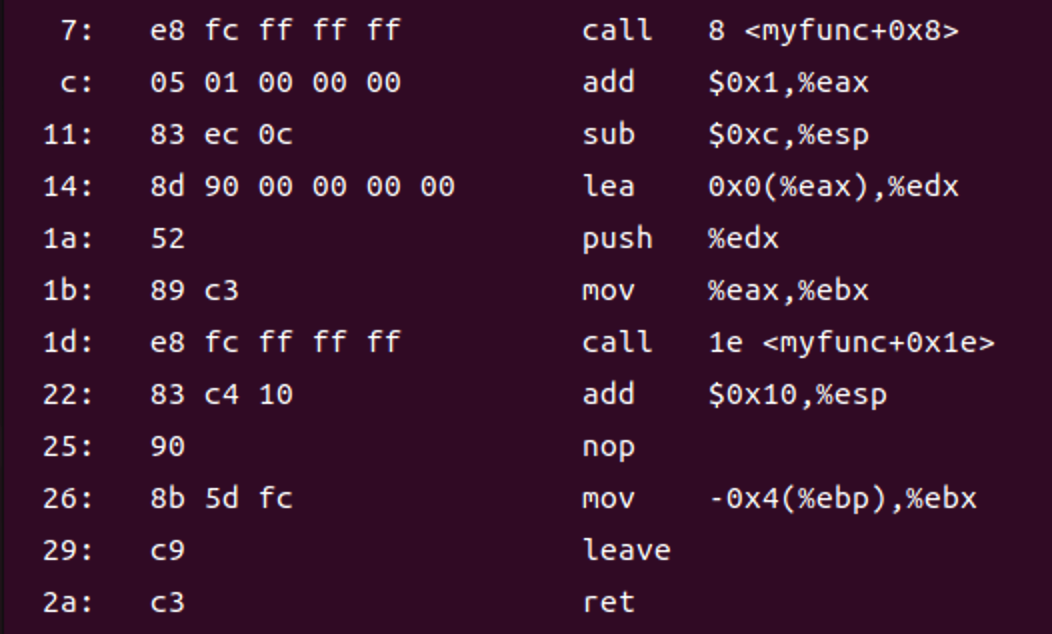


图 1-9 第2关修改phase2.o截图

**3、第3关 有参数的函数调用的机器指令修改**

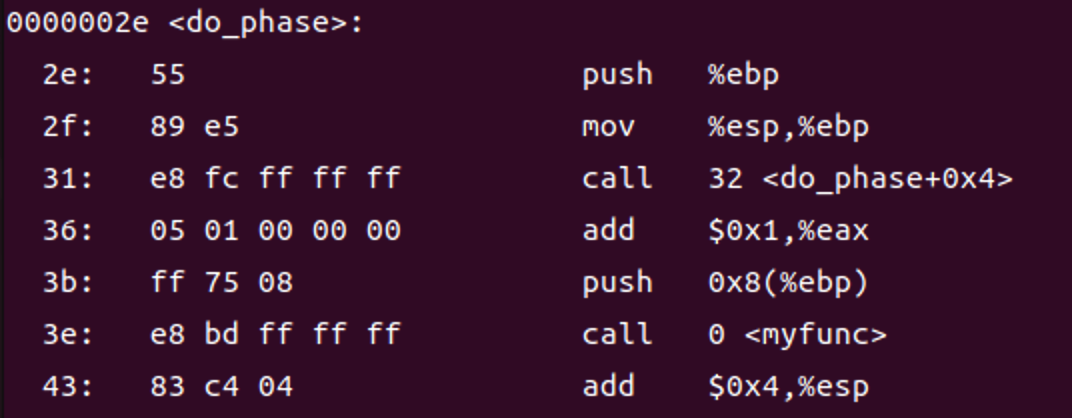


图 1-10 第3关修改phase3\_covered.o截图

**4、第4关 有局部变量的机器指令修改**

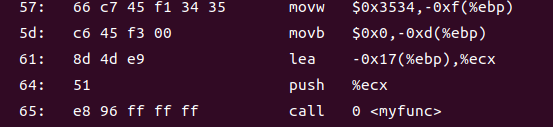


图 1-11 第4关修改phase4\_covered.o截图

**5、第5关 重定位表的修改**

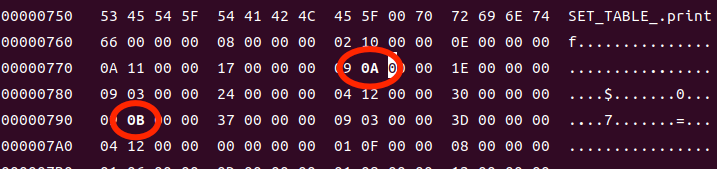


图 1-12 phase5.o修改部分截图

**6、第6关 强弱符号**

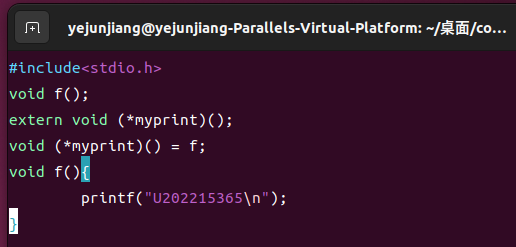


图 1-13 solve6.c内容截图

**7、第7关 只读数据节的修改**

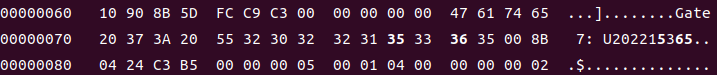


图 1-14 phase7.o修改截图

**（2） 描述修改各个文件的基本思想**

**1、第1关 数据节的修改**

我们先将main.c和phase1.o编译链接运行一下，得：

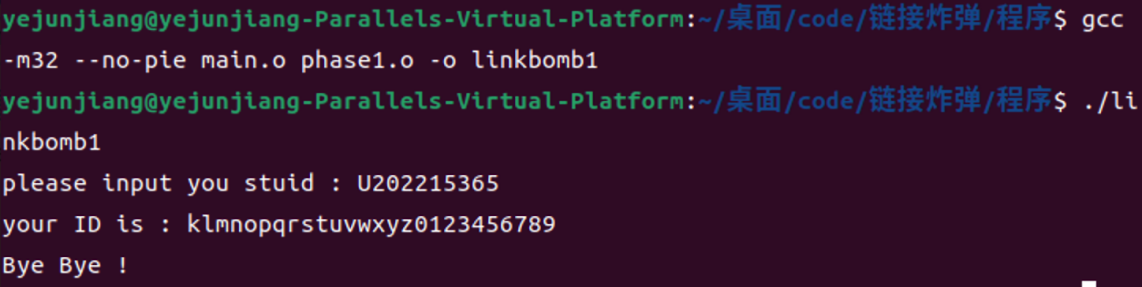


图 2-1 第1关初次运行linkbomb1截图

由此可见，我们应该要修改phase1.o中data节的内容才能使程序正常输出我们的学号。此时我们查看一下我们data节的内容。

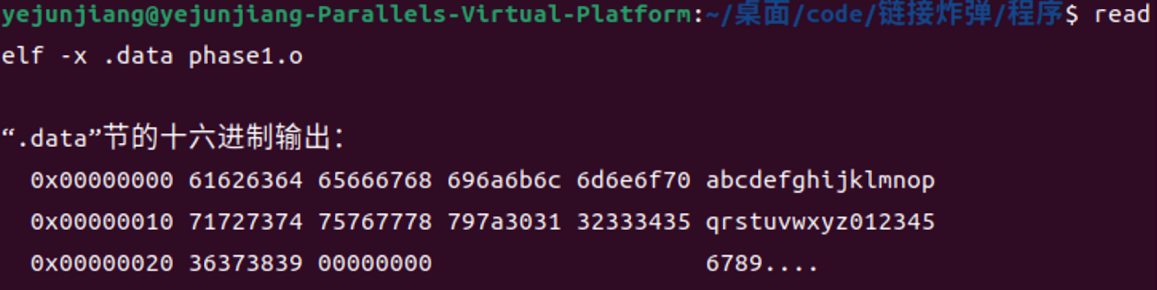


图 2-2 phase1.o初始.data节内容

在其中我们注意到.data节存在刚刚Terminal输出的字符串，故而猜测我们要将对应的字符串修改成我们的学号。那么我们先查看.data节在phase1.o文件中的偏移量。

