哈爾濱工業大學

实验报告

实 验（五）

题　　目 LinkLab

链接

专　　业 计算机

学　　号 1190202105

班　　级 1903002

学　　生 傅浩东

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

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

Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz；

16GB RAM；

1 TB SSD

1.2.2 软件环境

Windows 10 21H1；VirtualBox；Ubuntu 20.04 LTS

1.2.3 开发工具

EDB；GDB；CodeBlocks；vi/vim/gpedit+gcc；VSCode

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。

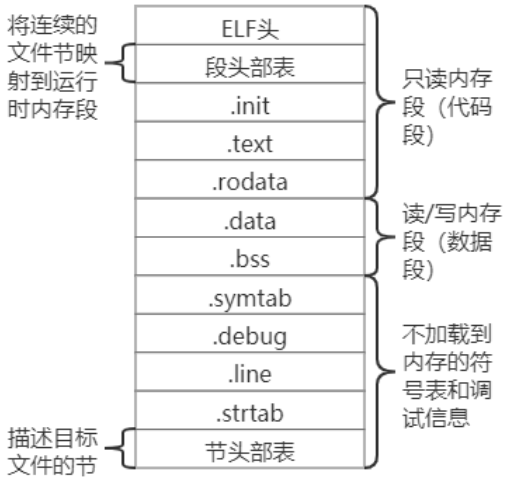
- gcc -m64 -o LinkAddress linkaddress.c

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：字段 e_entry 给出执行程序时第一条指令的地址

段头部表：将连续的文件映射到运行时的内存段

. init：定义了_init 函数，程序初始化代码会调用它

. 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

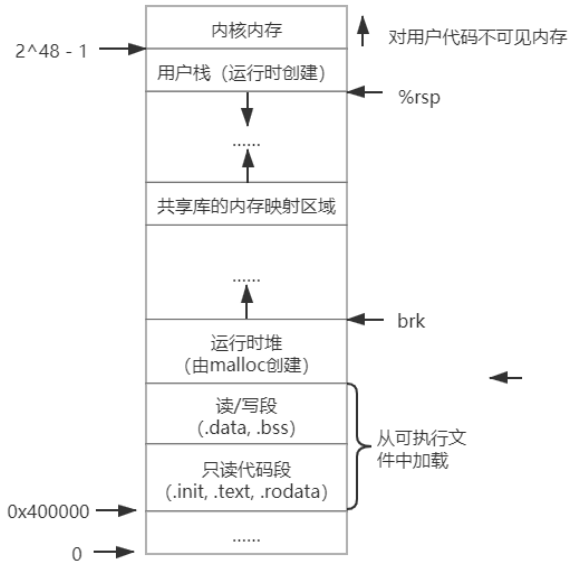
. line：原始 C 源程序的行号和.text 节中机器指令之间的映射

. 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 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）



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）

用户栈	nv 0x7ffd6b78beb0 140726406528688 env[0] *env 0x7ffd6b78c2d9 140726406529753 SHELL=/bin/bash env[1] *env 0x7ffd6b78c2e9 140726406529769 SESSION_MANAGER=local/ifu-VirtualBox:@/tmp/.ICE-unix/2048,unix/ifu-VirtualBox:/tmp/.ICE-unix/2048 env[2] *env 0x7ffd6b78c34b 140726406529867 QT_ACCESSIBILITY=1 env[3] *env 0x7ffd6b78c35e 140726406529886 COLORTERM=truecolor env[4] *env 0x7ffd6b78c372 140726406529906 XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg env[5] *env 0x7ffd6b78c39f 140726406529951 XDG_MENU_PREFIX=gnome- env[6] *env 0x7ffd6b78c3b6 140726406529974 GNOME_DESKTOP_SESSION_ID=this-is-deprecated env[7] *env 0x7ffd6b78c3e2 140726406530018 LANGUAGE=en_GB:en env[8] *env 0x7ffd6b78c3f4 140726406530036

