

**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 ： 图灵2301班**

**学 号 ： U202311239**

**姓 名 ： 刘星佳**

**指导教师 ： 王多强**

**2024 年 10 月 29 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 : Xu Xiangyang。

**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**

**1. 第1关**

在phase1.o中位置0x0000008e处起改为0x55,0x32,0x30,0x32,0x33,0x31,0x31,0x32,0x33,0x39,0x00。

1. **第2关**

在phase2.o中位置0x00000062处起改为0xe8,0xd9,0xff,0xff,0xff

1. **第3关**

在phase3.o中位置0x00000073处起改为0x8b,0x7d,0xfc,0xe8,0xc5,0xff,0xff,0xff

1. **第4关**

在phase4.o中位置0x0000008b处起改为0x55,0x32,0x30,0x32,0x33,0x31,0x31,0x32,0x48,0x89,0x45,0xed,0x66,0xc7,0x45,0xf5,0x33,0x39,0xc6,0x45,0xf7,0x00,0x48,0x8d,0x7d,0xed,0xe8,0x96,0xff,0xff,0xff

1. **第5关**

在phase5.o中位置0x00000404处改为0x0b，位置0x44c处改为0x0c。

1. **第6关**

增补代码phase6\_patch.c如下：

# include <stdio.h>

void printMyID(void);

void (\*myprint)(void) = printMyID;

void printMyID(void) {

puts("U202311239");

}

1. **第7关**

在phase7.o中位置0x00000070处起改为0x55,0x32,0x30,0x32,0x33,0x31,0x31,0x32,0x33,0x39,0x00。

**（2） 描述修改各个文件的基本思想**

**1. 第1关**

不做任何调整时运行linkbomb1，发现程序输出了opqrstuvwxyz0123456789，然后对二进制模块phase1.o反汇编，发现.data数据段中存储了一个常量字符串abcdefghijklmnopqrstuvwxyz0123456789，因此程序输出的结果是从这个字符串中截取的，截取位置与学号有关。所以只需要把o之后的字符串修改为想要的结果U202311239即可（字符串之后一个字节设置为0x00表示字符串结尾）

1. **第2关**

反编译linkbomb2发现do\_phase函数中有一大段代码空白段，因此可以在这里实现一个call命令调用myfunc函数，即call 0x40131d，变成机器码为e8 d9 ff ff ff（e8后面跟着目标函数相对下一行指令的偏移量，为-0x27），直接在二进制文件中0x90（nop）对应的位置修改为以上机器指令即可调用myfunc。

1. **第3关**

这关与第2关很类似，传入参数只需要把参数传入edi寄存器即可，而先前参数存储在-0x4（%rbp)的位置，因此我们只要在do\_phase的空代码段处加入以下汇编：

mov -0x4(%rbp), %edi

call <myfunc>

编译成二进制文件后转化成机器码，再插入到phase3.o空代码段对应的位置即可。

1. **第4关**

这关我们需要现在栈帧中存储学号字符串，然后将字符串的头指针作为参数传进myfunc中。观察linkbomb4反汇编的代码，发现原先的phase4.o已经在-0x13(%ebp)的位置设置好了一个字符串“U202212345”，因此只需要将设置这个字符串对应的机器码部分稍作修改改为自己的学号，然后再加入以下汇编代码：

