

**实验报告**

**实 验（五）**

题 目 LinkLab

链接

专 业 计算机类

学　　 号 1190202405

班　　 级 1903005

学 生 林逸灏

指 导 教 师 史先俊

实 验 地 点 G712

实 验 日 期 2021.05.19

**计算机科学与技术学院**

**目 录**

[第1章 实验基本信息 - 3 -](#_Toc8233)

[1.1 实验目的 - 3 -](#_Toc26186)

[1.2 实验环境与工具 - 3 -](#_Toc20826)

[1.2.1 硬件环境 - 3 -](#_Toc23554)

[1.2.2 软件环境 - 3 -](#_Toc3063)

[1.2.3 开发工具 - 3 -](#_Toc12797)

[1.3 实验预习 - 3 -](#_Toc27944)

[第2章 实验预习 - 4 -](#_Toc1828)

[2.1 请按顺序写出ELF格式的可执行目标文件的各类信息（5分） - 4 -](#_Toc7460)

[2.2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） - 4 -](#_Toc7161)

[2.3请运行“LinkAddress -u 学号 姓名” 按地址循序写出各符号的地址、空间。并按照Linux下X64内存映像标出其所属各区。 - 4 -](#_Toc22250)

[（5分） - 4 -](#_Toc20986)

[2.4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)（5分） - 4 -](#_Toc9870)

[第3章 各阶段的原理与方法 - 5 -](#_Toc15155)

[3.1 阶段1的分析 - 5 -](#_Toc23314)

[3.2 阶段2的分析 - 5 -](#_Toc25402)

[3.3 阶段3的分析 - 5 -](#_Toc20455)

[3.4 阶段4的分析 - 5 -](#_Toc1318)

[3.5 阶段5的分析 - 5 -](#_Toc4995)

[第4章 总结 - 6 -](#_Toc15951)

[4.1 请总结本次实验的收获 - 6 -](#_Toc23480)

[4.2 请给出对本次实验内容的建议 - 6 -](#_Toc3014)

[参考文献 - 7 -](#_Toc32173)

# 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结构与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

填写

# 第2章 实验预习

## 2.1 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：每个ELF文件都必须存在一个ELF\_Header,这里存放了很多重要的信息用来描述整个文件的组织,如: 版本信息,入口信息,偏移信息等。程序执行也必须依靠其提供的信息。

