

**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（链接炸弹）**

**院 系 ：计算机科学与技术**

**专业班级 ： CS2305**

**学 号 ： U202315677**

**姓 名 ： 岳皓**

**指导教师 ： 班鹏新**

**2025 年 4 月 18 日**

# 实验五 ELF文件与程序链接（链接炸弹）

**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**

**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。**

**第1关 静态数据与ELF数据节**

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。

**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!。

**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。

**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。

1. **实验步骤、记录及分析**

**第1关**

首先输入指令gcc -no-pie -o -linkbomb2 main.o phase2.o生成可执行文件linkbomb1，然后输入代码readelf -S phase1.o来显示各存储位的信息，显示的信息如图1。在信息中查找数据区的地址以备后用，由图可知数据区的首地址为80。



图 1 readelf查看phase1

再输入指令./linkbomb1来运行可执行文件文件，结果如图2，由图可知，其输出应为从k开始到z结束，因此在相应的位置找到k位置，并修改k到z之间的数据，即可运行得到想要的结果。

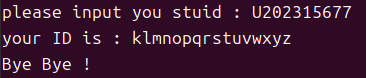


图 2 运行结果

输入指令hexdit phase1从而对文件phase1.o进行修改，找到k开始的位置，并对相应位置的数进行更改即可，按F2进行保存并退出从而实现对文件的修改，修改结果如图3。

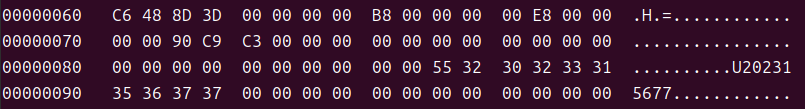


图 3 修改结果

通过上面的操作，再次输入指令gcc -no-pie -o -linkbomb2 main.o phase2.o和./linkbomb1可以生成新的可执行文件内容从而替换原先的文件内容并运行，结果如图4。

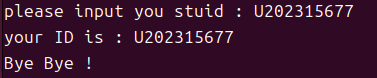


图 4 最终运行结果

**第2关**

首先输入指令gcc -no-pie -o -linkbomb2 main.o phase2.o生成可执行文件linkbomb2，再输入指令objdump -d linkbomb2 > bomb2.txt，在生成的bomb2.txt中查看函数myfunc和do\_phase的返汇编代码，如图5。从图中可知函数myfunc的入口地址为0x401385，而do\_phase函数在0x4013a4后面有一系列nop指令，需要对其进行替换。

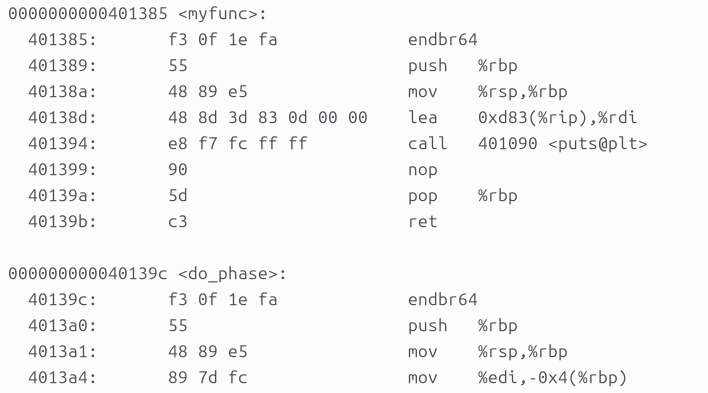


图 5 反汇编代码截图

由于call指令以及后面的偏移量占五个字节，故修改后的call指令之后的一个指令地址应为0x4013ac，而目标位置为myfunc函数地址，偏移量为0x401385-0x4013ac=0xffffd9，然后对相应的地方进行数值替换即可，其中90为nop指令，call指令为e8，同时注意存储方式为小端存储。

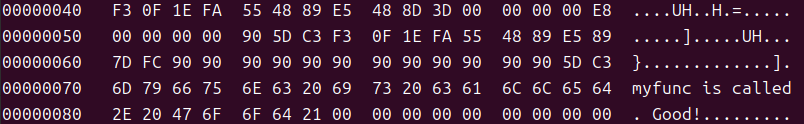


图 6 修改前的文件

QQ20250417-214607

图 7 修改结果

然后输入指令gcc -no-pie -o -linkbomb2 main.o phase2.o生成新的可执行文件，再输入指令objdump -d linkbomb2 > bomb2.txt将修改后的文件的返汇编代码放在bomb2.txt中，查看修改后的代码结果如图8。