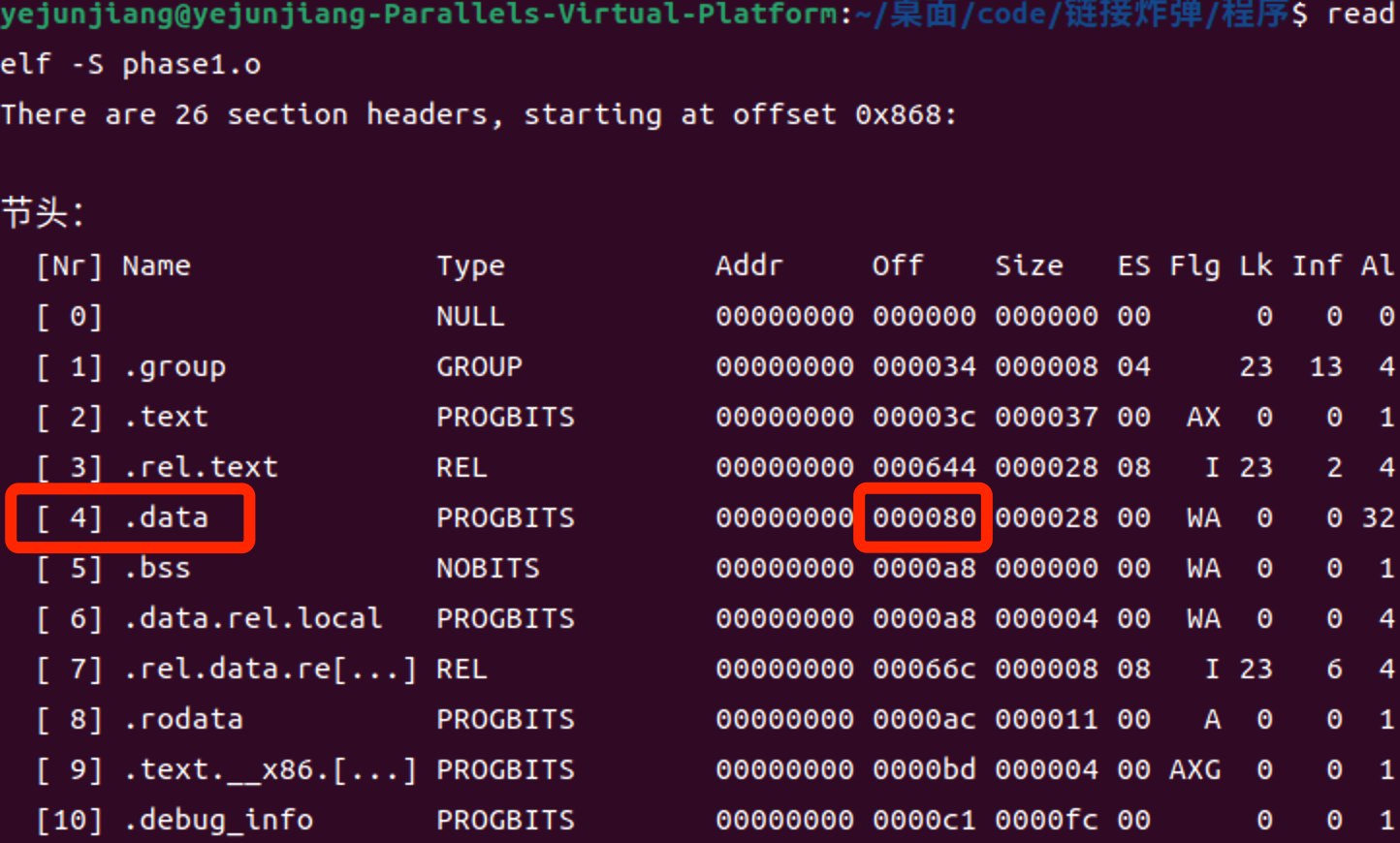


图 2-3 phase1.o各个节头的信息截图（部分）

从上图2-3可知，.data节头在phase1.o的偏移量为0x80，故而在0x80开始找起，到0x80+0x28的范围里根据ASCII码找对应的字符串内容，将对应的内容修改后完成第1关。

