

《操作系统》课程

第三章 内存管理

授课教师：孙海龙

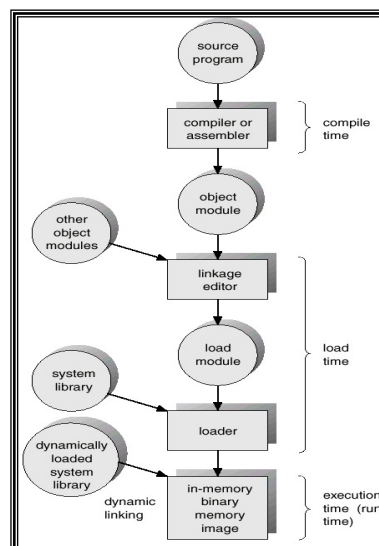
82339063, sunhl@buaa.edu.cn

2022年春季，北航计算机/软件学院

1

预备知识：程序的链接与装入

- 编译
- 链接
- 装入



2

program.c

```
int read_something (void);
int do_something (int);
void write_something (const char*);
int some_global_variable;
static int some_local_variable;

main () {
    int some_stack_variable;
    some_stack_variable = read_something ();
    some_global_variable = do_something (some_stack_variable);
    write_something ("I am done");
}
```

北京航空航天大学

计算机/软件学院

3

3

extras.c

```
#include <stdio.h>
extern int some_global_variable;
int read_something (void) {
    int res;
    scanf ("%d", &res);
    return res;
}
int do_something (int var) {
    return var + var;
}
void write_something (const char* str) {
    printf ("%s: %d\n", str, some_global_variable);
}
```

北京航空航天大学

计算机/软件学院

4

4

```

    int some_global_variable;
    globally visible by all modules (common, bss)

    static int some_local_variable;
    global/local: visible by all functions within the current module,
    but not outside the module (data)

    main () {
        int some_stack_variable;
        allocated on the stack, visible only within the block
        ...
    }

```

```

gcc program.c extras.c
./a.out

gcc -c program.c => program.o
gcc -c extras.c => extras.o
gcc program.o functions.o -o exe
./exe

```

gcc调用包含的几个工具

cc1: 预处理器和编译器

as: 汇编器

collect2: 链接器

```
gcc version 5.2.0 (crosstool-NG crosstool-ng-1.22.0)
COLLECT_GCC_OPTIONS='-o' 'exe' '-v' '-mabi=32' '-mllsc' '-mplt' '-mno-shared' '-EB'
/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/mips-malta-linux-gnu/5.2.0/cc1 -quiet -v -iprefix
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/ -isysroot
/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot program.c -meb -quiet -
dumpbase program.c -mabi=32 -mllsc -mplt -mno-shared -auxbase program -version -o /tmp/ccntenFQ.s
```

GNU C11 (crosstool-NG crosstool-ng-1.22.0) version 5.2.0 (mips-malta-linux-gnu)
compiled by GNU C version 4.6.3, GMP version 6.0.0, MPFR version 3.1.3, MPC version 1.0.3
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
ignoring duplicate directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/./lib/gcc/mips-malta-
linux-gnu/5.2.0/include"
ignoring nonexistent directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-
gnu/sysroot/home/wangluming/x-tools/mips-malta-linux-gnu/mips-malta-linux-gnu/sysroot/include"
ignoring duplicate directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/./lib/gcc/mips-malta-
linux-gnu/5.2.0/include-fixed"
ignoring duplicate directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/./lib/gcc/mips-malta-
linux-gnu/5.2.0/././././mips-malta-linux-gnu/include"
#include "...": search starts here:
#include <...> search starts here:
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/include
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/include-fixed
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/././././mips-malta-linux-
gnu/include
/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot/usr/include
End of search list.

编译

汇编

```
GNU C11 (crosstool-NG crosstool-ng-1.22.0) version 5.2.0 (mips-malta-linux-gnu)
  compiled by GNU C version 4.6.3, GMP version 6.0.0, MPFR version 3.1.3, MPC version 1.0.3
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
Compiler executable checksum: a0212981a25e6bcf7c0ea0e0513f0ef0
COLLECT_GCC_OPTIONS='-o' 'exe' '-v' '-mabi=32' '-mllsc' '-mplt' '-mno-shared' '-EB'
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/../../../../mips-malta-linux-
gnu/bin/as -v -EB -O1 -no-mdebug -mabi=32 -mno-shared -call_nonpic -o /tmp/cc0eljh2.o /tmp/ccntcnFQ.s
GNU assembler version 2.25.1 (mips-malta-linux-gnu) using BFD version (crosstool-NG crosstool-ng-
1.22.0) 2.25.1
COLLECT_GCC_OPTIONS='-o' 'exe' '-v' '-mabi=32' '-mllsc' '-mplt' '-mno-shared' '-EB'
/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/mips-malta-linux-gnu/5.2.0/cc1 -quiet -v -iprefix
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/ -isysroot
/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot extras.c -meb -quiet -
dumpbase extras.c -mabi=32 -mllsc -mplt -mno-shared -auxbase extras -version -o /tmp/ccntcnFQ.s
GNU C11 (crosstool-NG crosstool-ng-1.22.0) version 5.2.0 (mips-malta-linux-gnu)
  compiled by GNU C version 4.6.3, GMP version 6.0.0, MPFR version 3.1.3, MPC version 1.0.3
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
ignoring duplicate directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/../../../../lib/gcc/mips-malta-
linux-gnu/5.2.0/include"
ignoring nonexistent directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-
gnu/sysroot/home/vangluming/x-tools/mips-malta-linux-gnu/mips-malta-linux-gnu/sysroot/include"
ignoring duplicate directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/../../../../lib/gcc/mips-malta-
linux-gnu/5.2.0/include-fixed"
ignoring duplicate directory "/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/../../../../lib/gcc/mips-malta-
linux-gnu/5.2.0/../../../../mips-malta-linux-gnu/include"
```

编译

9

汇编

```
#include "..." search starts here:
#include <...> search starts here:
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/include
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/include-fixed
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/../../../../mips-malta-linux-
gnu/include
/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot/usr/include
End of search list.
GNU C11 (crosstool-NG crosstool-ng-1.22.0) version 5.2.0 (mips-malta-linux-gnu)
  compiled by GNU C version 4.6.3, GMP version 6.0.0, MPFR version 3.1.3, MPC version 1.0.3
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
Compiler executable checksum: a0212981a25e6bcf7c0ea0e0513f0ef0
COLLECT_GCC_OPTIONS='-o' 'exe' '-v' '-mabi=32' '-mllsc' '-mplt' '-mno-shared' '-EB'
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/../../../../mips-malta-linux-
gnu/bin/as -v -EB -O1 -no-mdebug -mabi=32 -mno-shared -call_nonpic -o /tmp/cc0fSkXd.o
/tmp/ccntcnFQ.s
GNU assembler version 2.25.1 (mips-malta-linux-gnu) using BFD version (crosstool-NG crosstool-ng-
1.22.0) 2.25.1
COMPILER_PATH=/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/mips-malta-linux-
gnu/5.2.0:/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/OSLAB/compiler/mips-malta-linux-
gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/../../../../mips-malta-linux-gnu/bin/
LIBRARY_PATH=/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-
gnu/5.2.0:/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/OSLAB/compiler/mips-malta-linux-
gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/../../../../mips-malta-linux-gnu/lib/OSLAB/compiler/mips-
malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot/lib/OSLAB/compiler/mips-malta-linux-
gnu/bin/../mips-malta-linux-gnu/sysroot/usr/lib/
```