	LC_ADDRESS=zh_CN.UTF-8 env[9] *env 0x7ffd6b78c40b 140726406530059 GNOME_SHELL_SESSION_MODE=ubuntu env[10] *env 0x7ffd6b78c42b 140726406530091 LC_NAME=zh_CN.UTF-8 env[11] *env 0x7ffd6b78c43f 140726406530111 SSH_AUTH_SOCK=/run/user/1000/keyring/ssh env[12] *env 0x7ffd6b78c468 140726406530152 XMODIFIERS=@im=ibus env[13] *env 0x7ffd6b78c47c 140726406530172 DESKTOP_SESSION=ubuntu env[14] *env 0x7ffd6b78c493 140726406530195 LC_MONETARY=zh_CN.UTF-8 env[15] *env 0x7ffd6b78c4ab 140726406530219 SSH_AGENT_PID=1864 env[16] *env 0x7ffd6b78c4be 140726406530238 GTK_MODULES=gail:atk-bridge env[17] *env 0x7ffd6b78c4da 140726406530266 DBUS_STARTER_BUS_TYPE=session env[18] *env 0x7ffd6b78c4f8 140726406530296 PWD=/mnt/Linux_Share/Lab5 env[19] *env 0x7ffd6b78c512 140726406530322 LOGNAME=ifu env[20] *env 0x7ffd6b78c51e 140726406530334 XDG_SESSION_DESKTOP=ubuntu env[21] *env 0x7ffd6b78c539 140726406530361 XDG_SESSION_TYPE=x11 env[22] *env 0x7ffd6b78c54e 140726406530382 GPG_AGENT_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1 env[23] *env 0x7ffd6b78c582 140726406530434 XAUTHORITY=/run/user/1000/gdm/Xauthority env[24] *env 0x7ffd6b78c5ab 140726406530475 WINDOWPATH=2 env[25] *env 0x7ffd6b78c5b8 140726406530488 HOME=/home/ifu env[26] *env 0x7ffd6b78c5c7 140726406530503 USERNAME=ifu env[27] *env 0x7ffd6b78c5d4 140726406530516 IM_CONFIG_PHASE=1 env[28] *env 0x7ffd6b78c5e6 140726406530534 LC_PAPER=zh_CN.UTF-8 env[29] *env 0x7ffd6b78c5fb 140726406530555 LANG=en_GB.UTF-8 env[30] *env 0x7ffd6b78c60c 140726406530572 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:c

d=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:*.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[31] *env 0x7ffd6b78cbee 140726406532078
XDG_CURRENT_DESKTOP=ubuntu:GNOME
env[32] *env 0x7ffd6b78cc0f 140726406532111
VTE_VERSION=6003
env[33] *env 0x7ffd6b78cc20 140726406532128
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/c68c6a5a_e6a5_4eb9_b2d2_8235188ac70f
env[34] *env 0x7ffd6b78cc76 140726406532214
INVOCATION_ID=25a1d7add6654ed0b53cec7f3d2f69b6
env[35] *env 0x7ffd6b78cca5 140726406532261
MANAGERPID=1408
env[36] *env 0x7ffd6b78ccb5 140726406532277
LESSCLOSE=/usr/bin/lesspipe %s %s
env[37] *env 0x7ffd6b78ccd7 140726406532311
XDG_SESSION_CLASS=user
env[38] *env 0x7ffd6b78ccee 140726406532334
TERM=xterm-256color
env[39] *env 0x7ffd6b78cd02 140726406532354
LC_IDENTIFICATION=zh_CN.UTF-8
env[40] *env 0x7ffd6b78cd20 140726406532384
LESSOPEN=| /usr/bin/lesspipe %s
env[41] *env 0x7ffd6b78cd40 140726406532416
USER=ifu
env[42] *env 0x7ffd6b78cd49 140726406532425
GNOME_TERMINAL_SERVICE=:1.80
env[43] *env 0x7ffd6b78cd66 140726406532454
DISPLAY=:0
env[44] *env 0x7ffd6b78cd71 140726406532465