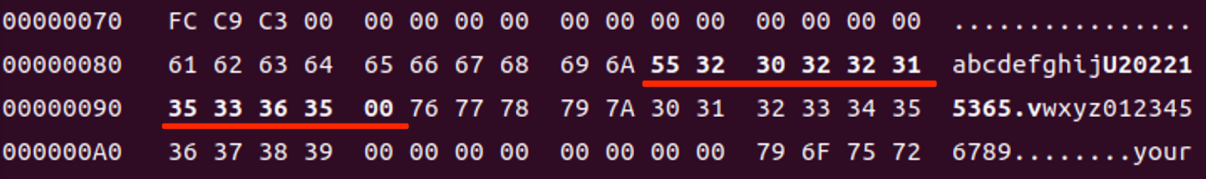


图 2-4 hexedit命令修改phase1.o截图

**2、第2关 简单的机器指令修改**

我们还是先链接运行：

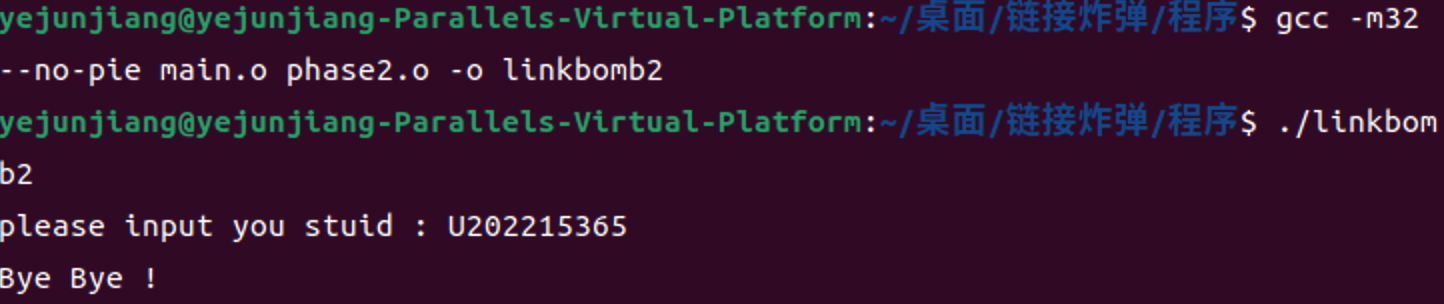


图 2-5 第2关初次链接phase2.o运行截图

由上图2-5发现运行输入学号后没输出什么信息便结束了，这时我们再反汇编一下phase2.o文件查看一下汇编代码。

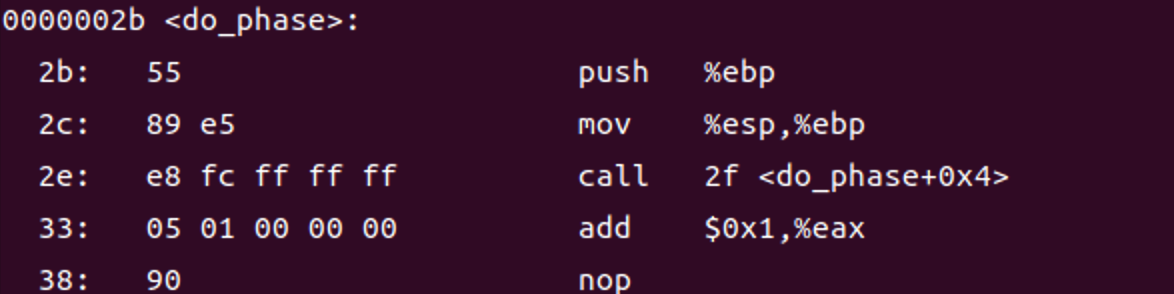


图 2-6 phase2.0<do\_phase>代码截图

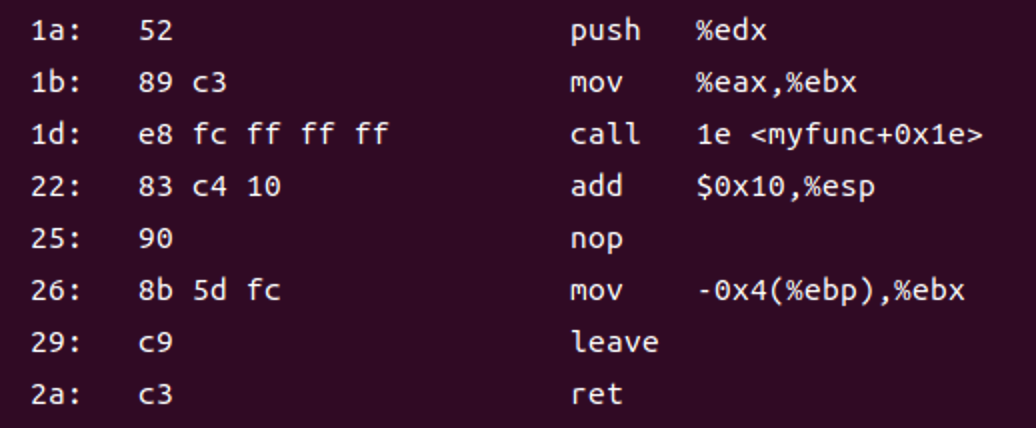


图 2-7 phase2.o<myfunc>代码截图

由上图2-6、图2-7中可见，<do\_phase>函数并没有什么具体实现，<myfunc>函数的偏移为0。又因为我们使用PC相对寻址，则可以计算新的重定位的内容并在add指令后添加一个call指令来跳转到<myfunc>函数。由转移目标地址=PC+偏移地址，PC为0x3d，目标地址为0，所以重定位的值为0-3d=-3d,转换为十六进制为0xffffffc3。故而我们要到.txt节中修改我们的重定位值。经查询，.txt节的偏移为0x3c，那么需要修改的位置是0x3c+0x3d处的值，修改后再次反汇编phase2.o代码。

