

**实验报告**

**实 验（五）**

题 目 Link Lab

链接

专 业 计算机类

学　　 号 1190200526

班　　 级 1903002

学 生 沈城有

指 导 教 师 郑贵滨

实 验 地 点 G709

实 验 日 期 2021.5.21

**计算机科学与技术学院**

**目 录**

[第1章 实验基本信息 - 3 -](#_Toc72505593)

[1.1 实验目的 - 3 -](#_Toc72505594)

[1.2 实验环境与工具 - 3 -](#_Toc72505595)

[1.2.1 硬件环境 - 3 -](#_Toc72505596)

[1.2.2 软件环境 - 3 -](#_Toc72505597)

[1.2.3 开发工具 - 3 -](#_Toc72505598)

[1.3 实验预习 - 3 -](#_Toc72505599)

[第2章 实验预习 - 4 -](#_Toc72505600)

[2.1 ELF文件格式解读 - 4 -](#_Toc72505601)

[2.2程序的内存映像结构 - 4 -](#_Toc72505602)

[2.3程序中符号的位置分析 - 4 -](#_Toc72505603)

[2.4程序运行过程分析 - 4 -](#_Toc72505604)

[第3章 各阶段的原理与方法 - 5 -](#_Toc72505605)

[3.1 阶段1的分析 - 5 -](#_Toc72505606)

[3.2 阶段2的分析 - 5 -](#_Toc72505607)

[3.3 阶段3的分析 - 5 -](#_Toc72505608)

[3.4 阶段4的分析 - 5 -](#_Toc72505609)

[3.5 阶段5的分析 - 5 -](#_Toc72505610)

[第4章 总结 - 6 -](#_Toc72505611)

[4.1 请总结本次实验的收获 - 6 -](#_Toc72505612)

[4.2 请给出对本次实验内容的建议 - 6 -](#_Toc72505613)

[参考文献 - 7 -](#_Toc72505614)

# 第1章 实验基本信息

## 1.1 实验目的

理解链接的作用与工作步骤；

掌握ELF结构、符号解析与重定位的工作过程；

熟练使用Linux工具完成ELF分析与修改。

## 1.2 实验环境与工具

### 1.2.1 硬件环境

X64 CPU；2GHz；2G RAM；256GHD Disk 以上

### 1.2.2 软件环境

Windows7 64位以上；VirtualBox/Vmware 11以上；Ubuntu 16.04 LTS 64位/优麒麟 64位

### 1.2.3 开发工具

Visual Studio 2010 64位以上；GDB/OBJDUMP；DDD/EDB等

## 1.3 实验预习

上实验课前，必须认真预习实验指导书（PPT或PDF）。

了解实验的目的、实验环境与软硬件工具、实验操作步骤，复习与实验有关的理论知识。

请按顺序写出ELF格式的可执行目标文件的各类信息。

请按照内存地址从低到高的顺序，写出Linux下X64内存映像。

请运行“LinkAddress -u 学号 姓名” 按地址顺序写出各符号的地址、空间。并按照Linux下X64内存映像结构，标出其所属各区。

gcc -m64 -o LinkAddress linkaddress.c

请按顺序写出LinkAddress从开始执行到main前/后执行的子程序的名字(gcc与objdump/GDB/EDB)。

# 第2章 实验预习

## 2.1 ELF文件格式解读

请按顺序写出ELF格式的可执行目标文件的各类信息（5分）

|  |
| --- |
| ELF头 |
| 段头部表 |
| .init |
| .text |
| .rodata |
| .data |
| .bss |
| .symtab |
| .debug |
| .line |
| .strtab |
| 节头部表 |

## 2.2程序的内存映像结构

请按照内存地址从低到高的顺序，写出Linux下X64内存映像（5分）

|  |
| --- |
| 内核内存 |
| 用户栈 |
| （运行时创建） |
| （栈向下生长） |
| … |
| （映射区域向上生长） |
| 共享库的内存映射区域 |
| … |
| （堆向上生长） |
| 运行时堆 |
| （由malloc创建） |
| 读/写段 |
| （.data, .bss） |
| 只读代码段 |
| （.init, .text, .rodata） |
| … |