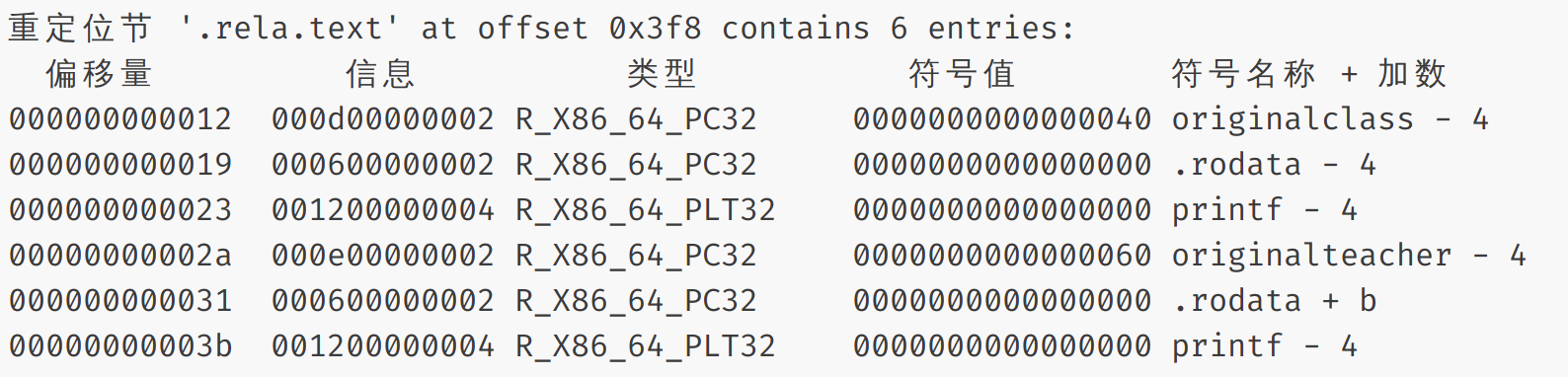
lea -0x13(%rbp), %rdi

call <myfunc>

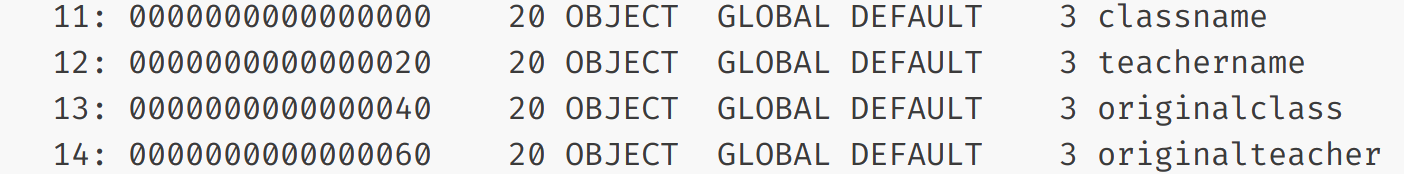
用gdb和objdump得到机器码后修改phase4.o即可。

1. **第5关**

首先通过readelf -r查看重定位节的内容：



发现printf对应的偏移量在汇编的do\_phase中正好是两个call，因此输出是在do\_phase中进行的，并且可以发现printf传入的参数对应的符号为originalclass和originalteacher。这时候用readelf -s看一眼符号表：



根据偏移量对照汇编中.data节内容，发现originalclass是c programming，而classname和teachername是Computer Foundation和Xu Xiangyang，意味着我们要修改重定位表使得输出的内容应当定位到符号classname和teachername处。

在重定位节中，可以发现r\_info（信息）字段的高16位为r\_sym，表示对应的符号在符号表中的编号；低32位为r\_type表示重定位类型。我们应该改变r\_sym，将偏移量0x12对应的r\_sym从000d改为000b（对应符号表11，classname），偏移量为0x2a对应的r\_sym从000e改为000c（对应符号表12，teachername）即可。

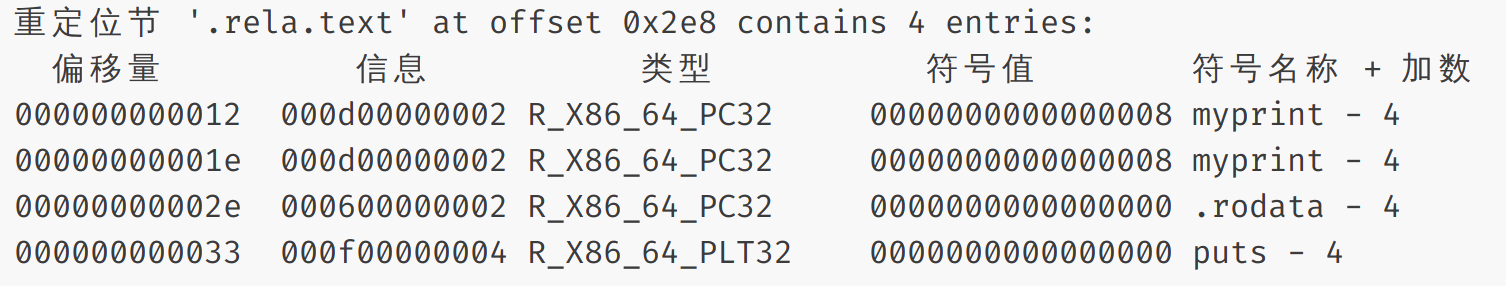
要着手修改，还需要知道重定位节在.o文件的位置（偏移量），使用readelf -S命令查看节头：