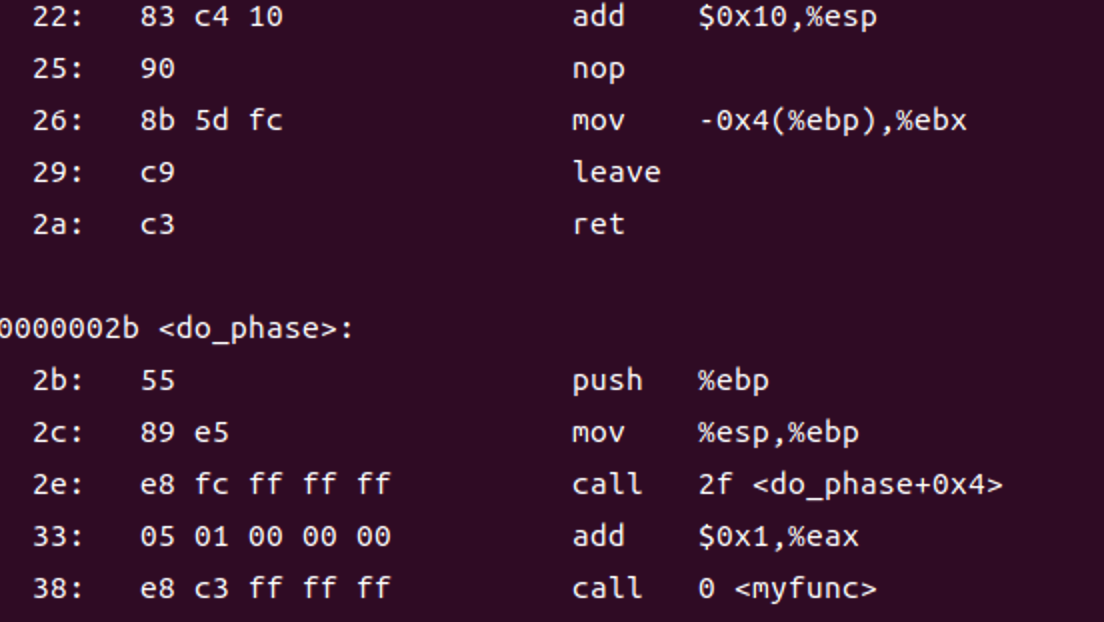


图 2-8 修改后的phase2.o截图

我们这时候再链接运行，便发现输出符合我们预期结果，第2关完成。

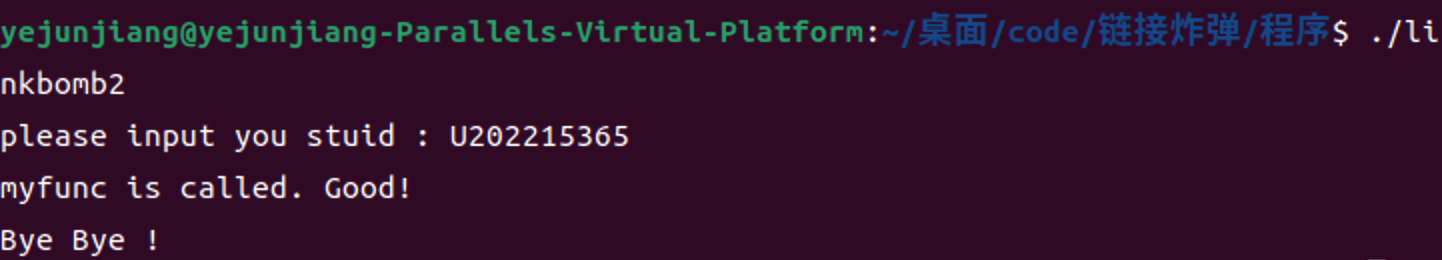


图 2-9 第2关linkbomb2运行截图

**3、第3关 有参数的函数调用的机器指令修改**

不知道是mac设备的问题还是虚拟机版本的问题，初始我们在试着链接或者反编译phase3.o时均报错文件格式不匹配，执行file命令后发现系统识别文件格式为data格式。后经查询，发现是拷贝到虚拟机中的phase3.o中ELF头地址存在偏移，偏移量为0xC00，故而虚拟机识别不出ELF格式，所以我们需要将前面无关的偏移量去掉才可以正常链接或者反编译，我们将去除偏移量后的文件命名为phase3\_covered.o（第四关中的phase4.o也存在同样问题，解决方案一致，生成文件为phase4\_covered.o，故不再赘述）。