SHLVL=1
env[45] *env 0x7ffd6b78cd79 140726406532473
LC_TELEPHONE=zh_CN.UTF-8
env[46] *env 0x7ffd6b78cd92 140726406532498
QT_IM_MODULE=ibus
env[47] *env 0x7ffd6b78cda4 140726406532516
LC_MEASUREMENT=zh_CN.UTF-8
env[48] *env 0x7ffd6b78cdbf 140726406532543
DBUS_STARTER_ADDRESS=unix:path=/run/user/1000/bus,guid=0a553ef00b6b541ef6ea5b3560b09f30
env[49] *env 0x7ffd6b78ce17 140726406532631
PAPERSIZE=a4
env[50] *env 0x7ffd6b78ce24 140726406532644
XDG_RUNTIME_DIR=/run/user/1000
env[51] *env 0x7ffd6b78ce43 140726406532675
LC_TIME=zh_CN.UTF-8
env[52] *env 0x7ffd6b78ce57 140726406532695
JOURNAL_STREAM=8:36424
env[53] *env 0x7ffd6b78ce6e 140726406532718
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
env[54] *env 0x7ffd6b78cec3 140726406532803
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
env[55] *env 0x7ffd6b78cf2b 140726406532907
GDMSESSION=ubuntu
env[56] *env 0x7ffd6b78cf3d 140726406532925
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus,guid=0a553ef00b6b541ef6ea5b3560b09f30
env[57] *env 0x7ffd6b78cf99 140726406533017
LC_NUMERIC=zh_CN.UTF-8
env[58] *env 0x7ffd6b78cfb0 140726406533040
OLDPWD=/mnt/Linux_Share/Lab5/linklab-1190202105
env[59] *env 0x7ffd6b78cfe0 140726406533088
_=./LinkAddr
local 0x7ffd6b78bd50 140726406528336
argc 0x7ffd6b78bd4c 140726406528332
argv 0x7ffd6b78be88 140726406528648
argv[0] 7ffd6b78c2b6
argv[1] 7ffd6b78c2c1
argv[2] 7ffd6b78c2c4
argv[3] 7ffd6b78c2cf
argv[0] 0x7ffd6b78c2b6 140726406529718
./LinkAddr
argv[1] 0x7ffd6b78c2c1 140726406529729
-u

	argv[2] 0x7ffd6b78c2c4 140726406529732 1190202105 argv[3] 0x7ffd6b78c2cf 140726406529743 傅浩东
共享库的内存映射区域	exit 0x7f56f43e0bc0 140011441621952 printf 0x7f56f43fbe10 140011441733136 malloc 0x7f56f4434260 140011441963616 free 0x7f56f4434850 140011441965136
运行时堆	p1 0x7f56e4396010 140011172880400 p3 0x7f56e4375010 140011172745232 p4 0x7f56a4374010 140010098999312 p5 0x7f5624373010 140007951511568
读/写段	big array 0x56411ffd3040 94837709549632 huge array 0x5640dffd3040 94836635807808 global 0x5640dffd302c 94836635807788 p2 0x564121a616b0 94837737395888
只读代码段	show_pointer 0x5640dfdd181a 94836633704474 useless 0x5640dfdd184d 94836633704525 main 0x5640dfdd1858 94836633704536

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）

main 执行之前	Ld-2.27.so!_dl_start 、 Ld-2.27.so!_dl_init Libc-2.27.so!_cxa_atexit Linkaddress!_init Linkaddress!_register_tm_clones Libc-2.27.so!_setjmp Libc2.27.so!__sigsetjmp Libc2.27.so!__sigjmpsave
main 执行之后	Linkaddress!puts@plt Linkaddress!useless@plt Linkaddress!showpointer@plt malloc Linkaddress!.plt Libc-2.27.so!exit

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：

```
ifu@ifu-VirtualBox:Lab5$ gcc -o linkbomb1 main.o phase1.o -no-pie
ifu@ifu-VirtualBox:Lab5$ ./linkbomb1
1190202105
ifu@ifu-VirtualBox:Lab5$
```

分析与设计的过程：

1. 链接 main.o 和 phase1.o 成 linkbomb1，运行查看输出字符串：

38AbQXAmjAjJKDhUZJZ6zg2wGsCCrGaDV5UY8id8ONmJ1prwX8pU04KoH Lta0pn PXstgSm4EZ40lsrVd7qXuZ0 5gbV gyuvjfO4X6D7kdof1Dacd2 qk7ds