注：内存地址从下向上增大

## 2.3程序中符号的位置分析

请运行“LinkAddress -u 学号 姓名” 按地址顺序写出各符号的地址，并按照Linux下X64内存映像标出其所属内存区段（5分）

|  |  |
| --- | --- |
| 内存区段 | 符号 地址（16进制） 地址（10进制） |
| 只读代码段  （.init, .text, .rodata） | show\_pointer 0x5573bfb16199 93955625673113  useless 0x5573bfb161d0 93955625673168  main 0x5573bfb161df 93955625673183 |
| 读/写段  （.data, .bss） | global 0x5573bfb1902c 93955625685036  huge array 0x5573bfb19040 93955625685056  big array 0x5573ffb19040 93956699426880 |
| 运行时堆  （由malloc创建） | p5 0x7f851a160010 140209645027344  p4 0x7f859a161010 140211792515088  p3 0x7f85da162010 140212866261008  p2 0x7f85da183010 140212866396176  p1 0x7f85da1a4010 140212866531344 |
| 共享库的内存映射区域 | exit 0x7f85ea1eebc0 140213135272896  printf 0x7f85ea209e10 140213135384080  malloc 0x7f85ea242260 140213135614560  free 0x7f85ea242850 140213135616080 |
| 用户栈  （运行时创建） | argc 0x7ffdb4e26b1c 140727638190876  local 0x7ffdb4e26b20 140727638190880  argv 0x7ffdb4e26c58 140727638191192  env 0x7ffdb4e26c80 140727638191232  argv[0] 0x7ffdb4e2888c 140727638198412  ./Linkaddress  argv[1] 0x7ffdb4e2889a 140727638198426  -u  argv[2] 0x7ffdb4e2889d 140727638198429  1190200526  argv[3] 0x7ffdb4e288a8 140727638198440  沈城有  env[0] \*env 0x7ffdb4e288b2 140727638198450  SHELL=/bin/bash  env[1] \*env 0x7ffdb4e288c2 140727638198466  SESSION\_MANAGER=local/ubuntu-20-04:@/tmp/.ICE-unix/1225,unix/ubuntu-20-04:/tmp/.ICE-unix/1225  env[2] \*env 0x7ffdb4e28920 140727638198560  QT\_ACCESSIBILITY=1  env[3] \*env 0x7ffdb4e28933 140727638198579  COLORTERM=truecolor  env[4] \*env 0x7ffdb4e28947 140727638198599  XDG\_CONFIG\_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg  env[5] \*env 0x7ffdb4e28974 140727638198644  XDG\_MENU\_PREFIX=gnome-  env[6] \*env 0x7ffdb4e2898b 140727638198667  GNOME\_DESKTOP\_SESSION\_ID=this-is-deprecated  env[7] \*env 0x7ffdb4e289b7 140727638198711  LANGUAGE=zh\_CN:en\_US:en  env[8] \*env 0x7ffdb4e289cf 140727638198735  LC\_ADDRESS=zh\_CN.UTF-8  env[9] \*env 0x7ffdb4e289e6 140727638198758  GNOME\_SHELL\_SESSION\_MODE=ubuntu  env[10] \*env 0x7ffdb4e28a06 140727638198790  LC\_NAME=zh\_CN.UTF-8  env[11] \*env 0x7ffdb4e28a1a 140727638198810  SSH\_AUTH\_SOCK=/run/user/1000/keyring/ssh  env[12] \*env 0x7ffdb4e28a43 140727638198851  XMODIFIERS=@im=ibus  env[13] \*env 0x7ffdb4e28a57 140727638198871  DESKTOP\_SESSION=ubuntu  env[14] \*env 0x7ffdb4e28a6e 140727638198894  LC\_MONETARY=zh\_CN.UTF-8  env[15] \*env 0x7ffdb4e28a86 140727638198918  SSH\_AGENT\_PID=1160  env[16] \*env 0x7ffdb4e28a99 140727638198937  GTK\_MODULES=gail:atk-bridge  env[17] \*env 0x7ffdb4e28ab5 140727638198965  DBUS\_STARTER\_BUS\_TYPE=session  【……篇幅原因，中间略去……】  env[55] \*env 0x7ffdb4e28fc3 140727638200259  LC\_NUMERIC=zh\_CN.UTF-8  env[56] \*env 0x7ffdb4e28fda 140727638200282  \_=./Linkaddress |

## 2.4程序运行过程分析

请按顺序写出LinkAddress从开始执行到main前/后执行的子程序的名字(使用gcc与objdump/GDB/EDB)（5分）

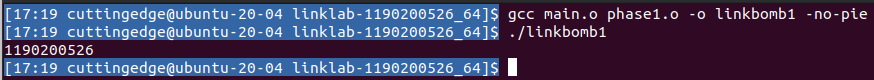
|  |  |
| --- | --- |
| 时间段 | 子程序名 |
| 开始执行至main函数前 | \_start()  \_\_libc\_start\_main@plt()  \_\_libc\_csu\_init()  \_init()  frame\_dummy()  register\_tm\_clones() |
| main函数执行过程中 | main()  puts@plt()  useless@plt()  showpointer@plt()  malloc@plt()  free@plt() |
| main函数后 | \_\_do\_global\_dtors\_aux()  deregister\_tm\_clones()  \_libc\_csu\_fini()  \_fini() |