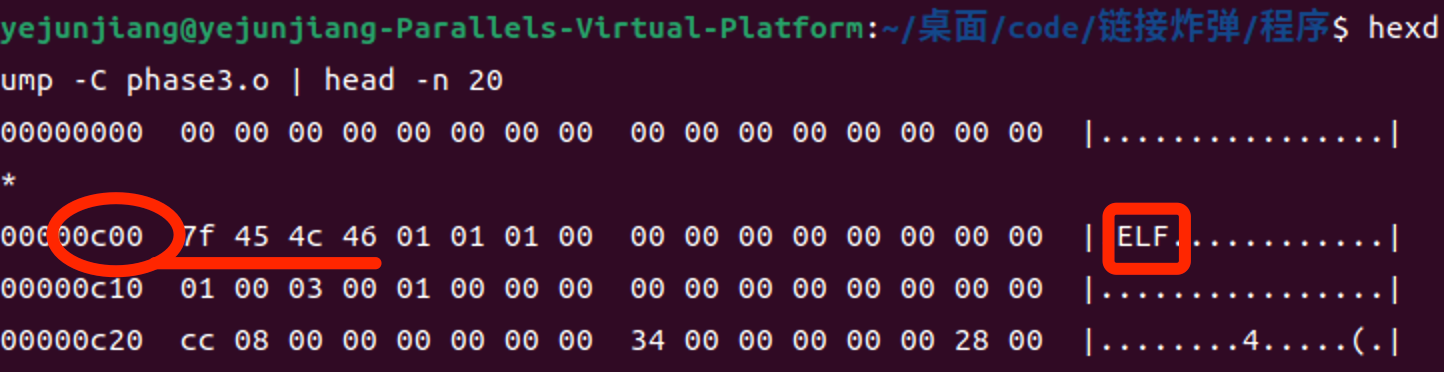


图 2-10 查询phase3.o中ELF头偏移量截图

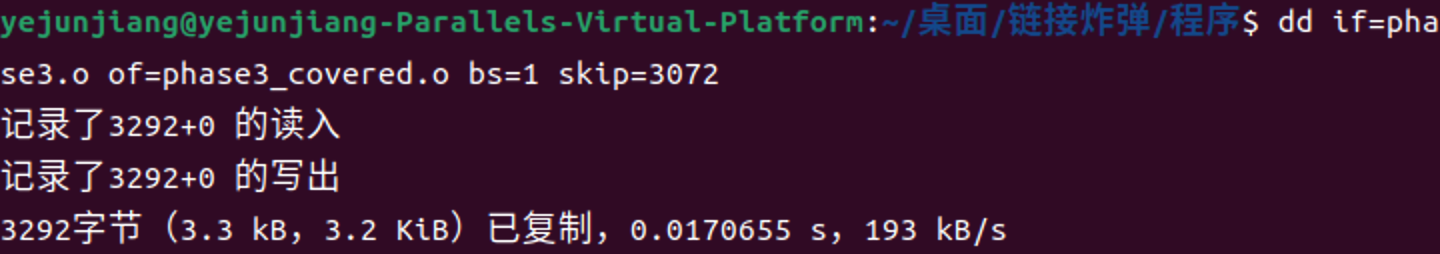


图 2-11 去除phase3.o中ELF偏移量截图

我们这时候再链接运行一下，发现linkbomb3也是没什么输出信息。

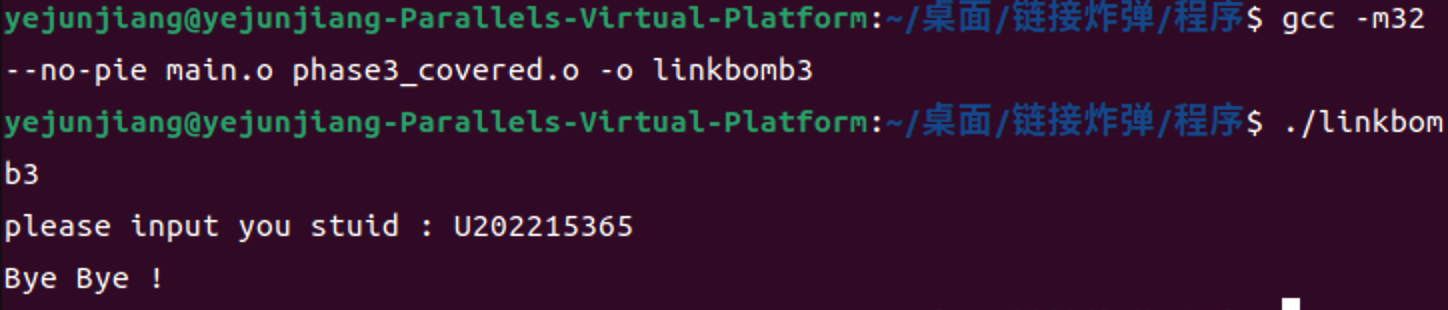


图 2-12 初次链接phase3\_covered.o运行截图

这时候我们反汇编一下phase3\_covered.o，发现和第2关一样<do\_phase>函数没有调用<myfunc>的实现。故而我们需要在<do\_phase>中添加带参数的<myfunc>函数。经查询，得知<do\_phase>中的参数保存8（%ebp）中，且在myfunc中也是调用8（%ebp）来调用参数。故而我们的编译代码可以如下图2-13所示编写。