10

链接

```
COLLECT_GCC_OPTIONS='-o' 'exe' '-v' '-mabi=32' '-mllsc' '-mplt' '-mno-shared' '-EB'
/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/mips-malta-linux-gnu/5.2.0/collect2 -plugin
/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/mips-malta-linux-gnu/5.2.0/liblto_plugin.so -
plugin-opt=/OSLAB/compiler/mips-malta-linux-gnu/bin/../libexec/gcc/mips-malta-linux-gnu/5.2.0/lto-
wrapper -plugin-opt=-fresolution=/tmp/ccvIN9Dp.res -plugin-opt=-pass-through=-lgcc -plugin-opt=-pass-
through=-lgcc_s -plugin-opt=-pass-through=-lc -plugin-opt=-pass-through=-lgcc -plugin-opt=-pass-
through=-lgcc_s --sysroot=/OSLAB/compiler/lib/ld.so.1 mips-malta-linux-gnu/bin/../mips-malta-linux-
gnu/sysroot --eh-frame-hdr -EB -dynamic-linker /-melf32btmip -o exe /OSLAB/compiler/mips-malta-
linux-gnu/bin/../mips-malta-linux-gnu/sysroot/usr/lib/crt1.o /OSLAB/compiler/mips-malta-linux-
gnu/bin/../mips-malta-linux-gnu/sysroot/usr/lib/crti.o /OSLAB/compiler/mips-malta-linux-
gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/crtbegin.o -L/OSLAB/compiler/mips-malta-linux-
gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0 -L/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc -
L/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/../../../../mips-malta-
linux-gnu/lib -L/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot/lib -
L/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot/usr/lib /tmp/cc0elj2.o
/tmp/cc0fSkXd.o -lgcc --as-needed -lgcc_s --no-as-needed -lc -lgcc --as-needed -lgcc_s --no-as-needed
/OSLAB/compiler/mips-malta-linux-gnu/bin/../lib/gcc/mips-malta-linux-gnu/5.2.0/crtend.o
/OSLAB/compiler/mips-malta-linux-gnu/bin/../mips-malta-linux-gnu/sysroot/usr/lib/crtn.o
```

11

```
.file 1 "program.c"
.section .mdebug.abi32
.previous
.nan legacy
.module fp=32
.module nooddspreg
.abicalls
.option pic0

.comm some_global_variable,4,4
.local some_local_variable
.comm some_local_variable,4,4
.rdata
.align 2
$LC0:
.ascii "I am done\000"
.text
.align 2
.globl main
.set nomips16
.set nomicromips
.ent main
.type main, @function
main:
.frame $fp,40,$31 # vars= 8, regs= 2/0, args= 16,
gp= 8
.mask 0xc0000000,-4
.fmask 0x00000000,0
北京航空航天大学

.set noreorder
.set nomacro
addiu $sp,$sp,-40
sw $31,36($sp)
sw $fp,32($sp)
move $fp,$sp
jal read_something
nop

sw $2,24($fp)
lw $4,24($fp)
jal do_something
nop
move $3,$2
lui $2,%hi(some_global_variable)
sw $3,%lo(some_global_variable)($2)
lui $2,%hi($LC0)
addiu $4,$2,%lo($LC0)
jal write_something
nop

move $2,$0
move $sp,$fp
lw $31,36($sp)

lw $fp,32($sp)
addiu $sp,$sp,40
j $31
nop

.set macro
.set reorder
.end main
.size main,.-main
.ident "GCC: (crosstool-
NG crosstool-ng-1.22.0) 5.2.0"
```

计算机/软件学院

12

12

生成的汇编文件

```
.file 1 "program.c"
.section .mdebug.abi32
.previous
.nan legacy
.module fp=32
.module nooddspreg
.abicalls
.option pic0

.comm some_global_variable,4,4
.local some_local_variable
.comm some_local_variable,4,4
.rdata
.align 2

$LCO:
.ascii "I am done\000"
.text
.align 2
.globl main
.set nomips16
.set nomicromips
.ent main
.type main, @function
main:
.frame $fp,40,$31
# vars= 8, regs= 2/0, args= 16, gp= 8
.mask 0xc0000000,-4
.fmask 0x00000000,0
```

北京航空航天大学

计算机/软件学院

13

13

生成的汇编文件

```
.set noreorder
.set nomacro
addiu $sp,$sp,-40
sw $31,36($sp)
sw $fp,32($sp)
move $fp,$sp
jal read_something
nop

sw $2,24($fp)
lw $4,24($fp)
jal do_something
nop
move $3,$2
lui $2,%hi(some_global_variable)
sw $3,%lo(some_global_variable)($2)
lui $2,%hi($LCO)
addiu $4,$2,%lo($LCO)

jal write_something
nop
move $2,$0
move $sp,$fp
lw $31,36($sp)
lw $fp,32($sp)
addiu $sp,$sp,40
j $31
nop

.set macro
.set reorder
.end main
.size main,.-main
.ident "GCC: (crosstool-NG
crosstool-ng-1.22.0) 5.2.0"
```