命令行：gcc -o linkbomb1 main.o phase1.o

```
ifu@ifu-VirtualBox:Lab5$ gcc -o linkbomb1 main.o phase1.o -no-pie
ifu@ifu-VirtualBox:Lab5$ ./linkbomb1
VnEO1Zi1LKIDR91 SfgfN8j68BKwtOf7DGrDBflq0jU0B6qfTIXkSCNyyHFxvLBt
        Y       Jq8XhSXHwy89UT
ifu@ifu-VirtualBox:Lab5$
```

2. Hexedit.exe 编辑 phase1.o，用学号替换上述描述中的输出的字符串，我的学号 1190202105 的 ASCII 码为 31 31 39 30 32 30 32 31 30 35，多余的位全部由 00 占位，如下操作。

```
E800 0000 0090 5DC3 0000 0000 0000 0000  ?....?]?........
6C31 4F6F 725A 507A 6E50 5709 6C6F 3267  llOorZPznPW.lo2g
566E 454F 315A 6931 4C4B 4944 5239 3120  VnEO1Zi1LKIDR91
5366 6766 4E38 6A36 3842 4B77 744F 6637  SfgfN8j68BKwtOf7
4447 7244 4266 6C71 306A 5530 4236 7166  DGrDBflq0jU0B6qf
5449 586B 5343 4E79 7948 4678 764C 4274  TIXkSCNyyHFxvLBt
0959 094A 7138 5868 5358 4877 7938 3955  .Y.Jq8XhSXHwy89U
5400 0000 0000 0000 0000 0000 0000 0000  T...............
0047 4343 3A20 2855 6275 6E74 7520 392E  .GCC: (Ubuntu 9.
322E 312D 3975 6275 6E74 7532 2920 392E  2.1-9ubuntu2) 9.
```

```
E800 0000 0090 5DC3 0000 0000 0000 0000  ?....?]?........
6C31 4F6F 725A 507A 6E50 5709 6C6F 3267  l1OorZPznPW.lo2g
3131 3930 3230 3231 3035 0000 0000 0000  1190202105......
0000 0000 0000 0000 0000 0000 0000 0000  ................
0000 0000 0000 0000 0000 0000 0000 0000  ................
0000 0000 0000 0000 0000 0000 0000 0000  ................
0000 0000 0000 0000 0000 0000 0000 0000  ................
0000 0000 0000 0000 0000 0000 0000 0000  ................
0047 4343 3A20 2855 6275 6E74 7520 392E  .GCC: (Ubuntu 9.
322E 312D 3975 6275 6E74 7532 2920 392E  2.1-9ubuntu2) 9.
```

3. 重新编译链接，输出结果为 1190202105，与预期结果相符合。

3.2 阶段 2 的分析

程序运行结果截图：

```
ifu@ifu-VirtualBox:Lab5$ gcc -o linkbomb2 main.o phase2.o -no-pie
ifu@ifu-VirtualBox:Lab5$ ./linkbomb2
1190202105
ifu@ifu-VirtualBox:Lab5$
```

分析与设计的过程：

1. 使用 objdump 工具，分析 do_phase 函数和 MOSBQwWk 函数的反汇编指令。任务是修改 phase2.o，构造调用输出函数（通过相对 PC 的偏移量）的机器指令，并替换 do_phase 函数中预留的 nop 指令。

```
0000000000000000 <MOSBQwWk>:
   0:   f3 0f 1e fa             endbr64
   4:   55                      push   %rbp
   5:   48 89 e5                mov    %rsp,%rbp
   8:   48 83 ec 10             sub    $0x10,%rsp
   c:   48 89 7d f8             mov    %rdi,-0x8(%rbp)
  10:   48 8b 45 f8             mov    -0x8(%rbp),%rax
  14:   be 00 00 00 00          mov    $0x0,%esi
  19:   48 89 c7                mov    %rax,%rdi
  1c:   e8 00 00 00 00          callq  21 <MOSBQwWk+0x21>
  21:   85 c0                   test   %eax,%eax
  23:   75 0e                   jne    33 <MOSBQwWk+0x33>
  25:   48 8b 45 f8             mov    -0x8(%rbp),%rax
  29:   48 89 c7                mov    %rax,%rdi
  2c:   e8 00 00 00 00          callq  31 <MOSBQwWk+0x31>
  31:   eb 01                   jmp    34 <MOSBQwWk+0x34>
  33:   90                      nop
  34:   c9                      leaveq
  35:   c3                      retq

0000000000000036 <do_phase>:
  36:   f3 0f 1e fa             endbr64
  3a:   55                      push   %rbp
  3b:   48 89 e5                mov    %rsp,%rbp
  3e:   90                      nop
  3f:   90                      nop
  40:   90                      nop
```