段头部表：结构数组，将连续的文件映射到运行时的内存段

.init：定义了\_init函数，该函数用来执行可执行目标文件开始执行时的初始化工作

.text：已编译程序的机器代码

.rodata：只读数据，比如printf语句中的格式串和开关语句的跳转表

. data ： 已初始化的全局和静态C变量

. bss ：    未初始化的全局和静态C变量

. symtab ：一个符号表，它存放在程序中定义和引用的函数和全局变量的信息

. debug ： 一个调试符号表，其条目时程序中定义的全局变量和类型定义，程序中定义和引用的全局变量，以及原始的C源文件。

. line ：   原始C源程序的行号和.text节中机器指令之间的映射

. strtab ： 一个字符串表，其内容包括 .symtab 和 .debug节中的符号表，以及节头部中的节名字。

节头部表：描述目标文件的节

## 2.2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）

图示

描述已自动生成

## 2.3请运行“LinkAddress -u 学号 姓名” 按地址循序写出各符号的地址、空间。并按照Linux下X64内存映像标出其所属各区。

## （5分）

|  |  |
| --- | --- |
| 所属区 | 各符号的地址、空间、（地址从小到大） |
| 只读代码段（.init,.text,.redata） | show\_pointer 0x401205 4198917  useless 0x4011f6 4198902  main 0x40123a 4198970  exit 0x401100 4198656  printf 0x4010e0 4198624  malloc 0x4010f0 4198640  free 0x4010a0 4198560  strcpy 0x4010b0 4198576 |
| 读写段（.data,.bss） | big array 0x404140 4211008  huge array 0x1404140 20988224  global 0x404080 4210816  gint0 0x41404140 1094730048  glong 0x404088 4210824  cstr 0x4040a0 4210848  pstr 0x402020 4202528  gc 0x40204c 4202572  cc 0x402060 4202592  local int 0 0x7ffef0c7986c 140732938033260  local int 1 0x7ffef0c79870 140732938033264  local static int 0 0x41404144 1094730052  local static int 1 0x404110 4210960  local astr 0x7ffef0c798b0 140732938033328  local pstr 0x4020d0 4202704 |
| 运行时堆（由malloc创建） | p1 0x7f2128506010 139780387004432  p2 0x428356b0 1115903664  p3 0x7f21284e5010 139780386869264  p4 0x7f20e84e4010 139779313123344  p5 0x7f20684e3010 139777165635600 |
| 用户栈（运行时创建） | argc 0x7ffef0c7985c 140732938033244  argv 0x7ffef0c79da8 140732938034600  argv[0] 7ffef0c7a36a  argv[1] 7ffef0c7a372  argv[2] 7ffef0c7a375  argv[3] 7ffef0c7a380  argv[0] 0x7ffef0c7a36a 140732938036074  ./a.out  argv[1] 0x7ffef0c7a372 140732938036082  -u  argv[2] 0x7ffef0c7a375 140732938036085  1190202405  argv[3] 0x7ffef0c7a380 140732938036096  林逸灏  env 0x7ffef0c79dd0 140732938034640  env[0] \*env 0x7ffef0c7a38a 140732938036106  SHELL=/bin/bash  env[1] \*env 0x7ffef0c7a39a 140732938036122  SESSION\_MANAGER=local/ubuntu:@/tmp/.ICE-unix/1544,unix/ubuntu:/tmp/.ICE-unix/1544  env[2] \*env 0x7ffef0c7a3ec 140732938036204  QT\_ACCESSIBILITY=1  env[3] \*env 0x7ffef0c7a3ff 140732938036223  COLORTERM=truecolor  env[4] \*env 0x7ffef0c7a413 140732938036243  XDG\_CONFIG\_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg  env[5] \*env 0x7ffef0c7a440 140732938036288  XDG\_MENU\_PREFIX=gnome-  env[6] \*env 0x7ffef0c7a457 140732938036311  GNOME\_DESKTOP\_SESSION\_ID=this-is-deprecated  env[7] \*env 0x7ffef0c7a483 140732938036355  GTK\_IM\_MODULE=fcitx  env[8] \*env 0x7ffef0c7a497 