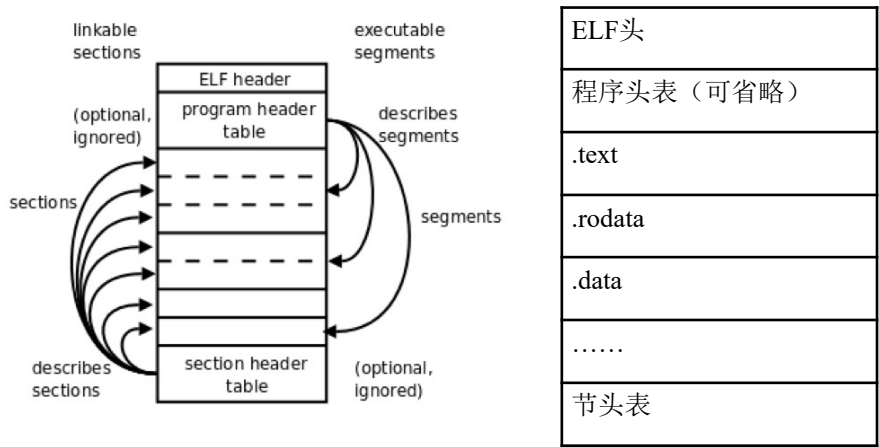
函数调用变成了汇编函数调用指令，
do_something只是标记

北京航空航天大学

计算机/软件学院

14

ELF(Executable and Linkable Format)-可执行文件格式



15

- .bss 此节存放用于程序内存映象的未初始化数据。此节类型是 SHT_NOBITS,因此不占文件空间。
- .comment 此节存放版本控制信息。
- .data和.data1 此节存放用于程序内存映象的初始化数据。
- .debug 此节存放符号调试信息。
- .dynamic 此节存放动态连接信息。
- .dynstr 此节存放动态连接所需的字符串，在大多数情况下，这些字符串代表的是与符号表项有关的名字。
- .dynsym 此节存放的是“符号表”中描述的动态连接符号表。
- .fini 此节存放与进程中指代码有关的执行指令。
- .got 此节存放全程偏移量表。
- .hash 此节存放一个符号散列表。
- .init 此节存放组成进程初始化代码的执行指令。
- .interp 此节存放一个程序解释程序的路径名。
- .line 此节存放符号调试中使用的行号信息，主要描述源程序与机器指令之间的对应关系。
- .note 此节存放供其他程序检测兼容性，一致性的特殊信息。
- .plt 此节存放过程连接表。
- .relname和.rela name 此节存放重定位信息。
- .rodata和.rodata1 此节存放进程映象中不可写段的只读数据。
- .shstrtab 此节存放节名。
- .strtab 此节存放的字符串标识与符号表项有关的名字。
- .symtab 此节存放符号表。
- .text 此节存放正文，也称程序的执行指令。

16

ELF文件头的定义

ELF头描述文件组成。

```
typedef struct {
    unsigned char    e_ident[16]; /* 标志本文件为目标文件，提供
                                   机器无关的数据，可实现对文
                                   件内容的译码与解释 */
    unsigned char    e_type[2];   /* 标识目标文件类型 */
    unsigned char    e_machine[2]; /* 指定必需的体系结构 */
    unsigned char    e_version[4]; /* 标识目标文件版本 */
    unsigned char    e_entry[4];  /* 指向起始虚地址的指针 */
    unsigned char    e_phoff[4];  /* 程序头表的文件偏移量 */
    unsigned char    e_shoff[4];  /* 节头表的文件偏移量 */
    unsigned char    e_flags[4];  /* 针对具体处理器的标志 */
    unsigned char    e_ehsize[2]; /* ELF头的大小 */
    unsigned char    e_phentsize[2]; /* 程序头表每项的大小 */
    unsigned char    e_phnum[2];  /* 程序头表项的个数 */
    unsigned char    e_shentsize[2]; /* 节头表每项的大小 */
    unsigned char    e_shnum[2];  /* 节头表项的个数 */
    unsigned char    e_shstrndx[2]; /* 与节名字符串表相关的节头表
                                   项的索引 */
} Elf32_Ehdr;
```

北京航空航天大学

计算机/软件学院

17

17

ELF文件头的定义

e_ident: 这一部分是文件的标志，用于表明该文件是一个ELF文件。ELF文件的头四个字节为magic number。

e_type: 用于标明该文件的类型，如可执行文件、动态连接库、可重定位文件等。

e_machine: 表明体系结构，如x86，x86_64，MIPS，PowerPC等等。

e_version: 文件版本

e_entry: 程序入口的虚拟地址

e_phoff: 程序头表在该ELF文件中的位置(具体地说是偏移)。ELF文件可以没有程序头表。

北京航空航天大学

计算机/软件学院

18

18

ELF文件头的定义

e_shoff:	节头表的位置。
e_elflags:	针对具体处理器的标志。
e_ehsize:	ELF 头的大小。
e_phentsize:	程序头表每项的大小。
e_phnum:	程序头表项的个数。
e_shentsize:	节头表每项的大小。
e_shnum:	节头表项的个数。
e_shstrndx:	与节名字符串表相关的节头表。

符号表

program.o

```
13: 00000004  4 OBJECT GLOBAL DEFAULT COM some_global_variable
14: 00000000 96 FUNC  GLOBAL DEFAULT  1 main
15: 00000000  0 NOTYPE GLOBAL DEFAULT UND read_something
16: 00000000  0 NOTYPE GLOBAL DEFAULT UND do_something
17: 00000000  0 NOTYPE GLOBAL DEFAULT UND write_something
```

extras.o

```
12: 00000000 68 FUNC  GLOBAL DEFAULT  1 read_something
13: 00000000  0 NOTYPE GLOBAL DEFAULT UND __isoc99_scanf
14: 00000044 52 FUNC  GLOBAL DEFAULT  1 do_something
15: 00000078 84 FUNC  GLOBAL DEFAULT  1 write_something
16: 00000000  0 NOTYPE GLOBAL DEFAULT UND some_global_variable
17: 00000000  0 NOTYPE GLOBAL DEFAULT UND printf
```

a.out

```
61: 00400744 52 FUNC GLOBAL DEFAULT 13 do_something
62: 00410a1c 4 OBJECT GLOBAL DEFAULT 26 some_global_variable
67: 00400700 68 FUNC GLOBAL DEFAULT 13 read_something
74: 00400778 84 FUNC GLOBAL DEFAULT 13 write_something
77: 00400990 0 FUNC GLOBAL DEFAULT [MIPS PLT] UND
    __libc_start_main@@GLIBC_
80: 00410a40 0 NOTYPE GLOBAL DEFAULT 27 _end
81: 00410a1c 0 NOTYPE GLOBAL DEFAULT 26 __bss_start
82: 004006a0 96 FUNC GLOBAL DEFAULT 13 main

Main.o
13: 00000004 4 OBJECT GLOBAL DEFAULT COM some_global_variable
14: 00000000 96 FUNC GLOBAL DEFAULT 1 main
15: 00000000 0 NOTYPE GLOBAL DEFAULT UND read_something
16: 00000000 0 NOTYPE GLOBAL DEFAULT UND do_something
17: 00000000 0 NOTYPE GLOBAL DEFAULT UND write_something
```

北京航空航天大学

计算机/软件学院

21

21

使用objdump反汇编ELF文件

```
program.o: file format elf32-tradbigmips
Disassembly of section .text:
00000000 <main>:
```

```
0: 27bdfdd8 addiu sp,sp,-40
4: afbf0024 sw ra,36(sp)
8: afbe0020 sw s8,32(sp)
c: 03a0f021 move s8,sp
10: 0c000000 jal 0 <main>
14: 00000000 nop
18: afc20018 sw v0,24(s8)
1c: 8fc40018 lw a0,24(s8)
20: 0c000000 jal 0 <main>
24: 00000000 nop
28: 00401821 move v1,v0
2c: 3c020000 lui v0,0x0
30: ac430000 sw v1,0(v0)
34: 3c020000 lui v0,0x0
38: 24440000 addiu a0,v0,0
3c: 0c000000 jal 0 <main>
```

偏移

汇编指令

机器码

北京航空航天大学

计算机/软件学院

22

22

使用objdump反汇编ELF文件

- 在源文件三处函数调用，对应到汇编文件里，就是三处jal指令。
- 三条jal所对应的机器码，头六位二进制数(000011)代表jal，而后面的一串0是操作数，也就是要跳转到的地址。

10: 0c000000 jal 0 <main>

该条机器指令的二进制表示：

$(0c000000)_{16} = (0000\ 1100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000)_2$

要跳转的函数地址都是0? !