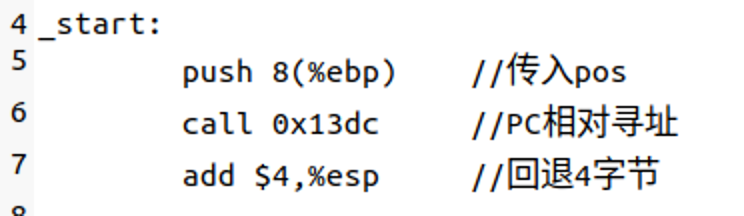


图 2-13 phase3\_covered.o待添加汇编代码截图

得到汇编代码后，将其转换成机器码添加到<do\_phase>中，有：

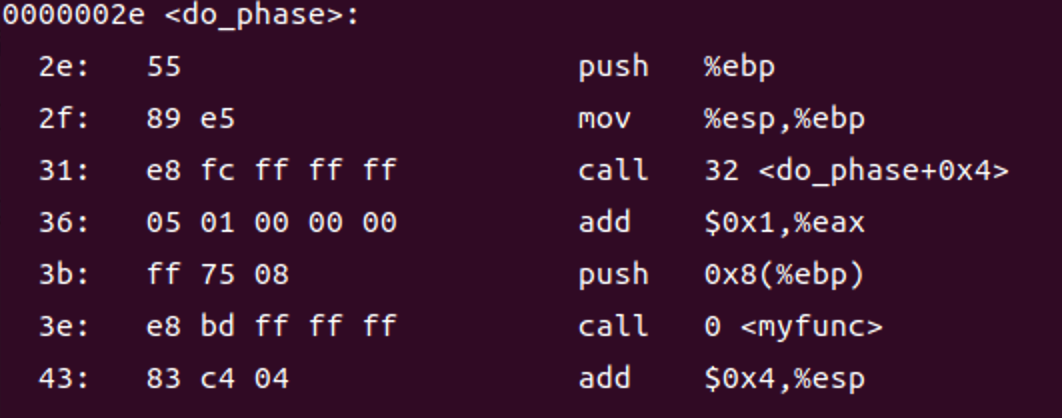


图 2-14 修改后的<do\_phase>截图

此时我们再运行linkbomb3，发现输出符合预期结果，第3关完成。

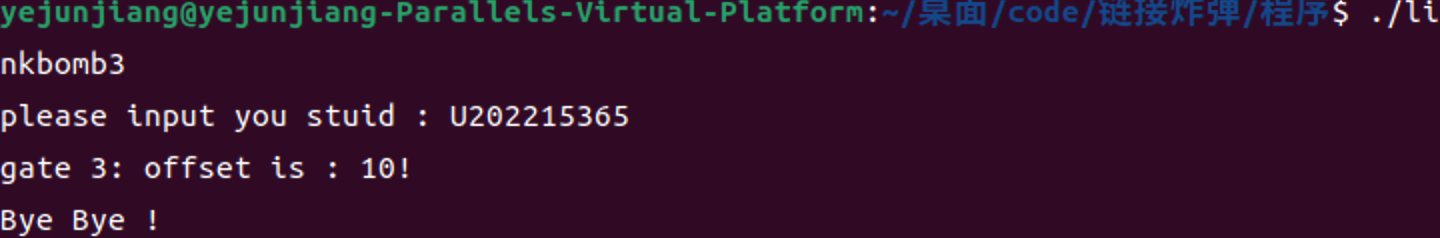


图 2-15 第3关linkbomb3运行截图

1. **第4关 有局部变量的机器指令修改**

我们链接运行，发现还是没有输出什么信息

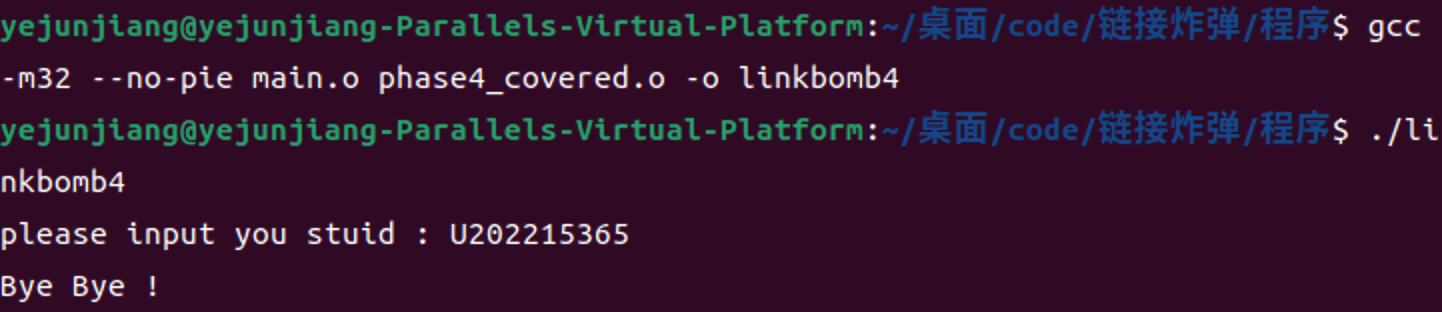


图 2-16 初次运行linkbomb4截图

这时候反汇编发现<do\_phase>函数发现局部变量存储的是一个字符串，里面应为学号，故而我们需将其内容修改成自己的学号并调用函数输出。

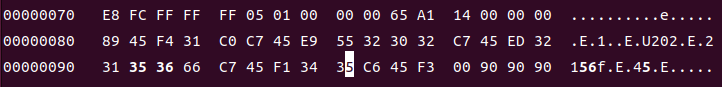


图 2-17 修改phase4\_covered.o的.data节

此时我们再编写汇编代码放入phase4\_covered.o中，计算一下重定位内容为-6a，十六进制为0xffffff96，故而最终phase4\_covered.o中的<do\_phase>函数应该如下图2-18所示。