140732938036375  QT4\_IM\_MODULE=fcitx  env[9] \*env 0x7ffef0c7a4ab 140732938036395  LC\_ADDRESS=zh\_CN.UTF-8  env[10] \*env 0x7ffef0c7a4c2 140732938036418  GNOME\_SHELL\_SESSION\_MODE=ubuntu  env[11] \*env 0x7ffef0c7a4e2 140732938036450  LC\_NAME=zh\_CN.UTF-8  env[12] \*env 0x7ffef0c7a4f6 140732938036470  SSH\_AUTH\_SOCK=/run/user/1000/keyring/ssh  env[13] \*env 0x7ffef0c7a51f 140732938036511  XMODIFIERS=@im=fcitx  env[14] \*env 0x7ffef0c7a534 140732938036532  DESKTOP\_SESSION=ubuntu  env[15] \*env 0x7ffef0c7a54b 140732938036555  LC\_MONETARY=zh\_CN.UTF-8  env[16] \*env 0x7ffef0c7a563 140732938036579  PrlDRIImportFD=10  env[17] \*env 0x7ffef0c7a575 140732938036597  SSH\_AGENT\_PID=1471  env[18] \*env 0x7ffef0c7a588 140732938036616  GTK\_MODULES=gail:atk-bridge  env[19] \*env 0x7ffef0c7a5a4 140732938036644  PWD=/home/lyhhao/Desktop  env[20] \*env 0x7ffef0c7a5bd 140732938036669  LOGNAME=lyhhao  env[21] \*env 0x7ffef0c7a5cc 140732938036684  XDG\_SESSION\_DESKTOP=ubuntu  env[22] \*env 0x7ffef0c7a5e7 140732938036711  XDG\_SESSION\_TYPE=x11  env[23] \*env 0x7ffef0c7a5fc 140732938036732  GPG\_AGENT\_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1  env[24] \*env 0x7ffef0c7a630 140732938036784  XAUTHORITY=/run/user/1000/gdm/Xauthority  env[25] \*env 0x7ffef0c7a659 140732938036825  GJS\_DEBUG\_TOPICS=JS ERROR;JS LOG  env[26] \*env 0x7ffef0c7a67a 140732938036858  WINDOWPATH=2  env[27] \*env 0x7ffef0c7a687 140732938036871  HOME=/home/lyhhao  env[28] \*env 0x7ffef0c7a699 140732938036889  USERNAME=lyhhao  env[29] \*env 0x7ffef0c7a6a9 140732938036905  IM\_CONFIG\_PHASE=1  env[30] \*env 0x7ffef0c7a6bb 140732938036923  LC\_PAPER=zh\_CN.UTF-8  env[31] \*env 0x7ffef0c7a6d0 140732938036944  LANG=en\_US.UTF-8  env[32] \*env 0x7ffef0c7a6e1 140732938036961  LS\_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:\*.tar=01;31:\*.tgz=01;31:\*.arc=01;31:\*.arj=01;31:\*.taz=01;31:\*.lha=01;31:\*.lz4=01;31:\*.lzh=01;31:\*.lzma=01;31:\*.tlz=01;31:\*.txz=01;31:\*.tzo=01;31:\*.t7z=01;31:\*.zip=01;31:\*.z=01;31:\*.dz=01;31:\*.gz=01;31:\*.lrz=01;31:\*.lz=01;31:\*.lzo=01;31:\*.xz=01;31:\*.zst=01;31:\*.tzst=01;31:\*.bz2=01;31:\*.bz=01;31:\*.tbz=01;31:\*.tbz2=01;31:\*.tz=01;31:\*.deb=01;31:\*.rpm=01;31:\*.jar=01;31:\*.war=01;31:\*.ear=01;31:\*.sar=01;31:\*.rar=01;31:\*.alz=01;31:\*.ace=01;31:\*.zoo=01;31:\*.cpio=01;31:\*.7z=01;31:\*.rz=01;31:\*.cab=01;31:\*.wim=01;31:\*.swm=01;31:\*.dwm=01;31:\*.esd=01;31:\*.jpg=01;35:\*.jpeg=01;35:\*.mjpg=01;35:\*.mjpeg=01;35:\*.gif=01;35:\*.bmp=01;35:\*.pbm=01;35:\*.pgm=01;35:\*.ppm=01;35:\*.tga=01;35:\*.xbm=01;35:\*.xpm=01;35:\*.tif=01;35:\*.tiff=01;35:\*.png=01;35