链接的过程

- 编译C程序的时候，是以.c文件作为编译单元的。
 - 编译：.c→.o
 - 编译时函数定义在不同文件，无法知道地址。
- 链接的过程：
 - 将这些.o文件链接到一起，形成最终的可执行文件。
 - 在链接时，链接器会扫描各个目标文件，将之前未填写的地址填写上，从而生成一个真正可执行的文件。
 - E重定位目标文件
 - U未解析符号
 - D已定义符号
- 重定位(Relocation)
 - 将之前未填写的地址填写的过程。

```
gcc -Wall -o link program.o
```

```
program.o: In function `main':
```

```
program.c:(.text+0x10): undefined reference to `read_something'
```

```
program.c:(.text+0x20): undefined reference to `do_something'
```

```
program.c:(.text+0x3c): undefined reference to `write_something'
```

```
collect2: error: ld returned 1 exit status
```

Relocation entry

```
typedef struct {
```

```
/*给出了使用重定位动作的地点。对重定位文件来说，  
它的值是从节起始处到受重定位影响的存储单元的字节偏移量；对可执行文件或共享目标文件来说，它的  
值是受重定位影响的存储单元的虚拟地址*/
```

```
Elf32_Addr r_offset;
```

```
/*给出了与重定位修改地点有关的符号表索引和所使用的  
的重定位的类型*/
```

```
Elf32_Word r_info;(symbol:24; type:8)
```

```
} Elf32_Rel;
```

Readelf读取重定位节

- Relocation section '.rel.text' at offset 0x348 contains 7 entries:

Offset	Info	Type	Sym.Value	Sym. Name
00000010	00000f04	R_MIPS_26	00000000	read_something
00000020	00001004	R_MIPS_26	00000000	do_something
0000002c	00000d05	R_MIPS_HI16	00000004	some_global_variable
00000030	00000d06	R_MIPS_LO16	00000004	some_global_variable
00000034	00000705	R_MIPS_HI16	00000000	.rodata
00000038	00000706	R_MIPS_LO16	00000000	.rodata
0000003c	00001104	R_MIPS_26	00000000	write_something

10: 0c000000 jal 0 <main>

北京航空航天大学

计算机/软件学院

27

27

链接后...

```
004006a0 <main>:
4006a0: 27bdfdd8    addiu sp,sp,-40
4006a4: afbf0024    sw ra,36(sp)
4006a8: afbe0020    sw s8,32(sp)
4006ac: 03a0f021    move s8,sp
4006b0: 0c1001c0    jal 400700 <read_something>
4006b4: 00000000    nop
4006b8: afc20018    sw v0,24(s8)
4006bc: 8fc40018    lw a0,24(s8)
4006c0: 0c1001d1    jal 400744 <do_something>
4006c4: 00000000    nop
4006c8: 00401821    move v1,v0
4006cc: 3c020041    lui v0,0x41
4006d0: ac430a1c    sw v1,2588(v0)
4006d4: 3c020040    lui v0,0x40
4006d8: 24440930    addiu a0,v0,2352
4006dc: 0c1001de    jal 400778 <write_something>
4006e0: 00000000    nop
4006e4: 00001021    move v0,zero
4006e8: 03c0e821    move sp,s8
4006ec: 8fbf0024    lw ra,36(sp)
4006f0: 8fbe0020    lw s8,32(sp)
4006f4: 27bd0028    addiu sp,sp,40
4006f8: 03e00008    jr ra
4006fc: 00000000    nop
```

```
0: 27bdfdd8    addiu sp,sp,-40
4: afbf0024    sw ra,36(sp)
8: afbe0020    sw s8,32(sp)
c: 03a0f021    move s8,sp
10: 0c000000    jal 0 <main>
14: 00000000    nop
18: afc20018    sw v0,24(s8)
1c: 8fc40018    lw a0,24(s8)
20: 0c000000    jal 0 <main>
24: 00000000    nop
28: 00401821    move v1,v0
2c: 3c020000    lui v0,0x0
30: ac430000    sw v1,0(v0)
34: 3c020000    lui v0,0x0
38: 24440000    addiu a0,v0,0
3c: 0c000000    jal 0 <main>
.....
```

北京航空航天大学

计算机/软件学院

28

28

重定位时链接地址的计算

Name	Symbol	Calculation
R_MIPS_26	Local	$((A \mid ((P + 4) \& 0xf0000000)) + S) \gg 2$
	External	$(\text{sign_extend}(A) + S) \gg 2$
R_MIPS_HI16	Any	$\%high(AHL + S)$ The $\%high(x)$ function is $(x - (\text{short})x) \gg 16$
R_MIPS_LO16	Any	$AHL + S$

A 附加值(addend)。
S 符号的地址。
AHL 地址的附加量(addend)。

北京航空航天大学

计算机/软件学院

29

29

链接地址的计算read_something

10: 0c000000 jal 0 <main> 编译后main.o

Offset	Info	Type	Sym.Value	Sym. Name			
00000010	00000f04	R_MIPS_26	00000000	read_something			
Symbol table '.symtab' contains 93 entries:							
Num:	Value	Size	Type	Bind	Vis	Ndx	Name
67:	00400700	68	FUNC	GLOBAL	DEFAULT	13	read_something

计算的公式为 $(\text{sign_extend}(A) + S) \gg 2$ ，其中， $A=0$ ， $S=00400700$ ，所以结果为1001c0，填写到jal指令的操作数的位置，得到的结果正是0c1001c0，与汇编器给出的一致。

0000 1100 0100 0000 0000 0111 0000 0000 右移2位→

0000 1100 0001 0000 0000 0001 1100 0000

4006b0: 0c1001c0 jal 400700 <read_something> 链接后

北京航空航天大学

计算机/软件学院

30

30

Extras.o

00000078 <write_something>:

78: 27bdffe0	addiu sp,sp,-32	b4: 03c0e821	move sp,s8
7c: afbf001c	sw ra,28(sp)	b8: 8fbf001c	lw ra,28(sp)
80: afbe0018	sw s8,24(sp)	bc: 8fbe0018	lw s8,24(sp)
84: 03a0f021	move s8,sp	c0: 27bd0020	addiu sp,sp,32
88: afc40020	sw a0,32(s8)	c4: 03e00008	jr ra
8c: 3c020000	lui v0,0x0	c8: 00000000	nop
90: 8c420000	lw v0,0(v0)	cc: 00000000	nop
94: 00000000	nop		
98: 00403021	move a2,v0		
9c: 8fc50020	lw a1,32(s8)		
a0: 3c020000	lui v0,0x0		
a4: 24440004	addiu a0,v0,4		
a8: 0c000000	jal 0 <read_something>		
ac: 00000000	nop		
b0: 00000000	nop		

```
void write_something(const char* str) {
    printf("%s: %d\n", str, some_global_variable);
}
```

北京航空航天大学

计算机/软件学院

31

31

exe

62: 00410a1c 4 OBJECT GLOBAL DEFAULT 26 some_global_variable

00400778 <write_something>:

400778: 27bdffe0	addiu sp,sp,-32	4007b8: 8fbf001c	lw ra,28(sp)
40077c: afbf001c	sw ra,28(sp)	4007bc: 8fbe0018	lw s8,24(sp)
400780: afbe0018	sw s8,24(sp)	4007c0: 27bd0020	addiu sp,sp,32
400784: 03a0f021	move s8,sp	4007c4: 03e00008	jr ra
400788: afc40020	sw a0,32(s8)	4007c8: 00000000	nop
40078c: 3c020041	lui v0,0x41	4007cc: 00000000	nop
400790: 8c420a1c	lw v0,2588(v0)		
400794: 00000000	nop		
400798: 00403021	move a2,v0		
40079c: 8fc50020	lw a1,32(s8)		
4007a0: 3c020040	lui v0,0x40		
4007a4: 24440944	addiu a0,v0,2372		
4007a8: 0c100260	jal 400980 <printf@plt>		
4007ac: 00000000	nop		
4007b0: 00000000	nop		
4007b4: 03c0e821	move sp,s8		

北京航空航天大学

计算机/软件学院

32

32

链接地址的计算some_global_variable

2c: 3c020000 lui v0, 0x0
30: ac430000 sw v1,0(v0) 编译后main.o

Offset	Info	Type	Sym.Value	Sym. Name
0000002c	00000d05	R_MIPS_HI16	00000004	some_global_variable
00000030	00000d06	R_MIPS_LO16	00000004	some_global_variable

Symbol table '.symtab' contains 93 entries:

Num:	Value	Size	Type	Bind	Vis	Ndx	Name
62:	00410a1c	4	OBJECT	GLOBAL	DEFAULT	26	some_global_variable

高16位的类型为R_MIPS_HI16, 计算公式为((AHL + S)-(short)(AHL + S))>>16, 此处AHL为0, S为00410a1c, 结果为41
低16位地址的类型为R_MIPS_LO16, 计算公式为AHL+S, 此处AHL为0, S为00410a1c。这里只保留16位, 因此, 结果为0a1c

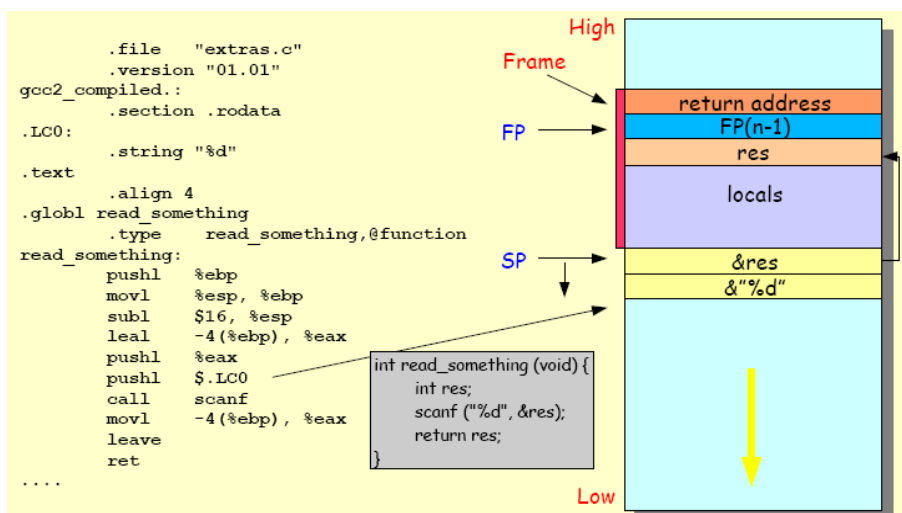
4006cc: 3c020041 lui v0, 0x41
4006d0: ac430a1c sw v0, 2588(v0) 链接后

北京航空航天大学

计算机/软件学院

33

33



北京航空航天大学

计算机/软件学院

34

34

```

00000000h: 7F 45 4C 46 01 01 01 00 00 00 00 00 00 00 00 00 ; ELF.....
00000010h: 01 00 03 00 01 00 00 00 00 00 00 00 00 00 00 00 ; .....
00000020h: C4 00 00 00 00 00 00 00 34 00 00 00 00 00 28 00 ; ?.....4....(.
00000030h: 09 00 06 00 55 89 E5 83 EC 08 83 E4 F0 B8 00 00 00 ; ...u文波.麻葛..
00000040h: 00 00 29 C4 E8 FC FF FF B8 00 00 00 00 C9 C3 ; ..)焉? ?...擅
00000050h: 01 00 00 00 02 00 00 00 00 47 43 43 3A 20 28 47 ; .....GCC: (G
00000060h: 4E 55 29 20 33 2E 32 2E 32 20 32 30 30 33 30 32 ; NU) 3.2.2 200302
00000070h: 32 32 20 28 52 65 64 20 48 61 74 20 4C 69 6E 75 ; 22 (Red Hat Linu
00000080h: 78 20 33 2E 32 2E 32 2D 35 29 00 00 2E 73 79 6D ; x 3.2.2-5)...sym
00000090h: 74 61 62 00 2E 73 74 72 74 61 62 00 2E 73 68 73 ; tab..strtab..shs
000000a0h: 74 72 74 61 62 00 2E 72 65 6C 2E 74 65 78 74 00 ; trtab..rel.text.
000000b0h: 2E 64 61 74 61 00 2E 62 73 73 00 2E 63 6F 6D 6D ; .data..bss..comm
000000c0h: 65 6E 74 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ent.....
000000d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; .....
000000e0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 1F 00 00 ; .....
000000f0h: 01 00 00 00 06 00 00 00 00 00 00 00 00 34 00 00 ; .....4...
00000100h: 1C 00 00 00 00 00 00 00 00 00 00 00 00 04 00 00 ; .....
00000110h: 00 00 00 00 1F 00 00 00 09 00 00 00 00 00 00 00 ; .....
00000120h: 00 00 00 00 D4 02 00 00 08 00 00 00 07 00 00 00 ; ....?.....
00000130h: 01 00 00 00 04 00 00 00 08 00 00 00 25 00 00 00 ; .....%...
00000140h: 01 00 00 00 03 00 00 00 00 00 00 00 50 00 00 00 ; .....P...
00000150h: 08 00 00 00 00 00 00 00 00 00 00 00 04 00 00 00 ; .....
00000160h: 00 00 00 00 2E 00 00 00 08 00 00 00 03 00 00 00 ; ....+.....
00000170h: 00 00 00 00 58 00 00 00 00 00 00 00 00 00 00 00 ; ....X.....
00000180h: 00 00 00 00 04 00 00 00 00 00 00 00 30 00 00 00 ; .....0...
00000190h: 01 00 00 00 00 00 00 00 00 00 00 00 58 00 00 00 ; .....X...
000001a0h: 33 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 ; 3.....
000001b0h: 00 00 00 00 11 00 00 00 03 00 00 00 00 00 00 00 ; .....
000001c0h: 00 00 00 00 8F 00 00 00 39 00 00 00 00 00 00 00 ; ....?.9.....
000001d0h: 00 00 00 00 01 00 00 00 00 00 00 00 01 00 00 00 ; .....
000001e0h: 02 00 00 00 00 00 00 00 00 00 00 00 2C 02 00 00 ; .....
000001f0h: 90 00 00 00 08 00 00 00 06 00 00 00 04 00 00 00 ; ?.....
00000200h: 10 00 00 00 09 00 00 00 03 00 00 00 00 00 00 00 ; .....
00000210h: 00 00 00 00 BC 02 00 00 16 00 00 00 00 00 00 00 ; ....?.....
00000220h: 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 ; .....