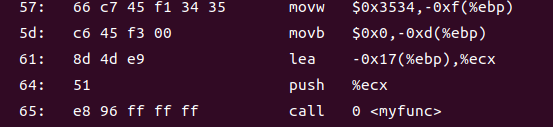


图 2-18 修改后的phase4\_covered.o汇编代码截图（部分）

修改后再重新链接运行，发现符合预期结果，第4关完成。

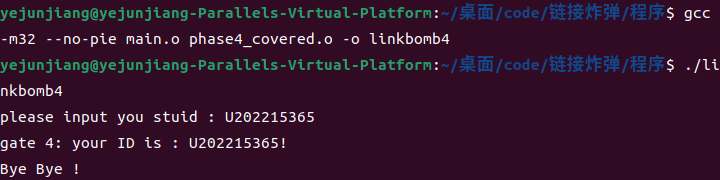


图 2-19 第4关linkbomb4运行截图

**5、第5关 重定位表的修改**

我们还是先链接运行一下。

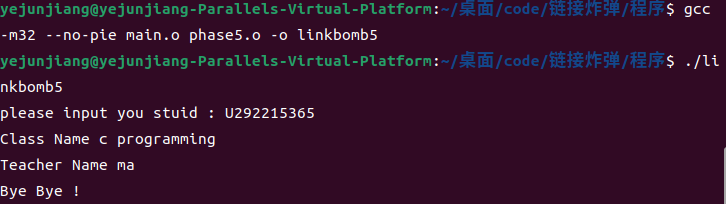


图 2-20 初次运行linkbomb5截图

我们这时候查看一下符号表，发现没什么信息。

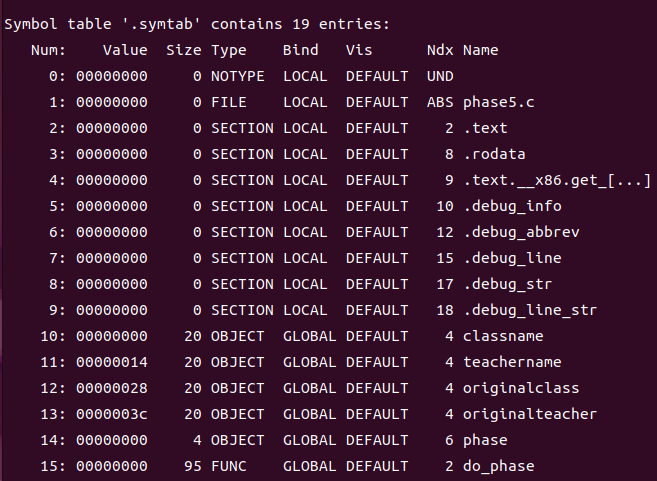


图 2-21 phase5.o符号表截图

这时我们先hexedit输出phase5.o的数据，我们在查看时发现originalclass和originalteacher的值是错的，并且刚刚好是图2-20的输出，而teachername和classname恰好是我们需要的答案，故而我们需要将重定位表中的内容定位到teachername和classname中。那么根据classname和teachername的编号分别为10和11，即十六进制的a和b，我们开始修改。经查询，得知重定位节的偏移为0新64，那么我们开始修改符号编号。