:\*.svg=01;35:\*.svgz=01;35:\*.mng=01;35:\*.pcx=01;35:\*.mov=01;35:\*.mpg=01;35:\*.mpeg=01;35:\*.m2v=01;35:\*.mkv=01;35:\*.webm=01;35:\*.webp=01;35:\*.ogm=01;35:\*.mp4=01;35:\*.m4v=01;35:\*.mp4v=01;35:\*.vob=01;35:\*.qt=01;35:\*.nuv=01;35:\*.wmv=01;35:\*.asf=01;35:\*.rm=01;35:\*.rmvb=01;35:\*.flc=01;35:\*.avi=01;35:\*.fli=01;35:\*.flv=01;35:\*.gl=01;35:\*.dl=01;35:\*.xcf=01;35:\*.xwd=01;35:\*.yuv=01;35:\*.cgm=01;35:\*.emf=01;35:\*.ogv=01;35:\*.ogx=01;35:\*.aac=00;36:\*.au=00;36:\*.flac=00;36:\*.m4a=00;36:\*.mid=00;36:\*.midi=00;36:\*.mka=00;36:\*.mp3=00;36:\*.mpc=00;36:\*.ogg=00;36:\*.ra=00;36:\*.wav=00;36:\*.oga=00;36:\*.opus=00;36:\*.spx=00;36:\*.xspf=00;36:  env[33] \*env 0x7ffef0c7acd0 140732938038480  XDG\_CURRENT\_DESKTOP=ubuntu:GNOME  env[34] \*env 0x7ffef0c7acf1 140732938038513  VTE\_VERSION=6200  env[35] \*env 0x7ffef0c7ad02 140732938038530  GNOME\_TERMINAL\_SCREEN=/org/gnome/Terminal/screen/10e2e303\_3f80\_4d64\_92c1\_d50ab92be88c  env[36] \*env 0x7ffef0c7ad58 140732938038616  CLUTTER\_IM\_MODULE=fcitx  env[37] \*env 0x7ffef0c7ad70 140732938038640  GJS\_DEBUG\_OUTPUT=stderr  env[38] \*env 0x7ffef0c7ad88 140732938038664  LESSCLOSE=/usr/bin/lesspipe %s %s  env[39] \*env 0x7ffef0c7adaa 140732938038698  XDG\_SESSION\_CLASS=user  env[40] \*env 0x7ffef0c7adc1 140732938038721  TERM=xterm-256color  env[41] \*env 0x7ffef0c7add5 140732938038741  LC\_IDENTIFICATION=zh\_CN.UTF-8  env[42] \*env 0x7ffef0c7adf3 140732938038771  LESSOPEN=| /usr/bin/lesspipe %s  env[43] \*env 0x7ffef0c7ae13 140732938038803  USER=lyhhao  env[44] \*env 0x7ffef0c7ae1f 140732938038815  GNOME\_TERMINAL\_SERVICE=:1.115  env[45] \*env 0x7ffef0c7ae3d 140732938038845  DISPLAY=:0  env[46] \*env 0x7ffef0c7ae48 140732938038856  SHLVL=1  env[47] \*env 0x7ffef0c7ae50 140732938038864  LC\_TELEPHONE=zh\_CN.UTF-8  env[48] \*env 0x7ffef0c7ae69 140732938038889  QT\_IM\_MODULE=fcitx  env[49] \*env 0x7ffef0c7ae7c 140732938038908  LC\_MEASUREMENT=zh\_CN.UTF-8  env[50] \*env 0x7ffef0c7ae97 140732938038935  XDG\_RUNTIME\_DIR=/run/user/1000  env[51] \*env 0x7ffef0c7aeb6 140732938038966  LC\_TIME=zh\_CN.UTF-8  env[52] \*env 0x7ffef0c7aeca 140732938038986  XDG\_DATA\_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop  env[53] \*env 0x7ffef0c7af1f 140732938039071  PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin  env[54] \*env 0x7ffef0c7af87 140732938039175  GDMSESSION=ubuntu  env[55] \*env 0x7ffef0c7af99 140732938039193  DBUS\_SESSION\_BUS\_ADDRESS=unix:path=/run/user/1000/bus  env[56] \*env 0x7ffef0c7afcf 140732938039247  LC\_NUMERIC=zh\_CN.UTF-8  env[57] \*env 0x7ffef0c7afe6 140732938039270  \_=./a.out |