```

北京航空航天大学

计算机/软件学院

35

35

```

00000000h: 7F 45 4C 46 01 01 01 00 00 00 00 00 00 00 00 00 ; ELF.....
00000010h: 02 00 03 00 01 00 00 00 40 83 04 08 34 00 00 00 ; .....8?4...
00000020h: 74 2A 00 00 00 00 00 00 34 00 20 00 06 00 28 00 ; t*.....4. ...(.
00000030h: 1D 00 1A 00 06 00 00 00 34 00 00 00 34 80 04 08 ; .....4...4E..
00000040h: 34 80 04 08 C0 00 00 C0 00 00 00 00 05 00 00 00 ; 4E...?.?...
00000050h: 04 00 00 00 03 00 00 00 F4 00 00 00 F4 80 04 08 ; .....?.刳..
00000060h: F4 80 04 08 13 00 00 00 13 00 00 00 04 00 00 00 ; 刳.....
00000070h: 01 00 00 00 01 00 00 00 00 00 00 00 80 04 08 ; .....E...
00000080h: 00 80 04 08 4D 05 00 00 4D 05 00 00 05 00 00 00 ; .E..M...M.....
00000090h: 00 10 00 00 01 00 00 00 50 05 00 00 50 95 04 08 ; .....P...P?..
000000a0h: 50 95 04 08 10 01 00 00 30 01 00 00 06 00 00 00 ; P?....0.....
000000b0h: 00 10 00 00 02 00 00 00 64 05 00 00 64 95 04 08 ; .....d...d?..
000000c0h: 64 95 04 08 C8 00 00 00 C8 00 00 00 06 00 00 00 ; d??.?.....
000000d0h: 04 00 00 00 04 00 00 00 08 01 00 00 08 81 04 08 ; .....?.
000000e0h: 08 81 04 08 20 00 00 00 20 00 00 00 04 00 00 00 ; .?. ...
000000f0h: 04 00 00 00 2F 6C 69 62 2F 6C 64 2D 6C 69 6E 75 ; .../lib/ld-linux
00000100h: 78 2E 73 6F 2E 32 00 00 04 00 00 00 10 00 00 00 ; x.so.2.....
00000110h: 01 00 00 00 47 4E 55 00 00 00 00 00 02 00 00 00 ; ...GNU.....
00000120h: 02 00 00 00 05 00 00 00 03 00 00 00 08 00 00 00 ; .....
00000130h: 07 00 00 00 04 00 00 00 06 00 00 00 00 00 00 00 ; .....
00000140h: 00 00 00 00 01 00 00 00 02 00 00 00 03 00 00 00 ; .....
00000150h: 00 00 00 00 05 00 00 00 00 00 00 00 00 00 00 00 ; .....
00000160h: 00 00 00 00 00 00 00 00 00 00 00 00 51 00 00 00 ; .....Q...
00000170h: E4 82 04 08 95 00 00 00 22 00 00 00 2A 00 00 00 ; 豔..?.."...*...
00000180h: F4 82 04 08 34 00 00 00 12 00 00 00 12 00 00 00 ; 豔..4.....
00000190h: 04 83 04 08 25 00 00 00 22 00 00 00 3F 00 00 00 ; .?.%..."...?..
000001a0h: 14 83 04 08 D3 00 00 00 12 00 00 00 0B 00 00 00 ; .?.?.....
000001b0h: 24 83 04 08 32 00 00 00 12 00 00 00 30 00 00 00 ; $?.2.....0...
000001c0h: 34 85 04 08 04 00 00 00 11 00 0E 00 67 00 00 00 ; 4?.....g...
000001d0h: 00 00 00 00 00 00 00 00 20 00 00 00 00 6C 69 62 ; .....lib
000001e0h: 63 2E 73 6F 2E 36 00 70 72 69 6E 74 66 00 5F 5F ; c.so.6.printf._
000001f0h: 64 65 72 65 67 69 73 74 65 72 5F 66 72 61 6D 65 ; deregister_frame
00000200h: 5F 69 6E 6E 6F 00 73 63 61 6E 66 00 5F 49 4F 5F ; _info.scanf._IO
00000210h: 73 74 64 69 6E 5F 75 73 65 64 00 5F 5F 6C 69 62 ; stdin_used._lib
00000220h: 63 5F 73 74 61 72 74 5F 6D 61 69 6E 00 5F 5F 72 ; c_start_main._r

```

北京航空航天大学

计算机/软件学院

36

36

ELF头
程序头表
.init
.text
.rodata
.data
.bss
.symtab
.debug
节头表

37

```

SIGNATURE
000 7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00
-- E L F 32 LE FW -----

ELF HEADER
010 01 00 03 00 01 00 00 00 00 00 00 00 00 00 00
    0001 0003 00000001 00000000 00000000
    ↑      target architecture      ↑      program header (executable module only)
    relocatable                      starting address for execution (executable module only)
    module

020 04 01 00 00 00 00 00 00 34 00 00 00 00 00 28 00
    00000104 00000000 0034 0000 0000 0028
    SHT offset  CPU flags  hdr  no PHT  SHT entry size
                        length

030 0b 00 08 00
    000b 0008
    ↑      index (into SHT) of the string section containing section names
    number of entries in SHT

SHT = Section Header Table
PHT = Program Header Table

```

38

- ./exe

程序入口点

ELF Header:

Magic: 7f 45 4c 46 01 02 01 00 01 00 00 00 00 00 00 00

Class: ELF32

Data: 2's complement, big endian

Version: 1 (current)

OS/ABI: UNIX - System V

ABI Version: 1

Type: EXEC (Executable file)

Machine: MIPS R3000

Version: 0x1

Entry point address: 0x4004c0

Start of program headers: 52 (bytes into file)

Start of section headers: 5520 (bytes into file)

Flags: 0x1005, noreorder, cpic, o32, mips1

Size of this header: 52 (bytes)

Size of program headers: 32 (bytes)

Number of program headers: 9

Size of section headers: 40 (bytes)

Number of section headers: 35

Section header string table index: 32

但是不等于Main的地址:
004006a0 ? ? ?