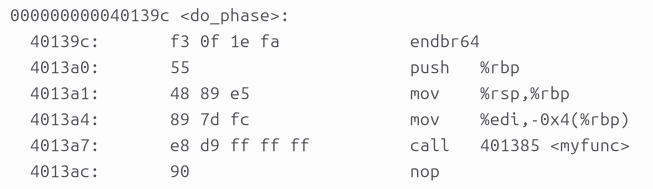


图 8 修改后的函数

通过上面的操作，再次输入指令./linkbomb2运行文件，结果如图9。

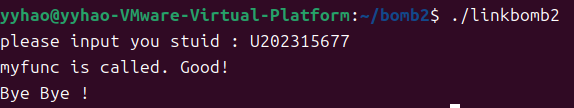


图 9 运行结果

**第3关**

首先输入指令gcc -no-pie -o -linkbomb3 main.o phase3.o生成可执行文件linkbomb3，再输入指令objdump -d linkbomb3 > bomb3.txt，在生成的bomb3.txt中查看函数myfunc和do\_phase的代码，如图10。从图中可知函数myfunc的地址分别为0x401385，而do\_phase函数在0x4013ad后面有一系列nop指令，需要对其中一条进行替换。

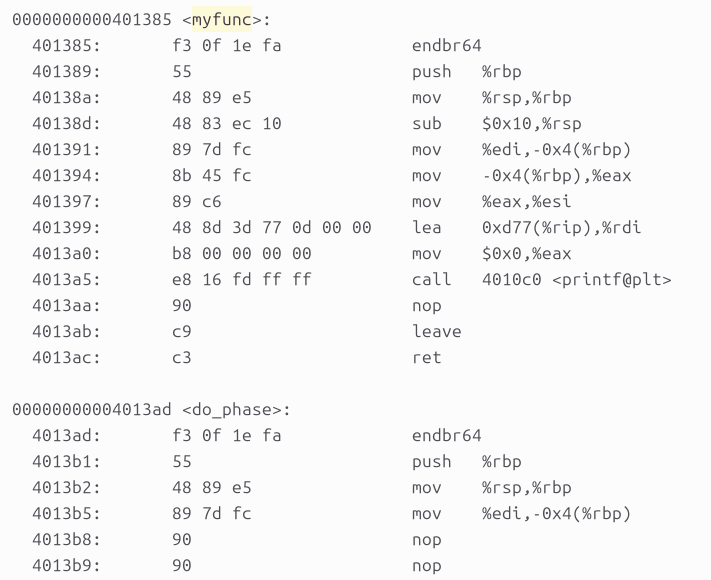


图 10 未修改前的文件返汇编代码截图

由于call指令以及后面的偏移量占五个字节，故修改后的call指令后面一个指令地址为0x4013bd，而目标位置为myfunc函数地址，偏移量为0x401385-0x4013bd=0xffffc8，然后对相应的地方进行数值替换即可，其中90为nop指令，call指令为e8，注意存储方式为小端存储。

QQ20250417-203224

图 11 修改前的文件

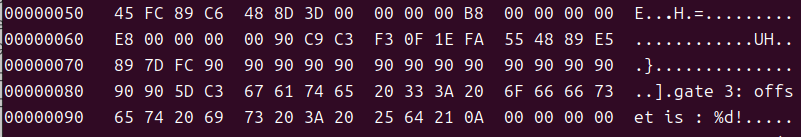


图 12 修改结果

然后输入指令gcc -no-pie -o -linkbomb3 main.o phase3.o生成新的可执行文件，再输入objdump -d linkbomb3 > bomb3.txt将修改后的文件的返汇编代码放在bomb3.txt中，查看修改后的函数结果如图13。