# 第3章 各阶段的原理与方法

每阶段40分，phasex.o 20分，分析20分，总分不超过80分

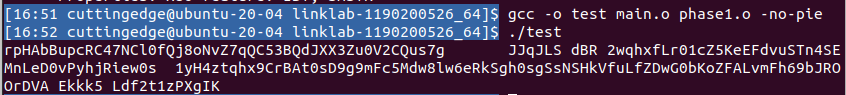
## 3.1 阶段1的分析

程序运行结果截图：

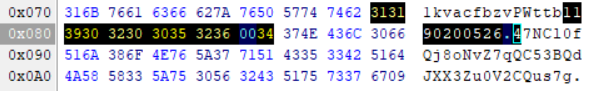


分析与设计的过程：

首先将main.o与未修改的phase1.o链接生成可执行文件，运行可执行文件得到以下输出（目的在于确定打印字符串的起始位置）：



使用HEXEdit工具，找到起始位置，并将字符串修改为学号并添加结尾的’\0’（即16进制00），如下图：



保存修改，重新链接main.o与phase1.o生成linkbomb1并运行，完成。

## 3.2 阶段2的分析

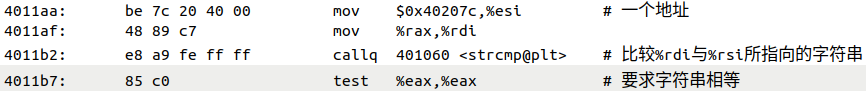
程序运行结果截图：



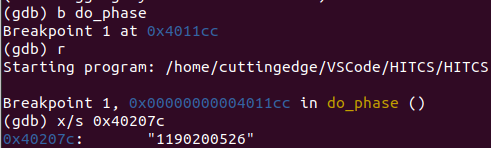
分析与设计的过程：

首先链接main.o与phase2.o生成可执行文件，并使用objdump查看其反汇编代码。

主要关注实验指导中提到的随机名称函数（XiyfHieL()，起始地址0x401196）部分，如下图：

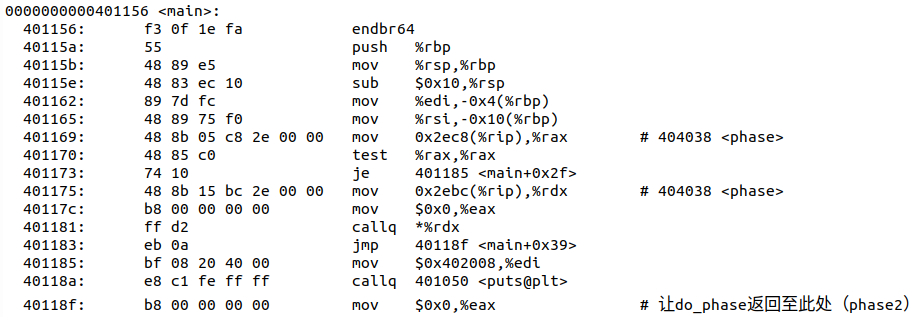


注意到地址0x40207c，使用GDB检查发现为学号字符串，过程如下图：



为使字符串比较检查通过，编写汇编代码并使用gcc和objdump转化为机器代码添加至phase2.o的do\_phase函数中。

下图为main函数返回位置，需要在汇编代码中使用：



汇编代码如下：

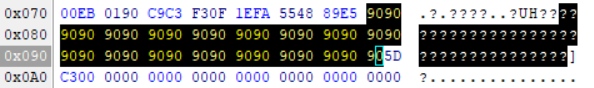
movq $0x40207c, %rdi # 让两个寄存器指向同一字符串，通过比较

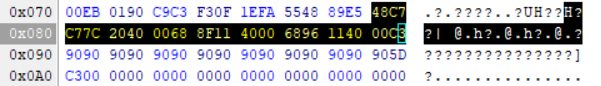
pushq $0x40118f # main函数地址压栈，避免段错误

pushq $0x401196 # 输出函数地址，用于跳转至输出函数

ret # 转移控制

使用HEXEdit工具，修改phase2.o中do\_phase函数的nop部分为以上汇编代码对应的机器代码，过程如下图（第一张为原图，第二张为修改后的）：





将修改后的phase2.o与main.o再次链接，生成可执行文件linkbomb2，执行输出学号。

## 3.3 阶段3的分析

程序运行结果截图：

分析与设计的过程：

## 3.4 阶段4的分析

程序运行结果截图：

分析与设计的过程：

## 3.5 阶段5的分析

程序运行结果截图：

分析与设计的过程：

# 第4章 总结

## 4.1 请总结本次实验的收获

## 4.2 请给出对本次实验内容的建议

注：本章为酌情加分项。

# 参考文献

**为完成本次实验你翻阅的书籍与网站等**

[1] 林来兴. 空间控制技术[M]. 北京：中国宇航出版社，1992：25-42.

[2] 辛希孟. 信息技术与信息服务国际研讨会论文集：A集[C]. 北京：中国科学出版社，1999.

[3] 赵耀东. 新时代的工业工程师[M/OL]. 台北：天下文化出版社，1998 [1998-09-26]. http://www.ie.nthu.edu.tw/info/ie.newie.htm（Big5）.

[4] 谌颖. 空间交会控制理论与方法研究[D]. 哈尔滨：哈尔滨工业大学，1992：8-13.

[5] KANAMORI H. Shaking Without Quaking[J]. Science，1998，279（5359）：2063-2064.

[6] CHRISTINE M. Plant Physiology: Plant Biology in the Genome Era[J/OL]. Science，1998，281：331-332[1998-09-23]. http://www.sciencemag.org/cgi/ collection/anatmorp.