Crt1.o

- g F.text 00000000 _start
- Call __libc_csu_init
- Call __libc_start_main
- Call main
- Call __libc_csu_fini

```
/usr/lib/crt1.o: file format elf32-tradbigmips
Disassembly of section .text:
00000000 <_start>:
0: 3c1c0000    lui    gp,0x0
4: 279c0000    addiu  gp,gp,0
8: 0000f821    move   ra,zero
c: 3c040000    lui    a0,0x0
10: 24840000    addiu  a0,a0,0
14: 8fa50000    lw     a1,0(sp)
18: 27a60004    addiu  a2,sp,4
1c: 2401fff8    li     at,-8
20: 03a1e824    and    sp,sp,at
24: 27bdffe0    addiu  sp,sp,-32
28: 3c070000    lui    a3,0x0
2c: 24e70000    addiu  a3,a3,0
30: 3c080000    lui    t0,0x0
34: 25080000    addiu  t0,t0,0
38: afa80010    sw     t0,16(sp)
3c: afa20014    sw     v0,20(sp)
40: afbd0018    sw     sp,24(sp)
44: 3c190000    lui    t9,0x0
48: 27390000    addiu  t9,t9,0
4c: 0320f809    jair   t9
...
```

程序入口点- _start 函数

程序入口点

Crt1.o 的定位表:

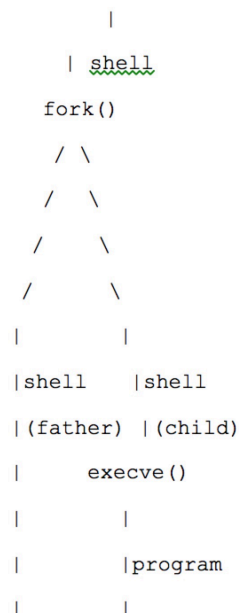
start_入口函数调用了main

Relocation section '.rel.text' at offset 0x42c contains 10 entries:

Offset	Info	Type	Sym.Value	Sym. Name
00000000	0000f05	R_MIPS_HI16	00000000	_gp
00000004	0000f06	R_MIPS_LO16	00000000	_gp
0000000c	00001305	R_MIPS_HI16	00000000	main
00000010	00001306	R_MIPS_LO16	00000000	main
00000028	00001205	R_MIPS_HI16	00000000	__libc_csu_init
0000002c	00001206	R_MIPS_LO16	00000000	__libc_csu_init
00000030	00001005	R_MIPS_HI16	00000000	__libc_csu_fini
00000034	00001006	R_MIPS_LO16	00000000	__libc_csu_fini
00000044	00001605	R_MIPS_HI16	00000000	__libc_start_main
00000048	00001606	R_MIPS_LO16	00000000	__libc_start_main

程序的装载和运行

- 执行程序的过程
 - shell调用fork()系统调用,
 - 创建出一个子进程
 - 子进程调用execve()加载program
- Fork()
- Execve(char *filename, char *argv[], char *envp)



程序的装载

装载前的工作：

- shell调用fork()系统调用，创建一个子进程。

装载工作：

- 子进程调用execve()加载program(即要执行的程序)。

程序如何被加载：

- 加载器在加载程序的时候只需要看ELF文件中和segment相关的信息即可。我们用readelf工具将segment读取出来：读出的信息分为两部分，一部分是各segment的具体信息，另一部分是section和segment之间的对应关系。
- 其中Type为Load的segment是需要被加载到内存中的部分。

程序的装载

文件中的偏移

起始虚地址

文件中的大小

内存中的大小

Program Headers:

Type	Offset	VirtAddr	PhysAddr	FileSiz	MemSiz	Flg	Align
PHDR	0x000034	0x00400034	0x00400034	0x00120	0x00120	R E	0x4
INTERP	0x000154	0x00400154	0x00400154	0x0000d	0x0000d	R	0x1
[Requesting program interpreter: /lib/ld.so.1]							
ABIFLAGS	0x000188	0x00400188	0x00400188	0x00018	0x00018	R	0x8
REGINFO	0x0001a0	0x004001a0	0x004001a0	0x00018	0x00018	R	0x4
LOAD	0x000000	0x00400000	0x00400000	0x009b4	0x009b4	R E	0x10000
LOAD	0x0009b4	0x004109b4	0x004109b4	0x00068	0x0008c	RW	0x10000
DYNAMIC	0x0001b8	0x004001b8	0x004001b8	0x000f8	0x000f8	R	0x4
NOTE	0x000164	0x00400164	0x00400164	0x00020	0x00020	R	0x4
NULL	0x000000	0x00000000	0x00000000	0x00000	0x00000		0x4

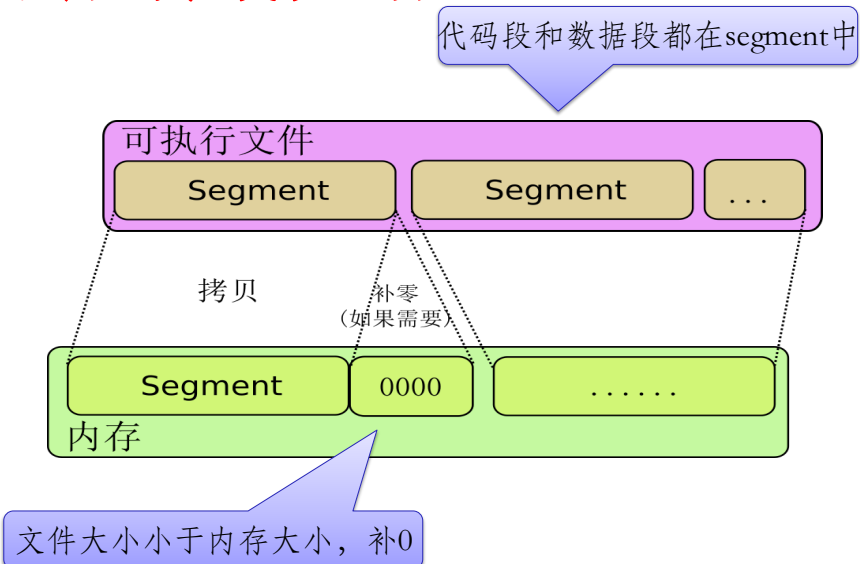
LOAD表示要加载到内存的部分

程序的装载

细节：

- 一个segment在文件中的大小是小于等于其在内存中的大小。
- 如果在文件中的大小小于在内存中的大小，那么在载入内存时通过**补零**使其达到其在内存中应有的大小。