2. 在.rodata 段找到了我的学号，链接 main.o 和 phase2.o 之后在地址 0x40207c 得到学号。因此，接下来只需要将该地址的内容作为函数参数，传入函数 MOSBQwWk。

```
Contents of section .rodata:
 0000 31313930 32303231 303500           1190202105.
```

```
Contents of section .rodata:
 402000 01000200 00000000 57656c63 6f6d6520  ........Welcome
 402010 746f2074 68697320 736d616c 6c206c61  to this small la
 402020 62206f66 206c696e 6b696e67 2e20546f  b of linking. To
 402030 20626567 696e206c 61622c20 706c6561   begin lab, plea
 402040 7365206c 696e6b20 74686520 72656c65  se link the rele
 402050 76616e74 206f626a 65637420 6d6f6475  vant object modu
 402060 6c652873 29207769 74682074 6865206d  le(s) with the m
 402070 61696e20 6d6f6475 6c652e00 31313930  ain module..1190
 402080 32303231 303500                      202105.
```

3. 将学号的地址作为参数传给 MOSBQwWk，代码为 mov $0x40207c,%rdi，对其编译和反汇编得到机器码 48 c7 c7 7c 20 40 00。

```
ifu@ifu-VirtualBox:Lab5$ gcc -c bomb2.s
ifu@ifu-VirtualBox:Lab5$ cat bomb2.s
mov $0x40207c,%rdi
ifu@ifu-VirtualBox:Lab5$ objdump -d bomb2.o

bomb2.o:     file format elf64-x86-64


Disassembly of section .text:

0000000000000000 <.text>:
   0:   48 c7 c7 7c 20 40 00    mov    $0x40207c,%rdi
ifu@ifu-VirtualBox:Lab5$
```

4. 将上述机器码替换最开始的 nop 代码，接下来 nop 从地址 0x 4011db 开始，通过相对 PC 的偏移量寻址使跳转到地址 0x 401196（调用 MOSBQwWk 函数）。相对寻址指令长五个字节，即从 0x 4011e0 减去 4a 到目的地址，因此 PC 相对地址差为 ff ff ff b6，填入代码 e8 b6 ff ff ff。

5. Hexedit.exe 编辑 phase2.o，用 48 c7 c7 7c 20 40 00 e8 b6 ff ff ff 代替 90.

```
00EB 0190 C9C3 F30F 1EFA 5548 89E5 48C7  .?.????..?UH??H?
C77C 2040 00E8 B6FF FFFF 9090 9090 9090  ?| @.??    ??????
9090 9090 9090 9090 9090 9090 9090 905D  ????????????????]
```

6. 重新编译链接，输出结果为 1190202105，与预期结果相符合。

3.3 阶段 3 的分析

程序运行结果截图：

```
ifu@ifu-VirtualBox:Lab5$ gcc -o linkbomb3 main.o phase3.o phase3_patch.o -no-pie
ifu@ifu-VirtualBox:Lab5$ ./linkbomb3
1190202105
ifu@ifu-VirtualBox:Lab5$
```