## 2.4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)（5分）

main执行前：

<\_init>:

<.plt>

<puts@plt>

<\_\_stack\_chk\_fail@plt>

<\_\_printf\_chk@plt>

<free@plt>

<malloc@plt>

<\_\_cxa\_finalize@plt>

<\_start>

<deregister\_tm\_clones>

<register\_tm\_clones>

<\_\_do\_global\_dtors\_aux>

<frame\_dummy>

<useless>

<show\_pointer>

main执行后：

<main>

<\_\_libc\_csu\_init>

<\_\_libc\_csu\_fini>

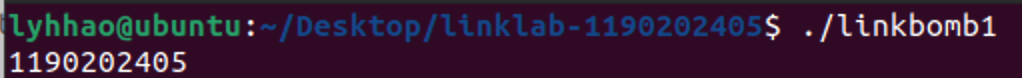
<\_fini>

# 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

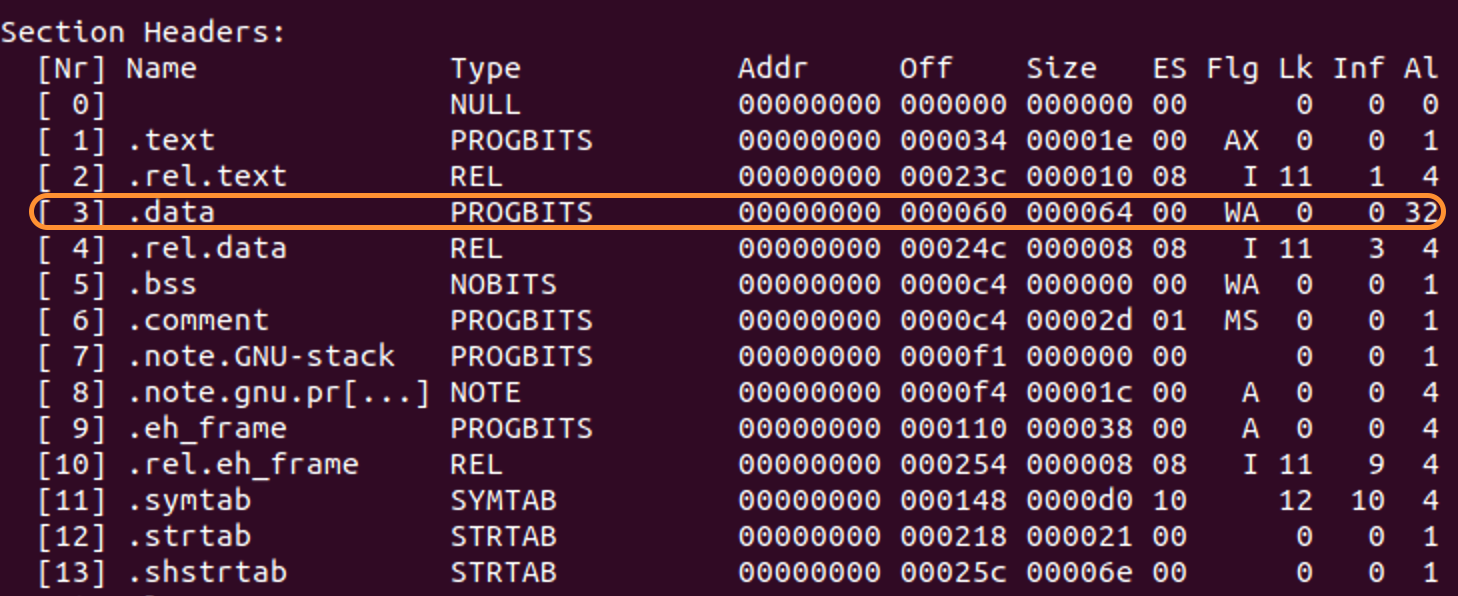
## 3.1 阶段1的分析

程序运行结果截图：

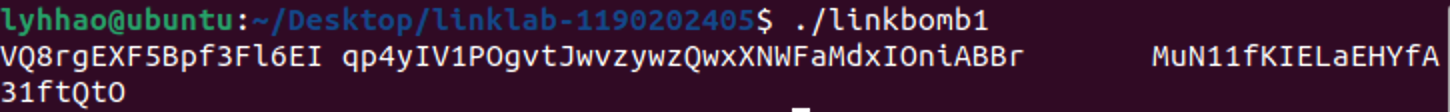


分析与设计的过程：

1.使用readelf -a phase1.o 命令查看elf文件的内容，可以看到字符串输出的起始地址在.data节中的偏移量为32



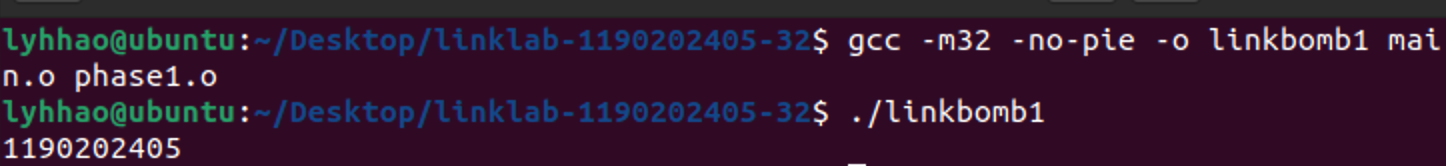
2. 使用命令 gcc -m32 -o linkbomb1 main.o phase1.o 将 main.o 和 phase.o 链接成 linkbomb1.o，然后运行 linkbomb1 程序，输出字符为VQ8rgEXF5Bpf3Fl6EI qp4yIV1POgvtJwvzywzQwxXNWFaMdxIOniABBr MuN11fKIELaEHYfA31ftQtO