程序的装载和运行



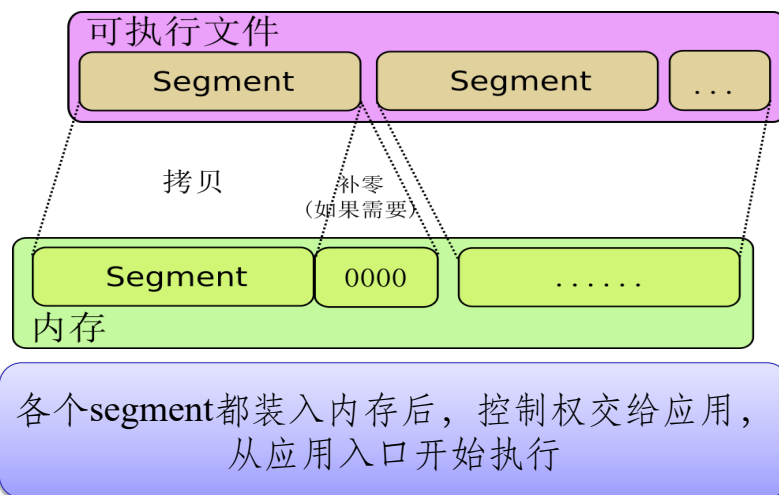
程序的装载流程

- 读取ELF头部的魔数(Magic Number)，以确认该文件确实是ELF文件。
 - ELF文件的头四个字节依次为'0x7f'、'E'、'L'、'F'。
 - 加载器会首先对比这四个字节，若不一致，则报错。
- 找到段表项。
 - ELF头部会给出的段表起始位置在文件中的偏移，段表项的大小，以及段表包含了多少项。根据这些信息可以找到每一个段表项。
- 对于每个段表项解析出各个段应当被加载的虚地址，在文件中的偏移。以及在内存中的大小和在文件中的大小。（段在文件中的大小小于等于内存中的大小）。

程序的装载流程

- 对于每一个段，根据其在内存中的大小，为其分配足够的物理页，并映射到指定的虚地址上。再将文件中的内容拷贝到内存中。
- 若ELF中记录的段在内存中的大小大于在文件中的大小，则多出来的部分用0进行填充。
- 设置进程控制块中的PC为ELF文件中记载的入口地址。
- 控制权交给进程开始执行！

程序的运行



北京航空航天大学

计算机/软件学院

51

51

- 系统调用： `execve()`
- 对应函数：
 - `int do_execve(char *filename, char **argv, char **envp, struct pt_regs *regs);`
 - `asmlinkage int sys_execve (struct pt_regs regs);`
- 主要数据结构：
 - `struct pt_regs`在系统调用时用于保存寄存器组；
 - `struct linux_binprm`用于存储执行该文件的一些参数；
 - `struct linux_binfmt`其中定义了一些用以载入二进制文件的函数。

北京航空航天大学

计算机/软件学院

52

52

```

struct linux_binprm{
    char buf[128];
    unsigned long page[MAX_ARG_PAGES];
    unsigned long p;
    int sh_bang;
    struct inode * inode;
    int e_uid, e_gid;
    int argc, envc;
    char * filename;      /* Name of binary */
    unsigned long loader, exec;
    int dont_iput;        /* binfmt handler has put inode */
};

```

```

struct linux_binfmt {
    struct linux_binfmt * next;
    long *use_count;
    int (*load_binary)(struct linux_binprm *, struct pt_regs *
regs);
    int (*load_shlib)(int fd);
    int (*core_dump)(long signr, struct pt_regs * regs);
};

```

- `sys_execve()`只是函数`do_execve()`的一个界面，实际的处理动作在`do_execve()`中完成。
- `regs.ebx`：指向程序文件名的指针；
- `regs.ecx`：指向传递给程序的参数的指针；
- `regs.edx`：程序运行的环境的地址。

do_execve

- 通过这个程序的文件名，找到该程序对应的可执行文件即i结点；
- 检查对该文件的接触权限（在打开i结点的函数`open_namei()`中实现）；
- 设置程序所需的参数结构`struct linux_binprm bprm`；
- 以`bprm`为参数，调用函数`prepare_binprm()`，进行一些其他的检查并读入该文件的前128字节；
- 使用函数`search_binary_handler()`，试着用多种方式将该可执行文件载入。Linux为它所支持的每一种格式的可执行文件都提供了一个函数（由函数指针`load_binary`指向）以载入文件。轮流调用这些函数，并判断其返回值是否为成功标识。如果成功，则可以判断该可执行文件的格式。如果都不成功，则返回错误标识`NOEXEC`；

do_load_elf_binary

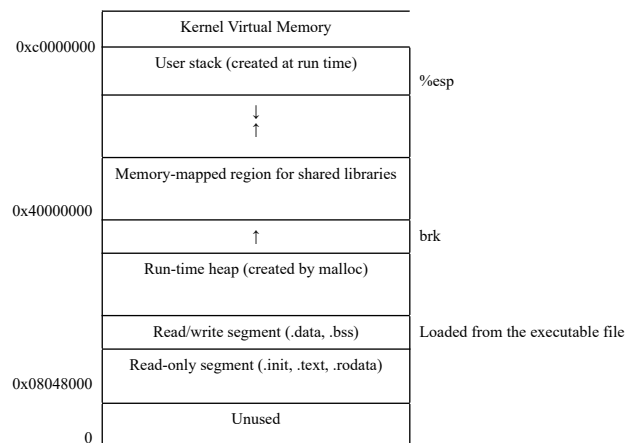
- 在参数bprm->buf中存储了文件的前128字节；
- 该函数先对欲载入文件的格式进行检查，判断是否正确。如果不正确，返回错误标识ENOEXEC，于是函数do_execve()可以继续进行检查。
- 如果判断结果是正确的，这个文件将被载入。然后，将原进程所分配的内存释放。
- 将文件的代码段和数据段使用do_mmap()函数载入内存。
- 使用set_brk()函数载入BSS段，接着将寄存器特别是指令寄存器初始化。
- 在系统调用execve结束时，使用start_thread()开始让进程从新的地址开始执行。

北京航空航天大学

计算机/软件学院

57

57



58