分析与设计的过程：

1. 链接 main.o 和 phase3.o 成 linkbomb3，利用 objdumnp 查看反汇编代码。分析 do_phase 函数反汇编指令，do_phase 会输出 10 个字符，COOKIE 值分别是：72 68 76 75 7a 6e 69 62 74 61.

```
00000000004011b6 <do_phase>:
  4011b6:  f3 0f 1e fa             endbr64
  4011ba:  55                      push   %rbp
  4011bb:  48 89 e5                mov    %rsp,%rbp
  4011be:  48 83 ec 20             sub    $0x20,%rsp
  4011c2:  64 48 8b 04 25 28 00            mov    %fs:0x28,%rax
  4011c9:  00 00
  4011cb:  48 89 45 f8             mov    %rax,-0x8(%rbp)
  4011cf:  31 c0                   xor    %eax,%eax
  4011d1:  48 b8 72 68 76 75 7a            movabs $0x62696e7a75766872,%rax
  4011d8:  6e 69 62
  4011db:  48 89 45 ed             mov    %rax,-0x13(%rbp)
  4011df:  66 c7 45 f5 74 61       movw   $0x6174,-0xb(%rbp)
  4011e5:  c6 45 f7 00             movb   $0x0,-0x9(%rbp)
  4011e9:  c7 45 e8 00 00 00 00            movl   $0x0,-0x18(%rbp)
  4011f0:  eb 24                   jmp    401216 <do_phase+0x60>
  4011f2:  8b 45 e8                mov    -0x18(%rbp),%eax
  4011f5:  48 98                   cltq
  4011f7:  0f b6 44 05 ed          movzbl -0x13(%rbp,%rax,1),%eax
  4011fc:  0f b6 c0                movzbl %al,%eax
  4011ff:  48 98                   cltq
  401201:  0f b6 80 80 40 40 00    movzbl 0x404080(%rax),%eax
  401208:  0f be c0                movsbl %al,%eax
  40120b:  89 c7                   mov    %eax,%edi
  40120d:  e8 4e fe ff ff          callq  401060 <putchar@plt>
  401212:  83 45 e8 01             addl   $0x1,-0x18(%rbp)
  401216:  8b 45 e8                mov    -0x18(%rbp),%eax
  401219:  83 f8 09                cmp    $0x9,%eax
  40121c:  76 d4                   jbe    4011f2 <do_phase+0x3c>
  40121e:  bf 0a 00 00 00          mov    $0xa,%edi
  401223:  e8 38 fe ff ff          callq  401060 <putchar@plt>
  401228:  90                      nop
  401229:  48 8b 45 f8             mov    -0x8(%rbp),%rax
  40122d:  64 48 33 04 25 28 00            xor    %fs:0x28,%rax
  401234:  00 00
  401236:  74 05                   je     40123d <do_phase+0x87>
  401238:  e8 43 fe ff ff          callq  401080 <__stack_chk_fail@plt>
  40123d:  c9                      leaveq
  40123e:  c3                      retq
  40123f:  90                      nop
```

2. 查看符号表，有唯一一个长度为 256 个字节的数组 imaXMLbBbt，起始地址为 0x 404080，与 do_phase 函数汇编指令相符合。

```
52: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND __stack_chk_fail@@GLIBC_2
53: 0000000000404080   256 OBJECT  GLOBAL DEFAULT   26 imaXMLbBbt
54: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND __libc_start_main@@GLIBC_
55: 0000000000404030     0 NOTYPE  GLOBAL DEFAULT   25 __data_start
```

3. 利用强弱全局符号的解析规则，在 patch 模块中定义同名且按输出要求正确初始化映射关系的数组变量——从而在链接时替换对原数组的引用。因此设计全局数组 imaXMLbBbt 的内容，要求部分位置有如下对映关系。

COOKIE	*114*	*104*	*118*	*117*	*122*	*110*	*105*	*98*	*116*	*97*
内容	1	1	9	0	2	0	2	1	0	5

```
ifu@ifu-VirtualBox:Lab5$ cat phase3_patch.c
char imaXMLbBbt[256]={
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', '5', '1', 'a', 'a', 'a', 'a', 'a', '1', '2', 'a', 'a', 'a', 'a', '0', 'a',
'a', 'a', '1', 'a', '0', '0', '9', 'a', 'a', 'a', '2', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a',
'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a'
};
```

4. 将 phase3_patch.c 编译得到 phase3_patch.o，再将生成的.c 文件与 main.o 和 phase3.o 三个文件链接在一起，运行 linkbomb3 与预期结果相符。

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

1. 更加熟悉 ELF 结构
2. 熟悉运用 readelf、objdump、hexedit 等工具

4.2 请给出对本次实验内容的建议

1. 希望说明 PPT 更加详细

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.