在0x3f8处修改。

1. **第6关**

首先查看重定位表：

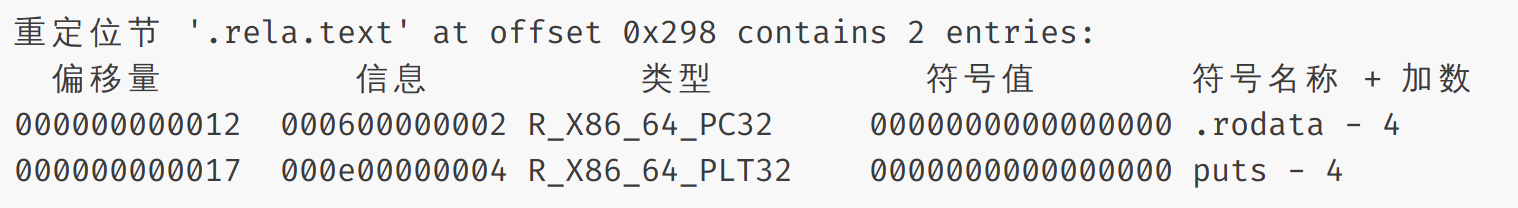


发现这里有一个陌生的符号myprint，结合题目要求可以认为这是一个我们需要在phase6\_patch.c中自己实现的函数。查看反汇编的结果，发现myprint作为一个指针（函数指针）在寄存器间传递，最终用call调用。同时，在调用这个函数之前也没有对往rdi寄存器中进行赋值，因此这个函数没有参数，调用这个函数之后也没有用到eax/rax寄存器的值，因此这个函数也没有返回值。所以，我们只需要先实现一个自己的函数void printMyID(void)，作用就是输出自己的学号，然后再赋值给extern函数指针myprint即可。

使用gcc -c phase6\_patch.c得到可重定位目标文件phase6\_patch.o并和phase6.o一起链接到main.c上。

1. **第7关**

仍然先看重定位表：



偏移量12对应的是传入给rdi寄存器即下一步调用的函数的参数，而下一步调用的函数位于偏移量17，总而言之就是用puts输出了位于.rodata节的数据。查看phase7.o的二进制文件发现.rodata节是一个字符串“Gate 7: U202212345”，因此将这里的学号改为自己的学号即可。

**四、体会**

通过这次《计算机系统基础》课程的实验，我对ELF文件与程序链接的理论知识有了更深刻的理解。实验不仅加深了我对可重定位目标文件格式、目标文件的生成以及链接过程的认识，而且通过实际操作，我学会了如何使用各种工具（如GCC、GDB、readelf等）来分析和修改二进制文件。

第1关让我学会了如何修改数据节来改变程序的输出，这不仅需要理解数据节的作用，还需要精确地定位到需要修改的位置。第2关和第3关则让我掌握了如何在代码节中插入机器指令，这对于理解程序的执行流程和函数调用机制非常有帮助。第4关难点在于理解局部变量和函数参数的传递，这让我对栈帧和函数调用约定有了更深入的认识。第5关花了我最长的时间，因为它涉及到重定位表的修改。这不仅需要理解重定位表的结构和作用，还需要精确地定位到需要修改的重定位条目。通过这个关卡，我对ELF文件的结构和链接过程的理解更加深入。第6关和第7关我学会了如何处理强弱符号和只读数据节的修改。

这次实验不仅提高了我的技术能力，也锻炼了我的问题解决能力。我学会了如何分析问题、查找资料、动手实践，并最终解决问题。这些经验对我的学习和未来都是非常宝贵的。我会将这些知识应用到更复杂的项目中，继续深化我的理解和技能。