3.使用hexedit工具进入phase1.o，用学号替换掉本应该输出的.data节里的字符串，用学号1190202405对应的ascii码31 31 39 30 32 30 32 34 30 35替换，多余的位用00作为字符串的结束，再进行链接，输出得到1190202405。表格

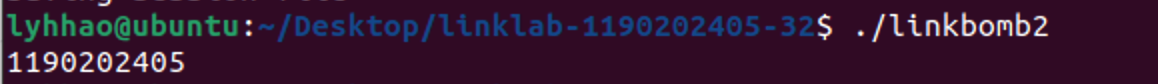
中度可信度描述已自动生成

4.编译输出



## 3.2 阶段2的分析

程序运行结果截图：



分析与设计的过程：

1.使用 gcc -m32 -o linkbomb2 main.o phase2.o 得到链接可执行文件linkbomb2，使用objdump进行反汇编，将反汇编代码存入一个文件中，便于查看。在反汇编代码中查看FqYGvbWI函数与do\_phase函数。

文本

描述已自动生成

表格

描述已自动生成

2.观察反汇编代码可知，我们需要补全do\_phase中的内容使输出结果为学号。链接器将FqYGvbWI内对strcmp，puts两个函数的调用进行了重定位。想要在屏幕中输出学号字符串的方式只有调用FqYGvbWI中的puts函数，因此需要在在do\_phase函数里调用上面这个FqYGvbWI函数。观察汇编代码，可知，该函数有两个参数，均为指针。函数体内需要做一次字符相等判断才能够调用puts函数输出学号字符串。

3.在函数FqYGvbWI中push $0x804a07c 查询该地址所存的值，可知该地址存储了学号1190202405的ascii码值。

蓝色的标志

描述已自动生成

根据PPT中的程序框架，可知，FqYGvbWI函数中通过比较输入的参数与内存地址0x804a07c的值进行比较，若相等，则输出id字符串，因此，我们需要在do\_phase中调用FqYGvbWI函数，且参数为”1190202405”。

图形用户界面, 文本, 应用程序

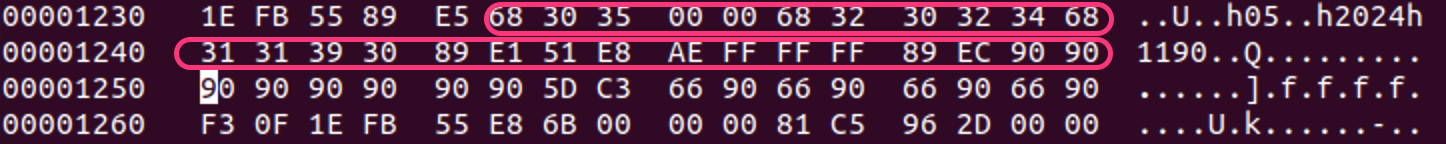
描述已自动生成

4.因此，我们在edb中编写汇编代码，可以知道所编写的汇编代码的机器语言。

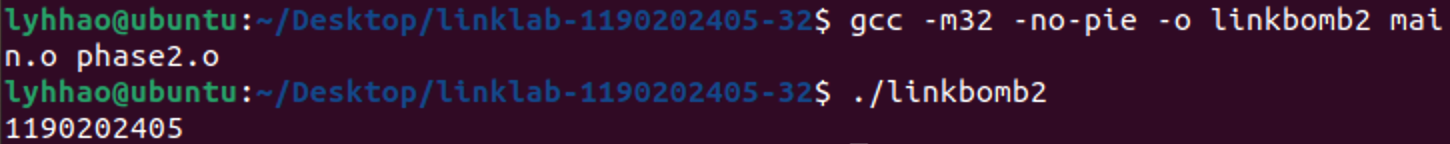
表格

描述已自动生成

5.将机器语言通过hexedit写入phase2.o中。



6.编译运行linkbomb2，输出正确结果。



## 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

经过本次实验，我对栈帧、链接、EDB等有了更加深入的了解。

## 4.2 请给出对本次实验内容的建议

无

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.