实验四 LinkLab实验

**一、实验目的**

1. 了解链接的基本概念和链接过程所要完成的任务。
2. 理解ELF目标代码和目标代码文件的基本概念和基本构成
3. 了解ELF可重定位目标文件和可执行目标文件的差别。
4. 理解符号表中包含的全局符号、外部符号和本地符号的定义。
5. 理解符号解析的目的和功能以及进行符号解析的过程。

**二、实验仪器设备/实验环境**

Linux Ubuntu 18.04 64-bit

笔记本电脑

VMware虚拟机

**三、实验内容**

每个实验阶段（共5个）考察ELF文件组成与程序链接过程的不同方面知识

阶段1：全局变量⬄数据节

阶段2：强符号与弱符号⬄数据节

阶段3：代码节修改

阶段4：代码与重定位位置

阶段5：代码与重定位类型

在实验中的每一阶段n（n=1,2,3,4,5…），**按照阶段的目标要求修改相应可重定位二进制目标模块phase[n].o后**，使用如下命令生成可执行程序linkbomb：

$ **gcc -o linkbomb main.o phase[n].o** [其他附加模块——见具体阶段说明]

正确性验证：如下运行可执行程序linkbomb，应输出符合各阶段期望的字符串：

$ ./linkbomb

$ 19210320303 [仅供示例，具体目标字符为每位学生学号]

实验结果：将修改后正确完成相应功能的各阶段模块（phase1.o, phase2.o, …）提交供评分。

**四、实验步骤**

1. 实验数据

学生实验数据包： linklab学号.tar

数据包中包含下面文件：

main.o：主程序的二进制可重定位目标模块（实验中无需修改）

phase1.o, phase2.o, phase3.o, phase4.o, phase5.o：各阶段实验所针对的二进制可重定位目标模块，需在相应实验阶段中予以修改。

**解压命令：tar xvf linklab学号.tar**

2. 实验工具

readelf：读取ELF格式的各.o二进制模块文件中的各类信息，如节（节名、偏移量及其中数据等）、符号表、字符串表、重定位记录等

objdump：反汇编代码节中指令并提供上述部分类似功能

hexedit：编辑二进制文件内容

**实验阶段1**

要求：修改二进制可重定位目标文件“phase1.o”的数据节内容，使其与main.o链接后能够运行输出（且仅输出）自己的学号：

$ gcc -o linkbomb main.o phase1.o -no-pie

$ ./linkbomb

学号

实验提示：

检查反汇编代码，获得printf（根据情况有可能被编译时转换为puts）输出函数的参数的（数据节中）地址 。

使用hexedit工具（或自己编写实现二进制ELF文件编辑程序），对phase1.o数据节中相应字节进行修改。

**实验阶段2**

要求：根据强符号与弱符号的原则，判断符号表中的符号以及其所对应的数据区域。利用符号解析规则，创建生成一个名为“phase2\_patch.o”的二进制可重定位目标文件（可以不修改phase2.o模块），使其与main.o、phase2.o链接后能够运行和输出（且仅输出）自己的学号：

$ gcc -o linkbomb main.o phase2.o phase2\_patch.o -no-pie

$ ./linkbomb

学号

实验提示：

Phase2.o模块的符号表中，包含了类型为COM的符号。此类符号的特点是：未被赋初值。所以，其在ELF的数据节中并不真实存在。所以需要另寻解决办法，创造出真实存在的数据并对其进行二进制编辑，以达到输出自己学号的目的。

解题需要运用的主要知识为强弱符号的解析规则。另外，层序中包含了一个数值转换过程，学生需要根据反汇编代码确定其修改规则，并根据修改规则进行“反制”。

**实验阶段3**

要求：修改二进制可重定位目标文件“phase3.o”的代码节内容，使其与main.o链接后能够运行输出（且仅输出）自己的学号：

$ gcc -o linkbomb main.o phase3.o -no-pie

$ ./linkbomb

学号

实验提示：

检查反汇编代码，定位模块中的各组成函数并推断其功能作用。 根据反汇编程序的执行逻辑，修改函数中的机器指令（用自己指令替换函数体中的nop指令）以实现期望的输出。

为了实现输出功能，自行编写获得的二进制程序（可以通过编写汇编代码然后使用gcc -c命令的方式实现）可以“借用”其他函数中的“有用代码或数据”，比如输出函数和数据引用等具体部分。

**实验阶段4**

要求：修改二进制可重定位目标文件“phase4.o”中重定位节和数据节中的内容，使其与main.o链接后能够运行输出（且仅输出）自己的学号：

$ gcc -o linkbomb main.o phase4.o -no-pie

$ ./linkbomb

学号

实验提示：

本阶段学生所拿到的.o文件中的“重定位位置”信息已经被抹除，学生需要根据实际情况确认冲重定位的发生位置，并根据重定位类型对位置信息进行恢复。若程序未能够正确修改重定位位置，则典型问题表现为段错误segmentation fault。此外，还需要学生根据程序所用到的数据情况进行数据部分的二进制修改。

**实验阶段5**

要求：修改二进制可重定位目标文件“phase5.o”中重定位节和数据节的内容，使其与main.o链接后能够正确输出（且仅输出）自己学号：

$ gcc -o linkbomb main.o phase5.o -no-pie

$ ./linkbomb

学号

实验提示：如果实验中对缺失重定位信息的恢复不完整或不正确的话，链接生成linkbomb程序时可能不报错，但运行程序可能得到以下结果之一：

**学生实验结果提交**

将修改完成的各阶段模块（phase1.o, phase2\_patch.o, phase3.o, phase4.o, phase5.o）和未改动的main.o、phase2.o模块一起用tar工具打包并命名为“linklib学号.tar”，打包命令如下：

tar cvf 学号.tar main.o phase1.o phase2.o phase3\_patch.o phase3.o phase4.o phase5.o

**五、实验注意事项**

注意：TAR文件中一定不要包含任何目录结构，即在上述命令行中指定待打包的源文件时，不要包含目录前缀（因此在运行tar命令前应cd到.o文件所在目录，或者使用“-C”命令选项）

提交结果tar文件。