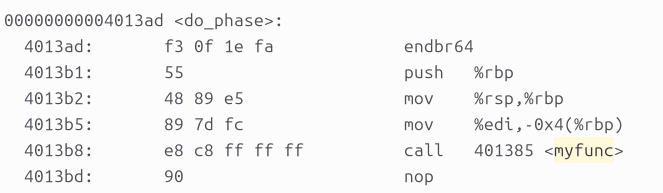


图 13 修改后的反汇编代码

通过上面的操作，再次输入指令./linkbomb3运行文件，结果如图14。

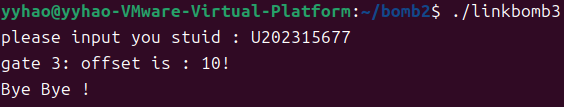


图 14 运行结果

**第4关**

首先输入指令gcc -no-pie -o -linkbomb4 main.o phase4.o生成可执行文件linkbomb4，再输入指令objdump -d linkbomb4 > bomb4.txt，在生成的bomb4.txt中查看函数myfunc和do\_phase的代码，如图15。从图中可知函数myfunc的地址分别为0x401385，而do\_phase函数在0x4013e2后面有一系列nop指令，需要对其中一条进行替换。

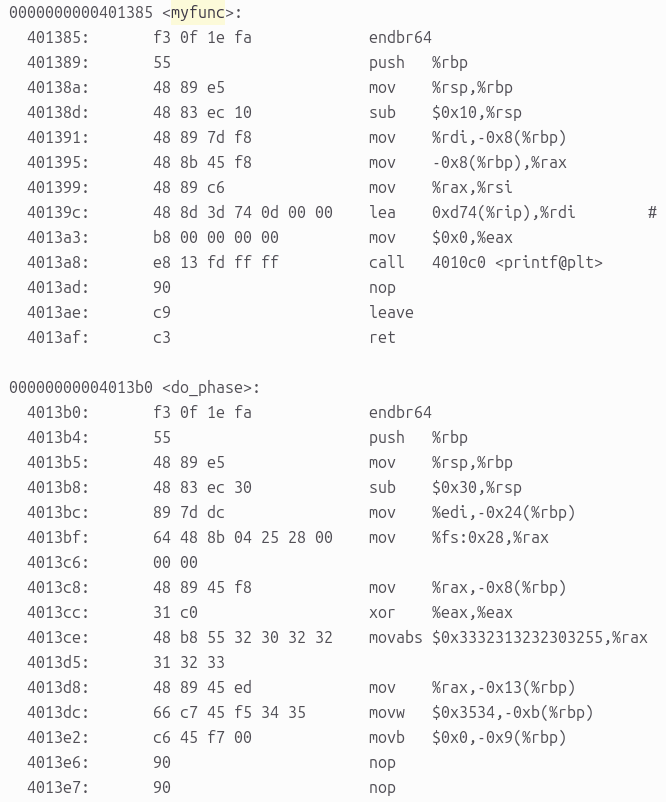


图 15 替换前的反汇编代码

QQ20250417-205031

图 16修改前的文件

QQ20250417-205109

图 17 添加指令后的文件

由于函数的参数使用的是字符指针，所以其输出时为数据位上的值，还需对相应位置的数进行替换，在文件中查找到相应的位置将其替换成自己的学号，替换结果如图18。

QQ20250417-205350

图 18 替换后的数据

然后输入指令gcc -no-pie -o -linkbomb4 main.o phase4.o生成新的可执行文件，再输入objdump -d linkbomb4 > bomb4.txt将修改后的文件的返汇编代码放在bomb4.txt中，查看修改后的函数结果如图19。



图 19 修改后的函数

通过上面的操作，再次输入指令./linkbomb4运行文件，结果如图20。

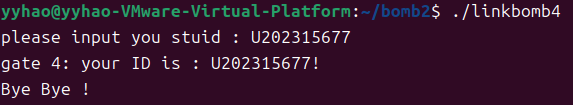


图 20 运行结果

**四、体会**

我深入认识了 ELF 文件结构与原理，其作为可执行文件标准格式，内部的文件头、节表和段表等结构，让我对程序编译和链接过程有了新理解，清晰地知晓程序构成与运行机制。

在实验中，我掌握了objdump、readelf、hexedit等解析和编辑 ELF 文件的工具。它们能查看汇编代码、头部信息和节表内容，还能修改文件，让我更了解程序内部结构，尤其是对 C 语言编译链接流程有了更全面认识。

同时，我体会到修改ELF文件有风险。实验中错误修改节表属性级以修改错误地址导致程序无法运行或者出错，这让我明白修改时要谨慎，要清楚后果，也深知编程中每个细节都关乎程序稳定性和正确性。

总之，这次实验加深了我对计算机系统基础知识的理解，提高了实践和问题解决能力，这些经验将对未来学习和工作产生深远影响。