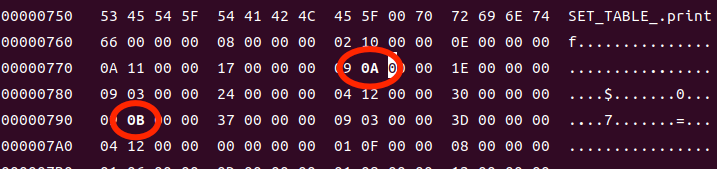


图 2-22 phase5.o修改部分截图

修改后我们再进行链接运行，发现得到预期结果，第5关完成。



图 2-23 第5关linkbomb5运行截图

**6、第6关 强弱符号**

这一关的话我们需要自己编写程序来实现。我们先反汇编一下main.o和phase6.o，查看一下phase变量和myprint变量。main.o的符号表中的phase信息如下图2-24所示，这里phase是一个全局变量，并且没有初始化（COM），所以这是一个弱符号。

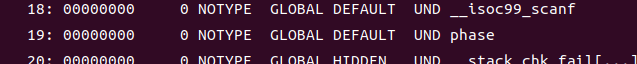


图 3-24 main.o符号表截图

phase6.o的符号表中的phase信息如下图2-24所示，这里说明phase在phase6.o中初始化，并且在这里我们能发现myprint变量是COM，说明myprint变量在phase6.o中声明但是未初始化，这是一个弱变量，故而我们可以在solve6.c中定义myprint并且初始化。

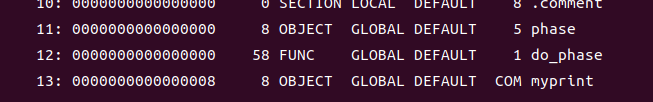


图 2-25 phase6.o符号表截图

我们编写的solve6.c内容如下图2-26所示。

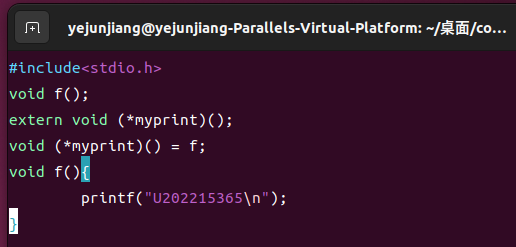


图 2-26 solve6.c截图

将main.c重新编译（64位），将他们链接运行发现得到预期结果，第6关完成。

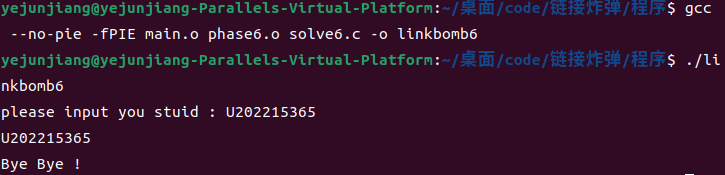


图 2-27 第6关linkbomb6运行截图

**7、第7关 只读数据节的修改**

我们还是链接运行一下，发现输出的学号后面几位与我们的学号不同，这时候我们使用readelf指令查看phase7.o中各个节对应的位置，找到我们的.rodata只读数据节对应的偏移，再利用hexedit指令在该数据节中依据右侧对应的值找到我们最终输出答案的存放位置，将错误的位改成我们的学号即可。

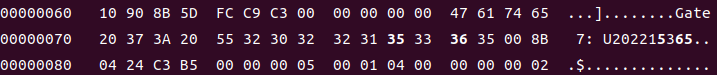


图 2-28 phase7.o修改截图

修改完后再链接运行，发现得到预期结果，第7关完成。

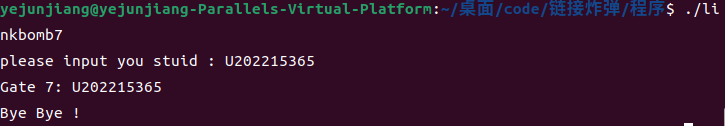


图 2-29 第7关linkbomb7运行截图

**四、体会**

这次实验是一次非常有挑战性和收获的体验，通过对可重定位目标文件（.o 文件）的修改和链接，深入理解了 ELF 文件格式、链接过程以及计算机系统底层操作的细节。在整个实验过程中，我体会到了二进制层次操作的精细与复杂，同时也大大提高了自己在处理底层系统编程问题上的能力。

在最开始做第1关第2关时都没有难度，直到做到第3关时发现我的设备上的phase3.o有问题，但是我当时不会处理而拖了很长时间。后来在查阅资料和老师的帮助下我才解决了这个问题，搞定这个问题之后后面也是顺风顺水，非常连贯的解决了本次实验的问题。

总的来说，这次实验不仅让我深入理解了 ELF 文件和链接过程，还大大提高了我在处理二进制文件和底层系统编程上的能力。在实验过程中，我学会了如何使用各种工具来分析和修改二进制文件，并且对计算机系统的底层操作有了更深的理解。这次实验的经验将对我未来的学习和工作带来巨大的帮助。